One-Pass AUC Optimization

Wei Gao[a], Lu Wang[a], Rong Jin[b], Shenghuo Zhu[c], Zhi-Hua Zhou[a,*]

[a]*National Key Laboratory for Novel Software Technology,*
Collaborative Innovation Center of Novel Software Technology and Industrialization
Nanjing University, Nanjing 210023, China
[b]*Department of Computer Science and Engineering,*
Michigan State University, East Lansing, USA
Institute of Data Science and Technologies at Alibaba Group, Seattle, USA
[c]*Institute of Data Science and Technologies at Alibaba Group, Seattle, USA*

Abstract

AUC is an important performance measure that has been used in diverse tasks, such as class-imbalanced learning, cost-sensitive learning, learning to rank, etc. In this work, we focus on one-pass AUC optimization that requires going through training data only once without having to store the entire training dataset. Conventional online learning algorithms cannot be applied directly to one-pass AUC optimization because AUC is measured by a sum of losses defined over pairs of instances from different classes. We develop a regression-based algorithm which only needs to maintain the first and second-order statistics of training data in memory, resulting in a storage requirement independent of the number of training data. To efficiently handle high-dimensional data, we develop two deterministic algorithms that approximate the covariance matrices. We verify, both theoretically and empirically, the effectiveness of the proposed algorithms.

Keywords: AUC, ROC Curve, online learning, large-scale learning, least square loss, random projection

1. Introduction

The Area Under the ROC (Receiver Operating Characteristics) Curve, or simply AUC (Metz, 1978; Hanley and McNeil, 1983), has been an important performance measure in many learning tasks such as class-imbalanced

[*] Email: zhouzh@lamda.nju.edu.cn

learning, cost-sensitive learning, information retrieval, etc., (Provost et al., 1998; Cortes and Mohri, 2004; Liu et al., 2009; Flach et al., 2011). AUC is preferable to accuracy as an evaluation measure in various real applications. For example, some categories may have more instances than others in class-imbalanced tasks such as face detection and collaborative filtering, and the level of imbalance (ratio of size of majority category to minority category) can be as high as 10^6 (Wu et al., 2008); therefore, accuracy is not suitable in such cases since the majority classifier will always perform well in terms of accuracy. AUC has also been used to measure the quality of ranking positive instances over negative ones for information retrieval and ranking problems. Many approaches have been developed to optimize AUC (Herschtal and Raskutti, 2004; Joachims, 2006; Rudin and Schapire, 2009; Kotlowski et al., 2011; Zhao et al., 2011; Gao et al., 2013).

In this work, we focus on AUC optimization that requires only one pass over training examples with storage independent of the data size. This is particularly important for applications involving big data or streaming data in which a large volume of data arrives in a short time period, making it infeasible to store the entire dataset in memory before an optimization procedure is applied. Although many online learning algorithms have been developed to find the optimal solution for certain performance measures by scanning the training data only once (Cesa-Bianchi and Lugosi, 2006), few efforts address one-pass AUC optimization.

Unlike the classical classification and regression problems where the loss function can be calculated on a single training example, AUC is measured by the losses defined over pairs of instances from different classes, making it challenging to develop algorithms for one-pass optimization. An online AUC optimization algorithm was proposed by Zhao et al. (2011). It is based on the idea of reservoir sampling, and achieves a regret bound by storing $\sqrt{T}$ instances, where T is the number of training examples. Wang et al. (2012) suggested an online learning algorithm with fixed-size buffer via a FIFO strategy, while Kar et al. (2013) made use of the reservoir idea and replacement-subsampling technique to develop another online algorithm for pairwise loss functions. Ideally, for one-pass approaches, the storage required by the learning process should be independent of the amount of training data, which is the goal of this work.

1.1. Our Contributions

This work develops the one-pass AUC algorithms, and verifies the effectiveness of the proposed algorithms both theoretically and empirically. The main contributions can be summarized as follows:

- We propose a regression-based algorithm for one-pass AUC optimization where the least square loss is used to measure the ranking error between two instances from different classes. The main advantage of using the least square loss lies in the fact that we only need to store the first and second-order statistics over the received training examples. Consequently, the storage requirement is reduced to $O(d^2)$, where d is the dimension of data, independent of the number of training examples.

- For high-dimensional dense data, we make use of the *frequent direction* algorithm (Liberty, 2013) to approximate the covariance matrix by low-rank matrix. For high-dimensional sparse data, we introduce another deterministic algorithm, called *sparse matrix algorithm*, where the basic idea is to approximate the covariance matrix by a sparse matrix that nullifies smaller elements.

- Theoretically, the pairwise least square loss is proved to be consistent with AUC in finite instance spaces. Then, we present regret bounds with respect to pairwise least square loss when the covariance matrices are provided and approximated, respectively. Finally, we present new generalization and online-to-batch bounds for the proposed algorithms.

- Extensive experiments show the effectiveness of the proposed methods.

A preliminary version of this work appeared in a conference paper (Gao et al., 2013). Compared with the original version, we introduce two new approaches for high-dimensional tasks, i.e., the frequent direction (Liberty, 2013) and the sparse approach are proposed to approximate the covariance matrices for dense and sparse datasets, respectively. We also give new regret, generalization and online-to-batch bounds for the proposed algorithms, and present better empirical performance.

1.2. Related Work

The study of AUC dates back to the 1970s in signal detection theory (Egan, 1975). AUC has been an important measure used in the machine learning literature (Provost et al., 1998; Provost and Fawcett, 2001; Elkan, 2001; Cortes and Mohri, 2004; Huang and Ling, 2005; Clemenćon et al., 2008; Hand, 2009; Flach et al., 2011). AUC can be estimated under parametric (Zhou et al., 2002), semi-parametric (Hsieh and Turnbull, 1996) or non-parametric (Hanley and McNeil, 1982) settings. The non-parametric estimation of AUC has been popular in machine learning since it is equivalent to the Wilcoxon-Mann-Whitney (WMW) statistic test of ranks (Hanley and McNeil, 1982).

Various algorithms have been developed to optimize AUC, such as boosting (Freund et al., 2003; Rudin and Schapire, 2009), SVM (Brefeld and Scheffer, 2005; Joachims, 2005, 2006), and a gradient descent algorithm (Herschtal and Raskutti, 2004). Moreover, Kotlowski et al. (2011) studied the use of univariate losses and Zhao et al. (2011) proposed the first online algorithm for AUC optimization. All those approaches require to store the entire or partial training data and scan the data multiple times.

Much theoretical work has been devoted to understanding the generalization of AUC approaches (Agarwal et al., 2005; Usunier et al., 2005; Cortes et al., 2007; Clemenćon et al., 2008; Agarwal and Niyogi, 2009; Rudin and Schapire, 2009; Wang et al., 2012; Kar et al., 2013). Agarwal and Roth (2005) presented separately a sufficient and necessary condition for AUC learnability. Gao and Zhou (2013) further proved stability as an equivalent condition. The consistency of AUC pairwise and univariate optimization has been studied in (Menon and Williamson, 2014; Gao and Zhou, 2015) and (Kotlowski et al., 2011; Agarwal, 2013), respectively.

1.3. Organization

Section 2 introduces some preliminaries. Sections 3 proposes the OPAUC (One-Pass AUC) framework. Section 4 presents the approximated OPAUC approaches for high-dimensional tasks. Section 5 provides theoretical justifications. Section 6 gives detailed proofs. Section 7 shows extensive experiments, and Section 8 concludes this work.

2. Preliminaries

Let $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} = \{+1, -1\}$ be the instance and label space, respectively. Denote $\mathcal{D}$ by an unknown (underlying) distribution over the product space $\mathcal{X} \times \mathcal{Y}$. Let $\mathcal{S} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_T, y_T)\}$ be a training sample, where each element is drawn identically and independently (i.i.d.) from distribution $\mathcal{D}$. For an integer $n > 0$ and a real $\alpha > 0$, let $[n] = \{1, 2, \ldots, n\}$, and denote $\lfloor \alpha \rfloor$ by the largest integer which is no more than α. For a set $\mathcal{A}$, let $|\mathcal{A}|$ denote its cardinality.

Let $f : \mathcal{X} \to \mathbb{R}$ be a real-valued function. Given a sample $\mathcal{S}$, the AUC of function f is defined as

$$\mathrm{AUC}(f, \mathcal{S}) = \sum_{i=1}^{T} \sum_{j=1}^{T} \frac{(\mathbb{I}[f(\mathbf{x}_i) > f(\mathbf{x}_j)] + \frac{1}{2}\mathbb{I}[f(\mathbf{x}_i) = f(\mathbf{x}_j)])\mathbb{I}[y_i > y_j]}{T_{\mathcal{S}}^{+} T_{\mathcal{S}}^{-}},$$

where $\mathbb{I}[\cdot]$ is the indicator function which returns 1 if the argument is true and 0 otherwise. Here

$$T_{\mathcal{S}}^{+} = |\{(\mathbf{x}_i, y_i) \in \mathcal{S} : y_i = +1\}| \quad \text{and} \quad T_{\mathcal{S}}^{-} = |\{(\mathbf{x}_i, y_i) \in \mathcal{S} : y_i = -1\}|.$$

Direct optimization of AUC often yields an NP-hard problem since it can be cast into a combinatorial optimization problem. A feasible solution in practice is to optimize some pairwise surrogate losses as follows:

$$\begin{aligned} \mathcal{L}(f, \mathcal{S}) &= \sum_{i=1}^{T} \sum_{j=1}^{T} \frac{\ell(f(\mathbf{x}_i) - f(\mathbf{x}_j))\mathbb{I}[y_i > y_j]}{T_{\mathcal{S}}^{+} T_{\mathcal{S}}^{-}} \\ &= \sum_{i=1}^{T} \sum_{j=1}^{i-1} \frac{\ell(y_i(f(\mathbf{x}_i) - f(\mathbf{x}_j)))\mathbb{I}[y_i \neq y_j]}{T_{\mathcal{S}}^{+} T_{\mathcal{S}}^{-}} \end{aligned}$$

where $\ell : \mathbb{R} \to \mathbb{R}^{+}$ is a convex function such as exponential loss $\ell(t) = e^{-t}$, hinge loss $\ell(t) = \max(0, 1 - t)$, logistic loss $\ell(t) = \log(1 + e^{-t})$, etc. The loss function $\ell(y_i(f(\mathbf{x}_i) - f(\mathbf{x}_j))$ is also called *pairwise surrogate loss* since it involves two instances from different classes.

Any distribution $\mathcal{D}$ can be specified exactly by the triplet $(\mathcal{D}^{+}, \mathcal{D}^{-}, p)$ as in (Menon and Williamson, 2014), where $\mathcal{D}^{+}(x) = \Pr[x|y = +1]$, $\mathcal{D}^{-}(x) = \Pr[x|y = -1]$ and $p = \Pr[y = +1]$. The expectation over $\mathcal{S}$ can be further decomposed into an expectation over random draws of $T_{\mathcal{S}}^{+}$ and $T_{\mathcal{S}}^{-}$ from $\mathrm{Binomial}(T, p)$, followed by an expectation over draw of samples from $\mathcal{D}^{+}$ and $\mathcal{D}^{-}$, respectively. Based on this decomposition, we have

Proposition 1. *Define the surrogate loss $\mathcal{L}(f, \mathcal{D}) = E_{\mathcal{S}}[\mathcal{L}(f, \mathcal{S})]$ with respect to distribution $\mathcal{D}$. We have*

$$\begin{aligned} \mathcal{L}(f, \mathcal{D}) &= E_{\mathbf{x}_i \sim \mathcal{D}^{+}, \mathbf{x}_j \sim \mathcal{D}^{-}}[\ell(f(\mathbf{x}_i) - f(\mathbf{x}_j))] && (1) \\ &= E_{(\mathbf{x}_i, y_i) \sim \mathcal{D}, (\mathbf{x}_j, y_j) \sim \mathcal{D}}[\ell(f(\mathbf{x}_i) - f(\mathbf{x}_j))|y_i > y_j]. && (2) \end{aligned}$$

The detailed proof is given in Section 6.1.

Many learning approaches and studies fall into this formulation (Herschtal and Raskutti, 2004; Cortes and Mohri, 2004; Agarwal et al., 2005; Brefeld and Scheffer, 2005; Rudin and Schapire, 2009). There has been previous work of (Wang et al., 2012; Kar et al., 2013), which considered the following formulation:

$$E_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}}\left[\ell(y_i(f(\mathbf{x}_i)-f(\mathbf{x}_j))\mathbb{I}[y_i\neq y_j])\right]. \tag{3}$$

Notice that

$$\mathcal{L}(f,\mathcal{D})\neq E_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}}\left[\ell(y_i(f(\mathbf{x}_i)-f(\mathbf{x}_j))\mathbb{I}[y_i\neq y_j])\right].$$

Thus, our formulation is different from the work of (Wang et al., 2012; Kar et al., 2013). Ying and Zhou (2015) considered the least square loss as surrogate loss and presented an online pairwise algorithm with kernels. However, this work requires to store the entire training sample for kernel tricks and follows the optimization formulation (given by Eqn. 3) as in the work of (Wang et al., 2012; Kar et al., 2013).

3. OPAUC

To address the challenge of one-pass AUC optimization, we propose to use the least square loss $\ell(t)=(1-t)^2$, and focus on a linear space $\mathcal{W}\subseteq\mathbb{R}^d$. Given a sample $\mathcal{S}$, we consider the following pairwise least square loss:

$$\mathcal{L}(\mathbf{w},\mathcal{S})=\frac{\lambda}{2}|\mathbf{w}|^2+\frac{1}{2}\sum_{i=1}^{T}\sum_{j=1}^{i-1}\frac{(1-y_i(\mathbf{x}_i-\mathbf{x}_j)^\top\mathbf{w})^2\mathbb{I}[y_i\neq y_j]}{T_\mathcal{S}^+T_\mathcal{S}^-} \tag{4}$$

where λ is a regularization parameter that controls the model complexity, and the constant $1/2$ is introduced for simplicity. Further, we define the pairwise least square loss with respect to distribution $\mathcal{D}$ as

$$\begin{aligned}\mathcal{L}(\mathbf{w},\mathcal{D}) &= E_\mathcal{S}[\mathcal{L}(\mathbf{w},\mathcal{S})]\\ &= \frac{\lambda}{2}|\mathbf{w}|^2+\frac{1}{2}E_{\mathbf{x}_i\sim\mathcal{D}^+,\,\mathbf{x}_j\sim\mathcal{D}^-}[(1-(\mathbf{x}_i-\mathbf{x}_j)^\top\mathbf{w})^2]\\ &= \frac{\lambda}{2}|\mathbf{w}|^2+\frac{1}{2}E_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}}[(1-(\mathbf{x}_i-\mathbf{x}_j)^\top\mathbf{w})^2|y_i>y_j].\end{aligned}$$

The main advantage of using the least square loss lies in the fact that it is sufficient to store the first and second-order statistics over training examples for optimization, leading to a memory requirement of $O(d^2)$, which is independent of the number of training examples. Another advantage is that the least square loss is consistent with AUC in finite instance space (Theorem 1 in Section 5), whereas loss functions such as hinge loss are proven to be inconsistent with AUC (Gao and Zhou, 2015).

In the online/stochastic setting, we will optimize a variant of the objective in Eqn. 4 that can be written as a sum of losses for individual training instance

$$\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_t(\mathbf{w}) \qquad \text{where}$$

$$\mathcal{L}_t(\mathbf{w}) = \frac{\lambda}{2}|\mathbf{w}|^2 + \frac{\sum_{i=1}^{t-1}\mathbb{I}[y_i \neq y_t](1 - y_t(\mathbf{x}_t - \mathbf{x}_i)^\top\mathbf{w})^2}{2|\{i \in [t-1] : y_i y_t = -1\}|}. \tag{5}$$

It is easy to see that $\mathcal{L}_t(\mathbf{w})$ is an unbiased estimator of $\mathcal{L}(\mathbf{w}, \mathcal{D})$ by the following proposition. The detailed proof is given in Section 6.2.

Proposition 2. *We have* $\mathcal{L}(\mathbf{w}, \mathcal{D}) = E_{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_t,y_t)\sim\mathcal{D}^t}[\mathcal{L}_t(\mathbf{w})]$.

Another main difference from (Wang et al., 2012; Kar et al., 2013) is the normalization term $|\{i \in [t-1] : y_i y_t = -1\}|$, which is dependent on the received data $\mathcal{S}_t$, and it is essential to the derivations of first and second-order statistics. In (Wang et al., 2012; Kar et al., 2013), however, this normalization term is fixed by $1/t$, which is only dependent on the time step t.

Let T_t^+ and T_t^- denote the cardinalities of positive and negative instances in $\mathcal{S}_t$, respectively. Further, we define $\mathcal{L}_t(\mathbf{w}) = 0$ for $T_t^+ T_t^- = 0$. If $y_t = 1$, we calculate the gradient as

$$\nabla\mathcal{L}_t(\mathbf{w}) = \lambda\mathbf{w} + \mathbf{x}_t\mathbf{x}_t^\top\mathbf{w} - \mathbf{x}_t + \sum_{i:\, i<t,\, y_i=-1}\frac{\mathbf{x}_i + (\mathbf{x}_i\mathbf{x}_i^\top - \mathbf{x}_i\mathbf{x}_t^\top - \mathbf{x}_t\mathbf{x}_i^\top)\mathbf{w}}{T_t^-}. \tag{6}$$

It is easy to observe that

$$c_t^- = \sum_{i:\, i<t,\, y_i=-1}\frac{\mathbf{x}_i}{T_t^-} \quad \text{and} \quad S_t^- = \sum_{i:\, i<t,\, y_i=-1}\frac{\mathbf{x}_i\mathbf{x}_i^\top - \mathbf{c}_t^-[\mathbf{c}_t^-]^\top}{T_t^-}$$

correspond to the mean and covariance matrix of negative instances, respectively; therefore, Eqn. 6 can be further simplified as

$$\nabla\mathcal{L}_t(\mathbf{w}) = \lambda\mathbf{w} - \mathbf{x}_t + \mathbf{c}_t^- + (\mathbf{x}_t - \mathbf{c}_t^-)(\mathbf{x}_t - \mathbf{c}_t^-)^\top\mathbf{w} + S_t^-\mathbf{w}. \tag{7}$$

Algorithm 1 The OPAUC algorithm

```
Input: The regularization parameter $\lambda > 0$ and stepsizes $\{\eta_t\}_{t=1}^{T}$.
Initialize: Set $T_0^+ = T_0^- = 0$, $\mathbf{c}_0^+ = \mathbf{c}_0^- = \mathbf{0}$, $\mathbf{w}_0 = \mathbf{0}$ and $\Gamma_0^+ = \Gamma_0^- = [\mathbf{0}]_{d\times u}$ for some $u > 0$
for $t = 1, 2, \dots, T$ do
    Receive a training example $(\mathbf{x}_t, y_t)$
    if $y_t = +1$ then
        $T_t^+ = T_{t-1}^+ + 1$ and $T_t^- = T_{t-1}^-$;
        $\mathbf{c}_t^+ = \mathbf{c}_{t-1}^+ + \frac{1}{T_t^+}(\mathbf{x}_t - \mathbf{c}_{t-1}^+)$ and $\mathbf{c}_t^- = \mathbf{c}_{t-1}^-$;
        Update $\Gamma_t^+$ and $\Gamma_t^-$;
        Calculate the gradient $\widehat{g}_t(\mathbf{w}_{t-1})$
    else
        $T_t^- = T_{t-1}^- + 1$ and $T_t^+ = T_{t-1}^+$;
        $\mathbf{c}_t^- = \mathbf{c}_{t-1}^- + \frac{1}{T_t^-}(\mathbf{x}_t - \mathbf{c}_{t-1}^-)$ and $\mathbf{c}_t^+ = \mathbf{c}_{t-1}^+$;
        Update $\Gamma_t^+$ and $\Gamma_t^-$;
        Calculate the gradient $\widehat{g}_t(\mathbf{w}_{t-1})$
    end if
    Update $\mathbf{w}_t = \mathbf{w}_{t-1} - \eta_t \widehat{g}_t(\mathbf{w}_{t-1})$
end for
Output: $\mathbf{w}_T$
```

In a similar manner for $y_t = -1$, we calculate the following gradient:

$$\nabla\mathcal{L}_t(\mathbf{w}) = \lambda\mathbf{w} + \mathbf{x}_t - \mathbf{c}_t^+ + (\mathbf{x}_t - \mathbf{c}_t^+)(\mathbf{x}_t - \mathbf{c}_t^+)^\top\mathbf{w} + S_t^+\mathbf{w} \tag{8}$$

where

$$\mathbf{c}_t^+ = \sum_{i:\, i<t,\, y_i=1} \frac{\mathbf{x}_i}{T_t^+} \quad \text{and} \quad S_t^+ = \sum_{i:\, i<t,\, y_i=1} \frac{\mathbf{x}_i\mathbf{x}_i^\top - \mathbf{c}_t^+[\mathbf{c}_t^+]^\top}{T_t^+}$$

denote the mean and covariance matrix of positive instances, respectively.

