# AUC Maximization in the Era of Big Data and AI: A Survey

TIANBAO YANG, Texas A&M University, USA
YIMING YING, University at Albany, SUNY, USA

Area under the ROC curve, a.k.a. AUC, is a measure of choice for assessing the performance of a classifier for imbalanced data. AUC maximization refers to a learning paradigm that learns a predictive model by directly maximizing its AUC score. It has been studied for more than two decades dating back to late 90s, and a huge amount of work has been devoted to AUC maximization since then. Recently, stochastic AUC maximization for big data and deep AUC maximization (DAM) for deep learning have received increasing attention and yielded dramatic impact for solving real-world problems. However, to the best our knowledge, there is no comprehensive survey of related works for AUC maximization. This article aims to address the gap by reviewing the literature in the past two decades. We not only give a holistic view of the literature but also present detailed explanations and comparisons of different papers from formulations to algorithms and theoretical guarantees. We also identify and discuss remaining and emerging issues for DAM and provide suggestions on topics for future work.

CCS Concepts: • **Computing methodologies** → **Machine learning algorithms**; • **Theory of computation** → **Continuous optimization**; **Stochastic control and optimization;**

Additional Key Words and Phrases: AUC, ROC, big data, deep learning

**ACM Reference format:**
Tianbao Yang and Yiming Ying. 2022. AUC Maximization in the Era of Big Data and AI: A Survey. *ACM Comput. Surv.* 55, 8, Article 172 (December 2022), 37 pages.
https://doi.org/10.1145/3554729



## 1 INTRODUCTION

**ROC (receiver operating characteristic)** curve is a curve of **true positive rate** (**TPR**, equivalently sensitivity or recall) versus **false positive rate** (**FPR**, equivalently fall-out) of a classifier by varying the threshold. The method was originally developed for operators of military radar receivers starting in 1941 [52]. ROC analysis has emerged as an important tool in many domains, e.g., medicine, radiology, biometrics, meteorology, forecasting of natural hazards, and is widely used in machine learning and artificial intelligence. A statistic measure associated with the ROC curve if the **area under the curve (AUC)**, which has been widely used for assessing the performance of a classifier. Another closely related measure is called partial AUC, which refers to AUC in a certain region that restricts the range of FPR and/or TPR.

T. Yang is supported by NSF Grant 2110545, NSF Career Award 1844403, and NSF Grant 1933212. Y. Ying is supported by NSF grants (IIS-1816227, IIS-2008532, IIS-2110546, and DMS-2110836).
Authors' addresses: T. Yang, Texas A&M University, College Station; email: tianbao-yang@uiowa.edu; Y. Ying, University at Albany, SUNY, Albany; email: yying@albany.edu.


0360-0300/2022/12-ART172
https://doi.org/10.1145/3554729

A standard approach in machine learning for learning a predictive model is to optimize some performance metric. A traditional performance metric of a classifier is the accuracy, i.e., the proportion of examples that are predicted correctly. However, accuracy can be misleading when the data is imbalanced, meaning that the number of data points from one class is much larger than the number of data points from the another class. In contrast, AUC is a more informative measure than accuracy for imbalanced data. However, studies show that algorithms that maximize accuracy of a model do not necessarily maximize the AUC score [28]. Hence, it is necessary to study algorithms for maximizing AUC directly.

AUC maximization in machine learning has a long history dating back to late '90s [68]. Tremendous studies have been devoted to this topic and various aspects have been studied ranging from formulations to algorithms and theories. Below, we give a brief overview with exemplar references. First, AUC maximization has been studied in the context of different learning paradigms, e.g., supervised learning [81, 152], semi-supervised learning [80, 165], **positive-unlabeled (PU)** learning [140, 143], active learning [30, 60], Bayesian learning [59], federated learning [55, 195], online learning [47, 201]. Second, models in different forms have been learned in the context of AUC maximization, including linear models [192], kernel models [68, 132], extreme learning machines [188], decision trees [45], neural networks [178], deep neural nets [194]. Third, various solvers based on different methodologies have been studied, e.g., linear programming [130], quadratic programming [68], cutting-plane methods [81], L-BFGS [94], evolutionary algorithms [106], gradient descent methods [69], stochastic gradient methods [192], other methods [13, 19]. Fourth, different theoretical guarantees have been examined, e.g., consistency [48], generalization error bounds [95], excess risk bounds [58, 193], regret bounds [201], convergence rates or sample complexities [101], stability [96, 184]. Last but not least, AUC maximization has been successfully investigated in a variety of applications [6, 9, 43, 61, 77, 150, 153, 166, 167, 176, 204, 208], e.g., medical image classification [196] and molecular properties prediction [171], to mention but a few.

A bulk of studies related to AUC maximization revolve around the development of the solver, i.e., optimization algorithms, for learning a predictive model. The reason is that, compared with the traditional metric of accuracy, the AUC score is non-decomposable over individual examples, which renders its optimization much more challenging, especially for big data. The research of AUC maximization algorithms has experienced four different ages in the long history of two decades, namely, full-batch-based methods for the first age (roughly 2000–2010), online methods for the second age (roughly 2011–2015), stochastic methods for the third age (roughly 2016–2019), and deep learning methods for the recent age (roughly 2020–present). The first three ages focus on learning linear models or kernelized models, and the last age focuses on deep neural networks. In each age, there have been seminal works in rigorous optimization algorithms that play important roles in the evolution of AUC maximization methods. The four ages are illustrated in Figure 1.

To the best of our knowledge, there is no comprehensive survey devoted to AUC maximization. The only related survey work is Reference [162], published in 2011. Nevertheless, it focuses on ordinal regression and does not provide a comprehensive survey of optimization algorithms for AUC maximization with theoretical guarantees. This article aims to address this gap by providing a comprehensive review of related works for AUC maximization, with a particular focus on the optimization algorithms. We will cover important works in all four ages about the optimization algorithms and discuss their properties. The remainder of this article is organized as follows:

- We provide some background for AUC and AUC estimators in Section 2. We give definitions for both AUC and partial AUC and derive their non-parametric estimators.
- In Section 3, we review different objective functions for AUC maximization and mainly discuss three families of objectives.

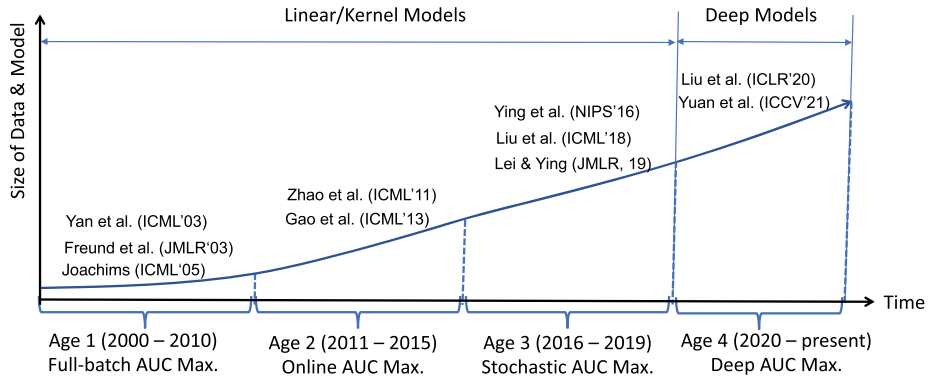


Fig. 1. Four ages of AUC maximization and exemplar works in each stage.

- We review full-batch-based methods for solving AUC maximization in the first age for both AUC maximization and partial AUC maximization in Section 4.
- In Section 5, we present two classes of online optimization methods for AUC maximization and discuss their properties.
- We present stochastic optimization methods in both offline setting and online setting in Section 6, and compare their properties.
- In Section 7, we survey recent papers about non-convex optimization for deep AUC and partial AUC maximization, and discuss their applications in the real world.
- In Section 8, we discuss remaining and emerging issues in deep AUC maximization and provide suggestions of topics for future work. Finally, we conclude in Section 9.

**Disclaimer.** Before ending this section, we would like to point out that we have done our best to include as many related works in machine learning as possible and may innocently miss some relevant papers in machine learning or other areas. We also emphasize that this article is about maximization of areas under ROC curves and does not cover the maximization of areas under Precision-Recall curves. Finally, we present a list of three fundamental papers of AUC, top 10 Cited Papers (as of 07/28/2022) related to AUC maximization, and two representative works for deep AUC maximization in Table 1.

## 2 BACKGROUND

**Notations.** Let $\mathbb{I}(\cdot)$ be an indicator function of a predicate, and $[n] = \{1, \ldots, n\}$. Let $\mathbf{z} = (\mathbf{x}, y)$ denote an input-output pair, where $\mathbf{x} \in \mathcal{X}$ denotes the input data and $y \in \{1, -1\}$ denotes its class label. Let $\mathcal{P}_+$ denote the distribution of positive examples and $\mathcal{P}_-$ denote the distribution of negative examples. Let $f(\mathbf{x}) : \mathcal{X} \to \mathbb{R}$ denote a predictive function to be learned. It is usually parameterized by a vector $\mathbf{w} \in \mathbb{R}^d$ and we use the notation $f_{\mathbf{w}}(\cdot)$ to emphasize that it is a parameterized model. Let $\ell(\mathbf{w}; \mathbf{x}, \mathbf{x}') = \ell(f_{\mathbf{w}}(\mathbf{x}') - f_{\mathbf{w}}(\mathbf{x}))$ denote a pairwise loss for a positive-negative pair $(\mathbf{x}, \mathbf{x}')$.

For a set of given training examples $\mathcal{S} = \{(\mathbf{x}_i, y_i), i \in [n]\}$ in the offline setting, let $\mathcal{S}_+$ and $\mathcal{S}_-$ be the subsets of $\mathcal{S}$ with only positive and negative examples, respectively, and let $n_+ = |\mathcal{S}_+|$ and $n_- = |\mathcal{S}_-|$ be the number of positive and negative examples, respectively. Denote by $\mathcal{S}^{\downarrow}[k_1, k_2] \subseteq \mathcal{S}$ the subset of examples whose rank in terms of their prediction scores in the descending order are in the range of $[k_1, k_2]$, where $k_1 \leq k_2$. Similarly, let $\mathcal{S}^{\uparrow}[k_1, k_2] \subseteq \mathcal{S}$ denote the subset of examples whose rank in terms of their prediction scores in the ascending order are in the range of $[k_1, k_2]$, where $k_1 \leq k_2$. We denote by $\mathbb{E}_{\mathbf{x}\sim\mathcal{S}}$ the average over $\mathbf{x} \in \mathcal{S}$. Let $D(\mathbf{p}, \mathbf{q})$ denote the KL divergence

Table 1. Three Fundamental Papers of AUC, Top 10 Cited Papers Related to AUC Maximization, and Two Representative Works for Deep AUC Maximization

| Title | Authors | Year | Citations | Venue | Reference |
|---|---|---|---|---|---|
| 1. The Meaning and Use of the Area under a Receiver Operating Characteristic | J. A. Hanley and B. J. McNeil. | 1982 | 22314 | Radiology | [63] |
| 2. Analyzing a Portion of the ROC Curve | D. Katzman McClish | 1989 | 769 | Med Decis Making | [110] |
| 3. Partial AUC Estimation and Regression | L. Dodd and M. Pepe | 2003 | 389 | Biometrics | [37] |
| 1. A Support Vector Method for Multivariate Performance Measures | T. Joachims | 2005 | 1022 | ICML | [81] |
| 2. AUC Optimization vs. Error Rate Minimization | C. Cortes and M. Mohri | 2003 | 721 | NIPS | [28] |
| 3. Optimizing Classifier Performance via an Approximation to the Wilcoxon-Mann-Whitney Statistic | L. Yan, R. Dodier, M. C. Mozer, and R. Wolniewicz | 2003 | 350 | ICML | [178] |
| 4. Optimising Area under the ROC Curve Using Gradient Descent | A. Herschtal and B. Raskutti | 2004 | 231 | ICML | [69] |
| 5. Online AUC Maximization | P. Zhao, S. C. H. Hoi, R. Jin, and T. Yang | 2011 | 217 | ICML | [201] |
| 6. AUC Maximizing Support Vector Learning | U. Brefeld, T. Scheffer | 2005 | 184 | ICML (workshop) | [18] |
| 7. The P-Norm Push: A Simple Convex Ranking Algorithm that Concentrates at the Top of the List | C. Rudin | 2009 | 179 | JMLR | [142] |
| 8. One-pass AUC Optimization | W. Gao, R. Jin, S. Zhu, and Z. Zhou | 2013 | 153 | ICML | [47] |
| 9. Efficient AUC Optimization for Classification | T. Calders and S. Jaroszewicz | 2007 | 128 | PKDD | [19] |
| 10. Stochastic Online AUC Maximization | Y. Ying, L. Wen, and S. Lyu | 2016 | 104 | NIPS | [192] |
| 1. Stochastic AUC Maximization with Deep Neural Networks | M. Liu, Z. Yuan, Y. Ying, and T. Yang. | 2020 | 35 | ICLR | [101] |
| 2. Robust Deep AUC Maximization: A New Surrogate Loss and Empirical Studies on Medical Image Classification | Z. Yuan, Y. Yan, M. Sonka and T. Yang. | 2021 | 13 | ICCV | [196] |

The citations data is from Google Scholar as of July 28, 2022, which is included for reference and is by no means the only metric to measure the influence of a paper. **Remark**: We do not include the highly cited paper [45] in the list due to that it was studied in Reference [28] for AUC maximization.

between two probability vectors. Let $\Delta$ denote a simplex of a proper dimension, and $\Pi_\Omega[\cdot]$ denote the standard Euclidean projection onto a set $\Omega$.

### 2.1 Definitions of AUC, Partial AUC, Two-way Partial AUC

In this subsection, we fix the predictive model $f(\cdot)$ and present the definitions and formulas for computing AUC of $f$. For a given threshold $t$, the **true positive rate (TPR)** can be written as $\text{TPR}(t) = \Pr(f(\mathbf{x}) > t | y = 1) = \mathbb{E}_{\mathbf{x}\sim\mathcal{P}_+}[\mathbb{I}(f(\mathbf{x}) > t)]$, and the **false positive rate (FPR)** can be written as $\text{FPR}(t) = \Pr(f(\mathbf{x}) > t | y = -1) = \mathbb{E}_{\mathbf{x}\sim\mathcal{P}_-}[\mathbb{I}(f(\mathbf{x}) > t)]$. Let $F_-(t) = 1 - \text{FPR}(t)$ denote the cumulative density function of the random variable $f(\mathbf{x})$ for $\mathbf{x} \sim \mathcal{P}_-$. Let $p_-(t)$ denotes its corresponding probability density function. Similarly, let $F_+(t) = 1 - \text{TPR}(t)$ and $p_+(t)$ denote the cumulative density function and the probability density function of $f(\mathbf{x})$ for $\mathbf{x} \sim \mathcal{P}_+$, respectively.

For a given $u \in [0, 1]$, let $\text{FPR}^{-1}(u) = \inf\{t \in \mathbb{R} : \text{FPR}(t) \leq u\}$. The ROC curve is defined as $\{u, \text{ROC}(u)\}$, where $u \in [0, 1]$ and $\text{ROC}(u) = \text{TPR}(\text{FPR}^{-1}(u))$. The AUC score of $f$ is given by

$$\begin{aligned}\text{AUC}(f) &= \int_0^1 \text{ROC}(u)du = \int_{-\infty}^{\infty} \text{TPR}(t)dF_-(t) = \int_{-\infty}^{\infty} \text{TPR}(t)p_-(t)dt \\ &= \int_{-\infty}^{\infty}\int_t^{\infty} p_+(s)dsp_-(t)dt = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p_+(s)p_-(t)\mathbb{I}(s > t)dsdt. \end{aligned} \tag{1}$$

The above expression also gives a probabilistic interpretation of AUC [63], i.e.,

$$\text{AUC}(f) = \Pr(f(\mathbf{x}_+) > f(\mathbf{x}_-)) = \mathbb{E}_{\mathbf{x}_+\sim\mathcal{P}_+, \mathbf{x}_-\sim\mathcal{P}_-}[\mathbb{I}(f(\mathbf{x}_+) > f(\mathbf{x}_-))]. \tag{2}$$

The normal AUC measure could be misleading when the data is highly imbalanced. In many applications (e.g., medical diagnostics), we would like to control the FPR in a certain range, e.g., $\text{FPR} \in (\alpha, \beta)$. Hence, another measure of interest is **partial AUC (pAUC)** with FRP restricted in the range $(\alpha, \beta)$, which is given by

$$\text{pAUC}(f, \alpha, \beta) = \int_{\text{FPR}^{-1}(\beta)}^{\text{FPR}^{-1}(\alpha)} \text{TPR}(t)dF_-(t) = \int_{\text{FPR}^{-1}(\beta)}^{\text{FPR}^{-1}(\alpha)} \int_{-\infty}^{\infty} p_+(s)p_-(t)\mathbb{I}(s > t)dsdt. \tag{3}$$

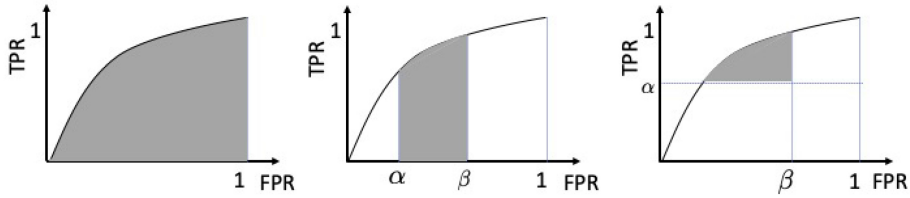


Fig. 2. From left to right: AUC, one-way pAUC, two-way pAUC.

This expression gives a probabilistic interpretation of pAUC, which was first shown in Reference [37], i.e.,

$$\text{pAUC}(f, \alpha, \beta) = \Pr(f(\mathbf{x}_+) > f(\mathbf{x}_-), f(\mathbf{x}_-) \in [\text{FPR}^{-1}(\beta), \text{FPR}^{-1}(\alpha)]). \tag{4}$$

In contrast to pAUC defined above that is also referred to as one-way pAUC, two-way pAUC has been also studied [181]. A two-way pAUC is defined by specifying an upper bound $\beta$ on the FPR and a lower bound on $\alpha$ on the TPR. Then, the two-way pAUC (TPAUC) is given by

$$\text{TPAUC}(f, \alpha, \beta) = \Pr(f(\mathbf{x}_+) > f(\mathbf{x}_-), f(\mathbf{x}_-) \geq \text{FPR}^{-1}(\beta), f(\mathbf{x}_+) \leq \text{TPR}^{-1}(\alpha)\}). \tag{5}$$

An illustration of AUC, one-way partial AUC and two-way partial AUC is given in Figure 2.

### 2.2 Non-Parametric Estimators

Given a set of examples $\mathcal{S} = \mathcal{S}_+ \cup \mathcal{S}_-$, how can we estimate AUC and pAUC? There are parametric estimators assuming the prediction scores following a particular distribution (e.g., normal distribution) [110], and non-parametric estimators that do not make any assumptions regarding the distribution of prediction scores. We will focus on non-parametric estimators below, since they are widely used for AUC maximization.

According to the probabilistic interpretation of AUC in Equation (2), a non-parametric estimator can be computed as follows that corresponds to the Mann-Whitney U-statistic [63]:

$$\text{AUC}(f; \mathcal{S}) = \frac{1}{n_+}\frac{1}{n_-}\sum_{\mathbf{x}_i \in \mathcal{S}_+}\sum_{\mathbf{x}_j \in \mathcal{S}_-} \mathbb{I}(f(\mathbf{x}_i) > f(\mathbf{x}_j)). \tag{6}$$

A (non-normalized) non-parametric estimator of pAUC can be computed by [37]:

$$\text{pAUC}(f, \alpha, \beta; \mathcal{S}) = \frac{1}{n_+}\frac{1}{n_-}\sum_{\mathbf{x}_i \in \mathcal{S}_+}\sum_{\mathbf{x}_j \in \mathcal{S}_-} \mathbb{I}(f(\mathbf{x}_i) > f(\mathbf{x}_j), f(\mathbf{x}_j) \in (q_\beta, q_\alpha)),$$

where $q_\alpha$ denotes the $\alpha$ quantile of $f(\mathbf{x}_-), \mathbf{x}_- \sim \mathcal{P}_-$. The quantiles $q_\alpha, q_\beta$ are usually replaced by their empirical estimations, which gives the following non-normalized estimator of pAUC:

$$\text{pAUC}(f, \alpha, \beta; \mathcal{S}) = \frac{1}{n_+}\frac{1}{n_-}\sum_{\mathbf{x}_i \in \mathcal{S}_+}\sum_{\mathbf{x}_j \in \mathcal{S}_-^{\downarrow}[k_1+1, k_2]} \mathbb{I}(f(\mathbf{x}_i) > f(\mathbf{x}_j)), \tag{7}$$

where $k_1 = \lceil n_-\alpha \rceil, k_2 = \lfloor n_-\beta \rfloor$. Similarly, a (non-normalized) non-parametric estimator for two-way pAUC is given by

$$\text{TPAUC}(f, \alpha, \beta; \mathcal{S}) = \frac{1}{n_+}\frac{1}{n_-}\sum_{\mathbf{x}_i \in \mathcal{S}_+^{\uparrow}[1, k_1]}\sum_{\mathbf{x}_j \in \mathcal{S}_-^{\downarrow}[1, k_2]} \mathbb{I}(f(\mathbf{x}_i) > f(\mathbf{x}_j)), \tag{8}$$

where $k_1 = \lfloor n_+\alpha \rfloor, k_2 = \lfloor n_-\beta \rfloor$.