The storage cost for keeping the class means ($\mathbf{c}_t^+$ and $\mathbf{c}_t^-$) and covariance matrices (S_{t-1}^+ and S_{t-1}^-) is $O(d^2)$. Once we compute the gradient $\nabla\mathcal{L}_t(\mathbf{w})$, by the theory of gradient descent, the classifier can be updated by

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \eta_t\nabla\mathcal{L}_t(\mathbf{w}_{t-1}),$$

where η_t is the stepsize in the t-th iteration.

Algorithm 1 presents a generic algorithm, which highlights the key steps. We initialize $\Gamma_0^- = \Gamma_0^+ = [\mathbf{0}]_{d\times d}$, where $u = d$. At each iteration, we denote $\Gamma_t^+ = S_t^+$ and $\Gamma_t^- = S_t^-$. In Line 6, we update $\Gamma_t^- = \Gamma_{t-1}^-$ and

$$\Gamma_t^+ = \Gamma_{t-1}^+ + \mathbf{c}_{t-1}^+[\mathbf{c}_{t-1}^+]^\top - \mathbf{c}_t^+[\mathbf{c}_t^+]^\top + (\mathbf{x}_t\mathbf{x}_t^\top - \Gamma_{t-1}^+ - \mathbf{c}_{t-1}^+[\mathbf{c}_{t-1}^+]^\top)/T_t^+,$$

whereas in Line 11, we update $\Gamma_t^+ = \Gamma_{t-1}^+$ and

$$\Gamma_t^- = \Gamma_{t-1}^- + \mathbf{c}_{t-1}^-[\mathbf{c}_{t-1}^-]^\top - \mathbf{c}_t^-[\mathbf{c}_t^-]^\top + (\mathbf{x}_t\mathbf{x}_t^\top - \Gamma_{t-1}^- - \mathbf{c}_{t-1}^-[\mathbf{c}_{t-1}^-]^\top)/T_t^-.$$

Finally, the gradient $\widehat{g}_t(\mathbf{w}_{t-1})$ of Lines 7 and 12 in Algorithm 1 are given by $\nabla\mathcal{L}_t(\mathbf{w}_{t-1})$ that are calculated by Eqs. 7 and 8, respectively.

Notice that $\Gamma_t^+ = S_t^+$ and $\Gamma_t^- = S_t^-$ are only specific to this section. Throughout this work, S_t^+ and S_t^- denote the covariance matrices of positive and negative instances, respectively, whereas Γ_t^+ and Γ_t^- take different values for different approximation algorithms in Section 4.

4. Handling High Dimensions

One limitation of the OPAUC algorithm is the $O(d^2)$ storage for two covariance matrices S_t^+ and S_t^-, making it unsuitable for high-dimensional data. A natural idea is to first project the high-dimensional data into a low-dimensional space by dimensionality reduction (PCA, hashing, random projection, etc.), and then apply the OPAUC algorithm. This strategy, however, does not work empirically (Section 7) because much information is lost in dimensionality reduction.

Let X_t^+ and X_t^- denote the matrices of positive and negative instances in $\mathcal{S}_t = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_t, y_t)\}$, respectively. Then, we have

$$S_t^+ = \frac{1}{T_t^+}X_t^+[X_t^+]^\top - c_t^+[c_t^+]^\top \text{ and } S_t^- = \frac{1}{T_t^-}X_t^-[X_t^-]^\top - c_t^-[c_t^-]^\top. \qquad (9)$$

Therefore, it suffices to approximate $X_t^+[X_t^+]^\top$ and $X_t^-[X_t^-]^\top$ since we can store the class means c_t^+ and c_t^- in memory. In this section, we will introduce two deterministic methods to approximate covariance matrices for high-dimensional dense and sparse tasks, respectively.

Algorithm 2 Frequent direction

```
Input: instance $\mathbf{x} \in \mathbb{R}^d$ and sketch matrices $Z \in \mathbb{R}^{d\times\tau}$
  Insert $\mathbf{x}$ into a column of $Z$ with zeros elements
  if $Z$ has no column with zeros elements then
    $[U, \Sigma, V] = \mathrm{SVD}(Z)$
    $\hat{\Sigma} = \sqrt{\max(\Sigma^2 - \mathbf{I}_{d\times\tau}\sigma^2_{\lfloor\tau/2\rfloor}, [\mathbf{0}]_{d\times\tau})}$, where $\max(\cdot)$ denotes an element-wise maximum.
    $Z = U\hat{\Sigma}$
  end if
Output: $Z$
```

4.1. High-Dimensional Dense Data

For high-dimensional dense data, we make use of the frequent direction method (Liberty, 2013) to approximate the covariance matrices, because this method works well in practice, and takes faster convergence rate for approximation error than random projection, hashing, etc. The basic idea is to maintain two $d \times \tau$ sketch matrices Z_t^+ and Z_t^-, and use $Z_t^+[Z_t^+]^\top$ and $Z_t^-[Z_t^-]^\top$ to approximate $X_t^+[X_t^+]^\top$ and $X_t^-[X_t^-]^\top$, respectively. Here τ is the sketch size.

More specifically, we receive an example $(\mathbf{x}_t, y_t)$ in the t-th iteration, and assume $y_t = +1$ (a similar procedure works for $y_t = -1$). If there are unfilled columns in Z_t^+, then we add $\mathbf{x}_t$ as a column vector to Z_t^+; otherwise, we nullify half of the columns in Z_t^+ and then add $\mathbf{x}_t$ as the $\lfloor\tau/2\rfloor + 1$ column in Z_t^+. The nullification procedure can be decomposed as (i) apply singular value decomposition (SVD) to compute the singular values and vectors of Z_t^+, and (ii) 'shrink' columns such that at least half of columns in Z_t^+ are zeros and therefore open to be filled.

Algorithm 2 presents a description of frequent direction. Let $\mathrm{SVD}(Z) = [U, \Sigma, V]$ be the singular value decomposition, i.e., $Z = U\Sigma V^\top$, $U^\top U = \mathbf{I}_d$, $V^\top V = \mathbf{I}_\tau$, and Σ is a rectangular diagonal matrix of size $d \times \tau$ with non-negative diagonal elements $\sigma_1, \sigma_2, \ldots, \sigma_\tau$ in non-increasing magnitude order. Here, $\mathbf{I}_\tau$ and $\mathbf{I}_d$ denote the identity matrix of size $\tau \times \tau$ and $d \times d$, respectively. Let $\mathbf{I}_{d\times\tau}$ denote a rectangular diagonal matrix whose diagonal elements are all 1. To shrink at least half of columns of Z, we set $\hat{\Sigma} = \left(\max(\Sigma^2 - \mathbf{I}_{d\times\tau}\sigma^2_{\lfloor\tau/2\rfloor}, [\mathbf{0}]_{d\times\tau})\right)^{1/2}$ and output $U\hat{\Sigma}$. Here $[\mathbf{0}]_{d\times\tau}$ denotes an $d \times \tau$ matrix of zeros and $\max(\cdot)$ denotes an element-wise maximum.

Algorithm 2 outputs $U\hat{\Sigma}$ rather than $U\hat{\Sigma}V^\top$ since we have, by $V^\top V = \mathbf{I}_\tau$,

$$U\hat{\Sigma}[U\hat{\Sigma}]^\top = U\hat{\Sigma}\hat{\Sigma}^\top U^\top = U\hat{\Sigma}V^\top V\hat{\Sigma}^\top U^\top = U\hat{\Sigma}V^\top[U\hat{\Sigma}V^\top]^\top.$$

Moreover, the matrix $U\hat{\Sigma}$ maintains at least half of zero columns.

Let Z_t^+ and Z_t^- be updated according to Algorithm 2, and we approximate covariance matrices S_t^+ and S_t^- by

$$\begin{aligned} \widehat{S}_t^+ &= Z_t^+[Z_t^+]^\top/T_t^+ - \mathbf{c}_t^+[\mathbf{c}_t^+]^\top, \qquad (10)\\ \widehat{S}_t^- &= Z_t^-[Z_t^-]^\top/T_t^- - \mathbf{c}_t^-[\mathbf{c}_t^-]^\top. \qquad (11)\end{aligned}$$

Based on approximated covariance matrices $\widehat{S}_t^+$ and $\widehat{S}_t^-$, the online optimization algorithm essentially tries to minimize $\sum_{t=1}^T \widehat{\mathcal{L}}_t(\mathbf{w})$, where

$$\begin{aligned}\widehat{\mathcal{L}}_t(\mathbf{w}) &= \mathbf{w}^\top(\mathbf{c}_t^- - \mathbf{x}_t) + \lambda|\mathbf{w}|^2/2\\ &\quad +(1+\mathbf{w}^\top\widehat{S}_t^-\mathbf{w})/2 + \mathbf{w}^\top(\mathbf{x}_t - \mathbf{c}_t^-)(\mathbf{x}_t - \mathbf{c}_t^-)^\top\mathbf{w}/2 \qquad (12)\end{aligned}$$

if $y_t = 1$; otherwise,

$$\begin{aligned}\widehat{\mathcal{L}}_t(\mathbf{w}) &= \mathbf{w}^\top(\mathbf{x}_t - \mathbf{c}_t^+) + \lambda|\mathbf{w}|^2/2\\ &\quad +(1+\mathbf{w}^\top\widehat{S}_t^+\mathbf{w})/2 + \mathbf{w}^\top(\mathbf{x}_t - \mathbf{c}_t^+)(\mathbf{x}_t - \mathbf{c}_t^+)^\top\mathbf{w}/2. \qquad (13)\end{aligned}$$

We do not intend to calculate and store the approximated covariance matrices $\widehat{S}_t^+$ and $\widehat{S}_t^-$ explicitly, but to maintain the matrices Z_t^+ and Z_t^- in memory. This is because the gradient $\widehat{g}_t(\mathbf{w})$ based on the approximate covariance matrices can be computed from Z_t^+ and Z_t^- directly, i.e.,

$$\begin{aligned}\widehat{g}_t(\mathbf{w}) &= \mathbf{c}_t^- - \mathbf{x}_t + (\mathbf{x}_t - \mathbf{c}_t^-)(\mathbf{x}_t - \mathbf{c}_t^-)^\top\mathbf{w}\\ &\quad +\lambda\mathbf{w} + (Z_t^-[Z_t^-]^\top/T_t^- - \mathbf{c}_t^-[\mathbf{c}_t^-]^\top)\mathbf{w} \qquad (14)\end{aligned}$$

if $y_t = 1$; otherwise

$$\begin{aligned}\widehat{g}_t(\mathbf{w}) &= \mathbf{x}_t - \mathbf{c}_t^+ + (\mathbf{x}_t - \mathbf{c}_t^+)(\mathbf{x}_t - \mathbf{c}_t^+)^\top\mathbf{w}\\ &\quad +\lambda\mathbf{w} + (Z_t^+[Z_t^+]^\top/T_t^+ - \mathbf{c}_t^+[\mathbf{c}_t^+]^\top)\mathbf{w}. \qquad (15)\end{aligned}$$

We require a memory of $O(\tau d)$ instead of $O(d^2)$ to calculate $\widehat{g}_t(\mathbf{w})$ by using the trick $A[A]^\top\mathbf{w} = A([A]^\top\mathbf{w})$, where $A \in \mathbb{R}^{d\times 1}$ or $\mathbb{R}^{d\times\tau}$.

To implement the approximate approach, we initialize $\Gamma_0^- = \Gamma_0^+ = [\mathbf{0}]_{d\times\tau}$ in Algorithm 1. In Line 6 of Algorithm 1, we update $\Gamma_t^- = \Gamma_{t-1}^-$, and update Γ_t^+ by Algorithm 2 with inputs x_t and Γ_{t-1}^+; in Line 11, we update $\Gamma_t^+ = \Gamma_{t-1}^+$, and update Γ_t^- by Algorithm 2 with inputs x_t and Γ_{t-1}^-. We calculate the gradient $\widehat{g}_t(\mathbf{w}_{t-1})$ of Lines 7 and 12 by Eqs. 14 and 15, respectively.

Algorithm 3 The OPAUCs algorithm

```
Input: The regularization parameter $\lambda > 0$ and stepsizes $\{\eta_t\}_{t=1}^{T}$.
Initialize: Set $T_0^+ = T_0^- = 0$, $\mathbf{c}_0^+ = \mathbf{c}_0^- = \mathbf{0}$, $\mathbf{w}_0 = \mathbf{0}$ and $\Gamma_0^+ = \Gamma_0^- = \mathbf{0}_{d\times d}$
for $t = 1, 2, \ldots, T$ do
    Receive a training example $(\mathbf{x}_t, y_t)$
    if $y_t = +1$ then
        $T_t^+ = T_{t-1}^+ + 1$ and $T_t^- = T_{t-1}^-$;
        $\mathbf{c}_t^+ = \mathbf{c}_{t-1}^+ + \frac{1}{T_t^+}(\mathbf{x}_t - \mathbf{c}_{t-1}^+)$ and $\mathbf{c}_t^- = \mathbf{c}_{t-1}^-$;
        Update $\Gamma_t^+ = \Gamma_{t-1}^+ + \mathbf{x}_t\mathbf{x}_t^\top$ and $\Gamma_t^- = \Gamma_{t-1}^-$;
        if $nnz(\Gamma_t^+) > d\tau$ then
            Keep only $d\tau$ largest elements in $\Gamma_t^+$ and nullify the others;
        end if
        Calculate the gradient $\widehat{g}_t(\mathbf{w}_{t-1})$;
    else
        $T_t^- = T_{t-1}^- + 1$ and $T_t^+ = T_{t-1}^+$;
        $\mathbf{c}_t^- = \mathbf{c}_{t-1}^- + \frac{1}{T_t^-}(\mathbf{x}_t - \mathbf{c}_{t-1}^-)$ and $\mathbf{c}_t^+ = \mathbf{c}_{t-1}^+$;
        Update $\Gamma_t^- = \Gamma_{t-1}^- + \mathbf{x}_t\mathbf{x}_t^\top$ and $\Gamma_t^+ = \Gamma_{t-1}^+$;
        if $nnz(\Gamma_t^-) > d\tau$ then
            Keep only $d\tau$ largest elements in $\Gamma_t^-$ and nullify the others;
        end if
        Calculate the gradient $\widehat{g}_t(\mathbf{w}_{t-1})$;
    end if
    Update $\mathbf{w}_t = \mathbf{w}_{t-1} - \eta_t\widehat{g}_t(\mathbf{w}_{t-1})$
end for
Output: $\mathbf{w}_T$
```

4.2. High-Dimensional Sparse Data

In this section, we introduce a *matrix sparsification* algorithm to handle high-dimensional sparse vectors, which shows better performance than random projection, hashing and frequent direction. Our basic idea is to seek two sparse PSD matrices Z_t^+ and Z_t^- which minimize

$$\min_{\mathrm{nnz}(Z_t^+)\le d\tau} \|X_t^+[X_t^+]^\top - Z_t^+\|_F \quad \text{and} \quad \min_{\mathrm{nnz}(Z_t^-)\le d\tau} \|X_t^-[X_t^-]^\top - Z_t^-\|_F$$

where $\|A\|_F$ denotes the Frobenius norm of matrix A, and $\mathrm{nnz}(A)$ denotes the cardinality of non-zero elements in matrix A. It is easy to find that the optimal solutions for the above problem are two sparse matrices Z_t^+ and Z_t^- which maintain $d\tau$ largest element in $X_t^+[X_t^+]^\top$ and $X_t^-[X_t^-]^\top$, respectively.

We will take Γ_t^+ and Γ_t^- as approximations for $X_t^+[X_t^+]^\top$ and $X_t^-[X_t^-]^\top$, respectively. Specifically, we receive an example $(\mathbf{x}_t, y_t)$ in the t-th iteration, and assume $y_t = +1$ (a similar procedure works for $y_t = -1$). We first update $\Gamma_t^+ = \Gamma_{t-1}^+ + \mathbf{x}_t[\mathbf{x}_t]^\top$, and then sparsify Γ_t^+ by only keeping the largest $d\tau$ elements in Γ_t^+. Therefore, the covariance matrices S_t^+ and S_t^- can be approximated, respectively, by

$$\widehat{S}_t^+ = \Gamma_t^+/T_t^+ - \mathbf{c}_t^+[\mathbf{c}_t^+]^\top \quad \text{and} \quad \widehat{S}_t^- = \Gamma_t^-/T_t^- - \mathbf{c}_t^-[\mathbf{c}_t^-]^\top.$$

Based on approximated covariance matrices $\widehat{S}_t^+$ and $\widehat{S}_t^-$, the gradient $\widehat{g}_t(\mathbf{w})$ can be computed as

$$\widehat{g}_t(\mathbf{w}) = \mathbf{c}_t^- - \mathbf{x}_t + (\mathbf{x}_t - \mathbf{c}_t^-)(\mathbf{x}_t - \mathbf{c}_t^-)^\top\mathbf{w} + \lambda\mathbf{w} + (\Gamma_t^-/T_t^- - \mathbf{c}_t^-[\mathbf{c}_t^-]^\top)\mathbf{w}, \quad (16)$$

for $y_t = 1$; otherwise

$$\widehat{g}_t(\mathbf{w}) = \mathbf{x}_t - \mathbf{c}_t^+ + (\mathbf{x}_t - \mathbf{c}_t^+)(\mathbf{x}_t - \mathbf{c}_t^+)^\top\mathbf{w} + \lambda\mathbf{w} + (\Gamma_t^+/T_t^+ - \mathbf{c}_t^+[\mathbf{c}_t^+]^\top)\mathbf{w}. \quad (17)$$

The detailed algorithm is described in Algorithm 3. We calculate the gradient $\widehat{g}_t(\mathbf{w}_{t-1})$ of Lines 10 and 18 in Algorithm 3 by Eqs. 16 and 17, respectively.

5. Theoretical Analysis

Section 5.1 provides the theoretical justification for pairwise least square loss. Sections 5.2 and 5.3 present regret bounds for the proposed algorithms based on covariance matrices and approximated covariance matrices, respectively. Section 5.4 gives new generalization and online-to-batch bounds.

5.1. Consistency Analysis

Many pairwise surrogate losses have been developed for AUC optimization as mentioned in Section 2. An important theoretical problem: what extent minimizing such a pairwise surrogate loss improves actual AUC; in other words, does the expected risk of learning with pairwise surrogate losses converge to the Bayes risk of AUC? Consistency implies that optimizing with a pairwise surrogate loss will yield an optimal solution. Formally, we define the *AUC consistency* as follows.

Definition 1. *The pairwise surrogate loss $\ell(f(\boldsymbol{x}) - f(\boldsymbol{x}'))$ is said to be consistent with AUC if for every sequence $\{f^{\langle n\rangle}(\mathbf{x})\}_{n\geq 1}$, the following holds over all distributions $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$*

$$\textit{if} \quad \mathcal{L}(f^{\langle n\rangle}, \mathcal{D}) \to \inf_f \mathcal{L}(f, \mathcal{D}) \quad \textit{then} \quad AUC(f^{\langle n\rangle}, \mathcal{D}) \to \inf_f AUC(f, \mathcal{D})$$

where the infimum takes over all measurable functions.

AUC consistency is defined on all measurable functions as in the work of (Kotlowski et al., 2011; Agarwal, 2013; Menon and Williamson, 2014). An interesting problem is to study AUC consistency on linear function spaces for further work.

Gao and Zhou (2015) gave a sufficient condition and a necessary condition for AUC consistency based on minimizing pairwise surrogate losses, but it remains open for pairwise least square loss. Menon and Williamson (2014) presented a general consistent analysis by composing sigmoidal link functions, whereas pairwise least square loss cannot be composed with a sigmoidal link. Therefore, previous studies did not provide the consistent analysis on pairwise least square loss.

We show the consistency of pairwise least square loss for finite instance spaces. The detailed proof is given in Section 6.3.

Theorem 1. *For finite instance spaces and least square loss $\ell(t) = (1-t)^2$, the pairwise surrogate loss $\ell(f(\boldsymbol{x}) - f(\boldsymbol{x}'))$ is consistent with AUC.*

5.2. Regret Bounds with Full Covariance Matrices

Let $\mathbf{w}_*$ and L^* be defined, respectively, as

$$\mathbf{w}_* = \arg\min_{\mathbf{w}} \sum_{t=1}^{T} \mathcal{L}_t(\mathbf{w}) \quad \text{and} \quad L^* = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_t(\mathbf{w}_*). \tag{18}$$

We provide the worst-case regret bounds when the full covariance matrices are provided, and it does not require a stochastic (data) sequence. The detailed proof is given in Section 6.4.