Table 2. Different Surrogate Loss Functions $\ell(s)$ Used in AUC Maximization

| Name | Form | Parameters | Remarks | References |
|---|---|---|---|---|
| Square | $\ell(s) = (s+c)^2$ | $c > 0$ | Consistent | [35, 47, 103, 149, 192] |
| Hinge | $\ell(s) = \max(0, c+s)$ | $c > 0$ | Non-consistent | [18, 86, 87, 169] |
| Squared Hinge | $\ell(s) = max(0, c+s)^2$ | $c > 0$ | Consistent | [18, 87, 178] |
| Logistic | $\ell(s) = -\log \frac{1}{1+\exp(cs)}$ | $c > 0$ | Consistent | [154] |
| Exponential Loss | $\ell(s) = \exp(cs)$ | $c > 0$ | Consistent | [45] |
| Barrier Hinge | $\ell(s) = \max(-b(r+s)+r, \max(b(s-r), r-s))$ | $r > 0, b > 0$ | Noisy Labels | [22] |
| Sigmoid | $\ell(s) = \frac{1}{1+\exp(-cs)}$ | $c > 0$ | Non-convex | [19, 69, 80, 159, 170] |
| Ramp Function | $\ell(s) = \max(0, 1+s) - \max(0, c+s)$ | $c < 0$ | Non-convex | [25] |
| CDF of Normal Distribution | $\ell(s) = \frac{1}{2}(1 + erf(\frac{s}{\sqrt{2}c}))$ | $c > 0$ | Non-convex | [90] |
| Exponential-type | $\ell(s) = 1 - \exp(-cs)$ | $c > 0$ | Non-convex | [157] |
| Chebyshev Polynomial | $\ell(s) = \sum_{k=0}^{m} c_k s^k$ | $m > 0$ | Decomposable | [19] |

## 3 SURROGATE OBJECTIVES FOR AUC MAXIMIZATION

**Objectives based on a Pairwise Surrogate Loss.** For AUC maximization, one often replaces the indicator function in the non-parametric estimators of AUCs defined above by a surrogate loss function $\ell(f_{\mathbf{w}}(\mathbf{x}_-) - f_{\mathbf{w}}(\mathbf{x}_+))$ of $\mathbb{I}(f_{\mathbf{w}}(\mathbf{x}_-) \geq f_{\mathbf{w}}(\mathbf{x}_+))$ to formulate the objective function. As a result, AUC maximization can be formulated as

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n_+ n_-} \sum_{\mathbf{x}_i \in \mathcal{S}_+} \sum_{\mathbf{x}_j \in \mathcal{S}_-} \ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i)), \tag{9}$$

and (one-way) pAUC maximization can be formulated as

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n_+}\frac{1}{n_-} \sum_{\mathbf{x}_i \in \mathcal{S}_+} \sum_{\mathbf{x}_j \in \mathcal{S}_-^{\downarrow}[k_1+1, k_2]} \ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i)), \tag{10}$$

and two-way pAUC maximization can be formulated as

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n_+}\frac{1}{n_-} \sum_{\mathbf{x}_i \in \mathcal{S}_+^{\uparrow}[1, k_1]} \sum_{\mathbf{x}_j \in \mathcal{S}_-^{\downarrow}[1, k_2]} \ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i)). \tag{11}$$

Regarding the pairwise surrogate loss $\ell(f_{\mathbf{w}}(\mathbf{x}_-) - f_{\mathbf{w}}(\mathbf{x}_+))$, different choices have been investigated in the literature. A list of different surrogate loss functions with a sampling of references are summarized in Table 2. An important property of pairwise surrogate loss for AUC maximization is its consistency [48]. Loosely speaking, a pairwise surrogate loss $\ell(\cdot)$ is called consistent if optimizing the surrogate loss with infinite amount of data gives a solution to optimizing the original AUC score. A rigorous definition is given in Reference [48]. A necessary condition for the consistency of a pairwise loss $\ell(f(\mathbf{x}') - f(\mathbf{x}))$ for a positive-negative pair $(\mathbf{x}, \mathbf{x}')$ is that $\ell$ is a convex, differentiable and non-decreasing function with $\ell'(0) > 0$ [48].

**Min-Max Objectives for AUC Maximization.** One issue of the pairwise loss-based objective is that it needs to explicitly construct the positive-negative pairs, which is not suitable for online learning where the data comes sequentially and distributed optimization where data is distributed over many different machines. To address this issue, Ying et al. [192] propose to formulate an equivalent min-max objective for using a **pairwise square loss**. Specially, when $\ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i)) = (c + f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i))^2$, the problem in Equation (9) is equivalent to

$$\min_{\mathbf{w}\in\mathbb{R}^d, (a,b)\in\mathbb{R}^2} \max_{\alpha\in\mathbb{R}} F(\mathbf{w}, a, b, \alpha) := \mathbb{E}_{\mathbf{z}}\left[F(\mathbf{w}, a, b, \alpha; \mathbf{z})\right], \tag{12}$$

where $\mathbb{E}_{\mathbf{z}}$ denotes the empirical average of training data (offline setting) or expectation over underlying distribution (online setting), $F(\mathbf{w}, a, b, \alpha; \mathbf{z})$ is given by

$$F(\mathbf{w}, a, b, \alpha; \mathbf{z}) = (1-p)\left(f_{\mathbf{w}}(\mathbf{x}) - a\right)^2 \mathbb{I}(y=1) + p(f_{\mathbf{w}}(\mathbf{x}) - b)^2 \mathbb{I}(y=-1) \\ - p(1-p)\alpha^2 + 2\alpha\left(p(1-p)c + pf_{\mathbf{w}}(\mathbf{x})\mathbb{I}(y=-1) - (1-p)f_{\mathbf{w}}(\mathbf{x})\mathbb{I}(y=1)\right), \tag{13}$$

and $p = \Pr(y=1)$ (online setting) or $p = n_+/n$ (offline setting). A benefit of this objective function is that it is decomposable over individual examples. Hence, it enables one to develop efficient stochastic algorithms for updating the model parameter $\mathbf{w}$ without explicitly constructing and handling positive-negative pairs. It is notable that a similar min-max formulation for AUC maximization was also independently examined in Reference [134] for the offline setting.

Recently, Yuan et al. [196] reveal some potential issues of optimizing pairwise square loss and its equivalent min-max objective. They demonstrate that optimizing the pairwise square loss or its equivalent min-max objective is sensitive to noisy data and also has adverse effect on easy data. To address these issues, they decompose the square loss-based objective into three components:

$$\mathbb{E}[(c - h_{\mathbf{w}}(\mathbf{x}) + h_{\mathbf{w}}(\mathbf{x}'))^2 | y=1, y'=-1] \\ = \mathbb{E}[(h_{\mathbf{w}}(\mathbf{x}) - a(\mathbf{w}))^2 | y=1] + \mathbb{E}[(h_{\mathbf{w}}(\mathbf{x}') - b(\mathbf{w}))^2 | y'=1] + (c - a(\mathbf{w}) + b(\mathbf{w}))^2, \tag{14}$$

where $a(\mathbf{w}) = \mathbb{E}_{\mathbf{x}}[f_{\mathbf{w}}(\mathbf{x})|y=1]$ and $b(\mathbf{w}) = \mathbb{E}_{\mathbf{x}}[f_{\mathbf{w}}(\mathbf{x})|y=-1]$, and they propose to replace the last term by using a squared hinge loss:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \mathbb{E}[(h_{\mathbf{w}}(\mathbf{x}) - a(\mathbf{w}))^2 | y=1] + \mathbb{E}[(h_{\mathbf{w}}(\mathbf{x}') - b(\mathbf{w}))^2 | y'=1] + (c - a(\mathbf{w}) + b(\mathbf{w}))_+^2, \tag{15}$$

whose objective is referred to as min-max margin loss [196]. For solving the above problem, they formulate the problem into an equivalent min-max optimization problem:

$$\min_{\mathbf{w}\in\mathbb{R}^d, (a,b)\in\mathbb{R}^2} \max_{\alpha\geq 0} f(\mathbf{w}, a, b, \alpha) := \mathbb{E}_{\mathbf{z}}\left[F(\mathbf{w}, a, b, \alpha; \mathbf{z})\right], \tag{16}$$

where $F(\mathbf{w}, a, b, \alpha; \mathbf{z})$ is the same as Equation (13). The difference between the above objective and Reference (12) is that there is a non-negative constraint on the dual variable $\alpha \geq 0$.

**Composite Objectives for AUC Maximization.** Recently, Zhu et al. [207] propose another family of objectives for AUC maximization, which subsumes min-max objective of the pairwise square loss and the min-max margin loss as special cases. The objective consists of three terms:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \mathbb{E}[(f_{\mathbf{w}}(\mathbf{x}) - a(\mathbf{w}))^2 | y=1] + \mathbb{E}[(f_{\mathbf{w}}(\mathbf{x}') - b(\mathbf{w}))^2 | y'=1] + \ell(c - a(\mathbf{w}) + b(\mathbf{w}))^2, \tag{17}$$

where $\ell(\cdot)$ is a surrogate loss. When $\ell(\cdot)$ is a square function, the above objective is equivalent to the pairwise square loss-based objective or the min-max objective in Equation (12). When $\ell(\cdot)$ is a squared hinge function, the above objective is equivalent to the min-max margin objective (16). Other choices of $\ell(\cdot)$ are possible [207]. For solving the above objective, it can be transformed into

$$\min_{\mathbf{w}\in\mathbb{R}^d, (a,b)\in\mathbb{R}^2} \mathbb{E}[(f_{\mathbf{w}}(\mathbf{x}) - a)^2 | y=1] + \mathbb{E}[(f_{\mathbf{w}}(\mathbf{x}') - b)^2 | y'=1] + \ell(c - a(\mathbf{w}) + b(\mathbf{w}))^2, \tag{18}$$

where the last term can be regarded as a two-level stochastic compositional function [49, 164]. Another way is to view all three terms in Equation (17) as compositional functions.

It is notable that a regularization term about $\mathbf{w}$ (e.g., $\ell_2$ norm square) can be added to the above objectives for improving generalization. In addition, formulations can be extended to the multi-class scenario following the one-vs.-all or one-vs.-one settings [62, 101, 186].

## 4 FULL BATCH BASED METHODS - THE FIRST AGE

Earlier works for AUC maximization use full batch-based methods, which process all training examples at each iteration in the algorithmic optimization. Notable optimization algorithms for AUC maximization include the quadratic programming, gradient decent methods, cutting plane algorithms, and boosting-type methods.

### 4.1 AUC Maximization

**Quadratic Programming.** To the best of our knowledge, the earliest work dates back to 1999 [68] and derives the dual problem of the **support vector machine (SVM)** formulation for ordinal regression in the kernel setting. Quadratic programming is then employed to obtain the optimal solution that can apply to AUC maximization. References [18, 138] use a similar optimization algorithm for AUC maximization with the hinge loss. Since the number of constraints and parameters grows quadratically in the number of examples, running such quadratic programming for AUC maximization is very computationally expensive for large-scale datasets. To mitigate such computational burden, in References [18, 138] heuristic tricks using k-means clustering and k-nearest neighborhood are proposed to reduce the number of constraints. However, such approximate solutions do not guarantee an optimal solution to the original AUC maximization problem.

**Gradient Descent Methods.** Gradient descent methods are used in References [19, 69, 178] for AUC maximization. Reference [178] is probably the first work that applies the gradient descent method for AUC maximization. They use the the hinge function with a power $p > 1$ as the surrogate loss. One year later, Reference [69] considers improving the gradient descent algorithm for AUC maximization, where they use the sigmoid function as a surrogate loss. They also propose a heuristic technique by reducing the number of positive-negative pairs used in the gradient descent methods. In particular, for each negative data they only construct a pairwise loss with only one positive data. However, the quality of such approximation highly depends on the properties of the dataset. When the examples have large intra-variance, their objective could yield poor performance. Reference [19] uses a different method to improve the scalability of the gradient descent method. In particular, they use the Chebyshev polynomial to approximate the indicator function in the original formulation of the AUC score given by Equation (6) and then a gradient descent method is employed to optimize such approximated AUC score, which only requires a linear scan of all examples at each iteration without explicitly working on all pairs.

**Cutting Plane and Accelerated Gradient-based Methods.** The seminal work by Joachims [81] uses the cutting plane algorithms to optimize a general multivariate performance measure including the AUC score. The basic principle behind this optimization algorithm is to, at each iteration, solve a quadratic programming problem subject to a selected subset of constraints. The sufficient subset of constraints is generated by gradually adding the currently most violated constraint in each iteration. The cutting plane methods converge with an iteration complexity of $O(\frac{1}{\lambda\epsilon})$ to find a $\epsilon$-accurate solution, where $\lambda$ is the regularization parameter in the formulation. Zhang et al. [198] work on the dual form of the formulation for optimizing the multivariate performance measure, which may be not smooth, and use the smoothing techniques [127] to smooth the empirical objective function. Then, the Nesterov's accelerated gradient method [126] is employed to optimize the smoothed objective function, which has an iteration complexity of $\max(O(\frac{1}{\epsilon}, \frac{1}{\sqrt{\lambda\epsilon}}))$.

**Boosting Methods.** Freund et al. [45] propose a boosting method named RankBoost for bipartite ranking, which is applicable to AUC maximization. The RankBoost algorithm is based on Freund and Schapire's AdaBoost algorithm [46] and its successor developed by Schapire and Singer [144]. RankBoost works by combining many "weak" rankings of the given instances to learn

a strong ranking model. The RankBoost algorithm was later applied to AUC maximization [28]. The boosting methods for AUC maximization have also been examined in Reference [105].

### 4.2 Partial AUC Maximization

Compared to AUC maximization, partial AUC maximization is much more challenging due to that it involves selection of examples whose prediction scores are in a certain range. We provide a survey of partial AUC maximization according to the chronological order and group them according to the underlying methodologies.

**Indirect Methods.** Wu et al. [173] propose a new **support vector machine (SVM)** named assymetric SVM, which aims to lower the false positive rate while maximizing the margin. To achieve this, it maximizes two margins, the core-margin (i.e., the margin between the negative class and the high confidence subset of the positive class), and the traditional class-margin. By enlarging the core-margin, it is able to enclose the core (i.e., high confident examples) of the positive class in a set. The authors employ **Sequential Minimal Optimization (SMO)** to solve the resulting objective.

Rudin [142] proposes the p-norm push method for bipartitie ranking, which is to optimize a measure focusing on the left end of the ROC curve aiming to the make the leftmost portion of ROC curve higher. The measure to be minimized is defined as a sum of p-norm of the heights of negative examples, where the height of a negative example is defined as the number of positive examples that are ranked lower than the negative example. The author proposes a boosting-type algorithm for optimizing the p-norm push objective.

Later, Agarwal [2] proposes the infinite-push method to minimize the maximal height of all negative examples, which can be considered as an empirical estimator of pAUC with the FPR controlled below $1/n_-$. The author proposes a gradient descent algorithm for solving the infinite-push objective, which suffers a higher per-iteration cost in the order of $O(n_+n_-d+n_+n_- \log(n_+n_-))$ and an iteration complexity of $O(1/\epsilon^2)$, where $d$ is the dimensionality of input data.

Rakotomamonjy [139] extends the infinite-push method to handle sparsity-inducing regularizers and proposes an ADMM-based algorithm for optimizing the problem, which has a per-iteration cost of $O(n_+n_-d + n_+n_- \log(n_+n_-))$ and an iteration complexity of $O(1/\epsilon)$. In 2014, Li, Jin, and Zhou [98] propose a method called TopPush for optimizing the infinity-push objective. The authors use a different formulation from that in Reference [2] where each positive example is only compared with the negative example with the highest score before computing the loss, which leads to a more efficient algorithm with a per-iteration cost of $O((n_+ + n_-)d)$. They employ the Nesterov's accelerated gradient method to optimize the dual objective with an iteration complexity of $O(1/\sqrt{\epsilon})$.

**Boosting-type methods.** Komori and Eguchi [90] propose a boosting-style algorithm named pAUCboost for partial AUC maximization. In this work, the indicator function $\mathbb{I}(f(\mathbf{x}_-) > f(\mathbf{x}_+))$ is approximated by the cumulative density function of the normal distribution. The weak leaner is defined by a natural cubic spline. To simplify the optimization for the weaker learner and its weight at each iteration, the algorithm first employs one-step Newton-Raphson to update the weight and then solves for the optimal weaker learner given its weight. However, it does not discuss complexity and efficiency in finding weaker learners for maximizing pAUC at each iteration. Takenouchi, Komori, and Eguchi [157] propose a more principled boosting method for pAUC maximization named pU-AUCBoost, where U stands for a surrogate function of the indicator $\mathbb{I}(f(\mathbf{x}_+) > f(\mathbf{x}_-))$. To address the inter-dependency issue between the weak learner and its weights, they derive a lower bound of the pAUC objective at each iteration, which decouples the weaker learner and its

weights. In these papers, the authors only conduct experiments on small scale datasets with few hundred or thousand examples.

**Heuristic Methods.** Wang and Chang [170] consider the marker (feature) selection problem via maximizing the partial AUC of linear risk scores. They propose a surrogate loss function for pAUC and show its non-asymptotic convergence and greedily select features for learning a linear classifier. There is no discussion on efficiency and complexity of how to solve the pAUC maximization problem. The authors have conducted experiments on some simulated data and real data with only few hundred examples. Ricamato and Tortorella [141] examine the problem of how to combine two or multiple classifiers to maximize partial AUC. The problem is reduced to optimizing a scalar combination weight, which is different from standard pAUC maximization methods for learning a classifier. For combining multiple classifiers, they use a greedy method to select which two classifiers to combine at each iteration. As a result, they derive a boosting algorithm similar to the classical Adaboost algorithm, which first finds the optimal base learner given previous combined learner and then optimizes the weight of the base learner.

**Structural SVM Methods.** Narasimhan and Agarwal [117] propose a structural SVM-based approach for learning a linear model by optimizing partial AUC inspired by Reference [81]. Their formulated optimization problem has an exponential number of constraints, one for each possible ordering of training data. To solve this problem, they use the cutting plane method, which is based on the fact that for any $\epsilon > 0$ a small subset of the constraints is sufficient to find an $\epsilon$-approximate solution to the problem. However, the bottleneck lies at finding the most violated constraint at each iteration, which could cost $O((n_+ + n_-)d + n_+n_- + n_- \log n_-)$ time complexity. In addition, the cutting-plane method could have a slow convergence with an iteration complexity of $O(1/\epsilon)$. In the extended version [119], the authors have managed to reduce the per-iteration time complexity to $O((n_+ + n_-)d + n_+n_-\beta + n_- \log n_-)$, where $\beta \in (0, 1)$ is the upper bound parameter of the FPR. In 2013, the same authors propose a tight surrogate loss for the partial AUC in the structural SVM framework [118]. In this article, the authors also present a projected gradient method, which suffers a per-iteration cost of $O((n_+ + n_-)d + n_- \log n_- + (n_+ + n_-\beta) \log(n_+ + n_-\beta))$ for learning a linear model of dimentionality of $d$, and an iteration complexity of $O(1/\epsilon^2)$. A DC programming approach is also presented in Reference [119] for optimizing pAUC with FPR restricted in a range $(\alpha_0, \alpha_1)$ where $\alpha_0 > 0$, which is computationally more expensive than the structural SVM approach due to requiring to solve an entire structural SVM optimization at each iteration. In these papers, the authors have conducted experiments on multiple datasets with size ranging from a few thousand to a few hundred thousand. The theoretical work cited in Reference [109] provides a statistical performance guarantee for algorithms of maximizing the empirical pAUC proposed in References [117–119].

**Constrained Optimization.** Maximizing the partial AUC can be reformulated as a constrained optimization problem that involves optimizing a non-decomposable evaluation metric with a certain thresholded form, while constraining another metric of interest. In particular, Reference [40] proposes to approximate the area under the ROC curve using a Riemann approximation while dividing the range of FPRs into a number of bins where each threshold is associated with a bin. This approach allows the reformulation of a constrained optimization problem where the objective is to maximize the sum of the TPRs at each threshold with constraints associated with threshold satisfying the FPRs. Replacing TPRs and FPRs with surrogate relaxations, it can be further shown to be equivalent to a Lagrangian (mini-max) problem and then vanilla stochastic gradient descent and ascent algorithms can be applied. The follow-up works [29, 120] have improved this approach using the surrogate relaxations for the primal updates. In Reference [92], the authors further improved this approach by expressing the threshold parameter as a function of the model parameters

Table 3. Comparison of Different Studies for pAUC Maximization

| Work | Category | Objective Functions | Models | Complexity/Convergence Analysis | Size of Data |
|---|---|---|---|---|---|
| [173] | Indirect Methods | SVM-like | Kernel | No | $10^5$ |
| [142] | Indirect Methods | P-norm Push | Linear | No | $10^4$ |
| [2] | Indirect Methods | Infinity Push | Linear | Yes | $10^3$ |
| [139] | Indirect Methods | Infinity Push | Linear | Yes | $10^3$ |
| [98] | Indirect Methods | Infinity Push | Linear | Yes | $10^6$ |
| [90] | Boosting-type | pAUC Surrogate | Cubic Spline | No | $10^3$ |
| [157] | Boosting-type | pAUC Surrogate | Decision Stump | No | $10^3$ |
| [170] | Heuristic Methods | pAUC Surrogate | Linear | No | $10^2$ |
| [141] | Heuristic Methods | pAUC Surrogate | Any | No | $10^5$ |
| [117–119] | Structural SVM | pAUC surrogate | Linear | Yes | $10^6$ |
| [29, 40, 92, 120] | Constrained Opt. | Riemann approximation | Linear/Non-linear | Convex Only | $10^5$ |
| [187] | Stochastic/Deep | Appr. Pairwise Surrogate | Deep Nets | No | $10^5$ |
| [206] | Stochastic/Deep | (10), (11), (23), (24) | Deep Nets | Yes | $10^5$ |
| [190] | Stochastic/Deep | (10) | Deep Nets | Yes | $10^5$ |

via the Implicit Function theorem [158]. The resulting optimization problem can be solved using standard gradient-based methods.

### 4.3 Summary

We compare different methods in Table 3 for pAUC maximization from different perspectives, where we also include deep partial AUC maximization methods reviewed in Section 7. The full-batch-based algorithms could suffer a quadratic time complexity in the worst-case or a super-linear (e.g., log-linear) time complexity per-iteration, which makes them not amenable for handling large-scale datasets. Most of them are for learning traditional models (e.g., linear models, kernel models) and algorithms for solving the underlying optimization problem are not scalable to large-scale datasets and not suitable for deep learning.

## 5 ONLINE AUC MAXIMIZATION - THE SECOND AGE

In contrast to the full-batch methods that need all training data beforehand, online learning algorithms [21] can update the model parameter upon receiving new datum and can efficiently handle streaming data where examples are presented in sequence. Online learning with point-wise loss has been studied extensively [65, 131, 147]. However, online learning for AUC maximization has different challenges due to that the pairwise loss does not naturally fit the streaming data. In the literature, there has been a wave of studies focusing on online learning for AUC maximization. Below, we will categorize them into two classes, namely, **online buffer-based methods, online statistics-based methods**. Revolving around these methods, we will discuss two theoretical properties, i.e., regret bounds and statistical error bounds.

We first provide some background on regret bounds and statistical error bounds. In the standard online learning setting, there is no statistical assumption on the data received, e.g., IID assumption. Hence, the measure of interest is the regret bound. Let $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)\}$ denote the sequence of data received in the stream. To measure the regret, let $L_t(\mathbf{w}_t, \mathbf{x}_t, y_t)$ denote the cost measure of the $t$th model $\mathbf{w}_t$ with respect to the received data $(\mathbf{x}_t, y_t)$ at the $t$th iteration, let $L(\mathbf{w}, \{\mathbf{x}_t, y_t\}_{t=1}^T)$ denote the cost measure defined on all data. Then the regret is defined as

$$R_T = \sum_{t=1}^{T} L_t(\mathbf{w}_t, \mathbf{x}_t, y_t) - \min_{\mathbf{w}} \sum_{t=1}^{T} L_t(\mathbf{w}, \mathbf{x}_t, y_t).$$

There are different ways to define the cost at each iteration for AUC maximization, which will be discussed in the following:

When the received data is assumed to follow the IID assumption, the statistical error bound is another performance guarantee of interest. There are two types of statistical error bounds, namely, generalization error bounds and excess risk bounds, where the former refers to the bounds of the difference between the expected risk of a learned model and the empirical risk, and the latter refers to the bounds of the difference between the expected loss of a learned model and the optimal expected loss. In particular, let $\mathcal{R}(\mathbf{w})$ denote the expected risk for AUC maximization, which is given by $\mathcal{R}(\mathbf{w}) = \mathbb{E}_{\mathbf{x}_+\sim\mathcal{P}_+,\mathbf{x}_-\sim\mathcal{P}_-}[\ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i))]$. Then the generalization error bounds usually take the form of $\mathcal{R}(\widehat{\mathbf{w}}_T) \leq \frac{1}{T}\sum_{t=1}^{T} L_t(\mathbf{w}_t, \mathbf{x}_t, y_t) + O(T^{\alpha})$ for some $\widehat{\mathbf{w}}_T$ and $\alpha > 0$, and the excess risk bounds usually take the form of $\mathcal{R}(\widehat{\mathbf{w}}_T) \leq \min_{\mathbf{w}\in\mathcal{W}} \mathcal{R}(\mathbf{w}) + O(T^{-\alpha})$ for some $\widehat{\mathbf{w}}_T$ and $\alpha$.

### 5.1 Online Buffered Gradient Descent for AUC Maximization

The most representative online buffer-based methods is the online buffer gradient descent method proposed in the seminal paper cited in Reference [201] in 2011 by Zhao, Hoi, Jin, and Yang. It is the first work that studies online AUC maximization and inspires many following studies. They propose online buffered gradient descent methods, whose algorithmic framework is shown in Algorithm 1. There are two key functions, i.e., UpdateBuffer and UpdateModel. In the paper, the authors define the following cost function for each iteration:

$$L_t(\mathbf{w}, \mathbf{x}_t, y_t) = \sum_{i<t, y_i=1} \mathbb{I}(y_t = -1)\ell(f_{\mathbf{w}}(\mathbf{x}_t) - f_{\mathbf{w}}(\mathbf{x}_i)) + \sum_{i<t, y_i=-1} \mathbb{I}(y_t = 1)\ell(f_{\mathbf{w}}(\mathbf{x}_i) - f_{\mathbf{w}}(\mathbf{x}_t)).$$

They update the buffer by using the "reservoir sampling" technique [161], which aims to simulate a uniform sampling of the received examples. They update the model parameter based on the gradient descent of the cost function $L_t$ by only using examples in the buffer, i.e., $(\mathbf{x}_i, y_i) \in \mathcal{B}_t$. They establish a regret bound in the order of $\sqrt{B_+T_+^3 + B_-T_-^3}$, where $B_+$ and $B_-$ denote the buffer size for positive samples and negative samples, respectively, $T_+$ and $T_-$ denote the number of received positive examples and negative examples over $T$ iterations, respectively. The authors provide an explanation regarding the optimal buffer size in the presence of the variance terms that have been ignored in the regret bound analysis, which gives an optimal buffer size $B_+ = \sqrt{T_+}$ and $B_- = \sqrt{T_-}$.

Later, the statistical error bounds of online buffer-based methods are established in References [86, 168]. Wang et al. [168] provide the generalization error bounds for any arbitrary online learner with an infinite buffer size and a finite buffer size for learning from $n$ examples. They use the covering number to bound the complexity of hypothesis and derive a generalization error bound of a tailed-averaged solution in the order of $O(\log(\mathcal{N}(\epsilon)n/\delta)/\sqrt{\min(T, B)})$ with a high probability $1-\delta$, where $\mathcal{N}(\epsilon)$ denotes the cardinality of $\epsilon$-net of the hypothesis space for a small value $\epsilon$, and $B$ denotes the buffer size. Kar et al. [86] improve the generalization error bound of the method with an infinite buffer by using the Rademacher complexity of the hypothesis space. Their error bound of the averaged solution is in the order of $O(C_d/\sqrt{T})$, where $C_d$ is a constant in the Rademacher complexity that is dependent only on the dimension $d$ of the input space. Based on the generalization error bound and the regret bound, they also establish an excess risk bound in the order of $R_T/T + O(\frac{C_d+\log(T/\delta)}{T})$, where $R_T$ is the regret bound and $\delta \in (0, 1)$ is the failure probability. In addition, they provide generalization error bounds and a high probability excess risk bound for online buffered gradient descent method with a finite-sized buffer, which has a dominating term of $O(C_d \log(T/\delta)/\sqrt{B})$, where $B$ is the buffer size.

Kar et al. [85] also study an online buffer-based method with an infinite buffer size for partial AUC maximization. In the paper, they define a different cost function for each iteration. Let $L(\mathbf{w}, \{\mathbf{x}_i, y_i\}_{i=1}^t)$ denote the pairwise loss summed over all pairs received in the first $t$ iterations. The cost function at the $t$th iteration is defined as $L_t(\mathbf{w}; \mathbf{x}_t, y_t) = L(\mathbf{w}, \{\mathbf{x}_i, y_i\}_{i=1}^t) - L(\mathbf{w}, \{\mathbf{x}_i, y_i\}_{i=1}^{t-1})$. In this way, the optimal model in hindsight $\min_{\mathbf{w}} L_t(\mathbf{w}; \mathbf{x}_t, y_t)$ indeed optimizes the objective of

**ALGORITHM 1:** Online Buffered Gradient Descent Method for AUC Maximization

Initialization: $\mathbf{w}_0 \in \mathbb{R}^d$, $\mathcal{B}_0 = \emptyset$,
**for** $t = 1, 2, \ldots, T$ **do**
  Receive a data $\{\mathbf{x}_t, y_t\}$
  Update the Buffer $\mathcal{B}_t = \text{UpdateBuffer}(\mathcal{B}_{t-1}, \mathbf{x}_t, y_t)$
  Update the model parameter $\mathbf{w}_t = \text{UpdateModel}(L_t, \mathbf{w}_{t-1}, \mathcal{B}_t, \mathbf{x}_t, y_t)$
**end for**

interest (e.g., pairwise-loss-based objective for AUC maximization). They employ the **Follow-the-Regularized-Leader (FTRL)** algorithm for updating the model and establish a regret bound in the order of $\sqrt{T_+T_-^2+T_-T_+^2}$. They also establish an excess risk bound for a modified FTRL method that uses $s$ samples per-iteration, which is in the order of $O(1/T^{1/4})$ for $s = \sqrt{T}$.

### 5.2 Online Statistics-based Methods

To address the issue of maintaining a large buffer size, Gao et al. [47] propose an online AUC maximization by leveraging the property of pairwise square loss for learning a linear model. They use the same definition of the cost function $L_t(\mathbf{w}; \mathbf{x}_t, y_t)$ as Reference [201]. By using the square loss for learning a linear model, they show that the gradient of the cost function $L_t$ can be computed based on first-order moments (mean vectors of positive and negative examples) and second-order moments (covariance matrices of positive and negative examples) of the received data before the $t$th iteration. Nevertheless, it also introduces high memory costs for maintaining the covariance matrix. To address this issue, the authors develop low-rank approximation methods and only update low-rank matrices for the second-order moments at each iteration. In the paper, the authors also establish the regret bounds for both full-rank and the low-rank approximation methods.

### 5.3 Online Non-linear Methods for AUC Maximization

Online nonlinear kernel methods based on AUC maximization have been proposed and studied in References [35, 70, 156] to address the non-separability of the data and the scalability issues. In particular, Ding et al. [35] extend the online buffered gradient descent method to learn non-linear kernel-based models. They employ two functional approximation strategies, i.e., **random fourier features (RFF)** [137] to approximate the shift invariant kernels and the Nyström method [172] to approximate the kernel matrix. For the two methods, the authors have established regret bounds in the order of $\sqrt{T}$. Nevertheless, it is claimed that the RFF-based method require $m = T$ random features for achieving a high probability bound.

Hu et al. [70] propose a different kernelized online AUC maximization method. They do not use RFF or the Nyström method to approximately compute the kernel similarities. Instead, they use the pairwise hinge loss or squared hinge loss as the surrogate loss and maintain support vectors of positive and negative classes in the online fashion, i.e., those examples whose contribution weights in the classifier are non-zero. They maintain and update two buffers for storing these support vectors and their contribution weights. The cost function at each iteration is defined similarly as in Reference [201] except that $k$-nearest examples to the received data in the buffer are used to compute the loss. They establish a regret bound in the order of $\sqrt{T}$. They also present an extension to the multiple kernel learning framework that can automatically determine a good kernel representation.

Szörényi et al. [156] propose a kNN-based online AUC maximization method by suggesting an algorithmic solution based on the kNN-estimate of the conditional probability function. They use an infinite buffer that stores all received examples.

Table 4. Comparison of Different Studies for Online AUC Maximization

| Work | Category | Loss Functions | Models | Regret/Generalization Analysis | Memory Costs |
|---|---|---|---|---|---|
| [201] | Buffer-based | Pairwise AUC Surrogate | Linear | Yes | $O(Bd)$ |
| [168] | Buffer-based | Pairwise AUC Surrogate | Linear | Yes | $O(Td)$ |
| [86] | Buffer-based | Pairwise AUC Surrogate | Linear | Yes | $O(Td)$ or $O(Bd)$ |
| [85] | Buffer-based | Pairwise pAUC Surrogate | Linear | Yes | $O(Td)$ |
| [35] | Buffer-based | Pairwise AUC Surrogate | Kernel | Yes | $O(Bd)$ |
| [70] | Buffer-based | Pairwise (squared) hinge loss | Kernel | Yes | $O(Bd)$ |
| [25] | Buffer-based | Pairwise Ramp loss | Linear | Yes | $O(Bd)$ |
| [156] | Buffer-based | NA | Non-Parametric | Yes | $O(Td)$ |
| [47] | Statistics-based | Pairwise square loss | Linear | Yes | $O(d^2)$ or $O(dr)$ |
| [36] | Statistics-based | Pairwise square loss | Linear | Yes | $O(d^2)$ |
| [103] | Statistics-based | Pairwise square loss | Linear | Yes | $O(d^2)$ |

Where $T$ is the Total Number of Iterations, $B$ is a Fixed Buffer Size, $r$ is a Constant Size of the Low Rank Approximation, $d$ is the Dimensionality of Input Data.

### 5.4 Adaptive Online AUC Maximization

References [25, 36, 103] propose and study adaptive online AUC maximization algorithms belonging to the two classes of online AUC maximization methods. Ding et al. [36] extend the online statistics-based method proposed in Reference [47] for AUC maximization by incorporating an online **adaptive gradient method (AdaGrad)** [39] for exploiting the knowledge of historical gradients. Cheng et al. [25] propose to use the Adam-style update in the framework of online buffered gradient descent methods, where the buffer is maintained in first-in-first-out fashion. In the paper, they use a non-convex ramp loss as the surrogate function of the indicator $\mathbb{I}(f_{\mathbf{w}}(\mathbf{x}_-) - f_{\mathbf{w}}(\mathbf{x}_+))$ and use **concave-convex procedure (CCCP)** to approximate the cost function at each iteration. Liu et al. [103] leverage the Adam-style update [88] in the framework of online statistics-based method for AUC maximization.

### 5.5 Summary

Two classes of methods, namely, online buffer-based methods and online statistics-based methods, have been proposed for online AUC maximization. Online buffer-based methods are more generic, which can be leveraged for learning both linear and non-linear classifiers for any possible pairwise surrogate losses, while online statistics-based methods are restricted to learning linear models and using pairwise square loss. Nevertheless, online buffer-based methods usually require a large buffer to achieve a good performance, and online statistics-based methods could enjoy a lower regret and a lower memory costs for low-dimensional data. We compare different works in Table 4 from different perspectives.

## 6 STOCHASTIC AUC MAXIMIZATION - THE THIRD AGE

Stochastic AUC maximization refers to a family of methods that only process one or a small mini-batch of examples at each iteration for updating the model parameters, which are amenable for handling big data. The difference from online AUC maximization is that the IID assumption of data is typically assumed in stochastic AUC maximization. In this section, we provide a review on works for learning linear and kernel-based models for AUC maximization, and present a review for deep AUC maximization in Section 7. We categorize the existing stochastic methods for AUC maximization into two classes, i.e., stochastic batch-based pairwise methods, stochastic primal-dual methods. The existing works consider two learning settings: online setting similar to stochastic approximation in conventional literature [148] and offline setting similar to stochastic average approximation in conventional literature [125]. In the online setting, the data $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t, \ldots\}$ is assumed to be i.i.d. from an unknown distribution and continuously arriving, i.e., streaming

data, and the goal is to minimize the expected loss in Equation (2). In the offline setting, a set of training data $\mathcal{S} = \{(\mathbf{x}_i, y_i), i \in [n]\}$ of size $n$ is given beforehand, and the goal is to minimize the empirical loss in Equation (9). There are two different errors that have been analyzed for different algorithms, namely, optimization error and statistical error. For optimizing the expected loss (2), the optimization error and the statistical error coincide.

### 6.1 Stochastic Batch-based Pairwise Methods

The idea of batch-based pairwise methods is to use a mini-batch of data points for computing a stochastic gradient estimator for updating the model parameter. Below, we discuss two categories of methods for the offline setting and the online setting, respectively.

**Offline setting.** A straightforward approach for designing stochastic AUC maximization algorithms is by using stochastic gradients of the pairwise loss function $\ell(\mathbf{w}; \mathbf{z}, \mathbf{z}') = \ell(f_{\mathbf{w}}(\mathbf{z}') - f_{\mathbf{w}}(\mathbf{z}))$ for the sampled positive-negative pairs $(\mathbf{z}, \mathbf{z}')$. Then the model parameter can be updated by any suitable stochastic algorithms, e.g., SGD. This approach has been adopted and studied in several papers with different aims [33, 53, 96, 184].

Gu et al. [53] focus on establishing statistical error in the order of $O(1/\sqrt{n})$ of a stochastic algorithm based on a finite training data set of size $n$. They propose a **doubly stochastic gradient algorithm (AdaDSG)** by solving regularized pairwise learning problems. Specifically, at each stage, AdaDSG uses an inner solver to solve a sampled sub-problem and then uses the solution obtained from this sub-problem as a warm start for the next larger problem with a doubled size of training samples. The inner solver simply uses the SGD method based on a randomly sampled positive-negative pair for updating the model parameter. Reference [33] proposes a triply stochastic functional gradient for AUC maximization problem for learning a kernelized model. At each iteration, this algorithm performs SGD update based on an unbiased functional gradient calculated from a random pair of examples using random Fourier features (the pair of examples and the random variable for constructing the Fourier features constitute the triplet). A convergence rate in the order $O(\frac{1}{T})$ for the optimization error was established for the strongly regularized empirical AUC maximization problem. Reference [149] also considers a similar algorithm for semi-supervised ordinal regression based on AUC optimization.

Recent works cited in References [95, 96] focus on establishing the statistical error for a specific type of SGD algorithms for pairwise learning in the offline setting. In particular, they study the following SGD-type algorithm for pairwise learning: At the each iteration, it randomly draws $(\mathbf{z}_{i_t}, \mathbf{z}_{j_t})$ from all possible $n(n-1)/2$ pairs of examples, and the model parameter is updated by $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla \ell(\mathbf{w}_t; \mathbf{z}_{i_t}, \mathbf{z}_{j_t})$. Reference [95] uses the the concept of uniform stability [15] and the corresponding high probability generalization bounds [16, 44] to derive the excess risk bound $O(\log n/\sqrt{n})$ in the convex case. Reference [96] further provides improved results by incorporating the variance information and show that, under an interpolation or a low noise assumption, the risk bounds can achieve $O(1/n)$ through exploiting the smoothness assumption.