Theorem 2. *Let $\|\mathbf{x}_t\| \leq 1$, and let $\mathbf{w}_*, L^*$ be defined in Eqn. 18. We have*

$$\sum_{t=1}^{T} \mathcal{L}_t(\mathbf{w}_{t-1}) - \sum_{t=1}^{T} \mathcal{L}_t(\mathbf{w}_*) \leq \frac{2(4+\lambda)}{\lambda} + 2\sqrt{\frac{4+\lambda}{\lambda} TL^*}$$

by setting the learning rate $\eta_t = 1/(4+\lambda+\sqrt{((4+\lambda)^2 + (4+\lambda)\lambda TL^*})$.

Theorem 2 theoretically shows that the performance of the OPAUC algorithm converges to the performance of a batch algorithm which observes all instances (in hindsight). This theorem gives an $O(1/T)$ bound when $L^* = 0$, and an $O(1/\sqrt{T})$ bound for worst cases, whereas Zhao et al. (2011) achieved at most $O(1/\sqrt{T})$. Our algorithm and regret bounds are independent of the ratio of positive and negative instances, while previous work (Kotlowski et al., 2011; Zhao et al., 2011) is heavily dependent. Wang et al. (2012, 2013) also obtained $O(1/T)$ and $O(1/\sqrt{T})$ regret bounds despite of different formulations and different techniques.

The faster convergence rate of our proposed algorithm owes to the smoothness of least square loss, an important property that has been explored in some studies of stochastic learning (Rakhlin et al., 2012) and generalization error bound analysis (Srebro et al., 2010). It is interesting to further exploit the λ-strongly convexity of $\mathcal{L}_t(\mathbf{w})$, which can lead to a tighter $O(\ln T/T)$ convergence rate by setting $\eta_t = 1/\lambda t$ as shown in (Hazan et al., 2006).

5.3. Regret Bounds with Approximated Covariance Matrices

This section studies the regret bounds when the covariance matrices are approximated. Recall that the covariance matrices are given by

$$S_t^+ = \frac{1}{T_t^+} X_t^+ [X_t^+]^\top - c_t^+ [c_t^+]^\top \text{ and } S_t^- = \frac{1}{T_t^-} X_t^- [X_t^-]^\top - c_t^- [c_t^-]^\top,$$

where X_t^+ and X_t^- denote the matrices of positive and negative instances, respectively. We try to approximate $X_t^+[X_t^+]^\top$ and $X_t^-[X_t^-]^\top$ as done in Algorithms 2 and 3, since it is easy to store the means c_t^+ and c_t^- in memory.

To unify our analysis, let $\hat{Z}_t^+$ and $\hat{Z}_t^-$ denote two positive semi-definite (PSD) matrices approximating $X_t^+[X_t^+]^\top$ and $X_t^-[X_t^-]^\top$, respectively. We write

$$\widehat{S}_t^+ = \hat{Z}_t^+/T_t^+ - c_t^+[c_t^+]^\top \quad \text{and} \quad \widehat{S}_t^- = \hat{Z}_t^-/T_t^- - c_t^-[c_t^-]^\top, \tag{19}$$

and denote

$$\widehat{\mathcal{L}}_t(\mathbf{w}) = \frac{\lambda|\mathbf{w}|^2}{2} + \mathbf{w}^\top(\mathbf{c}_t^- - \mathbf{x}_t) + \frac{1}{2} + \frac{1}{2}\big(\mathbf{w}^\top(\mathbf{x}_t - \mathbf{c}_t^-)(\mathbf{x}_t - \mathbf{c}_t^-)^\top\mathbf{w} + \mathbf{w}^\top\widehat{S}_t^-\mathbf{w}\big),$$

if $y_t = 1$; otherwise,

$$\widehat{\mathcal{L}}_t(\mathbf{w}) = \frac{\lambda|\mathbf{w}|^2}{2} + \mathbf{w}^\top(\mathbf{x}_t - \mathbf{c}_t^+) + \frac{1}{2} + \frac{1}{2}\big(\mathbf{w}^\top(\mathbf{x}_t - \mathbf{c}_t^+)(\mathbf{x}_t - \mathbf{c}_t^+)^\top\mathbf{w} + \mathbf{w}^\top\widehat{S}_t^+\mathbf{w}\big).$$

Notice that $\widehat{\mathcal{L}}_t(\mathbf{w})$ is an approximation of $\mathcal{L}_t(\mathbf{w})$. Let $\widehat{\mathbf{w}}_*$ be given by

$$\widehat{\mathbf{w}}_* = \arg\min_{\mathbf{w}} \sum_{t=1}^{T} \widehat{\mathcal{L}}_t(\mathbf{w}).$$

We define the *stable rank* of positive and negative instances, respectively, as

$$r_t^+ = \frac{1}{T_t^+}\mathrm{tr}(X_t^+[X_t^+]^\top) \quad \text{and} \quad r_t^- = \frac{1}{T_t^-}\mathrm{tr}(X_t^-[X_t^-]^\top),$$

and further denote

$$r = \max_{t=1}^{T}\{r_t^+, r_t^-\}. \tag{20}$$

We assume that the stable rank r is small in this work.

We will present a general result where we assume that there exists a non-increasing function $g(\tau)$ such that

$$\begin{aligned} \|X_t^+[X_t^+]^\top - \hat{Z}_t^+\| &\le g(\tau)\mathrm{tr}(X_t^+[X_t^+]^\top), \\ \|X_t^-[X_t^-]^\top - \hat{Z}_t^-\| &\le g(\tau)\mathrm{tr}(X_t^-[X_t^-]^\top), \end{aligned} \tag{21}$$

where τ is a parameter related to approximated methods. We will later instantiate $g(\tau)$ for the frequent direction method (Algorithm 2). The following theorem presents the worst-case regret bounds for approximated covariance matrices, and it does not require a stochastic (data) sequence.

Theorem 3. *Suppose $\|\mathbf{w}_*\| \le B$ and $\|\mathbf{x}_t\| \le 1$. Let L^*, $g(\tau)$ and r be defined as in Eqns. 18, 21 and 20, respectively. Denote $\beta = 1 + rg(\tau)/\lambda$, $\kappa = 4 + \lambda$ and select $\eta_t = 1/(\kappa + \sqrt{(\kappa^2 + \kappa T L^*/\beta/B^2})$. We have*

$$\sum_{t=1}^{T} \widehat{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \sum_{t=1}^{T} \mathcal{L}_t(\mathbf{w}_*) \le \frac{r}{\lambda} g(\tau) T L^* + 2\kappa\beta^2 B^2 + \beta B\sqrt{2\kappa\beta T L^*}.$$

The proof involves bounding the difference between the optimal model $\mathbf{w}_*$ and optimal approximated model $\widehat{\mathbf{w}}_*$, and then translate this to a bound on the difference between the cumulative losses of $\mathcal{L}_t(\mathbf{w}_*)$ and $\widehat{\mathcal{L}}_t(\widehat{\mathbf{w}}_*)$. The detailed proof is given in Section 6.5.

Theorem 3 theoretically shows the gap between the performance of an algorithm with approximated sample covariance matrices and the performance of a batch algorithm (in hindsight) under the assumption on $g(\tau)$. This theorem gives comparable regret bounds with Theorem 2 for $L^* = 0$ or for small

$rg(\tau)L^*/\lambda$. Also, the additional term $rg(\tau)L^*/\lambda$ can be viewed as the cost of using low-dimensional approximations to covariance matrices.

The function $g(\tau)$ depends on the choice of the approximated algorithm. We first investigate the approximations of $X_t^+[X_t^+]^\top$ and $X_t^-[X_t^-]^\top$ by frequent direction (Liberty, 2013), i.e., $X_t^+[X_t^+]^\top$ and $X_t^-[X_t^-]^\top$ are approximated by $Z_t^+[Z_t^+]^\top$ and $Z_t^-[Z_t^-]^\top$, respectively. Here Z_t^+ and Z_t^- are updated by Algorithm 2 (frequent direction). It is necessary to introduce a helpful lemma from (Liberty, 2013) as follows:

Lemma 1. *If Z_t^+ and Z_t^+ are the outputs of applying the frequent direction to matrices X_t^+ and X_t^-, respectively, then we have*

$$\begin{array}{lll} 0 \preceq Z_t^+[Z_t^+]^\top \preceq X_t^+[X_t^+]^\top & \text{and} & \left\|X_t^+[X_t^+]^\top - Z_t^+[Z_t^+]^\top\right\| \leq \frac{2}{\tau} tr(X_t^+[X_t^+]^\top) \\ 0 \preceq Z_t^-[Z_t^-]^\top \preceq X_t^-[X_t^-]^\top & \text{and} & \left\|X_t^-[X_t^-]^\top - Z_t^-[Z_t^-]^\top\right\| \leq \frac{2}{\tau} tr(X_t^-[X_t^-]^\top) \end{array}$$

where τ is the sketch size.

Therefore, we have $g(\tau) = 2/\tau$ for frequent direction. Combining Theorem 3 with Lemma 1, we derive the regret bounds when the covariance matrices are approximated by frequent direction.

Corollary 1. *Suppose $\|\mathbf{w}_*\| \leq B$ and $\|\mathbf{x}_t\| \leq 1$. Let L^* and r be defined in Eqns. 18 and 20, respectively, and let τ be the sketch size in Algorithm 2. Set $\beta = 1 + 2r/\tau\lambda$, $\kappa = 4 + \lambda$ and $\eta_t = 1/(\kappa + \sqrt{(\kappa^2 + \kappa TL^*/\beta/B^2}))$. We have*

$$\sum_{t=1}^T \widehat{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \sum_{t=1}^T \mathcal{L}_t(\mathbf{w}_*) \leq \frac{2r}{\tau\lambda} TL^* + 2\kappa\beta^2 B^2 + \beta B\sqrt{2\kappa\beta TL^*}.$$

For matrix sparsification of Algorithm 3, it is not easy to give specific expression of $g(\tau)$. However, we can observe empirically that $g(\tau)$ is much smaller than $2/\tau$ from Figure 4 (Section 7.3).

5.4. Generalization Analysis

Our framework and normalization are different from (Wang et al., 2012; Kar et al., 2013) as mentioned in Sections 2 and 3, and the normalization is data-dependent, which makes it difficult to extend the analysis of (Wang et al., 2012; Kar et al., 2013) to our work. This section presents new generalization bounds for our proposed algorithm, and an online-to-batch conversion follows from the generalization analysis.

Suppose that $\mathcal{W}$ is a compact function space, and let $\mathcal{N}(\mathcal{W},\epsilon)$ be the ϵ-covering number with respect to the L_2 norm. We denote $p = \Pr_{(\mathbf{x},y)\sim\mathcal{D}}[y = 1]$ to be the ratio of positive examples under distribution $\mathcal{D}$, and assume $p < 1/2$ without loss of generality. Throughout this section, we denote

$$\begin{aligned}\delta = \exp\Big(\frac{-T\epsilon^2}{16B_1^2}\Big) + 2T\mathcal{N}\Big(\mathcal{W},\frac{\epsilon}{B_2}\Big)\exp\Big(\frac{-Tp^2}{8}\Big)\\ +T\mathcal{N}\Big(\mathcal{W},\frac{\epsilon}{B_2}\Big)\exp\Big(\frac{-(\min\{p,2-3p\})^2T\epsilon^2}{2^{17}B_1^2}\Big)\end{aligned} \quad (22)$$

where $B_1 = ((1+2B)^2+\lambda B^2)/2$ and $B_2 = 16(1+(\lambda+2)B)$. We have

Theorem 4. *Let $\mathcal{W} = \{\mathbf{w}\colon \|\mathbf{w}\| \leq B\}$, $\|\mathbf{x}\| \leq 1$ and $T_0 = \lfloor T/2 \rfloor$. Suppose that $\mathbf{w}_{T_0},\ldots,\mathbf{w}_T \in \mathcal{W}$ are models output by OPAUC. For any $\epsilon > 0$ and sufficiently large T, the following holds with probability at least $1-\delta$ (δ is defined in Eqn. 22) over an i.i.d. sequence $\mathcal{S} = \{(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),\ldots,(\mathbf{x}_T,y_T)\}$*

$$\frac{1}{T-T_0}\sum_{t=T_0+1}^{T}\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D}) - \frac{1}{T-T_0}\sum_{t=T_0+1}^{T}\mathcal{L}_t(\mathbf{w}_{t-1}) < \epsilon.$$

The proof involves the decomposition of the excess risk into a martingale difference sequence and a residual term as in (Wang et al., 2012; Kar et al., 2013). The martingale sequence converges based on the Azuma-Hoeffding inequality, and the residual term is bounded by uniform convergence with cover numbers (Devroye et al., 1996; Rudin and Schapire, 2009).

We cannot make use of the techniques of (Wang et al., 2012) to prove Theorem 4 directly, because the normalization $|\{i \in [t-1] : y_i y_t = -1\}|$ is data-dependent, which may cause large variations of cumulative losses. For example, if $|\{i \in [t-1] : y_i y_t = -1\}|$ is very small even for large t, then the cumulative losses have large variations by randomly replacing an example, and some classical concentrations cannot be applied.

Our strategy is to partition the problems into two cases (based on) whether the fraction of positive instances (in the given sequence) is close to $p = \Pr_{(\mathbf{x},y)\sim\mathcal{D}}[y = 1]$ or not. We use the Hoeffding's inequality to deal with the case when the fraction of positive instances is far from p; this includes the special situation that $|\{i \in [t-1] : y_i y_t = -1\}|$ is small even for large t. For the other case, we show that the variations of cumulative losses are bounded. We finally combine the two cases by the law of total probability. The detailed proof is deferred to Section 6.6.

Theorem 4 presents theoretical analysis on the generalization performance of our proposed OPAUC algorithm. This theorem can be easily generalized to other bounded losses such as hinge loss, exponential loss, etc. In addition, this theorem considers the average of the last $T - T_0$ losses and the first T_0 losses are discarded because of technical reasons, and a similar strategy has been made in (Wang et al., 2012).

Another relevant work (Kar et al., 2013) presents the generalization error bounds based on generalized Rademacher complexity. However, our work cannot make similar extensions easily, because the normalization of cumulative losses is data-dependent, while previous normalizations of Rademacher complexities are all defined data-independently. In addition, it is difficult to make comparisons between the generalization bounds of Theorem 4 and those of (Wang et al., 2012; Kar et al., 2013) because of different framework and normalization.

In the following, we will derive the online-to-batch bounds for OPAUC algorithm. First, we say that an online learning algorithm has a regret bound $\mathfrak{R}_T$ if $\mathbf{w}_{T_0}, \mathbf{w}_{T_0+1}, \dots, \mathbf{w}_{T-1}$ are such that

$$\frac{1}{T - T_0} \sum_{t=T_0+1}^{T} \mathcal{L}_t(\mathbf{w}_{t-1}) \leq \inf_{\mathbf{w} \in \mathcal{W}} \frac{1}{T - T_0} \sum_{t=T_0+1}^{T} \mathcal{L}_t(\mathbf{w}) + \mathfrak{R}_T. \tag{23}$$

From Theorem 2, we can observe $\mathfrak{R}_T = O(1/\sqrt{T})$ for the OPAUC algorithm. This definition is helpful to derive the online-to-batch bounds from generalization bounds, and it has been used in (Kar et al., 2013).

Let $\mathbf{w}_*^* = \arg\min_{\mathbf{w} \in \mathcal{W}} \mathcal{L}(\mathbf{w}, \mathcal{D})$ denote the risk minimizer over the whole distribution $\mathcal{D}$. Similarly to the proof of Theorem 4, we have

$$\frac{1}{T - T_0} \sum_{t=T_0+1}^{T} \mathcal{L}_t(\mathbf{w}_*^*) - \mathcal{L}(\mathbf{w}_*^*, \mathcal{D}) < \epsilon \tag{24}$$

with probability at least $1 - \delta$. Based on Theorem 4 and Eqns. 23 and 24, we have

Theorem 5. *Let $\mathcal{W} = \{\mathbf{w} \colon \|\mathbf{w}\| \leq B\}$, $\|\mathbf{x}\| \leq 1$ and $T_0 = \lfloor T/2 \rfloor$. Suppose that $\mathbf{w}_{T_0}, \dots, \mathbf{w}_{T-1} \in \mathcal{W}$ are output models by OPAUC with a regret bound $\mathfrak{R}_T$. For any $\epsilon > 0$ and sufficient large T, the following holds with* *probability at least* $1 - 2\delta$ *(δ is defined in Eqn. 22) over i.i.d. sequence* $\mathcal{S} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_T, y_T)\}$,

$$\frac{1}{T - T_0} \sum_{t=T_0+1}^{T} \mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D}) - \mathcal{L}(\mathbf{w}_*^*, \mathcal{D}) < 2\epsilon + \mathfrak{R}_T.$$

This theorem presents the online-to-batch bounds for the OPAUC algorithm, and it is interesting to explore other techniques for the online-to-batch conversion. Based on this theorem, we present theoretical analysis on choosing a random stopping time as follows:

Corollary 2. *Let* $\mathcal{W} = \{\mathbf{w} \colon \|\mathbf{w}\| \leq B\}$*,* $\|\mathbf{x}\| \leq 1$ *and* $T_0 = \lfloor T/2 \rfloor$*. Suppose that* $\mathbf{w}_{T_0}, \ldots, \mathbf{w}_{T-1} \in \mathcal{W}$ *are output models by OPAUC with a regret bound* $\mathfrak{R}_T$*. For any* $\epsilon > 0$ *and sufficient large* T*, if we randomly select* $T_0 \leq t < T$*, then the following holds with probability at least* $2/3 - 2\delta$ *(δ is defined in Eqn. 22) over i.i.d. sequence* $\mathcal{S} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_T, y_T)\}$*,*

$$\mathcal{L}(\mathbf{w}_t, \mathcal{D}) - \mathcal{L}(\mathbf{w}_*^*, \mathcal{D}) < 6\epsilon + 3\mathfrak{R}_T.$$

This corollary shows that the output model could have good performance by a random stopping time, and the probability is about 2/3 for sufficient large T. In practice, we pick up the last output model $\mathbf{w}_T$ as in (Langford et al., 2009; Shalev-Shwartz et al., 2011), which gives good empirical performance by experiments (Section 7). The proof follows the technique of (Shalev-Shwartz et al., 2011), and we present the details for completeness in Section 6.7.

6. Proofs

In this section, we will present detailed proofs for our main results.

6.1. Proof of Proposition 1

We first have

$$\mathcal{L}(f, \mathcal{D}) = E_{\mathcal{S}}[\mathcal{L}(f, \mathcal{S})] = E_{\mathcal{S}} \left[\sum_{i=1}^{T} \sum_{j=1}^{T} \frac{\ell(f(\mathbf{x}_i) - f(\mathbf{x}_j)) \mathbb{I}[y_i > y_j]}{T_{\mathcal{S}}^{+} T_{\mathcal{S}}^{-}} \right].$$

The distribution $\mathcal{D}$ can be specified exactly by the triplet $(\mathcal{D}^+, \mathcal{D}^-, p)$, and assume that $\mathbf{x}_1, \ldots, \mathbf{x}_{T_{\mathcal{S}}^+}$ are selected i.i.d from distribution $\mathcal{D}^+$, and $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_{T_{\mathcal{S}}^-}$ are selected i.i.d. from distribution $\mathcal{D}^-$.