Yang et al. [184] propose a simple SGD-type algorithm for pairwise learning where, at the each iteration, it randomly draws $i_t \in [n]$ and the current example $\mathbf{z}_{i_t}$ is paired with previous one $\mathbf{z}_{i_{t-1}}$, and the model parameter is updated by the SGD based on the pair $(\mathbf{z}_{i_t}, \mathbf{z}_{i_{t-1}})$, i.e., $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla \ell(\mathbf{w}_t; \mathbf{z}_{i_t}, \mathbf{z}_{i_{t-1}})$. The authors have established excess risk bounds $O(\frac{1}{\sqrt{n}})$ for smooth and non-smooth convex losses and smooth non-convex losses under **Polyak-Lojasiewicz (PL)** condition, in different orders with different number of iterations.

**Online setting.** The online setting is more challenging due to that each iteration only receives or samples one data point. The challenge is that an unbiased stochastic gradient cannot be

computed based on one data point. To address the challenge, the received data will be stored in a buffer and will be used for computing a stochastic gradient, which is similar to online AUC optimization [86, 168, 169]. References [11, 58, 193] have considered SGD for pairwise learning in the stochastic setting with an infinite buffer. In particular, for such SGD-based pairwise learning algorithms, at each iteration, the current example $\mathbf{z}_t$ is paired with previous ones $\{\mathbf{z}_1, \ldots, \mathbf{z}_{t-1}\}$, the model parameter is updated by gradient descent based on the gradient of $L_t(\mathbf{w}_t; \mathbf{z}_t) = \frac{1}{t-1}\sum_{j=1}^{t-1} \ell(\mathbf{w}_t; \mathbf{z}_t, \mathbf{z}_j)$, i.e., $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla L_t(\mathbf{w}_t; \mathbf{z}_t)$ where $\eta_t$ is a step size. The authors of References [58, 193] prove the convergence of such SGD-based algorithm for learning a kernelized model with a convergence rate of $O(\frac{1}{t})$ for a strongly convex objective function and $O(\frac{1}{\sqrt{t}})$ for using the pairwise square loss without explicit regularization term. For learning a linear model, Boissier et al. [11] prove that a fast convergence rate $O(\frac{1}{t})$ is still possible for using the pairwise square loss. However, it is notable that the algorithms in References [11, 58, 193] are not scalable to large-scale datasets, since the buffer size increases as the number of sampled data. This issue can be addressed by the stochastic primal-dual methods discussed shortly.

### 6.2 Stochastic Primal-Dual (PD) Methods

The idea of stochastic primal-dual methods is to directly apply stochastic methods for addressing the min-max formulations of AUC maximization, e.g., Equation (12). The benefit of the min-max formulations is that the minimax objective is simply the average of individual data, which makes it suitable to the online setting. Nevertheless, the algorithms discussed below can be applied to both online and offline settings. It is notable that most of the algorithms discussed in this subsection are developed for solving the min-max formulation for the pairwise square loss. However, many of them can be easily extended for solving the min-max margin loss (16).

Ying et al. [192] are the first to propose the idea of solving a minimax objective (i.e., Equation (12)) by a stochastic algorithm for AUC maximization with a pairwise square loss. The authors propose to use the stochastic first-order primal-dual algorithm [124] for AUC maximization, which is referred to as SOLAM. The algorithm uses a stochastic gradient descent for updating the primal variables $\mathbf{w}, a, b$ and uses a stochastic gradient ascent for updating the dual variable $\alpha$. It enjoys a convergence rate $O(\frac{1}{\sqrt{T}})$ with a per-iteration complexity $O(d)$ for learning a linear model of dimensionality of $d$.

In the subsequent work cited in References [97, 121], the authors leverage the special formulation of the minimax objective for AUC maximization with a square loss to derive faster algorithms for learning a linear model, i.e., $f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$. In particular, Natole et al. [121] derive a closed form solution for $a, b, \alpha$ given $\mathbf{w}$ for Equation (12), i.e., $a(\mathbf{w}) = \mathbf{w}^\top \mathbb{E}[\mathbf{x}|y = 1]$, $b(\mathbf{w}) = \mathbf{w}^\top \mathbb{E}[\mathbf{x}|y = -1]$ and $\alpha(\mathbf{w}) = b(\mathbf{w}) + c - a(\mathbf{w})$.[1] Given that data statistics $\mathbb{E}[\mathbf{x}|y = 1]$, $\mathbb{E}[\mathbf{x}|y = -1]$ and the probability $p = \Pr(y = 1)$ can easily be estimated from training data, the authors propose a stochastic proximal gradient descent algorithm that only updates $\mathbf{w}$ while the auxiliary variables $a, b$, and $\alpha$ are subsequently computed from $\mathbf{w}$ using the updated data statistics. A fast convergence rate $O(\frac{1}{T})$ is proved for AUC maximization by leveraging the strong convexity of the regularization term, e.g., $\ell_2$ norm square regularization. Lei and Ying [97] give an alternative but self-contained proof for stochastic saddle point formulation in References [121, 192] by writing the objective function in Equation (12) with $c = 1$ as

$$
\begin{aligned}
&(1 - \mathbf{w}^\top(x - x'))^2 = ([1 + \alpha(\mathbf{w})] + [\mathbf{w}^\top x' - b(\mathbf{w})] - [\mathbf{w}^\top x - a(\mathbf{w})])^2 \\
&= ([1 + \mathbf{w}^\top(\mathbb{E}[\tilde{x}|\tilde{y} = -1] - \mathbb{E}[\tilde{x}|\tilde{y} = 1])] + [\mathbf{w}^\top(x' - \mathbb{E}[\tilde{x}|\tilde{y} = -1]) - \mathbf{w}^\top(x - \mathbb{E}[\tilde{x}|\tilde{y} = 1])])^2. \quad (19)
\end{aligned}
$$

[1] In their papers, $\alpha(\mathbf{w})$ is given by $b(\mathbf{w}) - a(\mathbf{w})$ due to a variable change.

Based on this important observation, they prove that AUC maximization (9) is equivalent to $\min_{\mathbf{w}} \mathbb{E}_{\mathbf{z}}[\widetilde{F}(\mathbf{w};\mathbf{z})] = f(\mathbf{w})$, where

$$\widetilde{F}(\mathbf{w};\mathbf{z}) = p(1-p) + (1-p)(\mathbf{w}^\top(x - \mathbb{E}[\mathbf{x}'|y'=1]))^2\mathbb{I}(y=1) + p(\mathbf{w}^\top(\mathbf{x} - \mathbb{E}[\mathbf{x}'|y'=-1]))^2\mathbf{I}_{[y=-1]}$$
$$+ 2p(1-p)\mathbf{w}^\top(\mathbb{E}[\mathbf{x}'|y'=-1] - \mathbb{E}[\mathbf{x}'|y'=1]) + p(1-p)(\mathbf{w}^\top(\mathbb{E}[\mathbf{x}'|y'=-1] - \mathbb{E}[\mathbf{x}'|y'=1]))^2.$$

From this key observation, they propose a **stochastic proximal stochastic gradient (SPAM)** that also only needs to update $\mathbf{w}$. In particular, the authors prove that, in either the unconstrained case without explicit regularizer or with a strong convex regularizer, SPAM can achieve a fast convergence rate $O(\frac{1}{T})$ with a linear per-iteration cost $O(d)$.

There are further studies trying to improve the convergence rate for solving the minimax objective (12) without assuming the strong convexity of the regularizer. Liu et al. [102] propose an improved stochastic algorithm for solving the minimax objective of AUC maximization. The idea is to leverage the strong concavity in terms of the dual variable $\alpha$ and a proved error bound condition of the primal objective function in terms of $\mathbf{w}, a, b$. Their algorithm needs to know the total number of iterations $T$ beforehand and divides the update into multiple stages according to $T$, and each stage calls a stochastic primal-dual method with a constant step size. After each stage, the step size is decreased by a constant factor. Their algorithm enjoys a convergence rate of $O(1/T)$ for $T$ iterations with one example per-iteration. Later on, Yan et al. [180] consider a more general minimax objective under an error bound condition of the primal objective and develop a stagewise stochastic algorithm without knowing the total number of iterations $T$ in advance. Their algorithm also enjoys a convergence rate of $O(1/T)$.

Stochastic algorithms with linear convergence for AUC maximization by using more advanced techniques, e.g., variance-reduction, have been considered in several later works, e.g., References [32, 123, 189]. Natole Jr et al. [123] propose a minibatch **stochastic primal-dual algorithm (SPDAM)** with a linear convergence rate. This algorithm is adapted from the mini-batch stochastic primal-dual coordinate method in Reference [200] to the problem of AUC maximization with the pairwise square loss and a strongly convex regularizer. The authors prove its linear convergence rate $e^{-c(n,m,\lambda)T}$ where $c(n,m,\lambda)$ depends on the size $m$ of the minibatch set, the size $n$ of training data, and the strong convexity parameter $\lambda$. Reference [32] further extends SPAM [121] by using the variance-reduction technique [83]. It enjoys a linear convergence rate $e^{-c(M,\beta,\eta)T}$ where $\beta$ is the strongly-convex parameter, $M$ is the strongly smooth parameter, and $\eta$ is the constant step size.

In References [189, 203], the authors develop efficient sparse AUC maximization algorithms with the pairwise square loss for analyzing the high dimensional data. Both studies use the minimax objective (e.g., Equation (12)) and the explicit solutions for the auxiliary variables $a, b$, and $\alpha$ as observed in References [97, 121]. In particular, Reference [189] uses the hard thresholding algorithms for AUC maximizatin and prove its linear convergence under the assumption of **restricted strong convexity (RSC)** and **restricted strong smoothness (RSS)** on the objective function. Reference [203] considers the application of AUC maximization for handling sparse high-dimensional datasets in the sense that the number of nonzero features $k$ in each example is far less than the total number of features $d$. Such datasets are abundant in online spam filtering [145], ad click prediction [112], and identifying malicious URLs [108]. They develop a generalized Follow-The-Regularized-Leader framework [111] for AUC maximization with a lazy update that only involves a per-iteration cost $O(k)$.

Recently, Reference [185] also proposes stochastic primal-dual algorithm for solving AUC maximization with a general convex pairwise loss. They propose to use Bernstein polynomials [135] to uniformly approximate a general loss. This reduction for AUC maximization with a general convex pairwise loss is equivalent to a weakly convex min-max problem (for learning a linear model).

Table 5. Comparison of Different Studies for Stochastic AUC Maximization Algorithms

| Work | Category | Objective | Model | Guarantee | Rate | Memory Cost |
|---|---|---|---|---|---|---|
| [58] | Stochastic BP | Pairwise hinge loss | kernel | Opt. Error | $O(1/T)$ | $O(T^2)$ |
| [193] | Stochastic BP | Pairwise square loss | kernel | Opt. Error | $O(1/T^{1/3})$ | $O(T^2)$ |
| [11] | Stochastic BP | Pairwise square loss | linear | Opt. Error | $O(1/T)$ | $O(d^2)$ |
| [33] | Stochastic BP | Pairwise loss | linear | Opt. Error | $O(1/T)$ | $O(d)$ |
| [53] | Stochastic BP | Pairwise loss | linear | Stat. Error | $O(1/\sqrt{n})$ | $O(Dd)$ |
| [96] | Stochastic BP | Pairwise loss | linear | Stat. Error | $O(1/\sqrt{n})$ | $O(d)$ |
| [184] | Stochastic BP | Pairwise loss | linear | Stat. Error | $O(1/\sqrt{n})$ | $O(d)$ |
| [192] | Stochastic PD | Minimax | linear | Opt. Error | $O(1/\sqrt{T})$ | $O(d)$ |
| [102] | Stochastic PD | Minimax (square loss) | linear | Opt. Error | $O(1/T)$ | $O(d)$ |
| [180] | Stochastic PD | Minimax | linear | Opt. Error | $O(1/T)$ | $O(d)$ |
| [121] | Stochastic PD | Minimax (square loss) | linear | Opt. Error | $O(1/T)$ | $O(d)$ |
| [97] | Stochastic PD | Minimax (square loss) | linear | Opt. Error | $O(1/T)$ | $O(d)$ |
| [123] | Stochastic PD | Minimax (square loss) | linear | Opt. Error | $O(1/T)$ | $O(d)$ |
| [189] | Stochastic PD | Minimax (square loss) | linear | Opt. Error | $e^{-c(\rho_k^+(B),\rho_k^-)T}$ | $O(Bd)$ |
| [32] | Stochastic PD | Minimax (square loss) | linear | Opt. Error | $e^{-c(\beta,\eta,M)T}$ | $O(d)$ |
| [185] | Stochastic PD | Minimax (general loss) | linear | Opt. Error | $O(1/\sqrt{m})$ | $O(md)$ |

Where $T$ is the Total Number of Iterations, $d$ denotes the Dimensionality of the Input Data, $B$ is the Batch Size used in Reference [189] on which the Parameter $\rho_k^+(B)$ is Dependent, $m$ is the Degree of Bernstein Polynomials to Approximate the General Convex Loss used in Reference [185], $\rho_k^+(B)$ and $\rho_k^-$ are RSC and RSS Parameters used in Reference [189], and $\beta$, $M$ and $\eta$, Respectively, Denote the Strong-convexity Parameter, the Smooth Parameter and the Constant Step Size in Reference [32]. Opt. is short for optimization and Stat. is short for statistical.

Then, the authors apply the stochastic proximal point-based method [136] for AUC maximization that has a per-iteration cost $O(md)$, where $m$ is the degree of Bernstein polynomials used to approximate the original convex surrogate loss. Despite its non-convexity, they have proved its global convergence by exploring the appealing convexity-preserving property [135] of Bernstein polynomials and the intrinsic structure of the min-max formulation. However, the final convergence in terms of the original objective function is of a slow rate $O(\frac{1}{\sqrt{m}})$.

### 6.3 Summary

Two main classes of methods have been proposed for stochastic AUC maximization: stochastic **batch-pairwise (BP)** methods and stochastic **primal-dual (PD)** methods. Stochastic batch-pairwise methods are generic, which depend on the strategy of pairing examples, while the stochastic PD methods explore the special problem structure that facilitates the design of fast stochastic optimization algorithms. We have compared different works in Table 5 from different perspectives.

## 7 DEEP AUC MAXIMIZATION (DAM): THE FOURTH AGE

Recently, there is a surge of interest in AUC maximization for learning deep neural networks, i.e., **deep AUC maximization (DAM)**. This problem has received much attention from the algorithmic perspective for solving the minimax objective of AUC maximization due to its advantage over the pairwise-loss-based objective for big data. Then, it is employed for solving real-world classification problems (e.g., medical image classification) and achieves great success [196]. Below, we will survey related works from algorithmic and practical perspectives. We would like to point out that all algorithms surveyed below are also stochastic algorithms. However, the differences from works in the third age in that (i) algorithms presented below are applicable to any deep neural networks; in contrast, many algorithms in the third age are developed for learning linear models by leveraging the special structure of the objective.; (ii) deep AUC maximization has faced some

**ALGORITHM 2:** A Unified Framework for solving $\min_{\mathbf{w}\in\mathbb{R}^d} \max_{\alpha\in\Omega} F(\mathbf{w},\alpha)$

Initialization: $\mathbf{w}_0 \in \mathbb{R}^d, \alpha_0 \in \Omega, \gamma, T_1, \eta_1$,
**for** $k = 1, 2, \ldots, K$ **do**
  Set $\mathbf{w}_0^k = \mathbf{w}_{k-1}$, $\alpha_0^k = \Lambda(\mathbf{w}_{k-1}, \alpha_{k-1})$ ◇$\Lambda$ is a certain function
  Construct $F_k(\mathbf{w},\alpha) = F(\mathbf{w},\alpha) + \frac{\gamma}{2}\|\mathbf{w} - \mathbf{w}_0^k\|^2$
  Solve $(\mathbf{w}_k, \alpha_k) = \mathcal{A}(F_k, \mathbf{w}_0^k, \alpha_0^k, \eta_k, T_k)$, ◇$\mathcal{A}$ is a stochastic algorithm
  Decrease $\eta_k$ appropriately, increase $T_k$ accordingly
**end for**
Return $\mathbf{w}_K$.

unique challenges, e.g., feature learning, regularization and normalization, which will be discussed in Section 8.

### 7.1 Non-convex Concave Min-Max Optimization

For deep learning, the prediction function $f_{\mathbf{w}}(\mathbf{x})$ is a non-linear function of the model parameter $\mathbf{w}$, which makes the objective in Equation (9) and the minimax objective in Equations (12) and (16) non-convex. Although standard stochastic methods (e.g., SGD, Adam) can be directly applied for solving the pairwise-loss-based objective in Equation (9) with provable convergence to a stationary point, these methods are not directly applicable to the minimax objective in Equation (12), which is more suitable for online learning and distributed optimization. The minimax objective in Equations (12) and (16) is a non-convex strongly concave problem. Below, we will focus on stochastic methods for solving non-convex min-max problems, and we categorize different stochastic methods into two classes, i.e., two-loop proximal point-based methods and single-loop stochastic primal-dual methods. Without loss of generality, we consider the following min-max optimization problem for discussion:

$$\min_{\mathbf{w}} F(\mathbf{w}) := \left\{\max_{\alpha\in\Omega} F(\mathbf{w},\alpha) = \mathbb{E}_{\mathbf{z}}[F(\mathbf{w},\alpha;\mathbf{z})]\right\}. \tag{20}$$

**Proximal Point-based Methods.** The proximal point-based methods follow a common framework as shown in Algorithm 2. This general framework has several unique features: (i) the algorithm is run in multiple stages $k = 1, \ldots, K$; (ii) at each stage a quadratic regularized function $F_k(\mathbf{w},\alpha)$ is constructed by adding a quadratic function $\frac{\gamma}{2}\|\mathbf{w}-\mathbf{w}_0^k\|^2$, where $\gamma > 0$ is a proper hyper-parameter; (iii) a proper stochastic algorithm $\mathcal{A}$ is employed for solving the regularized function with a step size $\eta_k$ and a number of iterations specified by $T_k$, whose output denoted by $\mathbf{w}_k, \alpha_k$ that are usually the last or the averaged solutions across all iterations in this stage; (iv) the step size $\eta_k$ and the number of iterations $T_k$ are changed appropriately for next stage. The following different methods differ in how to change $\eta_k, T_k$ and how to implement the function $\Lambda$ for computing $\alpha_0^k$:

Rafique et al. [136] are the first to study non-convex concave min-max optimization problems and to establish the convergence rate. In particular, they assume the objective function $F(\mathbf{w},\alpha)$ is weakly convex in terms of the primal variable $\mathbf{w}$ and is (strongly) concave in terms of the dual variable $\alpha$. A function is called weakly convex if it becomes a convex function by adding a quadratic function in term of the decision variable with a proper scaling factor. This is the motivation of adding $\frac{\gamma}{2}\|\mathbf{w}-\mathbf{w}_0^k\|^2$ to the objective at each stage, which can make the objective convex or strongly convex with an appropriate $\gamma > 0$. Since the objective function is non-convex and not necessarily smooth, they consider a convergence measure for weakly convex function, i.e., nearly stationary solution [34]. An $\epsilon$-level nearly stationary solution to a problem $\min_{\mathbf{w}} F(\mathbf{w})$ is defined as a point $\mathbf{w}$ such that there exists a point $\widehat{\mathbf{w}}$ satisfying $\|\mathbf{w}-\widehat{\mathbf{w}}\| \leq O(\epsilon)$ and $\text{Dist}(0, \partial F(\widehat{\mathbf{w}})) \leq \epsilon$, where $\text{Dist}(\cdot,\cdot)$

denotes the Euclidean distance from a point to a set. In this work, the authors consider both the online setting and the offline (a.k.a. finite-sum) setting for the objective function $F(\mathbf{w}, \alpha)$. For the online setting, they employ **stochastic mirror descent (SMD)** method for implementing $\mathcal{A}$. The parameters are set as $\eta_k \propto 1/\sqrt{k}, T_k \propto k^2$ when the objective is only concave in terms of the dual variable $\alpha$, and are set as $\eta_k \propto 1/k, T_k \propto k$ when the objective is strongly concave in terms of the dual variable. When the objective is weakly convex and concave, the sample complexity is in the order of $O(1/\epsilon^6)$ for finding an $\epsilon$-level nearly stationary solution to $F(\mathbf{w}) = \max_{\alpha\in\Omega} F(\mathbf{w}, \alpha)$, and when the objective is strongly concave, they improve the sample complexity to $O(1/\epsilon^4 + C/\epsilon^2)$ by considering a special class such that $\alpha_* = \arg\max_{\alpha\in\Omega} F(\mathbf{w}, \alpha)$ can be computed, where $C$ denotes the complexity for computing $\alpha_*$ given $\mathbf{w}$. To enjoy this improved complexity, they compute $\alpha_0^k$ by solving $\max_{\alpha\in\Omega} F(\mathbf{w}_0^k, \alpha)$ to the optimal solution for $\alpha$. For the finite-sum setting with $n$ components for the function $F(\mathbf{w}, \alpha)$, they improve the complexity to $O(n/\epsilon^2)$ when the objective is strongly concave in terms of $\alpha$.

Yan et al. [179] further improve the algorithm and complexity for solving weakly convex and strongly concave min-max problems. They do not assume certain structure of the objective function or the optimal dual variable can be easily computed given $\mathbf{w}$. Their algorithm is similar to the first algorithm proposed in Reference [136], i.e., the initial solution $\alpha_0^k$ is simply the averaged solution from last stage of running $\mathcal{A}$, i.e., $\Lambda(\mathbf{w}_{k-1}, \alpha_{k-1}) = \alpha_{k-1}$. They develop a novel analysis to prove the algorithm enjoys a sample complexity of $O(1/\epsilon^4)$ for finding an $\epsilon$-level nearly stationary solution to $F(\mathbf{w})$. The key challenge lies at tackling error $\|\alpha_0^k - \alpha(\mathbf{w}_k)\|^2$ in the upper bound for solving $\min_{\mathbf{w}} \max_{\alpha} F_k(\mathbf{w}, \alpha)$, where $\alpha(\mathbf{w}_k) = \arg\max_{\alpha\in\Omega} F(\mathbf{w}, \alpha)$. In Reference [136], the authors compute $\alpha_0^k = \alpha(\mathbf{w}_0^k)$, which reduces the dual error $\|\alpha_0^k - \alpha(\mathbf{w}_k)\|^2$ to $\|\mathbf{w}_{k-1} - \mathbf{w}_k\|^2$ due to the Lipchitz continuity of $\alpha(\mathbf{w})$, which is decreasing to zero. In contrast, the authors of Reference [179] avoid computing $\alpha_0^k = \alpha(\mathbf{w}_0^k)$ instead directly set $\alpha_0^k = \alpha_{k-1}$. As a result, they need to explicitly tackle the error $\|\alpha_0^k - \alpha(\mathbf{w}_k)\|^2$. To this end, they develop a novel analysis based on a new Lyapunov function to prove the convergence. In contrast to that in Reference [136], which uses the recursion of $F(\mathbf{w}_{k-1}) - F(\mathbf{w}_k)$, Yan et al. use both the recursions of the duality gap of the regularized function $F_k$ and of $F(\mathbf{w}_{k-1}) - F(\mathbf{w}_k)$. They are able to bound $\|\alpha_0^k - \alpha(\mathbf{w}_k)\|^2$ by the duality gap of the regularized function.

When the objective is just concave in terms of the dual variable, Zhao [202] develop a stagewise stochastic algorithm similar to Algorithm 1 except that the primal function is also smoothed by adding a strongly concave term on the dual variable, which has the same complexity as Reference [136].

Liu et al. [101] consider the deep AUC maximization explicitly and develop the first practical and provable stochastic algorithms for deep AUC maximization based on the min-max formulation of the pairwise square loss function, which enjoy a faster convergence rate. In particular, they assume that the primal objective function $F(\mathbf{w})$ satisfies a PL condition, i.e., there exists $\mu > 0$ such that $\|\nabla F(\mathbf{w})\|^2 \geq \mu(F(\mathbf{w}) - F_*)$, where $F_*$ denotes the global minimum of $F$. They show that two-layers neural network satisfy this PL condition. Based on this condition, they have shown that Algorithm 2 enjoys a faster convergence rate in the order of $O(1/(\mu^2\epsilon))$ for finding an $\epsilon$-level optimal solution. For $\alpha_0^k$, they compute it similarly to that in Reference [136] except that it is approximated by sampling a number of data. For the parameters $\eta_k, T_k$, they decrease $\eta_k$ geometrically and increase $T_k$ geometrically. For the stochastic algorithm $\mathcal{A}$, they employ both stochastic primal-dual gradient method and stochastic primal-dual adaptive gradient method, where the latter one could enjoy even faster convergence when the stochastic gradients have a slow growth.

Recently, Guo et al. [57] propose a family of **Proximal Epoch Stochastic (PES)** methods for more generic non-convex min-max optimization under a PL condition and establish several

**ALGORITHM 3:** A Single-loop Algorithmic Framework for solving $\min_{\mathbf{w}\in\mathbb{R}^d}\max_{\alpha\in\Omega} F(\mathbf{w},\alpha)$

```
Initialization: w_0 ∈ R^d, α_0 ∈ Ω, η_1, η_2, T,
for t = 0, 1, 2, ..., T do
    Compute a stochastic estimator of ∇_w F(w_t, α_t) by u_{t+1}
    Update w_{t+1} = w_t − η_{1,t} u_{t+1}
    Set ŵ_t appropriately
    Compute a stochastic estimator of ∇_α F(ŵ_t, α_t) by v_{t+1}
    Update α_{t+1} = ∏_Ω[α_t − η_{2,t} v_{t+1}]
end for
```

improved rates under different conditions, e.g., near convexity condition of the primal objective, and Lipchitz condition of stochastic gradients. Under these conditions, they can reduce the sample complexity to $O(1/(\mu\epsilon))$. They also analyze the convergence rates for multiple stochastic algorithms $\mathcal{A}$, including stochastic gradient descent ascent, stochastic optimistic gradient descent ascent, stochastic primal-dual STORM updates, and so on. In addition, they establish the PL condition of the primal objective for AUC maximization for learning over-parameterized neural networks.

Guo et al. [55, 195] also study the federated deep AUC maximization by solving the min-max formulations in a distributed fashion and establish both computation and communication complexity under a PL condition of the objective function. It is notable that Reference [195] claims that they achieve the optimal communication complexity.

**Single-loop Stochastic Primal-Dual Methods.** A generic framework of single-loop stochastic gradient descent ascent methods is shown in Algorithm 3. At each iteration, it computes a stochastic gradient estimator $\mathbf{u}_{t+1}$ of $\nabla_{\mathbf{w}}F(\mathbf{w}_t,\alpha_t)$ and then updates the primal variable based on this gradient estimator. Then it computes a stochastic gradient estimator $\mathbf{v}_{t+1}$ of $\nabla_{\alpha}F(\widehat{\mathbf{w}}_t,\alpha_t)$ and update, the dual variable based on $\mathbf{v}_{t+1}$ for some $\widehat{\mathbf{w}}_t$. Different methods differ from each other on how to compute the gradient estimators $\mathbf{u}_{t+1}$ and $\mathbf{v}_{t+1}$.

Lin et al. [100] are the first to analyze the single-loop primal-dual method (the basic stochastic gradient descent ascent method, i.e., SGDA) for non-convex concave min-max optimization problems, corresponding to Algorithm 3 with $\widehat{\mathbf{w}}_t = \mathbf{w}_t$. In the paper, they assume the objective function $F(\mathbf{w},\alpha)$ is smooth in terms of both $\mathbf{w}$ and $\alpha$. They compute $\mathbf{u}_{t+1} = \frac{1}{B}\sum_{\mathbf{z}_t\in\mathcal{B}}\nabla_{\mathbf{w}}F(\mathbf{w}_t,\alpha_t,\mathbf{z}_t)$ and $\mathbf{v}_{t+1} = \frac{1}{B}\sum_{\mathbf{z}_t\in\mathcal{B}}\nabla_{\alpha}F(\mathbf{w}_t,\alpha_t,\mathbf{z}_t)$ based on a batch of $B$ samples. However, their convergence results are unsatisfactory. In particular, for non-convex concave min-max problems, their analysis yields an $O(1/\epsilon^8)$ complexity for finding an $\epsilon$-stationary solution to $F(\mathbf{w})$; and for non-convex strongly concave min-max problems, their analysis requires a large mini-batch size in the order of $O(1/\epsilon^2)$ and yields an $O(1/\epsilon^4)$ sample complexity. It is worth to point out that the complexity for the former case is worse than that established in Reference [136] and the complexity for the latter case matches that in Reference [136] but requires a large mini-batch size, which is not required in Reference [136]. Recently, Boţ and Böhm [14] extend the analysis to stochastic alternating (proximal) gradient descent ascent method, which uses $\widehat{\mathbf{w}}_t = \mathbf{w}_{t+1}$ to compute the estimator $\mathbf{v}_{t+1}$. However, this algorithm suffers from the same issue of requiring a large mini-batch size and the worse complexity for non-convex concave min-max problems.

Recently, Guo et al. [56] develop a new stochastic primal-dual method for solving non-convex strongly concave min-max problems under the smoothness assumption of $F(\mathbf{w},\alpha)$. They address the issue of large mini-batch size requirement in References [14, 100]. The key improvement lies at using moving average to compute the estimator $\mathbf{u}_{t+1}$, i.e., $\mathbf{u}_{t+1} = (1-\beta_{1,t})\mathbf{u}_t + \beta_{1,t}O_{\mathbf{w}}(\mathbf{w}_t,\alpha_t)$ and simply use $\mathbf{v}_{t+1} = O_{\alpha}(\mathbf{w}_t,\alpha_t;\mathbf{z}_t)$, where $O_{\mathbf{w}}$ and $O_{\alpha}$ denote an unbiased stochastic estimator of

$\nabla_{\mathbf{w}} F(\mathbf{w}, \alpha)$ and $\nabla_{\alpha} F(\mathbf{w}, \alpha)$, respectively. The authors also establish the convergence using adaptive step sizes such as the Adam-style with a sample complexity in the order of $O(1/\epsilon^4)$. This is the first work that establishes the convergence Adam-style updates for solving non-convex min-max problems.

An improved complexity of $O(1/\epsilon^3)$ is achieved in several recent works under the Lipschitz continuous assumption for the stochastic gradient $\nabla_{\mathbf{w}} F(\mathbf{w}, \alpha; \mathbf{z})$ and $\nabla_{\alpha} F(\mathbf{w}, \alpha; \mathbf{z})$ [75, 107], which is a stronger condition than the smoothness condition of the objective function. Luo et al. [107] are the first to establish such an improved rate. Their algorithm called SREDA uses the SPIDER/SARAH technique [42, 129] to update the gradient estimators $\mathbf{u}_{t+1}$ and $\mathbf{v}_{t+1}$, i.e., $\mathbf{u}_{t+1} = \mathbf{u}_t + \frac{1}{B}\sum_{\mathbf{z}_t \in \mathcal{B}} \nabla_{\mathbf{w}} F(\mathbf{w}_t, \alpha_t, \mathbf{z}_t) - \frac{1}{B}\sum_{\mathbf{z}_t \in \mathcal{B}} \nabla_{\mathbf{w}} F(\mathbf{w}_{t-1}, \alpha_{t-1}, \mathbf{z}_t)$, where $B$ is in the order of $O(1/\epsilon)$. $\mathbf{u}_t$ and $\mathbf{v}_t$ are re-computed based on a large batch size in the order of $O(1/\epsilon^2)$ every $q = O(1/\epsilon)$ iterations. It is worth mentioning that SREDA is a double loop algorithm, where the inner loop is to mainly update the dual variable and the estimators $\mathbf{u}_{t+1}, \mathbf{v}_{t+1}$ with multiple iterations and the outer loop is to update the primal variable. This issue was addressed by Huang et al. [75], who propose a single-loop algorithm named AccMDA to enjoy a fast rate of $O(1/\epsilon^3)$ under the Lipschitz continuous assumption for the stochastic gradient. They use the STORM technique [31] to compute $\mathbf{u}_{t+1}, \mathbf{v}_{t+1}$, i.e., $\mathbf{u}_{t+1} = (1-\beta_t)\mathbf{u}_t + \beta \nabla_{\mathbf{w}} F(\mathbf{w}_t, \alpha_t, \mathbf{z}_t) - \beta_t \nabla_{\mathbf{w}} F(\mathbf{w}_{t-1}, \alpha_{t-1}, \mathbf{z}_t)$, similarly for $\mathbf{v}_{t+1}$. It is notable that both SREDA and AccMDA require computing two (batch) stochastic gradients at each iteration. It is notable that AccMDA has a worse dependence on the strong concavity parameter than that in References [56, 100]. It is likely that by simply computing $\mathbf{v}_{t+1} = \nabla_{\alpha} F(\mathbf{w}_t, \alpha_t, \mathbf{z}_t)$ in AccMDA, one should be able to improve the dependence on the strong concavity as in Reference [56].

Yang et al. [182] develop a single-loop algorithm for improving the convergence rate of non-convex min-max optimization under PL conditions. They consider a class of smooth non-convex non-concave problems, which satisfy both the dual-side PL condition (i.e., $F(\mathbf{w}, \cdot)$ satisfies a PL condition for any $\mathbf{w}$) and the primal-side PL condition (i.e., $F(\cdot, \alpha)$ satisfies a PL condition for any $\alpha$). They propose **stochastic alternating gradient descent ascent algorithm (Stoc-AGDA)** and establish a global convergence for a Lyapunov function $F(\mathbf{w}_t) - F_* + \lambda(F(\mathbf{w}_t) - F(\mathbf{w}_t, \alpha_t))$ for a constant $\lambda$ in the order of $O(1/\epsilon)$, which directly implies the convergence for the primal objective gap in the same order. Their algorithm uses a polynomially decreasing or very small step sizes. It is notable that the complexity of Stoc-AGDA is worse than that of PES established in Reference [57] under similar PL conditions but requiring the strong concavity of the objective function in terms of the dual variable, which makes PES more appropriate to deep AUC maximization. Without the primal-side PL condition, Stoch-AGDA and Smoothed-AGDA are also analyzed under the dual-side PL condition with a better dependence on the condition number [183].

**Improved Rates for the Offline (Finite-sum) Setting.** There are also multiple papers trying to improve the complexity of non-convex (strongly) concave min-max optimization in the finite-sum setting by leveraging the variance reduction techniques [107, 136, 182]. However, they usually require computing the gradient based on the full-batch or a large-batch that are less practical for deep learning with big data, see Table 6.

### 7.2 Deep Partial AUC Maximization

Deep pAUC maximization is challenging not only because of the non-differentiable selection operator but also due to non-convexity of the objective. Below, we discuss two classes of methods.

**Naive Mini-batch Approach.** Kar et al. [85] propose mini-batch-based stochastic methods for pAUC maximization, which is applicable to deep learning. At each iteration, a gradient estimator is simply computed based on the pAUC surrogate function of the mini-batch data. However, this

Table 6. Comparison of Different Stochastic Methods for Solving Non-convex Strongly Concave Min-max Optimization

| Method | Category | batch size | Sample Complexity | Oracle | Experiments for AUC Max. |
|---|---|---|---|---|---|
| SGDA [100] | Single-loop | $O(1/\epsilon^2)$ | $O(1/\epsilon^4)$ | General | No |
| PDAda [56] | Single-loop | $O(1)$ | $O(1/\epsilon^4)$ | General | Yes |
| AccMDA [75] | Single-loop | $O(1)$ | $O(1/\epsilon^3)$ | Lipschitz | No |
| Stoc-AGDA [182] | Single-loop | $O(1)$ | $O(1/(\mu^2\epsilon))$ | General | No |
| PG-SMD [136] | ProximalPoint | $O(1)$ | $O(1/\epsilon^4 + n/\epsilon^2)$ | General | No |
| Epoch-SGDA [179] | Proximal Point | $O(1)$ | $O(1/\epsilon^4)$ | General | No |
| PPD-SG [101] | Proximal Point | $O(1)$ | $O(1/\epsilon^4)$ | General | Yes |
| PPD-AdaGrad [101] | Proximal Point | $O(1)$ | $O(1/\epsilon^4)$ | General | Yes |
| PES-$\mathcal{A}$ [57] | Proximal Point | $O(1)$ | $O(1/\epsilon^4)$~$O(1/(\mu\epsilon))$ | General or Lipchitz | Yes |

$\epsilon$ denotes the target accuracy level for the objective gradient norm, i.e., $\mathbb{E}[\|\nabla F(\mathbf{w})\|] \leq \epsilon$ or the primal objective gap, i.e., $\mathbb{E}[F(\mathbf{w}) - F(\mathbf{w}_*)] \leq \epsilon$. The column "oracle" denotes the condition on the stochastic gradient. $\mu$ denotes a parameter in the assumed PL condition.

heuristic approach is not guaranteed to converge for minimizing the pAUC objective and its error scales as $O(1/\sqrt{B})$, where $B$ is the mini-batch size. Ueda and Fujino [159] consider partial AUC maximization for learning non-linear scoring functions, e.g., neural networks and probabilistic generative models. The paper claims to use the Adam optimizer [88] in Tensorflow for optimizing the partial AUC. However, it does not provide any discussion how the algorithm was implemented and what is the complexity and convergence of the optimization algorithm. We conjecture they use the naive mini-batch approach equipped with the Adam optimizer. For experiments, they have used an image dataset, namely, **Hyper Suprime-Cam (HSC)** dataset [116] with 487 real and 267,074 bogus optical transient objects collected with the HSC using the Subaru telescope.

**Reduction Approaches.** The idea is to reduce the objective into different formulations (equivalent or approximate), which facilitate the design of large-scale optimization algorithms.

Recently, Yang et al. [187] consider optimizing two-way pAUC with FPR less than $\beta$ and TPR larger than $1 - \alpha$. The paper focuses on simplifying the optimization problem that involves selection of top-ranked negative examples and bottom-ranked positive examples. They first formulate the problem into a bilevel optimization, where the upper-level objective function is a weighted average of pairwise surrogate loss, and the lower-level optimization problem is to compute the weights that accounts for selection of top-ranked negative examples and bottom-ranked positive examples. To address the computational challenge for solving the bilevel optimization problem, the authors propose to simplify the lower-level problem by relaxing the non-decoposable constraint on the decision variables into decomposable regularization. As a result, a simplified weighted pairwise loss minimization problem is derived, where the weights for each positive-negative pair is a product of two individual weights that are computed directly from the prediction scores of the positive and negative examples using a penalty function. Then any stochastic algorithms based on random positive-negative pairs can be employed for solving their formulation, e.g., SGD, Adam.

Zhu et al. [206] consider both pAUC maximization and two-way pAUC maximization. For pAUC maximization, they focus on that with FPR in a range $(0, \beta)$. They propose two formulations for pAUC maximization by leveraging distributionally robust optimization technique, and develop stochastic algorithms for optimizing both formulations for both pAUC and two-way pAUC. In particular, for pAUC they define a robust loss for each positive data by

$$\hat{L}_\phi(\mathbf{w}; \mathbf{x}_i) = \max_{\mathbf{p} \in \Delta_{n_-}} \sum_{\mathbf{x}_j \in \mathcal{S}_-} p_j \ell(f_\mathbf{w}(\mathbf{x}_j) - f_\mathbf{w}(\mathbf{x}_i)) - \lambda D_\phi(\mathbf{p}, 1/n_-),$$

where $\Delta_{n_-} = \{\mathbf{p} \in \mathbb{R}^{n_-} : \sum_j p_j = 1, p_j \geq 0\}$ is a simplex, $D_\phi(\mathbf{p}, 1/(n_-)) = \frac{1}{n_-}\sum_i \phi(n_- p_i)$ is a divergence measure defined by a function $\phi$. Then the following objective is used for one-way pAUC maximization:

$$\min_{\mathbf{w}} \frac{1}{n_+} \sum_{\mathbf{x}_i \in \mathcal{S}_+} \hat{L}_\phi(\mathbf{w}; \mathbf{x}_i). \tag{21}$$

They consider two functions $\phi$, i.e., the KL divergence $\phi_{kl}(t) = t \log t - t + 1$, which gives $D_\phi(\mathbf{p}, 1/n_-) = \sum_i p_i \log(n_- p_i)$, and the CVaR divergence $\phi_c(t) = \mathbb{I}(0 < t \leq 1/\beta)$ with a parameter $\beta \in (0, 1)$, which gives $D_\phi(\mathbf{p}, 1/n_-) = 0$ if $p_i \leq 1/(n_-\beta)$ and infinity otherwise. It is shown that if $\ell(\cdot)$ is non-decreasing, when using $\phi_c$ the objective (21) is equivalent to (10) for pAUC maximization, when using $\phi_{kl}$ it gives a soft estimator of pAUC.