Because the expectation over $\mathcal{S}$ can be decomposed into an expectation over random draws of $T_{\mathcal{S}}^{+}$ and $T_{\mathcal{S}}^{-}$ from Binomial(T, p), followed by an expectation over $\mathcal{D}^{+}$ and $\mathcal{D}^{-}$, respectively, $E_{\mathcal{S}}[\mathcal{L}(f, \mathcal{S})]$ is equal to

$$E\left[\frac{1}{T_{\mathcal{S}}^{+}T_{\mathcal{S}}^{-}}E_{\mathbf{x}_1\sim\mathcal{D}^{+},\ldots,\mathbf{x}_{T_{\mathcal{S}}^{+}}\sim\mathcal{D}^{+},\hat{\mathbf{x}}_1\sim\mathcal{D}^{-},\ldots,\hat{\mathbf{x}}_{T_{\mathcal{S}}^{-}}\sim\mathcal{D}^{-}}\left[\sum_{i=1}^{T_{\mathcal{S}}^{+}}\sum_{j=1}^{T_{\mathcal{S}}^{-}}\ell(f(\mathbf{x}_i)-f(\hat{\mathbf{x}}_j))\right]\right]$$

where the outer expectation is over $T_{\mathcal{S}}^{+}$ $\sim$Binomial(T, p) and $T_{\mathcal{S}}^{-} = T - T_{\mathcal{S}}^{+}$. By the linearity of expectation, we have

$$\mathcal{L}(f, \mathcal{D}) = E_{\mathcal{S}}[\mathcal{L}(f, \mathcal{S})] = E\left[\frac{1}{T_{\mathcal{S}}^{+}T_{\mathcal{S}}^{-}}\sum_{i=1}^{T_{\mathcal{S}}^{+}}\sum_{j=1}^{T_{\mathcal{S}}^{-}}E_{\mathbf{x}_i\sim\mathcal{D}^{+},\,\hat{\mathbf{x}}_j\sim\mathcal{D}^{-}}\ell(f(\mathbf{x}_i)-f(\hat{\mathbf{x}}_j))\right]$$

which completes the proof of Eqn. 1 since $E_{\mathbf{x}_i\sim\mathcal{D}^{+},\,\hat{\mathbf{x}}_j\sim\mathcal{D}^{-}}\ell(f(\mathbf{x}_i)-f(\hat{\mathbf{x}}_j))$ have the same values for all $i \in [T_{\mathcal{S}}^{+}]$ and $j \in [T_{\mathcal{S}}^{-}]$, and the denominator $T_{\mathcal{S}}^{+}T_{\mathcal{S}}^{-}$ can be cancelled. For Eqn. 2, we have

$$\begin{aligned}
&E_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}}[\ell(f(\mathbf{x}_i)-f(\mathbf{x}_j))|y_i > y_j]\\
=\;&E_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}}[\ell(f(\mathbf{x}_i)-f(\mathbf{x}_j))|y_i = 1, y_j = -1]\\
=\;&E_{\mathbf{x}_i\sim\mathcal{D}^{+},\,\mathbf{x}_j\sim\mathcal{D}^{-}}[\ell(f(\mathbf{x}_i)-f(\mathbf{x}_j))]
\end{aligned}$$

which completes the proof. □

6.2. Proof of Proposition 2

Let $\mathbf{x}_1, \ldots, \mathbf{x}_{T_{\mathcal{S}_{t-1}^{+}}}$ and $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_{T_{\mathcal{S}_{t-1}^{-}}}$ be drawn i.i.d from distributions $\mathcal{D}^{+}$ and $\mathcal{D}^{-}$, respectively. Recall that $p = \Pr_{(\mathbf{x},y)\sim\mathcal{D}}[y = 1]$. We have

$$\begin{aligned}
&E_{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_t,y_t)\sim\mathcal{D}^t}[\mathcal{L}_t(\mathbf{w})]\\
=\;&\frac{\lambda}{2}|\mathbf{w}|^2 + E_{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_t,y_t)\sim\mathcal{D}^t}\left[\frac{\sum_{i=1}^{t-1}\mathbb{I}[y_i \neq y_t](1-y_t(\mathbf{x}_t-\mathbf{x}_i)^{\top}\mathbf{w})^2}{2|\{i \in [t-1] : y_iy_t = -1\}|}\right]\\
=\;&\frac{\lambda}{2}|\mathbf{w}|^2 + pE_{\mathbf{x}_t\sim\mathcal{D}^{+},\,T_{\mathcal{S}_{t-1}^{-}}}\left[\sum_{i=1}^{T_{\mathcal{S}_{t-1}^{-}}}\frac{E_{\hat{\mathbf{x}}_i\sim\mathcal{D}^{-}}(1-(\mathbf{x}_t-\hat{\mathbf{x}}_i)^{\top}\mathbf{w})^2}{2T_{\mathcal{S}_{t-1}^{-}}}\right]\\
&+(1-p)E_{\mathbf{x}_t\sim\mathcal{D}^{-},\,T_{\mathcal{S}_{t-1}^{+}}}\left[\sum_{i=1}^{T_{\mathcal{S}_{t-1}^{+}}}\frac{E_{\mathbf{x}_i\sim\mathcal{D}^{+}}(1-(\mathbf{x}_i-\mathbf{x}_t)^{\top}\mathbf{w})^2}{2T_{\mathcal{S}_{t-1}^{+}}}\right]\\
=\;&\frac{\lambda}{2}|\mathbf{w}|^2 + \frac{1}{2}E_{\mathbf{x}\sim\mathcal{D}^{+},\,\hat{\mathbf{x}}\sim\mathcal{D}^{-}}[(1-(\mathbf{x}-\hat{\mathbf{x}})^{\top}\mathbf{w})^2]
\end{aligned}$$

which completes the proof. □

6.3. Proof of Theorem 1

We assume a finite instance space $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ with marginal probability $p_i = \Pr[x_i]$ and conditional probability $\xi_i = \Pr[y = +1|\mathbf{x}_i]$. The expected surrogate risk is given by

$$\begin{aligned}\mathcal{L}(f, \mathcal{D}) &= C_0 + C_1 \\ &\times \sum_{i \neq j} p_i p_j (\xi_i(1-\xi_j)\ell(f(\mathbf{x}_i) - f(\mathbf{x}_j)) + \xi_j(1-\xi_i)\ell(f(\mathbf{x}_j) - f(\mathbf{x}_i)))\end{aligned} \quad (25)$$

where $\ell(t) = (1-t)^2$, and C_0 and C_1 are constants and irrelevant to f. Notice that our proof does not rely on the scaled loss function, i.e., the property of consistency also holds for scaled loss function $\gamma\ell(\cdot)$ for any constant $\gamma > 0$, because scaled loss function $\gamma\ell(\cdot)$ corresponds to γC_1, which does not affect the optimal solution.

According to the analysis of (Gao and Zhou, 2015), it suffices to prove that, for every optimal solution f such that $\mathcal{L}(f, \mathcal{D}) = \inf_{f'} \mathcal{L}(f', \mathcal{D})$, we have $f(\mathbf{x}_i) > f(\mathbf{x}_j)$ for $\xi_i > \xi_j$ (equivalent to $f(\mathbf{x}_i) < f(\mathbf{x}_j)$ for $\xi_i < \xi_j$ by swapping i and j).

If $n = 2$, i.e., $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2\}$, then Eqn. 25 gives the expected risk as

$$\begin{aligned}\mathcal{L}(f, \mathcal{D}) = C_0 + C_1 p_1 p_2 \big(&\xi_1(1-\xi_2)(1 - f(\mathbf{x}_1) + f(\mathbf{x}_2))^2 \\ &+ \xi_2(1-\xi_1)(1 - f(\mathbf{x}_2) + f(\mathbf{x}_1))^2\big).\end{aligned} \quad (26)$$

Minimizing Eqn. 26 gives the optimal solution $f = (f(\mathbf{x}_1), f(\mathbf{x}_2))$ such that

$$f(\mathbf{x}_1) - f(\mathbf{x}_2) = \frac{\xi_1 - \xi_2}{\xi_1 + \xi_2 - 2\xi_1\xi_2} \quad \text{for} \quad \xi_1 \neq \xi_2.$$

Therefore, we have $f(\mathbf{x}_1) > f(\mathbf{x}_2)$ if $\xi_1 > \xi_2$; otherwise $f(\mathbf{x}_1) < f(\mathbf{x}_2)$. This shows the consistency of pairwise least square loss.

If $n \geq 3$ and $\xi_i(1-\xi_i) = 0$ for each $i \in [n]$, then each conditional probability satisfies

$$\xi_i = 0 \quad \text{or} \quad \xi_i = 1 \quad \text{for} \quad i \in [n].$$

Combining this with Eqn. 25, we have

$$\mathcal{L}(f, \mathcal{D}) = C_0 + C_1 \sum_{\xi_i = 1, \xi_j = 0} p_i p_j (1 - f(\mathbf{x}_i) + f(\mathbf{x}_j))^2.$$

Minimizing $\mathcal{L}(f, \mathcal{D})$ gives the optimal solution $f = (f(\mathbf{x}_1), f(\mathbf{x}_2), \cdots, f(\mathbf{x}_n))$ such that

$$f(\mathbf{x}_i) = f(\mathbf{x}_j) + 1 \quad \text{for } \xi_i = 1 \text{ and } \xi_j = 0$$

which also shows the consistency of pairwise least square loss.

If $n \geq 3$ and there exists some i_0 s.t. $\xi_{i_0}(1 - \xi_{i_0}) \neq 0$, then the subgradient conditions give optimal solution $f = (f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_n))$ such that

$$\sum_{k \neq i} p_k(\xi_i + \xi_k - 2\xi_i\xi_k)(f(\mathbf{x}_i) - f(\mathbf{x}_k)) = \sum_{k \neq i} p_k(\xi_i - \xi_k) \text{ for each } 1 \leq i \leq n.$$

Solving the above n linear equations, we obtain

$$f(\mathbf{x}_i) - f(\mathbf{x}_j) = (\xi_i - \xi_j) \frac{\prod_{k \neq i,j} \sum_{l=1}^{n} p_l(\xi_l + \xi_k - 2\xi_l\xi_k)}{\sum_{s_1 + \cdots + s_n = n-2, s_i \geq 0} p_1^{s_1} \cdots p_n^{s_n} \Gamma(s_1, s_2, \cdots, s_n)}$$

where Γ is a polynomial in $\xi_{k_1} + \xi_{k_2} - 2\xi_{k_1}\xi_{k_2}$ for $1 \leq k_1, k_2 \leq n$. In the following, we will give the specific expression for $\Gamma(s_1, s_2, \cdots, s_n)$. Let $\mathcal{A} = \{i \colon s_i \geq 1\}$ and $\mathcal{B} = \{i \colon s_i = 0\} = \{b_1, b_2, \cdots, b_{|\mathcal{B}|}\}$.

- If $|\mathcal{A}| = 1$, i.e., $\mathcal{A} = \{i_1\}$ for some $1 \leq i_1 \leq n$, then

$$\Gamma(s_1, s_2, \cdots, s_n) = \prod_{k \in \mathcal{B}} (\xi_{i_1} + \xi_k - 2\xi_{i_1}\xi_k).$$

- If $|\mathcal{A}| = 2$, i.e., $\mathcal{A} = \{i_1, i_2\}$ for some $1 \leq i_1, i_2 \leq n$, then we denote

$$\mathcal{A}_1 = \{s_{i_1} \odot i_1\} \bigcup \{s_{i_2} \odot i_2\}$$

where $\{s_{i_k} \odot i_k\}$ denotes the multi-set $\{i_k, i_k, \ldots, i_k\}$ of size s_{i_k} for $k = 1, 2$. It is clear that $|\mathcal{B}| = |\mathcal{A}_1| = n - 2$. Further, we denote $\mathcal{G}(\mathcal{A}_1)$ by the set of all permutations of $\mathcal{A}_1$. Therefore, we have

$$\Gamma(s_1, s_2, \cdots, s_n) = (\xi_{i_1} + \xi_{i_2} - 2\xi_{i_1}\xi_{i_2}) \sum_{\pi = \pi_1 \cdots \pi_{n-2} \in \mathcal{G}(\mathcal{A}_1)} \prod_{k=1}^{n-2} (\xi_{\pi_i} + \xi_{b_i} - 2\xi_{\pi_i}\xi_{b_i}).$$

- If $|\mathcal{A}| > 2$, then, for $i_1 \neq i_2$ and $i_1, i_2 \in \mathcal{A}$, we denote the multi-set

$$\mathcal{A}_1(i_1, i_2) = \{s_{i_1} \odot i_1\} \bigcup \{s_{i_2} \odot i_2\} \bigcup \Big(\bigcup_{k \in \mathcal{A} \backslash \{i_1, i_2\}} \{(s_k - 1) \odot k\} \Big),$$

and it is easy to derive $|\mathcal{A}_1| = |\mathcal{B}|$. Further, we denote $\mathcal{G}(\mathcal{A}\backslash\{i_1, i_2\})$ and $\mathcal{G}(\mathcal{A}_1)$ by the set of all permutations of $\mathcal{A}\backslash\{i_1, i_2\}$ and $\mathcal{A}_1$, respectively. Therefore, we set

$$\begin{aligned}
&\Gamma_1(i_1, i_2, \mathcal{A}) = \\
&\quad \sum_{\pi=\pi_1\pi_2\cdots\pi_{|\mathcal{A}|-2}\in\mathcal{G}(\mathcal{A}\backslash\{i_1,i_2\})} (\xi_{i_1} + \xi_{\pi_1} - 2\xi_{i_1}\xi_{\pi_1})(\xi_{\pi_1} + \xi_{\pi_2} - 2\xi_{\pi_1}\xi_{\pi_2}) \times \\
&\quad \cdots \times (\xi_{\pi_{|A|-3}} + \xi_{\pi_{|A|-2}} - 2\xi_{\pi_{|A|-3}}\xi_{\pi_{|A|-2}})(\xi_{\pi_{|A|-2}} + \xi_{i_2} - 2\xi_{i_2}\xi_{\pi_{|A|-2}}),
\end{aligned}$$

and we have

$$\begin{aligned}
\Gamma(s_1, s_2, \cdots, s_n) &= \sum_{i_1\neq i_2:\, i_1,i_2\in\mathcal{A}} \Gamma_1(i_1, i_2, \mathcal{A}) \\
&\quad \times \sum_{\pi=\pi_1\pi_2\ldots\pi_{|\mathcal{B}|}\in\mathcal{G}(\mathcal{A}_1)} \prod_{k=1}^{|B|} (\xi_{\pi_k} + \xi_{b_k} - 2\xi_{\pi_k}\xi_{b_k})
\end{aligned}$$

where $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$.

Since there is an i_0 s.t. $\xi_{i_0}(1 - \xi_{i_0}) \neq 0$, we have

$$\frac{\prod_{k\neq i,j} \sum_{l=1}^{n} p_l(\xi_l + \xi_k - 2\xi_l\xi_k)}{\sum_{s_1+\cdots+s_n=n-2, s_i\geq 0} p_1^{s_1}\cdots p_n^{s_n}\Gamma(s_1, s_2, \cdots, s_n)} > 0.$$

Therefore, we have $f(\mathbf{x}_i) > f(\mathbf{x}_j)$ if $\xi_i > \xi_j$, and this theorem holds. □

6.4. Proof of Theorem 2

First, we introduce the notion of *smoothness* as follows.

Definition 2. *Given a linear function space $\mathcal{W} \subseteq \mathbb{R}^d$, a function $f\colon \mathcal{W} \to \mathbb{R}$ is said to be μ-smooth if*

$$\|\nabla f(\mathbf{w}') - \nabla f(\mathbf{w})\| \leq \mu\|\mathbf{w}' - \mathbf{w}\| \textit{ for every } \mathbf{w}, \mathbf{w}' \in \mathcal{W}.$$

For smooth functions, we have a helpful lemma from (Nesterov, 2003, Theorem 2.1.5) as follows.

Lemma 2. *If f is μ-smooth, then, for every $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$, we have*

$$f(\mathbf{w}) - f(\mathbf{w}') \geq \langle\nabla f(\mathbf{w}'), \mathbf{w} - \mathbf{w}'\rangle + \frac{1}{2\mu}\|\mathbf{w} - \mathbf{w}'\|^2.$$

Proof of Theorem 2. We will exploit the smoothness to prove this theorem. The proof technique is motivated from the work of (Shalev-Shwartz, 2007; Srebro et al., 2010), and the detailed proof is presented for completeness.

We have defined $\mathcal{L}_t(\mathbf{w}) = 0$ for $T_t^+ T_t^- = 0$ in Section 3, and it is easy to analyze such cases; therefore, we consider $T_t^+ T_t^- \neq 0$ in the rest of our proof. Recall that

$$\mathcal{L}_t(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{\sum_{i=1}^{t-1} \mathbb{I}[y_i \neq y_t](1 - y_t\langle \mathbf{x}_t - \mathbf{x}_i, \mathbf{w}\rangle)^2}{2|\{i \in [t-1] : y_i \neq y_t\}|},$$

and it is easy to derive

$$\nabla\mathcal{L}_t(\mathbf{w}) = \lambda\mathbf{w} - \frac{\sum_{i=1}^{t-1} \mathbb{I}[y_i \neq y_t](1 - y_t\langle \mathbf{x}_t - \mathbf{x}_i, \mathbf{w}\rangle) y_t(\mathbf{x}_t - \mathbf{x}_i)}{|\{i \in [t-1] : y_i \neq y_t\}|}.$$

For $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$ and $\|\mathbf{x}_t\| \leq 1$, we have

$$\|\nabla\mathcal{L}_t(\mathbf{w}') - \nabla\mathcal{L}_t(\mathbf{w})\| \leq (4 + \lambda)\|\mathbf{w}' - \mathbf{w}\|,$$

which implies that $\mathcal{L}_t$ is $(4+\lambda)$-smooth. Denote

$$\mathbf{w}_{t*} = \arg\min_{\mathbf{w}} \mathcal{L}_t(\mathbf{w}),$$

and this gives $\nabla\mathcal{L}_t(\mathbf{w}_{t*}) = 0$ from the convex and differentiable loss $\mathcal{L}_t$. Based on Lemma 2 and $\mathcal{L}_t(\mathbf{w}_{t*}) \geq 0$, we have

$$\begin{aligned}\|\nabla\mathcal{L}_t(\mathbf{w}_{t-1})\|^2 &= \|\nabla\mathcal{L}_t(\mathbf{w}_{t-1}) - \nabla\mathcal{L}_t(\mathbf{w}_{t*})\|^2 \\ &\leq 2(\lambda+4)(\mathcal{L}_t(\mathbf{w}_{t-1}) - \mathcal{L}_t(\mathbf{w}_{t*})) \leq 2(\lambda+4)\mathcal{L}_t(\mathbf{w}_{t-1}). \end{aligned} \tag{27}$$

From the convexity of function $\mathcal{L}_{t-1}$, we have

$$\mathcal{L}_t(\mathbf{w}_{t-1}) - \mathcal{L}_t(\mathbf{w}_*) \leq \langle \nabla\mathcal{L}_t(\mathbf{w}_{t-1}), \mathbf{w}_{t-1} - \mathbf{w}_*\rangle. \tag{28}$$

Therefore, we have

$$\begin{aligned}\|\mathbf{w}_t - \mathbf{w}_*\|^2 = \|\mathbf{w}_{t-1} - \eta_t\nabla\mathcal{L}_t(\mathbf{w}_{t-1}) - \mathbf{w}_*\|^2 = \|\mathbf{w}_{t-1} - \mathbf{w}_*\|^2 \\ - 2\eta_t\langle\nabla\mathcal{L}_t(\mathbf{w}_{t-1}), \mathbf{w}_{t-1} - \mathbf{w}_*\rangle + \eta_t^2\|\nabla\mathcal{L}_t(\mathbf{w}_{t-1})\|^2.\end{aligned} \tag{29}$$

This implies that, by using Eqs. 27 and 28,

$$(1 - (4+\lambda)\eta_t)\mathcal{L}_t(\mathbf{w}_{t-1}) - \mathcal{L}_{t-1}(\mathbf{w}_*) \leq \frac{1}{2\eta_t}\|\mathbf{w}_{t-1} - \mathbf{w}_*\|^2 - \frac{1}{2\eta_t}\|\mathbf{w}_t - \mathbf{w}_*\|^2.$$

Summing over $t = 1, \dots, T$ and rearranging, we obtain

$$\begin{aligned}
&\sum_{t=1}^{T}(1-(4+\lambda)\eta_t)\mathcal{L}_t(\mathbf{w}_{t-1}) - \sum_{t=1}^{T}\mathcal{L}_t(\mathbf{w}_*) \\
&\quad \leq \quad \frac{1}{2\eta_1}\|\mathbf{w}_0 - \mathbf{w}_*\|^2 - \frac{1}{2\eta_T}\|\mathbf{w}_T - \mathbf{w}_*\|^2 + \sum_{t=1}^{T-1}(\frac{1}{2\eta_{t+1}} - \frac{1}{2\eta_t})\|\mathbf{w}_t - \mathbf{w}_*\|^2.
\end{aligned}$$

By setting $\eta_t = \eta$, we have

$$\frac{1}{2\eta_1}\|\mathbf{w}_0 - \mathbf{w}_*\|^2 - \frac{1}{2\eta_T}\|\mathbf{w}_T - \mathbf{w}_*\|^2 \leq \frac{1}{2\eta}\|\mathbf{w}_*\|^2 \leq \frac{1}{2\lambda\eta}$$

from $\mathbf{w}_0 = \mathbf{0}$ and $\|\mathbf{w}_*\| \leq 1/\sqrt{\lambda}$, and we finally get

$$\begin{aligned}
\sum_{t=1}^{T}\mathcal{L}_t(\mathbf{w}_{t-1}) - \sum_{t=1}^{T}\mathcal{L}_t(\mathbf{w}_*) \quad &\leq \quad \frac{1}{1-(4+\lambda)\eta}\left(\frac{1}{2\eta\lambda} + (4+\lambda)\eta\sum_{t=1}^{T}\mathcal{L}_t(\mathbf{w}_*)\right) \\
&\leq \quad \frac{1}{1-(4+\lambda)\eta}\left(\frac{1}{2\eta\lambda} + (4+\lambda)\eta T L^*\right).
\end{aligned}$$