For solving Equation (21) with CVaR divergence, they formulate the problem as a weakly convex optimization problem by introducing another set of variables, i.e.,

$$\min_{\mathbf{w}} \min_{\mathbf{s} \in \mathbb{R}^{n_+}} F(\mathbf{w}, \mathbf{s}) = \frac{1}{n_+} \sum_{\mathbf{x}_i \in \mathcal{S}_+} \left( s_i + \frac{1}{\beta} \frac{1}{n_-} \sum_{\mathbf{x}_j \in \mathcal{S}_-} (\ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i)) - s_i)_+ \right). \tag{22}$$

They develop an efficient stochastic algorithm named SOPA with a sample complexity of $O(1/\epsilon^4)$ for finding a nearly $\epsilon$-stationary point for $F(\mathbf{w}, \mathbf{s})$.

For solving Equation (21) with KL divergence, they formulate the problem as a novel finite-sum coupled compositional optimization problem, i.e.,

$$\min_{\mathbf{w}} \frac{1}{n_+} \sum_{\mathbf{x}_i \sim \mathcal{S}_+} \lambda \log \mathbb{E}_{\mathbf{x}_j \in \mathcal{S}_-} \exp\left( \frac{\ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i))}{\lambda} \right). \tag{23}$$

A stochastic algorithm named SOPA-s is proposed for solving Equation (23) with a sample complexity of $O(1/\epsilon^4)$ for finding an $\epsilon$-level stationary point.

For two-way pAUC such that FPR is less than $\beta$ and TPR is larger than $\alpha$, the authors further define a new objective:

$$F(\mathbf{w}; \phi, \phi') = \max_{\mathbf{q} \in \Delta_{n_+}} \sum_{\mathbf{x}_i \in \mathcal{S}_+} q_i \hat{L}_\phi(\mathbf{x}_i, \mathbf{w}) - \lambda' D_{\phi'}\left(\mathbf{q}, \frac{1}{n_+}\right). \tag{24}$$

They prove that when $\phi(t) = \mathbb{I}(0 < t \leq 1/\beta)$ and $\phi'(t) = \mathbb{I}(0 < t \leq 1/\alpha)$ the above objective is equivalent to Equation (11) if $\ell(\cdot)$ is non-decreasing. The authors develop two algorithms for solving the above objective with CVaR divergence and KL divergence, respectively, and establish their convergence. A sample complexity of $O(1/\epsilon^4)$ is established for the algorithm that optimizes the above objective with the KL-divergence to find a $\epsilon$-level stationary point. For optimizing the above objective with CVaR divergence, the sample complexity of their algorithm is $O(1/\epsilon^6)$. This is the first time that stochastic algorithms are developed for optimizing two-way pAUC for deep learning with convergence guarantee.

A concurrent work by Yao et al. [190] focuses on optimizing pAUC such that FPR is in a range $(\alpha, \beta)$. When $\ell(\cdot)$ is a non-decreasing function, they formulate Equation (21) as non-smooth different-of-convex problems:

$$F(\mathbf{w}) = \frac{1}{n_+} \sum_{\mathbf{x}_i \in \mathcal{S}_+} (F(\mathbf{w}; \mathbf{x}_i, k_2) - F(\mathbf{w}; \mathbf{x}_i, k_1)), \tag{25}$$

where

$$F(\mathbf{w}; \mathbf{x}_i, k) = \frac{1}{n_-} \sum_{\mathbf{x}_j \in \mathcal{S}_-^{\downarrow}[1,k]} \ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i)) = \min_{\lambda} \frac{\lambda k}{n_-} + \frac{1}{n_-} \sum_{\mathbf{x}_j \in \mathcal{S}_-} (\ell(f_{\mathbf{w}}(\mathbf{x}_j) - f_{\mathbf{w}}(\mathbf{x}_i)) - \lambda)_+.$$

They develop an efficient approximated gradient descent method based on the Moreau envelope smoothing technique, inspired by recent advances in non-smooth DC optimization [155]. To increase the efficiency of large data processing, they use an efficient stochastic block coordinate update for solving each sub-problem inexactly. A sample complexity of $O(1/\epsilon^6)$ is established for their algorithm to find a nearly $\epsilon$-stationary solution.

### 7.3 Applications of Deep AUC Maximization (DAM)

Due to the success of deep learning in various applications, DAM has also been applied to different domains with demonstrated success. Below, we review some applications of DAM.

**Medical Image Classification.** Sulam et al. [154] consider the classification of breast cancer based on imbalanced mammogram images. They learn a deep convolutional neural network by AUC maximization by using the online buffered gradient method proposed by Zhao et al [201]. Nevertheless, the issue of this approach is that it cannot scale to large datasets, as it requires a large buffer to store positive and negative samples at each iteration for computing an approximate AUC score. As a result, they only consider small-scale datasets. In particular, two datasets are used. The first one, named IMG, is a proprietary mammogram dataset comprising 796 patients, 80 of them defined as positive (164 images), and 716 negative (1,869 images) with both **Cranial-Caudal (CC)** and **Mediolateral-Oblique (MLO)** views, belonging to normal patients as well as benign findings. The second dataset is the public INbreast dataset, which consists of 115 cases with 410 images.

Yuan et al. [196] are the first to evaluate the performance of DAM on large-scale medical image data with hundreds of thousands of images for learning modern deep neural networks (e.g., ResNets, DenseNets). They propose a new minimax objective as in Equation (16) for robust AUC maximization to alleviate the issues of the square loss, namely, the sensitivity to noisy data and the adverse effect on easy data. The new objective is shown to be more robust than the commonly used square loss, while enjoying the same advantage in terms of large-scale stochastic optimization. The authors employ PESG [57] for solving the minimax objective. They conduct extensive empirical studies of DAM on four difficult medical image classification tasks discussed below.

- CheXpert Competition. CheXpert is a large-scale chest X-ray dataset for detecting chest and lung diseases, which is released through a medical AI competition [79]. The training data consists of 224,316 high-quality X-ray images from 65,240 patients with frontal and lateral views. The validation dataset consists of 234 images from 200 patients. The testing data has images for 500 patients, which is not released to the public. The model is only evaluated for predicting five selected diseases, i.e., Cardiomegaly, Edema, Consolidation, Atelectasis, Pleural Effusion, which have an average imbalance ratio (i.e., the proportion of positive examples) of 20.21% in the training set. AUC and NRBC are used for evaluation, where NRBC refers a number of radiologists out of 3 are beaten by AI algorithms. Yuan et al. [196] achieved first place using their DAM method in this competition in August 2020. Compared with the standard deep learning approach that minimizes the cross-entropy loss, their DAM method achieves 2% improvements.
- SIIM-ISIC Melanoma Classification. Melanoma is a skin cancer and is the major cause of skin cancer death [115]. Kaggle hold a competition for melanoma classification in 2020. The dataset consists of 33,126 training images with 584 malignant melanoma images and 10,892 testing images with an unknown number of malignant melanoma images. The testing set is split into a validation set with 30% images and the final testing set with 70% images. The raw images have high resolutions, e.g., 6,000 $\times$ 4,000. Yuan et al. [196] demonstrate the performance DAM for Melanoma Classification. They resize the the image into lower resolutions,

e.g., 384 × 384 and also use additional 12,859 images from previous competitions in their experiments. Their method achieves the 33rd place out 3,314 teams for the competition by ensemble over 10 models. A simple ensemble of a DAM model and a standard model learned by optimizing the cross-entropy loss beats the winning team by combining 18 models [196]. Compared with the standard deep learning approach that minimizes the cross-entropy loss, the DAM method achieves 1% improvements.

- Breast-Cancer Screening. For this task, they use the DDSM+ data, which is a combination of two datasets, namely, DDSM and CBIS-DDSM [17, 67]. The dataset consists of 55,000 mammographic images (224 × 224) taken at lower doses than usual X-rays for training with an imbalance ratio of 13% and 13,900 images for testing with an imbalance ratio of 4%. Compared with the standard deep learning approach that minimizes the cross-entropy loss, the DAM method achieves 1.5% improvements.
- Lymph Node Tumor Detection. They use the PathCamelyon dataset for this task, which consists of 294,912 color images (96 × 96) extracted from histopathologic scans of lymph node section for training and 32,768 images for testing with balanced class ratio [8, 160]. The authors manually construct an imbalanced dataset with an imbalance ratio of 1% for their experiments. Compared with the standard deep learning approach that minimizes the cross-entropy loss, the DAM method achieves 5% improvements.

Recently, Reference [66] investigated DAM for COVID-19 Chest X-ray Classification. Covid-19 is a global pandemic that broke out in 2020. Early detection of Covid-19 is crucial to contain the spread of the virus and is helpful for providing early treatment for the patients with Covid-19. The authors use the dataset (COVIDx8B), which consists of 15,952 chest X-ray images for training with 13.5% Covid-19 positive samples and 400 images for testing with balanced positive and negative samples. The authors use self-supervised training method discussed below for learning a backbone network and then use the LibAUC library [196] for fine-tuning the network with significant improvements observed over the baseline method.

**Molecular Property Predictions.** Molecular property prediction is one of the key tasks in cheminformatics and has applications in many fields, including quantum mechanics, physical chemistry, biophysics, and physiology. Multiple molecular datasets have been released, e.g., MoleculeNet benchmark datasets (e.g., PCBA, HIV, MUV, Tox21, ToxCast) [174], MIT AICURES Challenge dataset,[2] Stanford OGB benchmark datasets (e.g., OGBG-molhiv, OGBG-molpcba) [73]. Recently, Reference [171] has employed the LibAUC library [196] for solving the molecular property prediction and achieved the first place at MIT AICURES Challenge. Several research groups have also used LibAUC library for improving the performance on the OGBG-molhiv dataset.[3] The authors of Reference [206] also consider pAUC maximization on some of these molecular datasets and compare different methods for pAUC maximization.

**Fraud/Outlier Detection.** Identifying outliers in data is referred to as outlier or anomaly detection. It can be regarded as an extremely imbalanced classification problem where the object of interest (anomaly/outlier) is the minority class. AUC maximization can naturally be applied for outlier or anomaly detection. In Reference [36], multiple online AUC maximization algorithms, including OAM [201] and OPAUC [47], are applied to the benchmark datasets for outlier/anomaly detection such as Webspam [163], Sensor Faults [114], and Malware App [205]. The performance of different stochastic AUC maximization algorithms is compared in References [97, 122] for

[2] https://www.aicures.mit.edu/tasks.
[3] https://ogb.stanford.edu/docs/leader_graphprop/.

Table 7. Benchmark Datasets for Deep AUC and pAUC Maximization

| Dataset | Type | Order of Data Size | Imbalance | Source | References for DAM | Benchmark Result |
|---|---|---|---|---|---|---|
| STL10 | Natural Image | $10^3$ | Artificial | [27] | [196, 207] | - |
| CIFAR10 | Natural Image | $10^4$ | Artificial | [91] | [196, 207] | - |
| CIFAR100 | Natural Image | $10^4$ | Artificial | [91] | [196, 207] | - |
| Cat vs Dog | Natural Image | $10^4$ | Artificial | [41] | [196, 207] | - |
| Melanoma | Skin Lesion Image | $10^4$ | Natural | [1] | [196, 207] | 0.9505 |
| CheXpert | Chest X-ray Image | $10^5$ | Natural | [79] | [196, 207] | 0.9305 |
| DDSM+ | Mammographic Image | $10^4$ | Natural | [17, 67] | [196] | 0.9544 |
| PatchCamelyon | Microscopic Image | $10^5$ | Artificial | [8, 160] | [196] | - |
| MoleculeNet/HIV | Molecular Graph | $10^4$ | Natural | [174] | [207] | 0.770 |
| MoleculeNet/PCBA | Molecular Graph | $10^5$ | Natural | [174] | - | - |
| MoleculeNet/MUV | Molecular Graph | $10^4$ | Natural | [174] | [207] | 0.644 |
| MoleculeNet/Tox21 | Molecular Graph | $10^3$ | Natural | [174] | [207] | - |
| MoleculeNet/ToxCast | Molecular Graph | $10^3$ | Natural | [174] | [207] | - |
| ogbg-molhiv | Molecular Graph | $10^4$ | Natural | [72] | - | - |
| ogbg-molpcba | Molecular Graph | $10^5$ | Natural | [72] | [71] | 0.8406 |
| ogbg-molmuv | Molecular Graph | $10^4$ | Natural | [72] | [206] | - |
| ogbg-moltox21 | Molecular Graph | $10^3$ | Natural | [72] | [206] | - |
| ogbg-moltoxcast | Molecular Graph | $10^3$ | Natural | [72] | [206] | - |

The differences between the MoleculeNet and OGBG molecular graph datasets lie at the way of training/testing split. The numbers in the last column are the best results on testing data reported in the referred papers for AUC maximization. We do not include the results for partial AUC maximization, as it requires setting a threshold for FPR/TPR. We also do not present the results on datasets with manual construction of imbalanced data denoted by "Artificial" for the "Imbalance" column.

anomaly detection. In Reference [76], the authors consider AUC maximization for fraud detection on a graph. They consider learning both the parameters of a **graph neural network (GNN)** and a policy of edge pruning that affects prediction outputs of the GNN. For learning the parameters of GNN, they employ the PPD-SG algorithm [101] to solve the saddle point formulation. The learning of the edge pruner is formulated as a reinfocement learning problem and a classic policy gradient is used.

**Other Applications.** Besides medical image classification, molecular property prediction, and fraud/outlier detection, AUC/pAUC maximization have been investigated in other applications, which do not necessarily involve deep learning. Examples include points of interest recommendation [61], credit scoring for financial institutions [99], time series classification for medicine, manufacturing, and maintenance [177], protein disorder prediction [166], pedestrian detection [133], differentiated gene detection [104], and discovery of motifs [208].

**Benchmark and Library.** We present a summary of benchmark datasets in Table 7 for DAM and include references that provide benchmark results of deep AUC maximization and deep pAUC maximization. The author T. Yang's group has developed an open-source library for DAM called LibAUC,[4] which implements a set of efficient stochastic algorithms for deep AUC maximization, deep one-way pAUC maximization, and deep two-way pAUC maximization.

### 7.4 Summary

The research of deep AUC maximization springs from the studies for solving the non-convex min-max problems. The applicability in deep AUC maximization motivates a wave of studies for algorithmic design and theoretical analysis of non-convex strongly concave min-max problems. New formulations for AUC maximization and partial AUC maximization are also developed, for which

[4] www.libauc.org.

efficient stochastic algorithms are proposed. The algorithms are then employed for solving real-world applications with great success, e.g., medical image classification and molecular property prediction. However, there are still many challenges regarding deep (partial) AUC maximization to be addressed, which will be discussed in the next section.

## 8 OTHER ISSUES FOR DAM AND OUTLOOK FOR FUTURE WORK

Below, we discuss five remaining or emerging issues for DAM.

**Large-scale Stochastic Optimization.** Although large-scale optimization algorithms for DAM have been developed, there are still many open problems to be addressed. Below, we will list several important questions. (i) How to further improve the algorithms and theories for solving the composite objectives of AUC and for solving pAUC objectives? Zhu et al. [207] employ stochastic compositional algorithms for solving the composite objectives (18). There is still much room for improving the optimization for solving pAUC objectives, e.g., Equations (22)∼(25), or new formulations of pAUC. (ii) How to optimize AUC in the federated learning setting? Although federated deep AUC maximization for the minimax objective has been considered in References [55, 195], federated learning algorithms for optimizing other objectives remains to be developed. In particular, the objectives (22)∼(25) for pAUC maximization are much more challenging to be optimized in the federated learning setting.

**Network Structures.** Standard deep neural networks have been used for DAM in different applications. For example, deep convolutional neural networks such as VGG, ResNets, DenseNets, EfficientNets have been used for medical image classification [66, 154, 196]. Graph neural networks, e.g., **graph isomorphism network (GIN)** [74, 175], **Message-Passing Neural Network (MPNN)** [50, 171], have been used for molecular property prediction [171, 206]. It remains to be explored by using more advanced network structures, e.g., vision transformer [38] for medical image classification tasks in the context of DAM. Another interesting direction is to explore **neural architecture search (NAS)** [93] in the context of DAM. A natural question is if we use AUC as a performance measure of NAS, how would the found network be different from standard approaches that use accuracy as a performance measure?

**Regularization and Normalization.** Regularization is an important technique for improving generalization. A standard regularization technique is to use weight decay [51]. It was shown to be effective for DAM as well [207]. Nevertheless, more algorithmic regularization techniques should be considered for DAM, including explicit and implicit regularization. Recently, Zhu et al. [207] demonstrate that the explicit regularization by adding the quadratic term in the proximal point methods discussed in Section 7.1 is helpful for improving the generalization. Implicit regularization by gradient-based methods [128] for DAM is an interesting question to be explored. Normalization is another key technique for improving the training of deep neural networks, e.g., batch normalization [78], layer normalization [5], and so on. Of particular interest to DAM is the normalization in the output layer. Yuan et al. [196] have used a batch score normalization layer, which normalizes the non-activated scores in the mini-batch such that the $\ell_2$ norm of scores in the mini-batch is one. This is found to be better than not using any activation or normalization. Zhu et al. [207] show that using a sigmoid activation in the last layer also yields better performance than not using any activation or normalization. They also demonstrate that using the sigmoid activation is competitive with the batch score normalization, and they are better than the standard batch normalization.

**Data Sampling and Augmentation.** The standard data sampling for deep learning is to use data shuffling over all examples. However, different data sampling strategies have been considered for imbalanced data, e.g., oversampling and undersampling [82]. Recently, Zhu et al. [207]

demonstrate via empirical studies that oversampling for the minority class is helpful for improving the generalization performance of DAM. However, how can we incorporate more advanced oversampling or data augmentation techniques into DAM remains an interesting topic. One might consider synthetic oversampling method SMOTE [23] and MIXUP [197] for DAM.

**Feature Learning.** Feature learning is an important capability of deep learning for tackling unstructured data. It is shown that directly optimizing the AUC loss from scratch does not necessarily yield better feature representations [194]. A practice of DAM uses a two-stage approach: The first stage is to learn the encoder network by optimizing the traditional cross-entropy loss, and the second stage is to fine-tune the encoder network and to learn the classifier by DAM [196]. It is still not fully understood why optimizing the AUC loss in an end-to-end fashion does not yield better feature representations, and it remains an open problem how to learn better encoder networks by using DAM. Recently, Yuan et al. [194] proposed an end-to-end training method called compositional training for DAM. The idea is to solve a compositional objective $L_{\text{AUC}}(\mathbf{w} - \alpha \nabla L_{\text{CE}}(\mathbf{w}))$, where $L_{\text{AUC}}$ denotes an AUC loss and $L_{\text{CE}}$ denotes a standard cross-entropy loss. It is shown that the compositional training method for DAM yields much better feature representations than optimizing either the CE loss or the AUC loss from scratch. It remains an open problem how to understand this method theoretically. Another direction is to consider self-supervised pre-training methods on large-scale unlabeled medical datasets. This approach was recently explored in References [4, 151, 199]. The success on downstream tasks of using DAM has also been demonstrated for detecting COVID-19 based on X-ray images [66]. Nevertheless, we could consider pre-training on much larger medical datasets than those used in existing studies and demonstrate the performance of DAM on multiple downstream medical image classification tasks.

**Learning fair and interpretable AI models.** Building trustworthy AI is important for many domains, e.g., healthcare, in particular medical image classification. Two issues are of foremost importance, namely, fairness [7, 10, 20, 113] and interpretability [3, 24, 64]. Although these issues have received tremendous attention in the literature for medical image classification [26, 145, 146], developing fair and interpretable DAM methods remains to be explored. Some outstanding questions and work include (i) how to develop scalable in-processing algorithms for optimizing AUC under AUC-based fairness constraints [12, 84]; (ii) how to develop scalable and interpretable DAM methods; (iii) evaluating these fairness-aware and interpretable AUC optimization methods on large-scale medical image datasets.

**Out-of-distribution Robustness.** An emerging issue in machine learning that has attracted great attention is how to tackle the distributional shifts, i.e., the distribution of testing data differs from that of training data. While this issue has been investigated for traditional risk minimization, it has been rarely explored for AUC maximization. Given that AUC maximization is more aggressive in pushing positive examples ranked above negative examples [196], it might cause more severe performance degradation in the presence of distributional shifts. Different types of distributional shifts have been studied, e.g., domain generalization, subpopulation shift, covariate shift, concept drift, and so on. Accordingly, various benchmark datasets following different distributional shifts have been curated [54, 72, 89, 191]. Many of these datasets use AUROC as the performance measure. It remains an open problem how robust are existing DAM methods in the presence of distributional shifts and how to make them more robust.