By setting

$$\eta = \frac{1}{4+\lambda+\sqrt{(4+\lambda)^2+(4+\lambda)\lambda T L^*}}$$

and simple calculations, the theorem holds as desired. □

6.5. Proof of Theorem 3

It is sufficient to consider $T_t^+ T_t^- \neq 0$ as in the proof of Theorem 2. We rewrite $\mathcal{L}(\mathbf{w}; \mathcal{S}) = \frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_t(\mathbf{w})$ as

$$\mathcal{L}(\mathbf{w}; \mathcal{S}) = \frac{1}{2} + \mathbf{w}^\top \mathbf{a} + \frac{1}{2}\mathbf{w}^\top (A_1 + A_2)\,\mathbf{w},$$

where

$$\begin{aligned}
\mathbf{a} &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[y_t = 1](\mathbf{c}_{t-1}^{-} - \mathbf{x}_t) + \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[y_t = -1](\mathbf{c}_{t-1}^{+} - \mathbf{x}_t), \\
A_1 &= \lambda\mathbf{I}_d + \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[y_t = 1]S_{t-1}^{-} + \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[y_t = -1]S_{t-1}^{+}, \\
A_2 &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[y_t = 1](\mathbf{x}_t - \mathbf{c}_t^{-})(\mathbf{x}_t - \mathbf{c}_t^{-})^{\top} \\
& \quad + \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[y_t = -1](\mathbf{x}_t - \mathbf{c}_t^{+})(\mathbf{x}_t - \mathbf{c}_t^{+})^{\top}.
\end{aligned}$$

Similarly, we rewrite $\widehat{\mathcal{L}}(\mathbf{w}; \mathcal{S}) = \frac{1}{T}\sum_{t=1}^{T}\widehat{\mathcal{L}}_t(\mathbf{w})$ as

$$\widehat{\mathcal{L}}(\mathbf{w}; \mathcal{S}) = \frac{1}{2} + \mathbf{w}^{\top}\mathbf{a} + \frac{1}{2}\mathbf{w}^{\top}\big(\widetilde{A}_1 + A_2\big)\mathbf{w}$$

where

$$\widetilde{A}_1 = \lambda\mathbf{I}_d + \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[y_t = 1]\widehat{S}_t^{+} + \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}[y_t = -1]\widehat{S}_t^{-},$$

and $\widehat{S}_t^{+}$ and $\widehat{S}_t^{-}$ are defined by Eqn. 19. The optimal solutions minimizing $\mathcal{L}(\mathbf{w}; S)$ and $\widehat{\mathcal{L}}(\mathbf{w}; S)$ are given, respectively, by

$$\mathbf{w}_* = (A_1 + A_2)^{-1}\mathbf{a} \quad \text{and} \quad \widehat{\mathbf{w}}_* = (\widetilde{A}_1 + A_2)^{-1}\mathbf{a}.$$

From the assumption that $g(\tau)$ is a non-increasing function such that

$$\begin{aligned}
\left\|\hat{Z}_t^{+} - X_t^{+}[X_t^{+}]^{\top}\right\| &\leq g(\tau)\mathrm{tr}(X_t^{+}[X_t^{+}]^{\top}), \\
\left\|\hat{Z}_t^{-} - X_t^{-}[X_t^{-}]^{\top}\right\| &\leq g(\tau)\mathrm{tr}(X_t^{-}[X_t^{-}]^{\top}),
\end{aligned}$$

we have

$$\begin{aligned}
&\left\|(A_1 + A_2)^{1/2}(\widetilde{A}_1 + A_2)^{-1}(A_1 + A_2)^{1/2} - \mathbf{I}_d\right\| \\
&\quad = \left\|(\widetilde{A}_1 + A_2)^{-1/2}(A_1 - \widetilde{A}_1)(\widetilde{A}_1 + A_2)^{-1/2}\right\| \\
&\quad \leq \|A_1 - \widetilde{A}_1\|\|(\widetilde{A}_1 + A_2)^{-1}\| \leq rg(\tau)/\lambda. \qquad (30)
\end{aligned}$$

Denote $\Omega = (A_1 + A_2)^{1/2}(\widetilde{A}_1 + A_2)^{-1}(A_1 + A_2)^{1/2} - \mathbf{I}_d$, and it is easy to get

$$\begin{aligned}\|\widehat{\mathbf{w}}_* - \mathbf{w}_*\| = & \left\|\left((\widetilde{A}_1 + A_2)^{-1} - (A_1 + A_2)^{-1}\right)\mathbf{a}\right\| \\ = & \left\|(A_1 + A_2)^{-1/2}\Omega(A_1 + A_2)^{-1/2}\mathbf{a}\right\| \\ \leq & \frac{r}{\lambda}g(\tau)\|(A_1 + A_2)^{-1}\mathbf{a}\| \leq \frac{r}{\lambda}g(\tau)\|\mathbf{w}_*\|\end{aligned}$$

which implies, from $\|\mathbf{w}_*\| \leq B$, that

$$\|\widehat{\mathbf{w}}_*\| \leq \|\mathbf{w}_*\| + \|\widehat{\mathbf{w}}_* - \mathbf{w}_*\| \leq \beta B, \tag{31}$$

where $\beta = 1 + g(\tau)r/\lambda$. In addition, we have

$$\begin{aligned}& \left|\widehat{\mathcal{L}}(\mathbf{w}_*; \mathcal{S}) - \mathcal{L}(\mathbf{w}_*; \mathcal{S})\right| \\ = & \frac{3}{2}\left|\mathbf{a}^\top\left((\widetilde{A}_1 + A_2)^{-1} - (A_1 + A_2)^{-1}\right)\mathbf{a}\right| \\ = & \frac{3}{2}\left|\mathbf{a}^\top(A_1 + A_2)^{-1/2}\Omega(A_1 + A_2)^{-1/2}\mathbf{a}\right| \\ \leq & \frac{rg(\tau)}{\lambda}\left|\frac{3}{2}\mathbf{a}^\top(A_1 + A_2)^{-1}\mathbf{a}\right| \\ \leq & \frac{rg(\tau)}{\lambda}\left|\frac{1}{2} + \frac{3}{2}\mathbf{a}^\top(A_1 + A_2)^{-1}\mathbf{a}\right| = \frac{rg(\tau)}{\lambda}\mathcal{L}(\mathbf{w}_*)\end{aligned} \tag{32}$$

which yields that

$$\widehat{\mathcal{L}}(\widehat{\mathbf{w}}_*; \mathcal{S}) \leq \mathcal{L}(\mathbf{w}_*; \mathcal{S}) + |\widehat{\mathcal{L}}(\widehat{\mathbf{w}}_*; \mathcal{S}) - \mathcal{L}(\mathbf{w}_*; \mathcal{S})| \leq \beta T L^*. \tag{33}$$

Finally, we have

$$\begin{aligned}& \sum_{t=1}^T \widehat{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \sum_{t=1}^T \mathcal{L}_t(\mathbf{w}_*) \\ \leq & \sum_{t=1}^T \left(\widehat{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \widehat{\mathcal{L}}_t(\widehat{\mathbf{w}}_*)\right) + \sum_{t=1}^T \left(\widehat{\mathcal{L}}_t(\widehat{\mathbf{w}}_*) - \mathcal{L}_t(\mathbf{w}_*)\right).\end{aligned}$$

The second term in the above can be bounded by Eqn. 32. Similarly to the proof of Theorem 2, we have, by setting $\eta_t = 1/(\kappa + \sqrt{(\kappa^2 + \kappa T L^*/\beta/B^2})$,

$$\begin{aligned}\sum_{t=1}^T \widehat{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \sum_{t=1}^T \widehat{\mathcal{L}}_t(\widehat{\mathbf{w}}_*) \leq & 2\kappa\beta^2 B^2 + \beta B\sqrt{2\kappa\widehat{\mathcal{L}}(\widehat{\mathbf{w}}_*; S)} \\ \leq & 2\kappa\beta^2 B^2 + \beta B\sqrt{2\kappa\beta T L^*}\end{aligned}$$

where the last inequality holds from Eqn. 33. This completes the proof. □

6.6. Proof of Theorem 4

Theorem 6 (Detailed version of Theorem 4). *Let* $\mathcal{W} = \{\mathbf{w}\colon \|\mathbf{w}\| \leq B\}$, $\|\mathbf{x}\| \leq 1$ *and* $T_0 = \lfloor T/2 \rfloor$. *Suppose that* $\mathbf{w}_{T_0}, \ldots, \mathbf{w}_{T-1} \in \mathcal{W}$ *are models output by OPAUC. For any* $\epsilon > 0$ *and* $T \geq \max\big\{256, 16/p^2 \ln(64B_1/\epsilon), (2\ln 8/p^2)^{4/3}, (256B_1/\epsilon/\min\{p, 2-3p\})^4\big\}$, *the following holds with probability at least* $1-\delta$ *(*δ *is defined in Eqn. 22) over an i.i.d. sample* $\mathcal{S} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_T, y_T)\}$

$$\frac{1}{T-T_0}\sum_{t=T_0+1}^{T}\mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D}) - \frac{1}{T-T_0}\sum_{t=T_0+1}^{T}\mathcal{L}_t(\mathbf{w}_{t-1}) < \epsilon.$$

This theorem gives the exact expression that T needs to exceed for our bounds in contrast to Theorem 4.

Proof. We begin with an intermediate loss

$$\bar{\mathcal{L}}_t(\mathbf{w}_{t-1}) = E_{(\mathbf{x}_t, y_t)\sim\mathcal{D}}[\mathcal{L}_t(\mathbf{w}_{t-1})|(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_{t-1}, y_{t-1})], \tag{34}$$

as in the work of (Wang et al., 2012; Kar et al., 2013). We have

$$\begin{aligned} &\Pr\left[\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D}) - \mathcal{L}_t(\mathbf{w}_{t-1})}{T-T_0} \geq \epsilon\right] \\ &\leq \Pr\left[\sum_{t=T_0+1}^{T}\frac{\bar{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \mathcal{L}_t(\mathbf{w}_{t-1})}{T-T_0} \geq \frac{\epsilon}{2}\right] \\ &\quad + \Pr\left[\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D}) - \bar{\mathcal{L}}_t(\mathbf{w}_{t-1})}{T-T_0} \geq \frac{\epsilon}{2}\right]. \end{aligned} \tag{35}$$

For $\|\mathbf{x}_t\| \leq 1$ and $\|\mathbf{w}_{t-1}\| \leq B$, it is easy to see that

$$|\mathcal{L}_t(\mathbf{w}_{t-1})| \leq B_1 \quad \text{where} \quad B_1 = ((1+2B)^2 + \lambda B^2)/2.$$

Throughout this section, we denote by $E_t[\cdot] = E_{(\mathbf{x}_t, y_t)\sim\mathcal{D}}[\cdot]$. Our proof includes four parts as follows.

Step 1: Bounding the Martingale difference

We first observe that $\{(\bar{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \mathcal{L}_t(\mathbf{w}_{t-1}))/(T-T_0)\}_{t\geq T_0}$ is a martingale sequence, and it is bounded by $2B_1/(T-T_0)$. Based on the Hoeffding-Azuma inequality (Azuma, 1967), we have

$$\Pr\left[\sum_{t=T_0+1}^{T}\frac{\bar{\mathcal{L}}_t(\mathbf{w}_{t-1})-\mathcal{L}_t(\mathbf{w}_{t-1})}{T-T_0}\geq\frac{\epsilon}{2}\right]\leq\exp\Big(-\frac{T\epsilon^2}{16B_1^2}\Big).$$

Step 2: Symmetrization by a ghost sample
We begin with a ghost sample $\tilde{S}_T=\{(\tilde{\mathbf{x}}_1,\tilde{y}_1),(\tilde{\mathbf{x}}_2,\tilde{y}_2),\dots,(\tilde{\mathbf{x}}_T,\tilde{y}_T)\}$ drawn i.i.d. from distribution $\mathcal{D}$, and denote by

$$\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})=\frac{\lambda}{2}\|\mathbf{w}_{t-1}\|^2+\frac{\sum_{i=1}^{t-1}\mathbb{I}[\tilde{y}_i\neq y_t](1-y_t(\mathbf{x}_t-\tilde{\mathbf{x}}_i)^\top\mathbf{w}_{t-1})^2}{2|\{i\in[t-1]:\tilde{y}_iy_t=-1\}|}. \tag{36}$$

We further bound Eqn. 35 as follows:

Lemma 3. *For* $T\geq\max\big\{256,(2\ln 8/p^2)^{4/3},8/p^2\ln(64B_1/\epsilon),(256B_1/\epsilon/\min\{p,2-3p\})^4\big\}$, *we have*

$$\begin{aligned}&\Pr\left[\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-\bar{\mathcal{L}}_t(\mathbf{w}_{t-1})}{T-T_0}\geq\frac{\epsilon}{2}\right]\\ \leq\ & 2\Pr\left[\sum_{t=T_0+1}^{T}\frac{E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})-\bar{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T-T_0}\geq\frac{\epsilon}{4}\right].\end{aligned}$$

Recall that $p\leq 1/2$, and thus $\min\{p,2-3p\}>0$. Before the proof of Lemma 3, we first set $T_1=\lfloor T^{3/4}\rfloor$, and denote by

$$\tilde{S}_{T_1}=\{(\tilde{\mathbf{x}}_1,\tilde{y}_1),(\tilde{\mathbf{x}}_2,\tilde{y}_2),\dots,(\tilde{\mathbf{x}}_{T_1},\tilde{y}_{T_1})\}.$$

For any fixed $\tilde{S}_{T_1}$, we can see that $E[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]|\tilde{S}_{T_1}]$ converges to $\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})$ for sufficiently large T as follows:

Lemma 4. *For* $T\geq\max(256,8/p^2\ln(64B_1/\epsilon),(256B_1/\epsilon/\min\{p,2-3p\})^4)$, *we have*

$$\left|E\left[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]|\tilde{S}_{T_1}\right]-\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})\right|\leq\epsilon/8$$

where $E_t[\cdot]=E_{(\mathbf{x}_t,y_t)\sim\mathcal{D}}[\cdot]$ *and* $E[\cdot]=E_{(\tilde{\mathbf{x}}_{T_1+1},\tilde{y}_{T_1+1}),\dots,(\tilde{\mathbf{x}}_T,\tilde{y}_T)\sim\mathcal{D}^{T-T_1}}[\cdot]$.

Proof: It is easy to observe that, from Eqn. 36,

$$\begin{aligned}
& E\left[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]|\tilde{S}_{T_1}\right] \\
= \quad & \frac{\lambda}{2}\|\mathbf{w}_{t-1}\|^2 + E\left[\sum_{i=1}^{T_1} \frac{\mathbb{I}[\tilde{y}_i \neq y_t](1-y_t(\mathbf{x}_t - \tilde{\mathbf{x}}_i)^\top \mathbf{w}_{t-1})^2}{2|\{i \in [t-1] : \tilde{y}_i y_t = -1\}|}\middle| \tilde{S}_{T_1}\right] \\
& \qquad + E\left[\sum_{i=T_1+1}^{t-1} \frac{\mathbb{I}[\tilde{y}_i \neq y_t](1-y_t(\mathbf{x}_t - \tilde{\mathbf{x}}_i)^\top \mathbf{w}_{t-1})^2}{2|\{i \in [t-1] : \tilde{y}_i y_t = -1\}|}\middle| \tilde{S}_{T_1}\right]
\end{aligned}$$

and

$$\mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D}) = \frac{\lambda}{2}\|\mathbf{w}_{t-1}\|^2 + E\left[\frac{\sum_{i=T_1+1}^{t-1} \mathbb{I}[\tilde{y}_i \neq y_t](1-y_t(\mathbf{x}_t - \tilde{\mathbf{x}}_i)^\top \mathbf{w}_{t-1})^2}{2|\{T_1 < i < t : \tilde{y}_i y_t = -1\}|}\right].$$

Thus, we have

$$\begin{aligned}
& E\left[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]|\tilde{S}_{T_1}\right] - \mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D}) \\
= \quad & E\left[\frac{|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|}{|\{i \in [t-1] : \tilde{y}_i y_t = -1\}|}\sum_{i=1}^{T_1} \frac{\mathbb{I}[\tilde{y}_i \neq y_t](1-y_t(\mathbf{x}_t - \tilde{\mathbf{x}}_i)^\top \mathbf{w}_{t-1})^2}{2|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|} - \right. \\
& \left. \frac{|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|}{|\{i \in [t-1] : \tilde{y}_i y_t = -1\}|}\sum_{i=T_1+1}^{t-1} \frac{\mathbb{I}[\tilde{y}_i \neq y_t](1-y_t(\mathbf{x}_t - \tilde{\mathbf{x}}_i)^\top \mathbf{w}_{t-1})^2}{2|\{T_1 < i < t : \tilde{y}_i y_t = -1\}|}\middle| \tilde{S}_{T_1}\right].
\end{aligned}$$

For $\|\mathbf{x}_t\| \leq 1$ and $\|\mathbf{w}_{t-1}\| \leq B$, we have $(1 - y_t(\mathbf{x}_t - \tilde{\mathbf{x}}_i)^\top \mathbf{w}_{t-1})^2/2 \leq B_1$, and

$$\begin{aligned}
& \left|E\left[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]|\tilde{S}_{T_1}\right] - \mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D})\right| \\
\leq \quad & 2B_1 E\left[E_t\left[\frac{|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|}{|\{i \in [t-1] : \tilde{y}_i y_t = -1\}|}\right]\middle| \tilde{S}_{T_1}\right] \\
\leq \quad & 2B_1 E\left[E_t\left[\frac{|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|}{|\{i \in [T_0] : \tilde{y}_i y_t = -1\}|}\right]\middle| \tilde{S}_{T_1}\right] \text{ for } t \geq T_0.
\end{aligned}$$

We complete the proof by combining with Lemma 5. □

Lemma 5. *For $T \geq \max\left(256, 8/p^2 \ln(64B_1/\epsilon), (256B_1/\epsilon/\min\{p, 2-3p\})^4\right)$, we have*

$$E\left[E_t\left[\frac{|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|}{|\{i \in [T_0] : \tilde{y}_i y_t = -1\}|}\right]\middle| \tilde{S}_{T_1}\right] \leq \frac{\epsilon}{16B_1}$$

where $T_1 = \lfloor T^{3/4} \rfloor$ and $T_0 = \lfloor T/2 \rfloor$.

Proof: Let $\tilde{S}_{T_1+1:T_0} = \{(\tilde{\mathbf{x}}_{T_1+1}, \tilde{y}_{T_1+1}), (\tilde{\mathbf{x}}_{T_1+2}, \tilde{y}_{T_1+2}), \ldots, (\tilde{\mathbf{x}}_{T_0}, \tilde{y}_{T_0})\}$. Denote the set

$$\mathcal{A} = \left\{ \tilde{S}_{T_1+1:T_0} \colon \left| \sum_{i=T_1+1}^{T_0} \frac{I[\tilde{y}_i = 1]}{T_0 - T_1} - p \right| \geq \frac{p}{2} \right\}.$$

Based on the Hoeffding's inequality (Hoeffding, 1963), we have

$$\Pr\left[\tilde{S}_{T_1+1:T_0} \in \mathcal{A}\right] \leq 2\exp(-(T_0 - T_1)p^2/2) \leq \epsilon/(32B_1),$$

for $T > \max(256, 8/p^2 \ln(64B_1/\epsilon))$. For $\tilde{S}_{T_1+1:T_0} \notin \mathcal{A}$, we have

$$\begin{aligned} |\{T_1 < i \leq T_0 : y_t = 1\}| &> p(T_0 - T_1)/2 \geq Tp/8 \\ |\{T_1 < i \leq T_0 : y_t = -1\}| &> (T_0 - T_1)(1 - 3p/2) \geq T(2 - 3p)/8 \end{aligned}$$

for $T > 256$. Therefore, it holds that, for $T \geq (256B_1/\epsilon/\min\{p, 2-3p\})^4$,

$$E\left[E_t\left[\frac{|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|}{|\{i \in [T_0] : \tilde{y}_i y_t = -1\}|}\right] \middle| \tilde{S}_{T_1+1:T_0} \notin \mathcal{A}, \tilde{S}_{T_1}\right] \leq \frac{16T^{-1/4}}{\min(p, 2-3p)} \leq \frac{\epsilon}{32B_1}.$$

By the law of total expectation, i.e.,

$$E[\cdot] = E[\cdot|\tilde{S}_{T_1+1:T_0} \in \mathcal{A}]\Pr[\tilde{S}_{T_1+1:T_0} \in \mathcal{A}] + E[\cdot|\tilde{S}_{T_1+1:T_0} \notin \mathcal{A}]\Pr[\tilde{S}_{T_1+1:T_0} \notin \mathcal{A}],$$

we have

$$\begin{aligned} E\left[E_t\left[\frac{|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|}{|\{i \in [T_0] : \tilde{y}_i y_t = -1\}|}\right] \middle| \tilde{S}_{T_1}\right] &\leq \Pr[\tilde{S}_{T_1+1:T_0} \in \mathcal{A}] \\ + E\left[E_t\left[\frac{|\{i \in [T_1] : \tilde{y}_i y_t = -1\}|}{|\{i \in [T_0] : \tilde{y}_i y_t = -1\}|}\right] \middle| \tilde{S}_{T_1+1:T_0} \notin \mathcal{A}, \tilde{S}_{T_1}\right] &\leq \frac{\epsilon}{16B_1} \end{aligned}$$

which completes the proof. □

Proof of Lemma 3 It is easy to see that

$$\begin{aligned} &\Pr\left[\sum_{t=T_0+1}^{T} \frac{E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \bar{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T - T_0} \geq \frac{\epsilon}{4}\right] \\ \geq\ & E_{S_T}\left[I\left[\sum_{t=T_0+1}^{T} \frac{\mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D}) - \bar{\mathcal{L}}_t(\mathbf{w}_{t-1})}{T - T_0} \geq \frac{\epsilon}{2}\right]\right. \\ &\qquad \left. \times \Pr_{\tilde{S}_T}\left[\left|\sum_{t=T_0+1}^{T} \frac{\mathcal{L}(\mathbf{w}_{t-1}, \mathcal{D}) - E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T - T_0}\right| \leq \frac{\epsilon}{4} \middle| S_T\right]\right] \end{aligned}$$

and the lemma holds if

$$\Pr_{\tilde{S}_T}\left[\left|\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T-T_0}\right|\le\frac{\epsilon}{4}\middle|S_T\right]\ge\frac{1}{2}.$$

Denote the set

$$\mathcal{A}=\left\{\tilde{S}_{T_1}:\ \left|\frac{1}{T_1}\sum_{t=1}^{T_1}I[\tilde{y}_t=1]-p\right|>p/2\right\}.$$

From Hoeffding's inequality (Hoeffding, 1963), we have

$$\Pr[\tilde{S}_{T_1}\in\mathcal{A}]\le\exp\left(-T_1p^2/2\right)\le 1/4\ \text{ for }\ T>(2\ln 8/p^2)^{4/3}.$$

By the law of total probability, we have

$$\begin{aligned}&\Pr\left[\left|\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T-T_0}\right|>\frac{\epsilon}{4}\middle|S_T\right]\\ \le&\ \frac{1}{4}+\Pr\left[\left|\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T-T_0}\right|>\frac{\epsilon}{4}\middle|S_T,\tilde{S}_{T_1}\notin\mathcal{A}\right].\end{aligned}$$

For $T\ge\max\left(256,8/p^2\ln(64B_1/\epsilon),(256B_1/\epsilon/\min\{p,2-3p\})^4\right)$, it holds that, from Lemma 4,

$$\left|E\left[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]|\tilde{S}_{T_1}\right]-\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})\right|\le\epsilon/8,$$

which yields that

$$\begin{aligned}&\Pr\left[\left|\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T-T_0}\right|>\frac{\epsilon}{4}\middle|S_T,\tilde{S}_{T_1}\notin\mathcal{A}\right]\\ \le&\ \Pr\left[\left|\sum_{t=T_0+1}^{T}\frac{E[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]|\tilde{S}_{T_1}]-E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T-T_0}\right|>\frac{\epsilon}{8}\middle|S_T,\tilde{S}_{T_1}\notin\mathcal{A}\right]\\ \le&\ \frac{64}{\epsilon^2}\mathrm{Var}\left(\sum_{t=T_0+1}^{T}\frac{E_t\left[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})|\tilde{S}_{T_1}\right]}{T-T_0}\right)\quad\text{(by Chebyshev's inequality).}\end{aligned}$$

If $\tilde{S}_{T_1} \notin \mathcal{A}$, then each $E_t\left[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})|\tilde{S}_{T_1}\right]$ changes by at most $2B_1/\min\{p, 2-3p\}/T^{3/4}$ by randomly replacing any example $(\tilde{\mathbf{x}}_t, \tilde{y}_t)$ for $t \geq T_1$. Based on the work of (Devroye et al., 1996, Theorem 9.3), we have

$$\begin{aligned} & \frac{64}{\epsilon^2}\mathrm{Var}\left(\sum_{t=T_0+1}^{T} \frac{E_t\left[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})|\tilde{S}_{T_1}\right]}{T-T_0}\right) \\ \leq \; & \frac{16}{\epsilon^2}\sum_{i=T_1+1}^{T} \frac{B_1^2}{(\min\{p/2, 1-3p/2\})^2 T^{3/2}} \\ \leq \; & \frac{4B_1/\epsilon^2}{(\min\{p, 2-3p\})^2 T^{1/2}} \leq \frac{1}{4}, \end{aligned}$$

for $T > (256B_1/\epsilon/\min\{p, 2-3p\})^4 > (16B_1/\epsilon^2/(\min\{p, 2-3p\})^2)^2$. This completes the proof. □

Step 3: Uniform convergence
For any fixed $\mathbf{w} \in \mathcal{W}$, we have

Lemma 6. *For $t \geq T_0$, $\mathbf{w} \in \mathcal{W}$ and $T \geq 16/p^2 \ln(128B_1/\epsilon)$, we have*

$$\begin{aligned} & \Pr\left[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}) - \mathcal{L}_t(\mathbf{w})] \geq \epsilon/8\right] \\ \leq \; & 2\exp\left(-\frac{1}{8}Tp^2\right) + \exp\left(-\frac{(\min\{p, 2-3p\})^2 T\epsilon^2}{2^{17}B_1^2}\right). \end{aligned}$$

Proof: Given $T_2 = \lfloor T/4 \rfloor$, let

$$\begin{aligned} \tilde{S}_{T_2} &= \{(\tilde{\mathbf{x}}_1, \tilde{y}_1), (\tilde{\mathbf{x}}_2, \tilde{y}_2), \ldots, (\tilde{\mathbf{x}}_{T_2}, \tilde{y}_{T_2})\}, \\ S_{T_2} &= \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_{T_2}, y_{T_2})\}, \end{aligned}$$

i.e., the first T_2 examples in $\tilde{S}_T$ and S_T, respectively. Denote the sets

$$\begin{aligned} \mathcal{A}_1 &= \left\{\tilde{S}_{T_2} \colon \left|\frac{1}{T_2}\sum_{t=1}^{T_2} I[\tilde{y}_t = 1] - p\right| > p/2\right\}, \\ \mathcal{A}_2 &= \left\{S_{T_2} \colon \left|\frac{1}{T_2}\sum_{t=1}^{T_2} I[y_t = 1] - p\right| > p/2\right\}. \end{aligned}$$

By using the Hoeffding inequality (Hoeffding, 1963) again, we have

$$\Pr\left[\tilde{S}_{T_2} \in \mathcal{A}_1\right] \leq \exp(-Tp^2/8), \quad \Pr\left[S_{T_2} \in \mathcal{A}_2\right] \leq \exp(-Tp^2/8) \quad \text{and}$$

$$\Pr\left[\tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2\right] \geq 1 - 2\exp(-Tp^2/8) \geq 1/2 \text{ for } T \geq 16/p^2.$$

For simplicity, we denote $\Delta = E_t[\tilde{\mathcal{L}}_t(\mathbf{w}) - \mathcal{L}_t(\mathbf{w})]$. By the law of total probability, we have

$$\begin{aligned}
&\Pr\left[\Delta \geq \frac{\epsilon}{8}\right] \\
\leq\ & \Pr[\tilde{S}_{T_2} \in \mathcal{A}_1 \text{ or } S_{T_2} \in \mathcal{A}_2] + \Pr\left[\Delta \geq \frac{\epsilon}{8}\middle| \tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2\right] \\
\leq\ & 2\exp(-Tp^2/8) + \Pr\left[\Delta \geq \frac{\epsilon}{8}\middle| \tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2\right].
\end{aligned}$$

It is easy to observe $\mathcal{L}(\mathbf{w}, \mathcal{D}) = E[E_t[\tilde{\mathcal{L}}_t(\mathbf{w})]] = E[E_t[\mathcal{L}_t(\mathbf{w})]]$, and we have

$$E[\Delta] = E[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}) - \mathcal{L}_t(\mathbf{w})]] = 0,$$

and by the law of total expectation, it holds that

$$\begin{aligned}
E[\Delta] = & E[\Delta|\tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2]\Pr[\tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2] \\
& + E[\Delta|\tilde{S}_{T_2} \in \mathcal{A}_1 \text{ or } S_{T_2} \in \mathcal{A}_2]\Pr[\tilde{S}_{T_2} \in \mathcal{A}_1 \text{ or } S_{T_2} \in \mathcal{A}_2].
\end{aligned}$$

This yields that, for $T \geq 16/p^2 \ln(\epsilon/128B_1)$,

$$\begin{aligned}
&\frac{1}{2}\left|E[\Delta]|\tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2]\right| \\
&\quad \leq \left|E[\Delta|\tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2]\right| \Pr[\tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2] \\
&\quad = \left|E[\Delta|\tilde{S}_{T_2} \in \mathcal{A}_1 \text{ or } S_{T_2} \in \mathcal{A}_2]\right| \Pr[\tilde{S}_{T_2} \in \mathcal{A}_1 \text{ or } S_{T_2} \in \mathcal{A}_2] \\
&\quad \leq 2B_1 \Pr[\tilde{S}_{T_2} \in \mathcal{A}_1 \text{ or } S_{T_2} \in \mathcal{A}_2] \leq 4B_1\exp(-Tp^2/8) \leq \epsilon/32
\end{aligned}$$

which implies

$$\left|E[\Delta|\tilde{S}_{T_2} \notin \mathcal{A}_1 \text{ and } S_{T_2} \notin \mathcal{A}_2]\right| \leq \epsilon/16.$$

Therefore, we have

$$\begin{aligned}
&\Pr\left[\Delta \geq \frac{\epsilon}{8}\middle| \tilde{S}_{T_2} \notin \mathcal{A}_1, S_{T_2} \notin \mathcal{A}_2\right] \\
\leq\ & \Pr\left[\Delta - E[\Delta|\tilde{S}_{T_2} \notin \mathcal{A}_1, S_{T_2} \notin \mathcal{A}_2] \geq \frac{\epsilon}{16}\middle| \tilde{S}_{T_2} \notin \mathcal{A}_1, S_{T_2} \notin \mathcal{A}_2\right].
\end{aligned}$$

For $\tilde{S}_{T_2} \notin \mathcal{A}_1$ and $S_{T_2} \notin \mathcal{A}_2$, Δ has a bounded variation of $16B_1/\min\{p, 2-3p\}/T$ if each $(\tilde{\mathbf{x}}_t, \tilde{y}_t)$ and $(\mathbf{x}_t, y_t)$ vary for $t > T_2$. By using the McDiarmid's inequality (McDiarmid, 1989), we have

$$\Pr\left[\Delta - E\left[\Delta|\tilde{S}_{T_2} \notin \mathcal{A}_1, S_{T_2} \notin \mathcal{A}_2\right] \geq \frac{\epsilon}{16}\middle| \tilde{S}_{T_2} \notin \mathcal{A}_1, S_{T_2} \notin \mathcal{A}_2\right]$$
$$\leq \exp\left(-\frac{(\min\{p, 2-3p\})^2 T^2 \epsilon^2}{2^{17}(t-T_2)B_1^2}\right) \leq \exp\left(-\frac{(\min\{p, 2-3p\})^2 T \epsilon^2}{2^{17}B_1^2}\right)$$

where the last inequality holds for $t < T$. □

We next use uniform convergence techniques based on cover numbers; as a first step, we show that the objective is Lipschitz as follows:

$$|E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_1) - \mathcal{L}_t(\mathbf{w}_1)] - E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_2) - \mathcal{L}_t(\mathbf{w}_2)]| \leq 2(1 + (\lambda+2)B)\|\mathbf{w}_1 - \mathbf{w}_2\|.$$

for $\mathbf{w}_1 \in \mathcal{W}$ and $\mathbf{w}_2 \in \mathcal{W}$. Let $m = \mathcal{N}(\mathcal{W}, \epsilon/16(1 + (\lambda+2)B))$, and assume that $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m$ is a covering of $\mathcal{W}$, i.e., for any $\mathbf{w} \in \mathcal{W}$, there is $\mathbf{w}_k$ s.t. $\|\mathbf{w}_k - \mathbf{w}\| \leq \epsilon/16(1 + (\lambda+2)B))$. Therefore, we have

$$\begin{aligned}
&\Pr\left[\sum_{t=T_0+1}^{T} \frac{E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1}) - \mathcal{L}_t(\mathbf{w}_{t-1})]}{T - T_0} \geq \frac{\epsilon}{4}\right] \\
\leq &\sum_{t=T_0+1}^{T} \Pr\left[\sup_{\mathbf{w} \in \mathcal{W}} [E_t[\tilde{\mathcal{L}}_t(\mathbf{w}) - \mathcal{L}_t(\mathbf{w})]] \geq \frac{\epsilon}{4}\right] \\
\leq &\sum_{t=T_0+1}^{T} \sum_{k=1}^{m} \Pr\left[[E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_k) - \mathcal{L}_t(\mathbf{w}_k)]] \geq \frac{\epsilon}{8}\right] \\
\leq &\ mT \exp\left(-\frac{1}{8}Tp^2\right) + \frac{1}{2}mT \exp\left(-\frac{(\min\{p, 2-3p\})^2 T \epsilon^2}{2^{17}B_1^2}\right) \qquad (37)
\end{aligned}$$

where the last inequality holds from Lemma 6.

Step 4: Putting it all together
Based on previous analyses, we have

$$\begin{aligned}
&\Pr\left[\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-\mathcal{L}_t(\mathbf{w}_{t-1})}{T-T_0}\geq\epsilon\right]\\
\leq\ &\exp\Big(-\frac{T\epsilon^2}{16B_1^2}\Big)+\Pr\left[\sum_{t=T_0+1}^{T}\frac{\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-\bar{\mathcal{L}}_t(\mathbf{w}_{t-1})}{T-T_0}\geq\frac{\epsilon}{2}\right]\\
\leq\ &\exp\Big(-\frac{T\epsilon^2}{16B_1^2}\Big)+2\Pr\left[\sum_{t=T_0+1}^{T}\frac{E_t[\tilde{\mathcal{L}}_t(\mathbf{w}_{t-1})-\bar{\mathcal{L}}_t(\mathbf{w}_{t-1})]}{T-T_0}\geq\frac{\epsilon}{4}\right]
\end{aligned}$$

where the first inequality holds from Eqn. 35 and Step 1, and the second inequality holds from Lemma 3. We complete the proof by combining Eqn. 37 with $m=\mathcal{N}(\mathcal{W},\epsilon/16(1+(\lambda+2)B))$. □

6.7. Proof of Corollary 2

We first define a random variable

$$\psi=\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-\mathcal{L}(\mathbf{w}_*^*,\mathcal{D})$$

where the randomness is over the selection of $T_0\leq t\leq T$. It is easy to observe that $\psi\geq 0$ from the definition of $\mathbf{w}_*^*$, and

$$E[\psi]=\frac{1}{T-T_0}\sum_{t=T_0+1}^{T}\mathcal{L}(\mathbf{w}_{t-1},\mathcal{D})-\mathcal{L}(\mathbf{w}_*^*,\mathcal{D}). \tag{38}$$

By Markov inequality, we have

$$\Pr\left[\psi\geq 3E[\psi]\right]\leq 1/3,$$

which completes the proof by combining Theorem 5 with Eqn. 38. □

7. Experiments

In this section, we evaluate the performance of OPAUC on benchmark datasets in Section 7.1, and present an evaluation on high-dimensional dense and sparse datasets in Sections 7.2 and 7.3, respectively. Finally, we analyze the parameter influence in Section 7.4.

Table 1: Benchmark datasets

datasets	#inst	#feat	datasets	#inst	#feat
diabetes	768	8	w8a	49,749	300
fourclass	862	2	kddcup04	50,000	65
german	1,000	24	mnist	60,000	780
splice	3,175	60	connect-4	67,557	126
usps	9,298	256	acoustic	78,823	50
letter	15,000	16	ijcnn1	141,691	22
magic04	19,020	10	epsilon	400,000	2,000
a9a	32,561	123	covtype	581,012	54

7.1. Comparisons on Benchmark Data

We conduct our experiments on sixteen benchmark datasets[1,2,3] as summarized in Table 1. Some datasets have been used in previous studies on AUC optimization, whereas the others are large datasets requiring a one-pass procedure. The features have been scaled to $[-1, 1]$ for all datasets. Multi-class datasets have been transformed into binary ones by randomly partitioning classes into two groups, where each group contains the same number of classes.

We compare with a series of approaches as follows:

- **OAM$_{\textbf{seq}}$**: An online AUC optimization with a sequential updating method (Zhao et al., 2011);

- **OAM$_{\textbf{gra}}$**: An online AUC optimization with a gradient descent updating method (Zhao et al., 2011);

- **Online Uni-Exp**: An online gradient descent algorithm which optimizes the (weighted) univariate exponential loss (Kotlowski et al., 2011);

- **Online Uni-Log**: An online gradient descent algorithm which optimizes the (weighted) univariate logistic loss (Kotlowski et al., 2011);

[1] http://www.sigkdd.org/kddcup/

[2] http://www.ics.uci.edu/~mlearn/MLRepository.html

[3] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/

Table 2: Testing AUC (mean±std.) of OPAUC with online algorithms on benchmark datasets. •/◦ indicates that OPAUC is significantly better/worse than the corresponding method (pairwise t-tests at 95% significance level).

datasets	OPAUC	OAM_{seq}	OAM_{gra}	online Uni-Exp	online Uni-Log	online Uni-Squ
diabetes	.8309±.0350	.8264±.0367	.8262±.0338	.8215±.0309•	.8260±.0360	.8258±.0354
fourclass	.8310±.0251	.8306±.0247	.8295±.0251	.8281±.0305	.8288±.0304	.8292±.0304
german	.7978±.0347	.7747±.0411•	.7723±.0358•	.7908±.0367	.7914±.0361	.7899±.0349
splice	.9232±.0099	.8594±.0194•	.8864±.0166•	.8931±.0213•	.9160±.0131•	.9153±.0132•
usps	.9620±.0040	.9310±.0159•	.9348±.0122•	.9538±.0045•	.9574±.0039•	.9563±.0041•
letter	.8114±.0065	.7549±.0344•	.7603±.0346•	.8113±.0074	.8080±.0068	.8053±.0081•
magic04	.8383±.0077	.8238±.0146•	.8259±.0169•	.8354±.0099•	.8353±.0097•	.8344±.0086•
a9a	.9002±.0047	.8420±.0174•	.8571±.0173•	.9005±.0024	.9006±.0023	.8949±.0025•
w8a	.9633±.0035	.9304±.0074•	.9418±.0070•	.9093±.0986•	.9204±.0114•	.8847±.0130•
kddcup04	.7912±.0039	.6918±.0412•	.7097±.0420•	.7851±.0050•	.7859±.0045•	.7850±.0042•
mnist	.9242±.0021	.8615±.0087•	.8643±.0112•	.7932±.0245•	.9164±.0018•	.9156±.0027•
connect-4	.8760±.0023	.7807±.0258•	.8128±.0230•	.8702±.0025•	.8708±.0026•	.8685±.0033•
acoustic	.8192±.0032	.7113±.0590•	.7711±.0217•	.8171±.0034•	.8206±.0035	.8193±.0035
ijcnn1	.9269±.0021	.9209±.0079•	.9100±.0092•	.9264±.0035	.9243±.0047•	.9022±.0041•
epsilon	.9550±.0007	.8816±.0042•	.8659±.0176•	.9488±.0012•	.9406±.0011•	.9480±.0021•
covtype	.8244±.0014	.7361±.0317•	.7403±.0289•	.8236±.0017	.8253±.0014	.8236±.0020
win/tie/loss		**14/2/0**	**14/2/0**	**10/6/0**	**9/7/0**	**11/5/0**

- **Online Uni-Squ**: An online gradient descent algorithm which optimizes the (weighted) univariate least square loss;
- **SVM-perf**: A batch algorithm which directly optimizes AUC (Joachims, 2005);
- **Batch SVM-OR**: A batch algorithm which optimizes the pairwise hinge loss (Joachims, 2006);
- **Batch LS-SVM**: A batch algorithm which optimizes the pairwise least square loss;
- **Batch Uni-Log**: A batch algorithm which optimizes the (weighted) univariate logistic loss (Kotlowski et al., 2011);
- **Batch Uni-Squ**: A batch algorithm which optimizes the (weighted) univariate least square loss.