Finally, we would like to point out that the above list of issues is not complete. There must be some other issues related to DAM or in the context of DAM to be addressed in the future. While this issue has been studied for traditional risk optimization, it has been rarely explored for AUC maximization.

## 9 CONCLUSIONS

In this article, we have presented a comprehensive survey of AUC maximization methods in the past 20 years with a focus on recent research and development of stochastic AUC maximization and deep AUC maximization. We have compared different methods from different perspectives, e.g., formulations, per-iteration complexity, sample complexities, optimization error, statistical error, empirical performance, and so on. We also discuss remaining and emerging issues in deep AUC maximization and provide suggestions of topics for future work.

## ACKNOWLEDGMENTS

We thank the editors and anonymous reviewers for their constructive comments.

## REFERENCES

[1] Kaggle Competition: SIIM-ISIC Melanoma Classification. Retrieved from https://www.kaggle.com/c/siim-isic-melanoma-classification.

[2] Shivani Agarwal. 2011. The infinite push: A new support vector ranking algorithm that directly optimizes accuracy at the absolute top of the list. In *SIAM International Conference on Data Mining*.

[3] Sercan O. Arık and Tomas Pfister. 2020. ProtoAttend: Attention-based prototypical learning. *J. Mach. Learn. Res.* 21 (2020), 1–35.

[4] Shekoofeh Azizi, Basil Mustafa, Fiona Ryan, Zachary Beaver, Jan Freyberg, Jonathan Deaton, Aaron Loh, Alan Karthikesalingam, Simon Kornblith, Ting Chen, Vivek Natarajan, and Mohammad Norouzi. 2021. Big Self-Supervised Models Advance Medical Image Classification. arXiv:eess.IV/2101.05224.

[5] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *ArXiv* abs/1607.06450.

[6] Ioannis Bargiotas, Argyris Kalogeratos, Myrto Limnios, Pierre-Paul Vidal, Damien Ricard, and Nicolas Vayatis. 2020. Multivariate two-sample hypothesis testing through AUC maximization for biomedical applications. In *11th Hellenic Conference on Artificial Intelligence (SETN'20)*. Association for Computing Machinery, New York, NY, 56–59. DOI: https://doi.org/10.1145/3411408.3411422

[7] Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2019. *Fairness and Machine Learning*. Retrieved from fairmlbook.org.

[8] Babak Ehteshami Bejnordi, Mitko Veta, Paul Johannes Van Diest, Bram Van Ginneken, Nico Karssemeijer, Geert Litjens, Jeroen A. W. M. Van Der Laak, Meyke Hermsen, Quirine F. Manson, Maschenka Balkenhol, et al. 2017. Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *Jama* 318, 22 (2017), 2199–2210.

[9] Gowtham Bellala, Jason Stanley, Clayton Scott, and Suresh K. Bhavnani. 2012. Active diagnosis via AUC maximization: An efficient approach for multiple fault identification in large scale, noisy networks. *CoRR* abs/1202.3701.

[10] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John T. Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. 2018. AI fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. *CoRR* abs/1810.01943.

[11] Martin Boissier, Siwei Lyu, Yiming Ying, and Ding-Xuan Zhou. 2016. Fast convergence of online pairwise learning algorithms. In *Artificial Intelligence and Statistics*. PMLR, 204–212.

[12] Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. In *World Wide Web Conference*. 491–500.

[13] Henrik Boström. 2004. Pruning and exclusion criteria for unordered incremental reduced error pruning.

[14] Radu Ioan Boţ and Axel Böhm. 2020. Alternating proximal-gradient steps for (stochastic) nonconvex-concave minimax problems. *arXiv preprint arXiv:2007.13605*.

[15] Olivier Bousquet and André Elisseeff. 2002. Stability and generalization. *J. Mach. Learn. Res.* 2 (2002), 499–526.

[16] Olivier Bousquet, Yegor Klochkov, and Nikita Zhivotovskiy. 2020. Sharper bounds for uniformly stable algorithms. In *Conference on Learning Theory*. PMLR, 610–626.

[17] K. Bowyer, D. Kopans, W. P. Kegelmeyer, R. Moore, M. Sallam, K. Chang, and K. Woods. 1996. The digital database for screening mammography. In *3rd International Workshop on Digital Mammography*. 27.

[18] Ulf Brefeld and Tobias Scheffer. 2005. AUC maximizing support vector learning. In *ICML Workshop on ROC Analysis in Machine Learning*.

[19] T. Calders and S. Jaroszewicz. 2007. Efficient AUC optimization for classification. In *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 42–53.

[20] Simon Caton and Christian Haas. 2020. Fairness in machine learning: A survey. *CoRR* abs/2010.04053.
[21] Nicolo Cesa-Bianchi and Gábor Lugosi. 2006. *Prediction, Learning, and Games*. Cambridge University Press.
[22] Nontawat Charoenphakdee, Jongyeong Lee, and Masashi Sugiyama. 2019. On Symmetric Losses for Learning from Corrupted Labels. arXiv:stat.ML/1901.09314.
[23] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Int. Res.* 16, 1 (June 2002), 321–357.
[24] Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su, and Cynthia Rudin. 2018. This looks like that: Deep learning for interpretable image recognition. *arXiv preprint arXiv:1806.10574*.
[25] Fan Cheng, Xia Zhang, Chuang Zhang, Jianfeng Qiu, and Lei Zhang. 2018. An adaptive robust online method for AUC maximization. *IEEE Access* 6 (2018), 52004–52013.
[26] Valeriia Cherepanova, Vedant Nanda, Micah Goldblum, John P. Dickerson, and Tom Goldstein. 2021. Technical challenges for training fair neural networks. *CoRR* abs/2102.06764.
[27] Adam Coates, Andrew Ng, and Honglak Lee. 2011. An analysis of single-layer networks in unsupervised feature learning. In *14th International Conference on Artificial Intelligence and Statistics*. PMLR, 215–223.
[28] Corinna Cortes and Mehryar Mohri. 2003. AUC optimization vs. error rate minimization. In *Conference on Advances in Neural Information Processing Systems*.
[29] Andrew Cotter, Heinrich Jiang, and Karthik Sridharan. 2019. Two-player games for efficient non-convex constrained optimization. In *Algorithmic Learning Theory*. PMLR, 300–332.
[30] Matt Culver, Deng Kun, and Stephen Scott. 2006. Active learning to maximize area under the ROC curve. In *6th International Conference on Data Mining (ICDM'06)*. 149–158. DOI: https://doi.org/10.1109/ICDM.2006.12
[31] Ashok Cutkosky and Francesco Orabona. 2019. Momentum-based variance reduction in non-convex SGD. In *Advances in Neural Information Processing Systems*. 15210–15219.
[32] Soham Dan and Dushyant Sahoo. 2021. Variance reduced stochastic proximal algorithm for AUC maximization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 184–199.
[33] Zhiyuan Dang, Xiang Li, Bin Gu, Cheng Deng, and Heng Huang. 2020. Large-scale nonlinear AUC maximization via triply stochastic gradients. *IEEE Trans. Pattern Anal. Mach. Intell.* (2020).
[34] Damek Davis and Dmitriy Drusvyatskiy. 2019. Stochastic model-based minimization of weakly convex functions. *SIAM J. Optim.* 29, 1 (2019), 207–239. DOI: https://doi.org/10.1137/18M1178244
[35] Yi Ding, Chenghao Liu, Peilin Zhao, and Steven C. H. Hoi. 2017. Large scale kernel methods for online AUC maximization. In *IEEE International Conference on Data Mining (ICDM)*. IEEE, 91–100.
[36] Yi Ding, Peilin Zhao, Steven Hoi, and Yew-Soon Ong. 2015. An adaptive gradient method for online auc maximization. In *AAAI Conference on Artificial Intelligence*.
[37] Lori Dodd and Margaret Pepe. 2003. Partial AUC estimation and regression. *Biometrics* 59 (10 2003), 614–23. DOI: https://doi.org/10.1111/1541-0420.00071
[38] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.
[39] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* 12, 7 (2011).
[40] Elad Eban, Mariano Schain, Alan Mackey, Ariel Gordon, Ryan Rifkin, and Gal Elidan. 2017. Scalable learning of non-decomposable objectives. In *Artificial Intelligence and Statistics*. PMLR, 832–840.
[41] Jeremy Elson, John (J. D.) Douceur, Jon Howell, and Jared Saul. 2007. Asirra: A CAPTCHA that exploits interest-aligned manual image categorization. In *14th ACM Conference on Computer and Communications Security (CCS)*.
[42] Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. 2018. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Conference on Advances in Neural Information Processing Systems*. 689–699.
[43] Asghar Feizi. 2020. Hierarchical detection of abnormal behaviors in video surveillance through modeling normal behaviors based on AUC maximization. *Soft Comput.* 24 (2020), 10401–10413.
[44] Vitaly Feldman and Jan Vondrak. 2018. Generalization bounds for uniformly stable algorithms. *Adv. Neural Inf. Process. Syst.* 31 (2018).
[45] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.* 4 (Dec. 2003), 933–969.
[46] Yoav Freund and Robert E. Schapire. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. In *2nd European Conference on Computational Learning Theory (EuroCOLT'95)*. Springer-Verlag, Berlin, 23–37.

[47] Wei Gao, Rong Jin, Shenghuo Zhu, and Zhi-Hua Zhou. 2013. One-pass AUC optimization. In *International Conference on Machine Learning*. 906–914.
[48] Wei Gao and Zhi-Hua Zhou. 2015. On the consistency of AUC pairwise optimization. In *24th International Conference on Artificial Intelligence (IJCAI'15)*. AAAI Press, 939–945.
[49] Saeed Ghadimi, Andrzej Ruszczynski, and Mengdi Wang. 2020. A single timescale stochastic approximation method for nested stochastic optimization. *SIAM J. Optim.* 30, 1 (2020), 960–979.
[50] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *34th International Conference on Machine Learning*. PMLR, 1263–1272.
[51] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. The MIT Press, Cambridge, MA.
[52] David M. Green and John A. Swets. 1966. *Signal Detection Theory and Psychophysics*. Wiley, New York.
[53] Bin Gu, Zhouyuan Huo, and Heng Huang. 2019. Scalable and efficient pairwise learning to achieve statistical accuracy. In *AAAI Conference on Artificial Intelligence*. 3697–3704.
[54] Shurui Gui, Xiner Li, Limei Wang, and Shuiwang Ji. 2022. GOOD: A Graph Out-of-Distribution Benchmark.
[55] Zhishuai Guo, Mingrui Liu, Zhuoning Yuan, Li Shen, Wei Liu, and Tianbao Yang. 2020. Communication-efficient distributed stochastic AUC maximization with deep neural networks. In *37th International Conference on Machine Learning*. 3864–3874.
[56] Zhishuai Guo, Yi Xu, Wotao Yin, Rong Jin, and Tianbao Yang. 2021. On stochastic moving-average estimators for non-convex optimization. arXiv:math.OC/2104.14840
[57] Zhishuai Guo, Zhuoning Yuan, Yan Yan, and Tianbao Yang. 2020. Fast objective and duality gap convergence for non-convex strongly-concave min-max problems. *CoRR* abs/2006.06889.
[58] Zheng-Chu Guo, Yiming Ying, and Ding-Xuan Zhou. 2017. Online regularized learning with pairwise loss functions. *Adv. Computat. Math.* 43, 1 (2017), 127–150.
[59] Mehmet Gönen. 2016. AUC maximization in Bayesian hierarchical models. In *European Association for Artificial Intelligence*. 21–27.
[60] Guang Han and Chunxia Zhao. 2010. AUC maximization linear classifier based on active learning and its application. *Neurocomputing* 73, 7–9 (Mar. 2010), 1272–1280. DOI: https://doi.org/10.1016/j.neucom.2010.01.001
[61] Peng Han, Shuo Shang, Aixin Sun, Peilin Zhao, Kai Zheng, and Panos Kalnis. 2019. AUC-MF: Point of interest recommendation with AUC maximization. In *IEEE 35th International Conference on Data Engineering (ICDE)*. 1558–1561. DOI: https://doi.org/10.1109/ICDE.2019.00141
[62] David J. Hand and Robert J. Till. 2001. A simple generalisation of the area under the ROC curve for multiple class classification problems. *Mach. Learn.* 45, 2 (Oct. 2001), 171–186. DOI: https://doi.org/10.1023/A:1010920819831
[63] James A. Hanley and Barbara J. McNeil. 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* 143, 1 (1982), 29–36.
[64] Peter Hase, Chaofan Chen, Oscar Li, and Cynthia Rudin. 2019. Interpretable image recognition with hierarchical prototypes. In *AAAI Conference on Human Computation and Crowdsourcing*. 32–40.
[65] Elad Hazan. 2019. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207*.
[66] Siyuan He, Pengcheng Xi, Ashkan Ebadi, Stephane Tremblay, and Alexander Wong. 2021. Performance or trust? Why not both. Deep AUC maximization with self-supervised learning for COVID-19 chest x-ray classifications. *arXiv preprint arXiv:2112.08363*.
[67] Michael Heath, Kevin Bowyer, Daniel Kopans, P. Kegelmeyer, Richard Moore, Kyong Chang, and S. Munishkumaran. 1998. Current status of the digital database for screening mammography. In *Digital Mammography*. Springer, 457–460.
[68] Ralf Herbrich, Thore Graepel, and Klause Obermayer. 1999. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*. The MIT Press, 115–132.
[69] A. Herschtal and B. Raskutti. 2004. Optimising area under the ROC curve using gradient descent. In *21st International Conference on Machine Learning*. ACM, 49.
[70] Junjie Hu, Haiqin Yang, Michael R. Lyu, Irwin King, and Anthony Man-Cho So. 2017. Online nonlinear AUC maximization for imbalanced data sets. *IEEE Trans. Neural Netw. Learn. Syst.* 29, 4 (2017), 882–895.
[71] Quanqi Hu, Yongjian Zhong, and Tianbao Yang. 2022. Multi-block min-max bilevel optimization with applications in multi-task deep AUC maximization. *CoRR* (2022).
[72] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*.
[73] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *CoRR* abs/2005.00687.
[74] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2020. Strategies for pre-training graph neural networks. In *7th International Conference on Learning Representations*.

[75] Feihu Huang, Shangqian Gao, Jian Pei, and Heng Huang. 2020. Accelerated zeroth-order momentum methods from mini to minimax optimization. *arXiv preprint arXiv:2008.08170*.
[76] Mengda Huang, Yang Liu, Xiang Ao, Kuan Li, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. 2022. AUC-oriented graph neural network for fraud detection. In *ACM Web Conference*. 1311–1321.
[77] Kyu-Baek Hwang, Beom-Yong Ha, Sanghun Ju, and Sangsoo Kim. 2013. Partial AUC maximization for essential gene prediction using genetic algorithms. *BMB Rep.* 46, 1 (2013), 41–46.
[78] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning*. PMLR, 448–456.
[79] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. 2019. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *AAAI Conference on Artificial Intelligence*. 590–597.
[80] Tomoharu Iwata, Akinori Fujino, and Naonori Ueda. 2020. Semi-supervised learning for maximizing the partial AUC. In *AAAI Conference on Artificial Intelligence*.
[81] Thorsten Joachims. 2005. A support vector method for multivariate performance measures. In *22nd International Conference on Machine Learning*. ACM, 377–384.
[82] Justin M. Johnson and Taghi M. Khoshgoftaar. 2019. Survey on deep learning with class imbalance. *J. Big Data* 6, 1 (2019), 27.
[83] Rie Johnson and Tong Zhang. 2013. Accelerating stochastic gradient descent using predictive variance reduction. *Adv. Neural Inf. Process. Syst.* 26 (2013), 315–323.
[84] Nathan Kallus and Angela Zhou. 2019. The fairness of risk scores beyond classification: Bipartite ranking and the xAuc metric. *Adv. Neural Inf. Process. Syst.* 32 (2019), 3438–3448.
[85] Purushottam Kar, Harikrishna Narasimhan, and Prateek Jain. 2014. Online and stochastic gradient methods for non-decomposable loss functions. In *Adv. Neural Inf. Process. Syst. 27th International Conference on Neural Information Processing Systems (NIPS'14)*. The MIT Press, Cambridge, MA, 694–702.
[86] Purushottam Kar, Bharath Sriperumbudur, Prateek Jain, and Harish Karnick. 2013. On the generalization ability of online learning algorithms for pairwise loss functions. In *30th International Conference on Machine Learning*.
[87] Majdi Khalid, Indrakshi Ray, and Hamidreza Chitsaz. 2018. Scalable nonlinear auc maximization methods. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 292–307.
[88] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
[89] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. 2020. WILDS: A benchmark of in-the-wild distribution shifts. DOI: https://doi.org/10.48550/ARXIV.2012.07421
[90] Osamu Komori and Shinto Eguchi. 2010. A boosting method for maximizing the partial area under the ROC curve. *BMC Bioinf.* 11 (2010), 314–314.
[91] Alex Krizhevsky. 2009. Learning multiple layers of features from tiny images. (2009), 32–33.
[92] Abhishek Kumar, Harikrishna Narasimhan, and Andrew Cotter. 2021. Implicit rate-constrained optimization of non-decomposable objectives. In *International Conference on Machine Learning*. PMLR, 5861–5871.
[93] George Kyriakides and Konstantinos G. Margaritis. 2020. An introduction to neural architecture search for convolutional networks. *CoRR* abs/2005.11074.
[94] Erin LeDell, Mark Laan, and Maya Peterson. 2016. AUC-maximizing ensembles through metalearning. *Int. J. Biostatist.* 12 (5 2016), 203–218. DOI: https://doi.org/10.1515/ijb-2015-0035
[95] Yunwen Lei, Antoine Ledent, and Marius Kloft. 2020. Sharper generalization bounds for pairwise learning. In *NeurIPS*.
[96] Yunwen Lei, Mingrui Liu, and Yiming Ying. 2021. Generalization guarantee of SGD for pairwise learning. In *35th Conference on Neural Information Processing Systems*.
[97] Yunwen Lei and Yiming Ying. 2021. Stochastic proximal AUC maximization. *J. Mach. Learn. Res.* 22, 61 (2021), 1–45.
[98] Nan Li, Rong Jin, and Zhi-Hua Zhou. 2014. Top rank optimization in linear time. In *Conference on Neural Information Processing Systems*.
[99] Zhou Ligang, Kin Keung Lai, and Jerome Yen. 2009. Credit scoring models with AUC maximization based on weighted SVM. *Int. J. Inf. Technol. Decis. Mak.* 8 (12 2009), 677–696. DOI: https://doi.org/10.1142/S0219622009003582
[100] Tianyi Lin, Chi Jin, and Michael I. Jordan. 2020. On gradient descent ascent for nonconvex-concave minimax problems. In *37th International Conference on Machine Learning (ICML)*. 6083–6093.
[101] Mingrui Liu, Zhuoning Yuan, Yiming Ying, and Tianbao Yang. 2020. Stochastic AUC maximization with deep neural networks. In *8th International Conference on Learning Representations (ICLR)*.
[102] Mingrui Liu, Xiaoxuan Zhang, Zaiyi Chen, Xiaoyu Wang, and Tianbao Yang. 2018. Fast stochastic AUC maximization with $O(1/n)$-convergence rate. In *International Conference on Machine Learning*. 3189–3197.