Table 3: Testing AUC (mean±std.) of OPAUC with batch algorithms on benchmark datasets. •/◦ indicates that OPAUC is significantly better/worse than the corresponding method (pairwise t-tests at 95% significance level).

datasets	OPAUC	SVM-perf	batch SVM-OR	batch LS-SVM	batch Uni-Log	batch Uni-Squ
diabetes	.8309±.0350	.8325±.0220	.8326±.0328	.8325±.0329	.8330±.0322	.8332±.0323
fourclass	.8310±.0251	.8221±.0381	.8305±.0311	.8309±.0309	.8288±.0307	.8297±.0310
german	.7978±.0347	.7952±.0340	.7935±.0348	.7994±.0343	.7995±.0344	.7990±.0342
splice	.9232±.0099	.9235±.0091	.9239±.0089	.9245±.0092◦	.9208±.0107•	.9211±.0107•
usps	.9620±.0040	.9600±.0054•	.9630±.0047◦	.9634±.0045◦	.9637±.0041◦	.9617±.0043
letter	.8114±.0065	.8028±.0074•	.8144±.0064◦	.8124±.0065◦	.8121±.0061	.8112±.0061
magic04	.8383±.0077	.8427±.0078◦	.8426±.0074◦	.8379±0.0078	.8378±.0073	.8338±.0073•
a9a	.9002±.0047	.9033±.0039	.9009±.0036	.8982±.0028•	.9033±.0025◦	.8967±.0028•
w8a	.9633±.0035	.9626±.0042	.9495±.0082•	.9495±.0092•	.9421±.0062•	.9075±.0104•
kddcup04	.7912±.0039	.7935±.0037◦	.7903±.0039•	.7898±.0039•	.7900±.0039•	.7926±.0038
mnist	.9242±.0021	.9338±.0022◦	.9340±.0020◦	.9336±.0025◦	.9334±.0021◦	.9279±.0021◦
connect-4	.8760±.0023	.8794±.0024◦	.8749±.0025•	.8739±.0026•	.8784±.0026◦	.8760±.0024
acoustic	.8192±.0032	.8102±.0032•	.8262±.0032◦	.8210±.0033◦	.8253±.0032◦	.8222±.0031◦
ijcnn1	.9269±.0021	.9314±.0025◦	.9337±.0024◦	.9320±.0037◦	.9282±.0023◦	.9038±.0025•
epsilon	.9550±.0007	.8640±.0049•	.8643±.0053•	.8644±.0050•	.8647±.0150•	.8653±.0073•
covtype	.8244±.0014	.8271±.0011◦	.8248±.0013	.8222±.0014•	.8246±.0010	.8242±.0012
win/tie/loss		**4/6/6**	**4/6/6**	**6/4/6**	**4/6/6**	**6/8/2**

Here the weighted univariate losses mean that the losses are weighted by class priors as done in (Kotlowski et al., 2011).

All experiments are performed with Matlab 7 on a node of compute cluster with 16 CPUs (Intel Xeon Due Core 3.0GHz) running RedHat Linux Enterprise 5 with 48GB main memory. Due to memory limitation, we uniformly select 8,000 training examples at random (without replacement) over the whole training data for batch algorithms if training size exceeds 8,000, whereas only 2,000 training examples are selected in a similar manner for the epsilon dataset because of its high dimension. For all online approaches, we go through the entire training data only once.

Five-fold cross-validation is executed on training sets to determine the learning rate $\eta_t \in 2^{[-12:10]}$ for online algorithms, the regularized parameter $\lambda \in 2^{[-10:2]}$ for OPAUC and $\lambda \in 2^{[-10:10]}$ for batch algorithms. For OAM_{seq} and OAM_{gra}, the buffer sizes are fixed to be 100 as done in (Zhao et al., 2011). Theorem 2 shows that the optimal learning rate η_t depends on the optimal

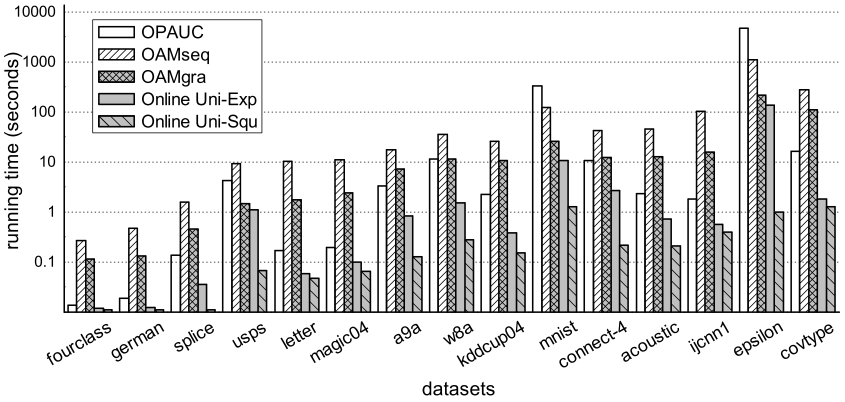

Figure 1: Comparison of the running time (in seconds) of OPAUC and online learning algorithms on benchmark datasets. Notice that the y-axis is in log-scale.

loss L^* yet it is unknown. Practically, we can set η_t to be $O(1/\sqrt{T})$ or a small constant as in (Zhang, 2004) to approach to the optimal learning rate, because our algorithm is insensitive to smaller η_t (e.g., $\eta_t < 1/16$) as shown in Figure 5, and this also keeps the one-pass property. In the following, cross-validation is executed to select η_t because we try to make fair comparisons with the first online AUC optimization work (Zhao et al., 2011).

The performances of the compared methods are evaluated by five trials of 5-fold cross validation, where the AUC values are obtained by averaging over these 25 runs. Table 2 shows that OPAUC is significantly better than the other four online algorithms OAM_{seq}, OAM_{gra}, online Uni-Exp and online Uni-Squ, particularly for large datasets. The win/tie/loss counts show that OPAUC is clearly superior to these online algorithms, as it wins for most times and never loses.

Table 3 shows that OPAUC is highly competitive to the other five batch learning algorithms; this is impressive because these batch algorithms require storing the whole/partial training dataset whereas OPAUC does not store training data. We also notice that those batch algorithms use smaller datasets because of memory limitation, and have potential for better performance. Additionally, batch LS-SVM which optimizes the pairwise least square loss is comparable to the other batch algorithms, verifying our argument that least square loss is effective for AUC optimization.

We also compare the running time of OPAUC and the online algorithms

Table 4: Testing AUC (mean±std.) of $\mathrm{OPAUC}_{\mathrm{fd}}$ with online methods on high-dimensional dense datasets. ●/○ indicates that $\mathrm{OPAUC}_{\mathrm{fd}}$ is significantly better/worse than the corresponding method (pairwise t-tests at 95% significance level). 'N/A' indicates that $\mathrm{OPAUC}^{\mathrm{pca}}$ runs out of memory because of PCA.

datasets	fourclass	magic04	letter	ijcnn1	acoustic
# features	50,002	50,010	50,016	50,022	50,050
$\mathrm{OPAUC}_{\mathrm{fd}}$	.8757±.0367	.8967±.0086	.9379±.0043	.9811±.0093	.8589±.0027
$\mathrm{OPAUC}_{\mathrm{s}}$	.8622±.0301●	.8895±.0075●	.9336±.0074●	.9740±.0098●	.8325±.0374●
$\mathrm{OAM}_{\mathrm{seq}}$	.8657±.0423●	.8898±.0096●	.9370±.0046	.9802±.0099	.8141±.0102●
$\mathrm{OAM}_{\mathrm{gra}}$	.8644±.0388●	.8830±.0146●	.9171±.0381●	.9615±.0102●	.8282±.0092●
online Uni-Exp	.8452±.0534●	.8805±.0064●	.9241±.0062●	.9653±.0051●	.8374±.0051●
online Uni-Log	.8555±.0329●	.8911±.0074●	.9293±.0094●	.9688±.0038●	.8387±.0039●
online Uni-Squ	.8507±.0439●	.8860±.0200●	.9350±.0159●	.9491±.0259●	.8494±.0077●
$\mathrm{OPAUC}^{\mathrm{f}}$	.8431±.0311●	.8193±.0082●	.7869±.0062●	.8793±.0126●	.8030±.0028●
$\mathrm{OPAUC}^{\mathrm{rp}}$	.8453±.0320●	.8219±.0103●	.7885±.0116●	.8640±.0236●	.8256±.0029●
$\mathrm{OPAUC}^{\mathrm{pca}}$	.8492±.0323●	N/A	.8452±.0055●	N/A	N/A
OPAUCr	.8631±.0498●	.8863±.1591●	.9302±.0352●	.9778±.0106●	.8062±.0853●
$\mathrm{OPAUC}_{\mathrm{h}}$	.8574±.0395●	.8588±.1413●	.9270±.0570●	.9653±.0214●	.8080±.0961●

$\mathrm{OAM}_{\mathrm{seq}}$, $\mathrm{OAM}_{\mathrm{gra}}$, online Uni-Exp and online Uni-Squ, and the average CPU time (in seconds) are shown in Figure 1. As expected, online Uni-Squ and online Uni-Exp take the least time cost because they optimize on single-instance (univariate) loss, whereas the other algorithms work by optimizing pairwise loss. On most datasets, the running time of OPAUC is competitive to $\mathrm{OAM}_{\mathrm{seq}}$ and $\mathrm{OAM}_{\mathrm{gra}}$, except on the mnist and epsilon datasets which have the highest dimension in Table 1.

7.2. Comparisons on High-Dimensional Dense Data

In this section, we study the empirical performance for high-dimensional dense datasets. For convenience, we denote $\mathrm{OPAUC}_{\mathrm{fd}}$ by the OPAUC algorithm where the covariance matrices are approximated by frequent direction algorithm as shown in Algorithm 2, and recall that OPAUCs represents the OPAUC algorithm where the covariance matrices are approximated by sparse matrices as illustrated in Section 4.2.

To verify the effectiveness of $\mathrm{OPAUC}_{\mathrm{fd}}$ on high-dimensional dense data, we select five datasets that have a small number of features in Table 1, i.e., fourclass, letter, magic04, acoustic and ijcnn1, and then add 50,000 extra features using random fourier features (Ali and Benjamin, 2007).

Besides four online algorithms OAM$_{\text{seq}}$, OAM$_{\text{gra}}$, online Uni-Exp and online Uni-Squ, as mentioned in the previous section, we also evaluate three variants of OPAUC, whose basic idea is to project high-dimensional data to low-dimensional data and then work with OPAUC as mentioned in Section 4. In addition, we also compare with two algorithms where the covariance matrices are approximated by random projection and hashing, respectively. The detailed description is given as follows.

- **OPAUC$^{\mathbf{f}}$**: Randomly selects $1,000$ features, and then works with OPAUC;
- **OPAUC$^{\mathbf{rp}}$**: Projects into a $1,000$-dim feature space by random projection, and then works with OPAUC;
- **OPAUC$^{\mathbf{pca}}$**: Projects into a $1,000$-dim feature space by principle component analysis, and then works with OPAUC.
- **OPAUCr**: The OPAUC algorithm where the covariance are approximated by random projection, which has been suggested in our preliminary work (Gao et al., 2013);
- **OPAUC$_{\mathbf{h}}$**: The OPAUC algorithm where the covariance are approximated by using hashing technique.

Similar to Section 7.1, five-fold cross validation is executed on training sets to determine the learning rate $\eta_t \in 2^{[-12:10]}$ and the regularization parameter $\lambda \in 2^{[-10:2]}$. Due to the memory and computational limit, the buffer sizes are set to 50 for OAM$_{\text{seq}}$ and OAM$_{\text{gra}}$, and the sketch size τ is also set to 50 for OPAUC$_{\text{fd}}$, OPAUCs, OPAUCr and OPAUC$_{\text{h}}$. The performance of the methods is evaluated by five trials of 5-fold cross validation, where the AUC values are obtained by averaging over these 25 runs.

The comparison results are summarized in Table 4, and we can see clearly that the proposed OPAUC$_{\text{fd}}$ approach is superior to the other compared methods, since the pairwise t-test shows that OPAUC$_{\text{fd}}$ wins most times and never loses. Compared with Table 2, we find that the performance can be significantly improved by using random Fourier features (Ali and Benjamin, 2007). The average CPU time (in seconds) is shown in Figure 2. As can be seen, OPAUC$^{\text{pca}}$ takes the highest cost in time, and it runs out of memory for larger datasets such as magic04, acoustic and ijcnn1. The proposed OPAUC$_{\text{fd}}$ approach takes higher cost in time than other methods but with best performance.

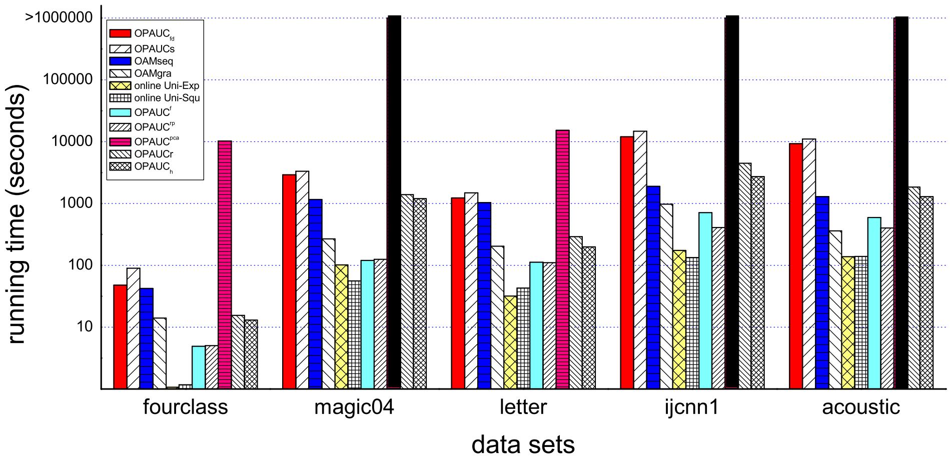

Figure 2: Comparison of the running time on high dimensional dense datasets, and full black columns indicates that OPAUC$^{\text{pca}}$ runs out of memory because of PCA.

Table 5: High-dimensional datasets ordered by feature dimensions

datasets	# inst	# feat	datasets	# inst	# feat
real-sim	72,309	20,985	sector	9,619	55,197
rcv	20,278	47,236	news20	15,935	62,061
rcv1v2	23,149	47,236	ecml2012	456,886	98,519
sector.lvr	9,619	55,197	news20.binary	19,996	1,355,191

7.3. Comparison on High-Dimensional Sparse Data

Now we investigate the empirical performance for high-dimensional sparse tasks. Eight real sparse datasets[4,5] are shown in Table 5. The news20.binary dataset contains two classes, different from news20 dataset. The original news20 and sector are multi-class datesets; in our experiments, we randomly group the multiple classes into two meta-classes each containing the same number of classes, and we also use the sector.lvr dataset which regards the largest class as positive and the union of other classes as negative. The original ecml2012 and rcv1v2 are multi-label datasets; in our experiments, we

[4] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/

[5] http://www.ecmlpkdd2012.net/discovery-challenge

Table 6: Testing AUC (mean±std.) of OPAUCs with online methods on high-dimensional datasets. ●/○ indicates that OPAUCs is significantly better/worse than the corresponding method (pairwise t-tests at 95% significance level). 'N/A' means that no result was obtained after running out 10^6 seconds (about 11.6 days).

datasets	real-sim	rcv	rcv1v2	sector.lvr
OPAUCs	.9910±.0013	.9906±.0015	.9787±.0022	.9980±.0040
OPAUC$_{\mathrm{fd}}$	.9745±.0011●	.9802±.0026●	.9633±.0031●	.9965±.0358●
OAM$_{\mathrm{seq}}$	.9840±.0061●	.9885±.0010●	.9686±.0026●	.9965±.0064●
OAM$_{\mathrm{gra}}$	.9762±.0062●	.9852±.0019●	.9604±.0025●	.9955±.0059●
online Uni-Exp	.9914±.0011	.9907±.0012	.9822±.0042○	.9969±.0093●
online Uni-Log	.9888±.0007●	.9895±.0014	.9770±.0017●	.9776±.0249●
online Uni-Squ	.9920±.0009○	.9918±.0010○	.9818±.0014○	.9669±.0260●
OPAUC$^{\mathrm{f}}$	.8105±.0042●	.7297±.0069●	.6875±.0101●	.6813±.0444●
OPAUC$^{\mathrm{rp}}$	.9444±.0036●	.9450±.0039●	.9353±.0053●	.9863±.0258●
OPAUC$^{\mathrm{pca}}$	.9834±.0009●	.9796±.0020●	.9752±.0020●	.9893±.0288●
OPAUCr	.9789±.0010●	.9831±.0016●	.9686±.0029●	.9962±.0011●
OPAUC$_{\mathrm{h}}$	.9791±.0015●	.9837±.0048●	.9700±.0032●	.9956±.0465●

datasets	sector	news20	ecml2012	news20.binary
OPAUCs	.9520±.0063	.9223±.0048	.9834±.0004	.6402±.0094
OPAUC$_{\mathrm{fd}}$	.9296±.0103●	.8840±.0059●	.9630±.0012●	.6406±.0088
OAM$_{\mathrm{seq}}$	.9163±.0087●	.8543±.0099●	N/A	.6314±.0131●
OAM$_{\mathrm{gra}}$	.9043±.0100●	.8346±.0094●	.9657±.0055●	.6351±.0135●
online Uni-Exp	.9215±.0034●	.8880±.0047●	.9820±.0016●	.6347±.0092●
online Uni-Log	.9528±.0054	.9200±.0050●	.9657±.0032●	.6340±.0032●
online Uni-Squ	.9203±.0043●	.8878±.0066●	.9530±.0041●	.6237±.0104●
OPAUC$^{\mathrm{f}}$	.6228±.0145●	.5958±.0118●	.6601±.0036●	.5068±.0086●
OPAUC$^{\mathrm{rp}}$	.7286±.0619●	.7885±.0079●	.9355±.0047●	.6212±.0072●
OPAUC$^{\mathrm{pca}}$	.8853±.0114●	.8878±.0115●	N/A	N/A
OPAUCr	.9292±.0081●	.8871±.0083●	.9828±.0008	.6389±.0136●
OPAUC$_{\mathrm{h}}$	.9265±.0218●	.8890±.0082●	.9742±.0013●	.6148±.0274●

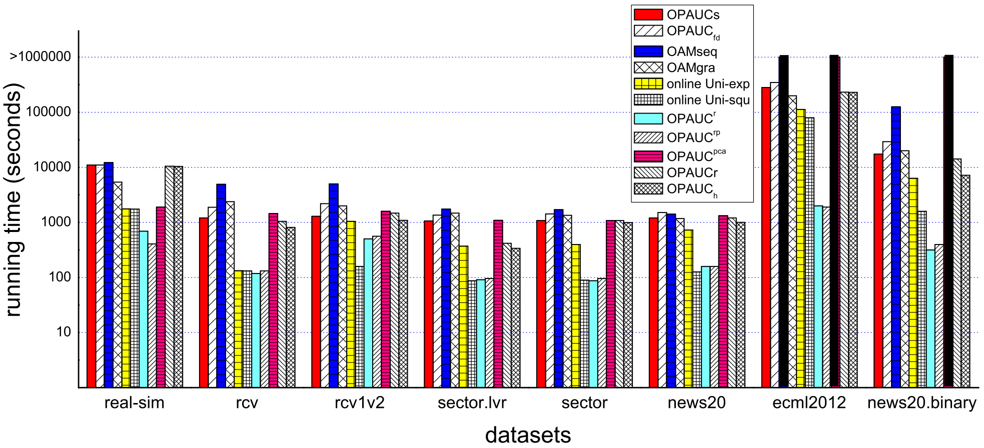

Figure 3: Comparison of the running time on high-dimensional sparse datasets. Full black columns imply that no results were returned after running out the maximal running time.

only consider the largest label and remove the features in ecml2012 dataset that take zero values for all instances.

Similar to Section 7.2, five-fold cross validation is executed on training sets to decide the learning rate $\eta_t \in 2^{[-12:10]}$ and the regularization parameter $\lambda \in 2^{[-10:2]}$. Due to memory and computational limit, the buffer sizes are set to 50 for $\mathrm{OAM}_{\mathrm{seq}}$ and $\mathrm{OAM}_{\mathrm{gra}}$, and the sketch size τ is also set to 50 for $\mathrm{OPAUC}_{\mathrm{fd}}$, OPAUCs, OPAUCr and $\mathrm{OPAUC}_{\mathrm{h}}$. The performance of the methods is evaluated by five trials of 5-fold cross validation, where the AUC values are obtained by averaging over these 25 runs.

The comparison results are summarized in Table 6 and the average CPU time (seconds) is shown in Figure 3. These results clearly show that our approximate OPAUCs approach is superior to the other compared methods. Compared with $\mathrm{OAM}_{\mathrm{seq}}$, $\mathrm{OAM}_{\mathrm{gra}}$, $\mathrm{OPAUC}_{\mathrm{fd}}$, OPAUCr and $\mathrm{OPAUC}_{\mathrm{h}}$, the time costs are comparable whereas the performance of OPAUCs is better. Online Uni-Squ and Uni-Exp are more efficient than OPAUCs because they optimize a univariate loss, but the performance of OPAUCs is highly competitive or better, except on real-sim, rcv and rcv1v2, the three datasets with less than 50,000 features and with class balance between positive and negative instances. Compared with the three variants, $\mathrm{OPAUC}^{\mathrm{f}}$ and $\mathrm{OPAUC}^{\mathrm{rp}}$ are more efficient, but with much worse performances. $\mathrm{OPAUC}^{\mathrm{pca}}$ achieves a worse performance on all datasets; particularly, on the two datasets with the largest number of features, $\mathrm{OPAUC}^{\mathrm{pca}}$ cannot return results even after

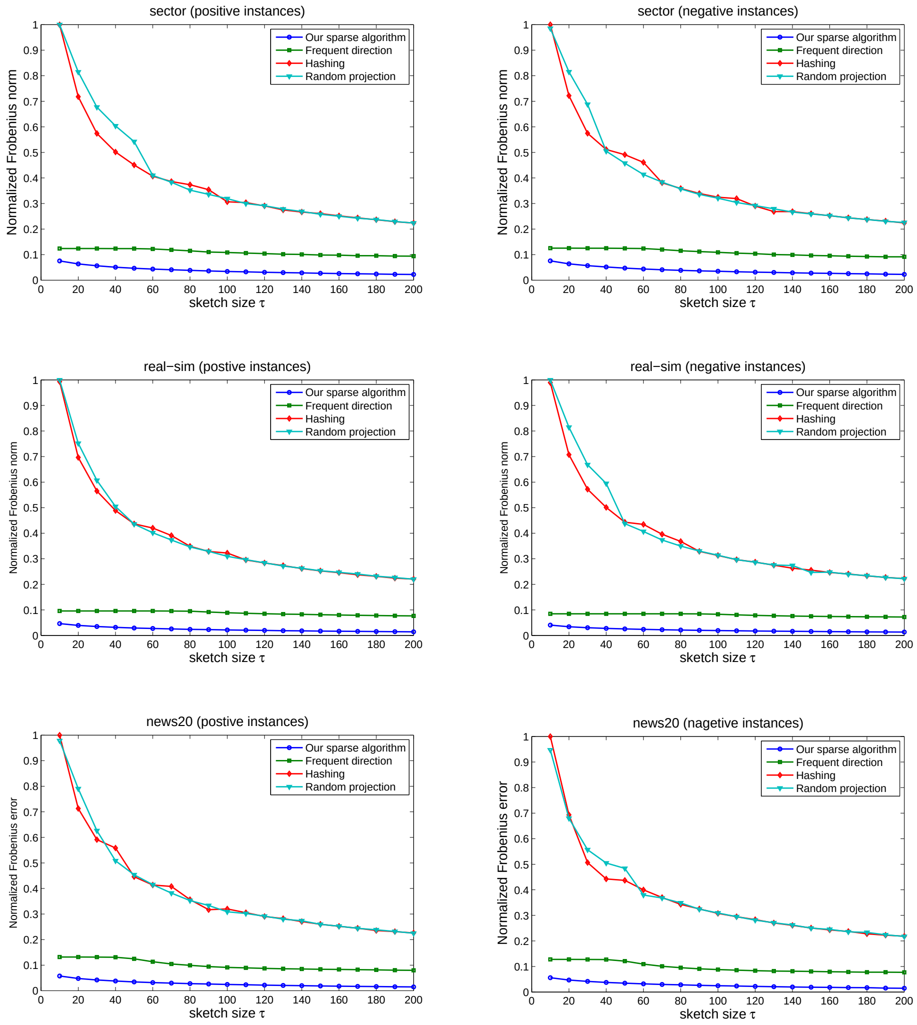

Figure 4: The comparisons of approximation error on frequent direction, random projection, hashing and our sparse algorithm.

running out 10^6 seconds (almost 11.6 days). These observations validate the effectiveness of OPAUCs for handing high-dimensional sparse data.

Finally, we try to understand why OPAUCs works better than $\mathrm{OPAUC_{fd}}$, OPAUCr and $\mathrm{OPAUC_h}$ on high-dimensional sparse datasets, although the basic idea for those approaches is to use low-rank matrices to approximate the covariance matrices. We measure the approximation error of each method by Frobenius norm. More precisely, for original matrix A and output sketch ma-

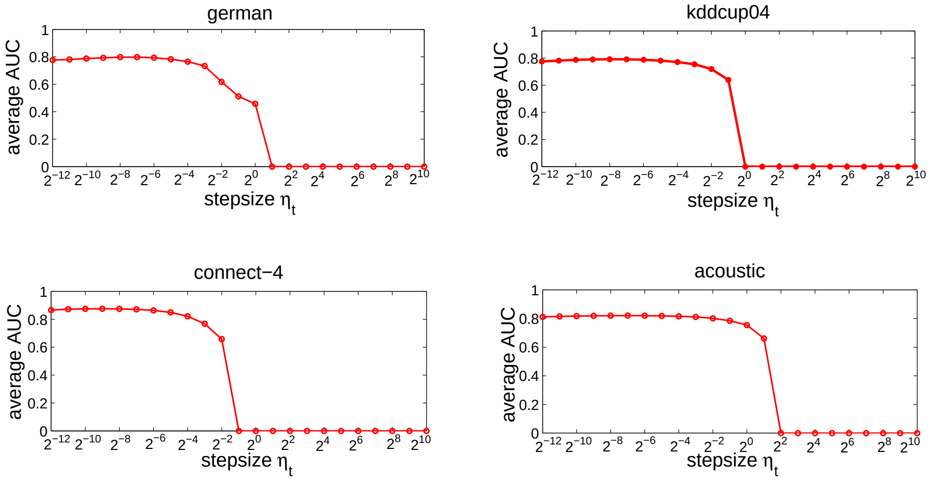

Figure 5: Influence of stepsize η_t

trix B, the empirical performance is measured by $\|A^\top A - B\|_F$ for OPAUCs, and $\|A^\top A - B^\top B\|_F$ for OPAUC$_{\text{fd}}$, OPAUCr and OPAUC$_{\text{h}}$.

Figure 4 shows the approximation error comparisons on datasets sector, news20 and real-sim. As can be seen, our proposed sparse algorithm (shown in Section 4.2) takes the best approximation to positive/negative covariance matrices, and those are rather stable. Frequent direction has better performance than random projection and hashing. This verifies the effectiveness of our sparse algorithms to high-dimensional sparse datasets.

7.4. Parameter Influence

We study the influence of parameters in this section. Figure 5 shows that stepsize η_t should not be set to values bigger than 1, whereas there is a relatively big range between $[2^{-12}, 2^{-4}]$ where OPAUC achieves good results. Figures 6 shows that OPAUC is not sensitive to the value of regularization parameter λ given that it is not set with a big value. Figure 7 studies the influence of the iterations for OPAUC, OAM$_{\text{seq}}$ and OAM$_{\text{gra}}$, and it is observable that OPAUC converges faster than the other two algorithms, which verifies our theoretical argument in Section 5.

8. Conclusion

This paper investigates the one-pass AUC optimization that requires going through the training data only once, without storing the entire dataset.

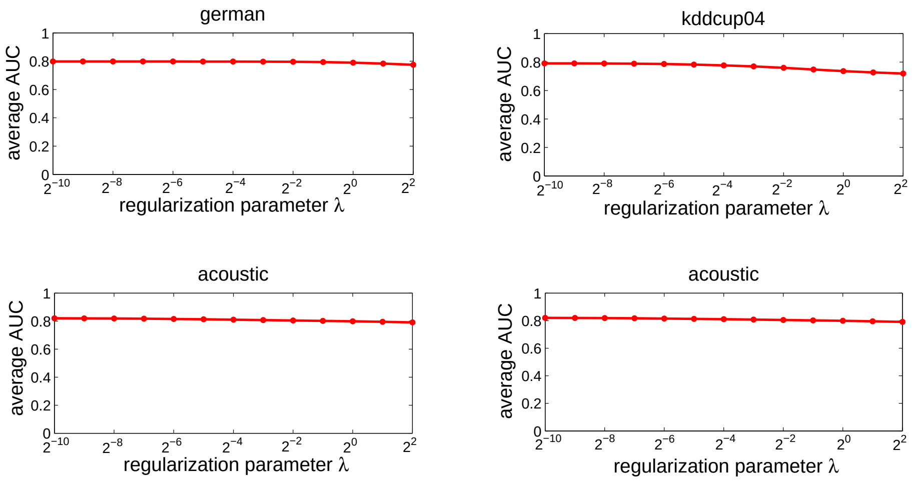

Figure 6: Influence of regularization parameter λ

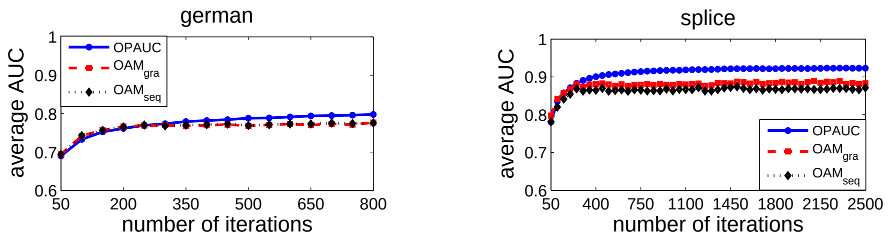

Figure 7: Convergence comparisons of OPAUC, OAM$_{\text{seq}}$ and OAM$_{\text{gra}}$

Here, a big challenge lies in the fact that AUC is measured by a sum of losses defined over pairs of instances from different classes. We propose the OPAUC approach, which employs the least square loss and requires the storing of only the first and second-statistics for the received training examples. A nice property of OPAUC is that its storage requirement is $O(d^2)$, where d is the dimension of data, independent of the number of training examples. To handle high-dimensional tasks, we develop two deterministic strategies to approximate the covariance matrices for dense and sparse datasets, respectively. The effectiveness of our proposed approach is verified both theoretically and empirically. In particular, the performance of OPAUC is significantly better than online AUC optimization approaches, and is even competitive to batch learning approaches; the approximate OPAUC is significantly better than all compared methods. An interesting future issue is to develop one-pass

AUC optimization approaches not only with a performance comparable to batch approaches, but also with an efficiency comparable to univariate loss optimization approaches.

Acknowledgement

The authors want to thank the associate editor and anonymous reviewers for helpful comments and suggestions. This research is supported by National Science Foundation of China (61333014, 61503179), the National Key Basic Research Program of China (2014CB340501), and the Jiangsu Science Foundation (BK20150586).

References

Agarwal, S., 2013. Surrogate regret bounds for the area under the ROC curve via strongly proper losses. In: Proceedings of the 26th Annual Conference on Learning Theory. Princeton, NJ, pp. 338–353.

Agarwal, S., Graepel, T., Herbrich, R., Har-Peled, S., Roth, D., 2005. Generalization bounds for the area under the roc curve. Journal of Machine Learning Research 6, 393–425.

Agarwal, S., Niyogi, P., 2009. Generalization bounds for ranking algorithms via algorithmic stability. Journal of Machine Learning Research 10, 441–474.

Agarwal, S., Roth, D., 2005. Learnability of bipartite ranking functions. In: Proceedings of the 18th Annual Conference on Learning Theory. Bertinoro, Italy, pp. 16–31.

Ali, R., Benjamin, R., 2007. Random features for large-scale kernel machines. In: Advances in Neural Information Processing Systems 3. MIT Press, Cambridge, MA, pp. 1177–1184.

Azuma, K., 1967. Weighted sums of certain dependent random variables. Tohoku Mathematical Journal 19 (3), 357–367.

Brefeld, U., Scheffer, T., 2005. Auc maximizing support vector learning. In: Proceedings of the 22nd International Conference on Machine Learning Workshop on ROC Alalysis. Bonn, Germany.

Cesa-Bianchi, N., Lugosi, G., 2006. Prediction, learning, and games. Cambridge University Press.

Clemenćon, S., Lugosi, G., Vayatis, N., 2008. Ranking and empirical minimization of U-statistics. Annals of Statistics 36 (2), 844–874.

Cortes, C., Mohri, M., 2004. AUC optimization vs. error rate minimization. In: Thrun, S., Saul, L., Schölkopf, B. (Eds.), Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, MA, pp. 313–320.

Cortes, C., Mohri, M., Rastogi, A., 2007. Magnitude-preserving ranking algorithms. In: Proceedings of the 24th Annual International Conference on Machine Learning. Corvallis, Oregon, pp. 169–176.

Devroye, L., Gyorfi, L., Lugosi, G., 1996. A Probabilistic Theory of Pattern Recognition. Springer, New York.

Egan, J., 1975. Signal detection theory and ROC curve, Series in Cognition and Perception. Academic Press, New York.

Elkan, C., 2001. The foundations of cost-sensitive learning. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence. Seattle, WA, pp. 973–978.

Flach, P. A., Hernández-Orallo, J., Ramirez, C. F., 2011. A coherent interpretation of AUC as a measure of aggregated classification performance. In: Proceedings of the 28th International Conference on Machine Learning. Bellevue, WA, pp. 657–664.

Freund, Y., Iyer, R., Schapire, R. E., Singer, Y., 2003. An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research 4, 933–969.

Gao, W., Jin, R., Zhu, S., Zhou, Z.-H., 2013. One-pass auc optimization. In: Proceedings of the 30th International Conference on Machine Learning. Atlanta, GA, pp. 906–914.

Gao, W., Zhou, Z.-H., 2013. Uniform convergence, stability and learnability for ranking problems. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence. Beijing, China, pp. 1337–1343.

Gao, W., Zhou, Z.-H., 2015. On the consistency of auc pairwise optimization. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence. Buenos Aires, Argentina, pp. 939–945.

Hand, D., 2009. Measuring classifier performance: a coherent alternative to the area under the roc curve. Machine Learning 77 (1), 103–123.

Hanley, J., McNeil, B., 1982. The meaning and use of the area under a receiver operating characteristic (roc) curve. Radiology 143, 29–36.

Hanley, J. A., McNeil, B. J., 1983. A method of comparing the areas under receiver operating characteristic curves derived from the same cases. Radiology 148 (3), 839–843.

Hazan, E., Kalai, A., Kale, S., Agarwal, A., 2006. Logarithmic regret algorithms for online convex optimization. In: Proceedings of the 19th Annual Conference on Learning Theory. Pittsburgh, PA, pp. 499–513.

Herschtal, A., Raskutti, B., 2004. Optimising area under the roc curve using gradient descent. In: Proceedings of the 21st International Conference on Machine Learning. Alberta, Canada.

Hoeffding, W., 1963. Probability inequalities for sums of bounded random variables. Journal of the American statistical association 58 (301), 13–30.

Hsieh, F., Turnbull, B., 1996. Nonparametric and semiparametric estimation of the receiver operating characteristic curve. Annals of Statistics 24 (1), 25–40.

Huang, J., Ling, C., 2005. Using auc and accuray in evaluating learing algorithms. IEEE Transactions on Knowledge and Data Engineering 17 (3), 299–310.

Joachims, T., 2005. A support vector method for multivariate performance measures. In: Proceedings of the 22nd International Conference on Machine Learning. Bonn, Germany, pp. 377–384.

Joachims, T., 2006. Training linear svms in linear time. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge Discovery and Data Mining. Philadelphia, PA, pp. 217–226.

Kar, P., Sriperumbudur, B., Jain, P., Karnick, H., 2013. On the generalization ability of online learning algorithms for pairwise loss functions. In: Proceedings of the 30th International Conference on Machine Learning. Atlanta, GA, pp. 441–449.

Kotlowski, W., Dembczynski, K., Hüllermeier, E., 2011. Bipartite ranking through minimization of univariate loss. In: Proceedings of the 28th International Conference on Machine Learning. Bellevue, WA, pp. 1113–1120.

Langford, J., Li, L., Zhang, T., 2009. Sparse online learning via truncated gradient. Journal of Machine Learning Research 10, 719–743.

Liberty, E., 2013. Simple and deterministic matrix sketching. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Chicago, IL, pp. 581–588.

Liu, X.-Y., Wu, J., Zhou, Z.-H., 2009. Exploratory undersampling for class-imbalance learning. IEEE Transactions on Systems, Man, and Cybernetics - B 39 (2), 539–550.

McDiarmid, C., 1989. On the method of bounded differences. In: Surveys in Combinatorics. Cambridge University Press, Cambridge, UK, pp. 148–188.

Menon, A. K., Williamson, R. C., 2014. Bayes-optimal scorers for bipartite ranking. In: Proceedings of the 27th Annual Conference on Learning Theory. Barcelona, Spain, pp. 68–106.

Metz, C. E., 1978. Basic principles of ROC analysis. Seminars in Nuclear Medicine 8 (4), 283–298.

Nesterov, Y., 2003. Introductory lectures on convex optimization: A basic course. Springer.

Provost, F. J., Fawcett, T., 2001. Robust classification for imprecise environments. Machine Learning 42 (3), 203–231.

Provost, F. J., Fawcett, T., Kohavi, R., 1998. The case against accuracy estimation for comparing induction algorithms. In: Proceedings of the 15th International Conference on Machine Learning. Madison, Wisconsin, pp. 445–453.

Rakhlin, A., Shamir, O., Sridharan, K., 2012. Making gradient descent optimal for strongly convex stochastic optimization. In: Proceedings of the 29th International Conference on Machine Learning. Edinburgh, Scotland, pp. 449–456.

Rudin, C., Schapire, R. E., 2009. Margin-based ranking and an equivalence between AdaBoost and RankBoost. Journal of Machine Learning Research 10, 2193–2232.

Shalev-Shwartz, S., 2007. Online learning: Theory, algorithms, and applications. Ph.D. thesis, Hebrew University of Jerusalem.

Shalev-Shwartz, S., Singer, Y., Srebro, N., Cotter, A., 2011. Pegasos: Primal estimated sub-gradient solver for svm. Mathematical Programming, Series B 127 (1), 3–30.

Srebro, N., Sridharan, K., Tewari, A., 2010. Smoothness, low noise and fast rates. In: Advances in Neural Information Processing Systems 24. MIT Press, Cambridge, MA, pp. 2199–2207.

Usunier, N., Amini, M. R., Gallinari, P., 2005. A data-dependent generalisation error bound for the auc. In: Proceedings of the 22nd International Conference on Machine Learning Workshop on ROC Alalysis. Bonn, Germany.

Wang, Y., Khardon, R., Pechyony, D., Jones, R., 2012. Generalization bounds for online learning algorithms with pairwise loss functions. In: Proceedings of the 25th Annual Conference on Learning Theory. pp. 13.1–13.22.

Wang, Y., Khardon, R., Pechyony, D., Jones, R., 2013. Generalization bounds for online learning algorithms with pairwise loss functions. CoRR/abstract 1305.2505.

Wu, J., Brubaker, S., Mullin, M., Rehg, J., 2008. Fast asymmetric learning for cascade face detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (3), 369–382.

Ying, Y., Zhou, D.-X., 2015. Online pairwise learning algorithms with kernels. CoRR/abstract 1502.07229.

Zhang, T., 2004. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In: Proceedings of the 21st International Conference on Machine Learning. Alberta, Canada.

Zhao, P., Hoi, S., Jin, R., Yang, T., 2011. Online AUC maximization. In: Proceedings of the 25th International Conference on Machine Learning. Bellevue, WA, pp. 233–240.

Zhou, X., Obuchowski, N., McClish, D., 2002. Statistical Methods in Diagnestie Medicine. John Wiley and Sons, New York.