[103] Xin Liu, Zhisong Pan, Haimin Yang, Xingyu Zhou, Wei Bai, and Xianghua Niu. 2019. An adaptive moment estimation method for online AUC maximization. *PLoS One* 14, 4 (2019), e0215426.
[104] Zhenqiu Liu and Terry Hyslop. 2010. Partial AUC for differentiated gene detection. In *IEEE International Conference on BioInformatics and BioEngineering*. IEEE, 310–311.
[105] Phil Long and Rocco Servedio. 2007. Boosting the area under the ROC curve. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc.
[106] Xiaofen Lu, Ke Tang, and Xin Yao. 2010. Evolving neural networks with maximum AUC for imbalanced data classification. In *5th International Conference on Hybrid Artificial Intelligence Systems (HAIS'10)*. Springer-Verlag, Berlin, 335–342.
[107] Luo Luo, Haishan Ye, Zhichao Huang, and Tong Zhang. 2020. Stochastic recursive gradient descent ascent for stochastic nonconvex-strongly-concave minimax problems. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[108] Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. 2009. Identifying suspicious URLs: An application of large-scale online learning. In *26th Annual International Conference on Machine Learning*. 681–688.
[109] Andreas Maurer and Massimiliano Pontil. 2020. Estimating weighted areas under the ROC curve. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 7733–7742.
[110] Donna Katzman McClish. 1989. Analyzing a portion of the ROC curve. *Med. Decis. Mak.* 9, 3 (1989), 190–195. DOI: https://doi.org/10.1177/0272989X8900900307
[111] H. Brendan McMahan. 2017. A survey of algorithms and analysis for adaptive online learning. *J. Mach. Learn. Res.* 18, 1 (2017), 3117–3166.
[112] H. Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: A view from the trenches. In *19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1222–1230.
[113] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2019. A survey on bias and fairness in machine learning. *CoRR* abs/1908.09635.
[114] Michalis P. Michaelides and Christos G. Panayiotou. 2009. SNAP: Fault tolerant event location estimation in sensor networks using binary data. *IEEE Trans. Comput.* 58, 9 (2009), 1185–1197.
[115] Arlo J. Miller and Martin C. Mihm Jr. 2006. Melanoma. *New Eng. J. Med.* 355, 1 (2006), 51–65.
[116] Mikio Morii, Shiro Ikeda, Nozomu Tominaga, Masaomi Tanaka, Tomoki Morokuma, Katsuhiko Ishiguro, J. Yamato, Naonori Ueda, Nao Suzuki, Naoki Yasuda, and Naoki Yoshida. 2016. Machine-learning selection of optical transients in the subaru/hyper suprime-cam survey. *Publicat. Astron. Societ. Japan* 68 (2016), 104.
[117] Harikrishna Narasimhan and Shivani Agarwal. 2013. A structural SVM based approach for optimizing partial AUC. In *30th International Conference on Machine Learning*. PMLR, 516–524.
[118] Harikrishna Narasimhan and Shivani Agarwal. 2013. SVMpAUCtight: A new support vector method for optimizing partial AUC based on a tight convex upper bound. In *19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*. Association for Computing Machinery, New York, NY, 167–175. DOI: https://doi.org/10.1145/2487575.2487674
[119] Harikrishna Narasimhan and Shivani Agarwal. 2017. Support vector algorithms for optimizing the partial area under the ROC curve. *Neural Computat.* 29 (2017), 1919–1963.
[120] Harikrishna Narasimhan, Andrew Cotter, Yichen Zhou, Serena Wang, and Wenshuo Guo. 2020. Approximate heavily-constrained learning with lagrange multiplier models. *Adv. Neural Inf. Process. Syst.* 33 (2020), 8693–8703.
[121] Michael Natole, Yiming Ying, and Siwei Lyu. 2018. Stochastic proximal algorithms for AUC maximization. In *International Conference on Machine Learning*. 3710–3719.
[122] Michael Natole Jr. 2020. *Fast Optimization Algorithms for AUC Maximization*. Ph.D. Dissertation. State University of New York.
[123] Michael Natole Jr, Yiming Ying, and Siwei Lyu. 2019. Stochastic AUC optimization algorithms with linear convergence. *Front. Appl. Math. Statist.* 5 (2019), 30.
[124] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Stochastic approximation approach to stochastic programming. In *SIAM J. Optim.* Citeseer.
[125] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. 2009. Robust stochastic approximation approach to stochastic programming. *SIAM J. Optim.* 19, 4 (Jan. 2009), 1574–1609. DOI: https://doi.org/10.1137/070704277
[126] Yurii Nesterov. 1983. A method for unconstrained convex minimization problem with the rate of convergence O (1/kˆ 2). In *Doklady an ussr*, Vol. 269. 543–547.
[127] Yu Nesterov. 2005. Smooth minimization of non-smooth functions. *Math. Program.* 103, 1 (2005), 127–152.
[128] Behnam Neyshabur. 2017. Implicit regularization in deep learning. *CoRR* abs/1709.01953.
[129] Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takac. 2017. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *34th International Conference on Machine Learning*. 2613–2621.

[130] Matthew Norton and Stan Uryasev. 2018. Maximization of AUC and buffered AUC in binary classification. *Math. Program.* 174 (7 2018), 1–38. DOIhttps://doi.org/10.1007/s10107-018-1312-2
[131] Francesco Orabona. 2019. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*.
[132] Tapio Pahikkala, Antti Airola, Hanna Suominen, Jorma Boberg, and Tapio Salakoski. 2008. Efficient AUC maximization with regularized least-squares. In *Search-oriented Conversational AI Conference*.
[133] Sakrapee Paisitkriangkrai, Chunhua Shen, and Anton Van Den Hengel. 2013. Efficient pedestrian detection by directly optimizing the partial area under the ROC curve. In *IEEE International Conference on Computer Vision*. 1057–1064.
[134] Balamurugan Palaniappan and Francis R. Bach. 2016. Stochastic variance reduction methods for saddle-point problems. In *Annual Conference on Neural Information Processing Systems*. 1408–1416.
[135] Michael James David Powell et al. 1981. *Approximation Theory and Methods*. Cambridge University Press.
[136] Hassan Rafique, Mingrui Liu, Qihang Lin, and Tianbao Yang. 2020. Non-convex min-max optimization: Provable algorithms and applications in machine learning. *Optim. Meth. Softw.* (2020).
[137] Ali Rahimi, Benjamin Recht, et al. 2007. Random features for large-scale kernel machines. In *Conference on Neural Information Processing Systems*.
[138] Alain Rakotomamonjy. 2004. *Support Vector Machines and Area under ROC Curves*. Technical Report.
[139] Alain Rakotomamonjy. 2012. Sparse support vector infinite push. In *International Conference on Machine Learning*.
[140] Ke Ren, Haichuan Yang, Yu Zhao, Mingshan Xue, Hongyu Miao, Shuai Huang, and Ji Liu. 2018. A robust AUC maximization framework with simultaneous outlier detection and feature selection for positive-unlabeled classification. *CoRR* abs/1803.06604.
[141] Maria Teresa Ricamato and Francesco Tortorella. 2011. Partial AUC maximization in a linear combination of dichotomizers. *Pattern Recog.* 44 (2011), 2669–2677.
[142] Cynthia Rudin. 2009. The p-norm push: A simple convex ranking algorithm that concentrates at the top of the list. *J. Mach. Learn. Res.* 10, 78 (2009), 2233–2271.
[143] Tomoya Sakai, Gang Niu, and Masashi Sugiyama. 2018. Semi-supervised AUC optimization based on positive-unlabeled learning. *Mach. Learn.* 107, 4 (2018), 767–794. DOI: https://doi.org/10.1007/s10994-017-5678-9
[144] Robert E. Schapire and Yoram Singer. 1998. Improved boosting algorithms using confidence-rated predictions. In *11th Annual Conference on Computational Learning Theory (COLT'98)*. Association for Computing Machinery, New York, NY, 80–91. DOI: https://doi.org/10.1145/279943.279960
[145] Kathryn Schutte, Olivier Moindrot, Paul Hérent, Jean-Baptiste Schiratti, and Simon Jégou. 2021. Using StyleGAN for visual interpretability of deep learning models on medical images. arXiv:eess.IV/2101.07563.
[146] Laleh Seyyed-Kalantari, Guanxiong Liu, Matthew B. A. McDermott, and Marzyeh Ghassemi. 2020. CheXclusion: Fairness gaps in deep chest X-ray classifiers. *CoRR* abs/2003.00827.
[147] Shai Shalev-Shwartz et al. 2011. Online learning and online convex optimization. *Found. Trends Mach. Learn.* 4, 2 (2011), 107–194.
[148] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczynski. 2014. *Lectures on Stochastic Programming: Modeling and Theory, Second Edition*. Society for Industrial and Applied Mathematics.
[149] Wanli Shi, Bin Gu, Xiang Li, and Heng Huang. 2020. Quadruply stochastic gradient method for large scale nonlinear semi-supervised ordinal regression AUC optimization. In *AAAI Conference on Artificial Intelligence*. 5734–5741.
[150] Dongjin Song and David A. Meyer. 2015. Recommending positive links in signed social networks by optimizing a generalized AUC. In *29th AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 290–296.
[151] Hari Sowrirajan, Jingbo Yang, Andrew Y. Ng, and Pranav Rajpurkar. 2020. MoCo pretraining improves representation and transferability of chest x-ray models. *CoRR* abs/2010.05352.
[152] Harald Steck. 2007. Hinge rank loss and the area under the ROC curve. In *European Conference on Machine Learning*.
[153] J. Sulam, R. Ben-Ari, and P. Kisilev. 2017. Maximizing AUC with deep learning for classification of imbalanced mammogram datasets. In *Eurographics Workshop on Visual Computing for Biology and Medicine (VCBM'17)*. Eurographics Association, Goslar, DEU, 131–135.
[154] Jeremias Sulam, Rami Ben-Ari, and Pavel Kisilev. 2017. Maximizing AUC with deep learning for classification of imbalanced mammogram datasets. In *Conference on Visual Computing for Biology and Medicine*. 131–135.
[155] Kaizhao Sun and Xu Andy Sun. 2021. Algorithms for difference-of-convex (DC) programs based on difference-of-Moreau-envelopes smoothing. *arXiv preprint arXiv:2104.01470*.
[156] Balázs Szörényi, Snir Cohen, and Shie Mannor. 2017. Non-parametric online AUC maximization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 575–590.
[157] Takashi Takenouchi, Osamu Komori, and Shinto Eguchi. 2012. An extension of the receiver operating characteristic curve and AUC-optimal classification. *Neural Computat.* 24 (6 2012), 2789–824.
[158] Loring W. Tu. 2011. An introduction to manifolds. second. New York, US: Springer (2011).

[159] Naonori Ueda and Akinori Fujino. 2018. Partial AUC maximization via nonlinear scoring functions. *ArXiv* abs/1806.04838.
[160] Bastiaan S. Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. 2018. Rotation equivariant CNNs for digital pathology. In *International Conference on Medical Image Computing and Computer-assisted Intervention*. Springer, 210–218.
[161] Jeffrey S. Vitter. 1985. Random sampling with a reservoir. *ACM Trans. Math. Softw.* 11, 1 (1985), 37–57.
[162] Willem Waegeman and Bernard De Baets. 2011. A survey on ROC-based ordinal regression. *Pref. Learn.* (1 2011). DOI: https://doi.org/10.1007/978-3-642-14125-6_7
[163] De Wang, Danesh Irani, and Calton Pu. 2012. Evolutionary study of web spam: Webb spam corpus 2011 versus Webb spam corpus 2006. In *8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom)*. IEEE, 40–49.
[164] Mengdi Wang, Ethan X. Fang, and Han Liu. 2017. Stochastic compositional gradient descent: Algorithms for minimizing compositions of expected-value functions. *Math. Program.* 161, 1–2 (2017), 419–449.
[165] Shijun Wang, Diana Li, Nicholas Petrick, Berkman Sahiner, Marius George Linguraru, and Ronald Summers. 2015. Optimizing area under the ROC curve using semi-supervised learning. *Pattern Recog.* 48 (1 2015), 276–287.
[166] Sheng Wang, Jianzhu Ma, and Jinbo Xu. 2016. AUCpreD: Proteome-level protein disorder prediction by AUC-maximized deep convolutional neural fields. *Bioinformatics* 32, 17 (8 2016), i672–i679. DOI: https://doi.org/10.1093/bioinformatics/btw446
[167] Sheng Wang, Siqi Sun, and Jinbo Xu. 2016. AUC-maximized deep convolutional neural fields for protein sequence labeling. In *ECML/PKDD (2) (Lecture Notes in Computer Science)*, Vol. 9852. Springer, 1–16.
[168] Yuyang Wang, Roni Khardon, Dmitry Pechyony, and Rosie Jones. 2012. Generalization bounds for online learning algorithms with pairwise loss functions. In *Conference on Learning Theory*. JMLR Workshop and Conference Proceedings, 13–1.
[169] Yuyang Wang, Roni Khardon, Dmitry Pechyony, and Rosie Jones. 2013. Online learning with pairwise loss functions. *arXiv preprint arXiv:1301.5332*.
[170] Zhu Wang and Y.-C. I. Chang. 2011. Marker selection via maximizing the partial area under the ROC curve of linear risk scores. *Biostatistics* 12 2 (2011), 369–85.
[171] Zhengyang Wang, Meng Liu, Youzhi Luo, Zhao Xu, Yaochen Xie, Limei Wang, Lei Cai, Qi Qi, Zhuoning Yuan, Tianbao Yang, and Shuiwang Ji. 2020. Advanced graph and sequence neural networks for molecular property prediction and drug discovery. *arXiv preprint arXiv:2012.01981*.
[172] Christopher Williams and Matthias Seeger. 2001. Using the Nyström method to speed up kernel machines. In *14th Annual Conference on Neural Information Processing Systems*. 682–688.
[173] Shan-Hung Wu, Keng-Pei Lin, Chung-Min Chen, and Ming-Syan Chen. 2008. Asymmetric support vector machines: Low false-positive learning under the user tolerance. In *14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08)*. Association for Computing Machinery, 749–757.
[174] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. 2018. MoleculeNet: A benchmark for molecular machine learning. *Chem. Sci.* 9, 2 (2018), 513–530.
[175] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? In *7th International Conference on Learning Representations*.
[176] Akihiro Yamaguchi, Shigeru Maya, Kohei Maruchi, and Ken Ueno. 2020. LTSpAUC: Learning time-series shapelets for partial AUC maximization. *Big Data* 8, 5 (2020), 391–411.
[177] Akihiro Yamaguchi, Shigeru Maya, Kohei Maruchi, and Ken Ueno. 2020. LTSpAUC: Learning time-series shapelets for partial AUC maximization. *Big Data* 8 (10 2020), 391–411. DOI: https://doi.org/10.1089/big.2020.0069
[178] Lian Yan, Robert Dodier, Michael C. Mozer, and Richard Wolniewicz. 2003. Optimizing classifier performance via an approximation to the Wilcoxon-Mann-Whitney statistic. In *International Conference on Machine Learning*. 848–855.
[179] Yan Yan, Yi Xu, Qihang Lin, Wei Liu, and Tianbao Yang. 2020. Optimal epoch stochastic gradient descent ascent methods for min-max optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[180] Yan Yan, Yi Xu, Qihang Lin, Lijun Zhang, and Tianbao Yang. 2019. Stochastic primal-dual algorithms with faster convergence than $O(1/\sqrt{T})$ for problems without bilinear structure. *CoRR* abs/1904.10112.
[181] Hanfang Yang, Kun Lu, Xiang Lyu, and Feifang Hu. 2019. Two-way partial AUC and its properties. *Statist. Meth. Med. Res.* 28, 1 (2019), 184–195. DOI: https://doi.org/10.1177/0962280217718866
[182] Junchi Yang, Negar Kiyavash, and Niao He. 2020. Global convergence and variance reduction for a class of nonconvex-nonconcave minimax problems. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[183] Junchi Yang, Antonio Orvieto, Aurélien Lucchi, and Niao He. 2021. Faster single-loop algorithms for minimax optimization without strong concavity. *CoRR* abs/2112.05604.
[184] Zhenhuan Yang, Yunwen Lei, Puyu Wang, Tianbao Yang, and Yiming Ying. 2021. Simple stochastic and online gradient descent algorithms for pairwise learning. In *Advances in Neural Information Processing Systems*.

[185] Zhenhuan Yang, Wei Shen, Yiming Ying, and Xiaoming Yuan. 2020. Stochastic AUC optimization with general loss. *Commun. Pure Appl. Anal.* 19, 8 (2020).
[186] Z. Yang, Q. Xu, S. Bao, X. Cao, and Q. Huang. 5555. Learning with multiclass AUC: Theory and algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.* 1 (July 5555), 1–1. DOI: https://doi.org/10.1109/TPAMI.2021.3101125
[187] Zhiyong Yang, Qianqian Xu, Shilong Bao, Yuan He, Xiaochun Cao, and Qingming Huang. 2021. When all we need is a piece of the pie: A generic framework for optimizing two-way partial AUC. In *38th International Conference on Machine Learning*. 11820–11829.
[188] Zhiyong Yang, Taohong Zhang, Jingcheng Lu, Dezheng Zhang, and Dorothy Kalui. 2017. Optimizing area under the ROC curve via extreme learning machines. *Knowl.-based Syst.* 130 (2017), 74–89.
[189] Zhenhuan Yang, Baojian Zhou, Yunwen Lei, and Yiming Ying. 2020. Stochastic hard thresholding algorithms for AUC maximization. In *IEEE International Conference on Data Mining (ICDM)*. IEEE, 741–750.
[190] Yao Yao, Qihang Lin, and Tianbao Yang. 2022. Large-scale optimization of partial AUC in a range of false positive rates. *arXiv preprint arXiv:2203.01505*.
[191] Nanyang Ye, Kaican Li, Haoyue Bai, Runpeng Yu, Lanqing Hong, Fengwei Zhou, Zhenguo Li, and Jun Zhu. 2021. OoD-Bench: Quantifying and understanding two dimensions of out-of-uistribution generalization.
[192] Yiming Ying, Longyin Wen, and Siwei Lyu. 2016. Stochastic online AUC maximization. In *Conference on Advances in Neural Information Processing Systems*. 451–459.
[193] Yiming Ying and Ding-Xuan Zhou. 2016. Online pairwise learning algorithms. *Neural Computat.* 28, 4 (2016), 743–777.
[194] Zhuoning Yuan, Zhishuai Guo, Nitesh Chawla, and Tianbao Yang. 2022. Compositional training for end-to-end deep AUC maximization. In *International Conference on Learning Representations*.
[195] Zhuoning Yuan, Zhishuai Guo, Yi Xu, Yiming Ying, and Tianbao Yang. 2021. Federated deep AUC maximization for hetergeneous data with a constant communication complexity. In *38th International Conference on Machine Learning*. PMLR, 12219–12229.
[196] Zhuoning Yuan, Yan Yan, Milan Sonka, and Tianbao Yang. 2020. Robust deep AUC maximization: A new surrogate loss and empirical studies on medical image classification. In *International Conference on Computer Vision*. arXiv:2012.03173.
[197] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.
[198] X. Zhang, A. Saha, and S. V. N. Vishwanathan. 2012. Smoothing multivariate performance measures. *J. Mach. Learn. Res.* 13 (2012), 3623–3680.
[199] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D. Manning, and Curtis P. Langlotz. 2020. Contrastive learning of medical visual representations from paired images and text. *CoRR* abs/2010.00747.
[200] Yuchen Zhang and Xiao Lin. 2015. Stochastic primal-dual coordinate method for regularized empirical risk minimization. In *International Conference on Machine Learning*. PMLR, 353–361.
[201] Peilin Zhao, Steven C. H. Hoi, Rong Jin, and Tianbao Yang. 2011. Online AUC maximization. In *International Conference on Machine Learning*. 233–240.
[202] Renbo Zhao. 2020. A primal dual smoothing framework for max-structured nonconvex optimization. arXiv:math.OC/2003.04375.
[203] Baojian Zhou, Yiming Ying, and Steven Skiena. 2020. Online AUC optimization for sparse high-dimensional datasets. In *IEEE International Conference on Data Mining (ICDM)*. IEEE, 881–890.
[204] Ligang Zhou, Kin Keung Lai, and Jerome Yen. 2009. Credit scoring models with AUC maximization based on weighted SVM. *Int. J. Inf. Technol. Decis. Mak.* 8, 4 (2009), 677–696.
[205] Yajin Zhou and Xuxian Jiang. 2012. Dissecting Android malware: Characterization and evolution. In *IEEE Symposium on Security and Privacy*. IEEE, 95–109.
[206] Dixian Zhu, Gang Li, Bokun Wang, Xiaodong Wu, and Tianbao Yang. 2022. When AUC meets DRO: Optimizing partial AUC for deep learning with non-convex convergence guarantee. In *International Conference on Machine Learning*.
[207] Dixian Zhu, Xiaodong Wu, and Tianbao Yang. 2022. Benchmarking deep AUROC optimization: Loss functions and algorithmic choices. *arXiv preprint* (2022).
[208] Lin Zhu, Hong-Bo Zhang, and De-Shuang Huang. 2017. Direct AUC optimization of regulatory motifs. *Bioinformatics* 33, 14 (7 2017), i243–i251.

Received 28 March 2022; revised 15 July 2022; accepted 25 July 2022