Learning With Multiclass AUC: Theory and Algorithms

Zhiyong Yang, Qianqian Xu, Senior Member, IEEE, Shilong Bao, Xiaochun Cao, Senior Member, IEEE, and Qingming Huang, Fellow, IEEE

Abstract—The Area under the ROC curve (AUC) is a well-known ranking metric for problems such as imbalanced learning and recommender systems. The vast majority of existing AUC-optimization-based machine learning methods only focus on binary-class cases, while leaving the multiclass cases unconsidered. In this paper, we start an early trial to consider the problem of learning multiclass scoring functions via optimizing multiclass AUC metrics. Our foundation is based on the M metric, which is a well-known multiclass extension of AUC. We first pay a revisit to this metric, showing that it could eliminate the imbalance issue from the minority class pairs. Motivated by this, we propose an empirical surrogate risk minimization framework to approximately optimize the M metric. Theoretically, we show that: (i) optimizing most of the popular differentiable surrogate losses suffices to reach the Bayes optimal scoring function asymptotically; (ii) the training framework enjoys an imbalance-aware generalization error bound, which pays more attention to the bottleneck samples of minority classes compared with the traditional $O(\sqrt{1/N})$ result. Practically, to deal with the low scalability of the computational operations, we propose acceleration methods for three popular surrogate loss functions, including the exponential loss, squared loss, and hinge loss, to speed up loss and gradient evaluations. Finally, experimental results on 11 real-world datasets demonstrate the effectiveness of our proposed framework. The code is now available at https://github.com/joshuaas/Learning-with-Multiclass-AUC-Theory-and-Algorithms.

Index Terms—AUC optimization, machine learning

1 INTRODUCTION

AUC (Area Under the ROC Curve), which measures the probability that a positive instance has a higher score than a negative instance, is a well-known performance metric for a scoring function’s ranking quality. AUC often comes up as a more appropriate performance metric than accuracy in various applications due to its appealing properties, e.g., insensitivity toward label distributions and costs [20], [30].

Zhiyong Yang is with the School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing, 101408, China. E-mail: joshua.zhiyongyang@gmail.com.

Qianqian Xu is with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China. E-mail: xuqianqian@ict.ac.cn.

Shilong Bao is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: baoshilong@iie.ac.cn.

Xiaochun Cao is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China, and also with the School of Cyber Science and Technology, Sun Yat-sen University, Shenzhen 518100, China. E-mail: caoxiaochun@ieee.org.cn.

Qingming Huang is with the School of Computer Science and Technology, Key Laboratory of Big Data Mining and Knowledge Management (BDKM), University of Chinese Academy of Sciences, Beijing 101408, China, and with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with the Peng Cheng Laboratory, Shenzhen 518055, China. E-mail: qmhuang@ucas.ac.cn.

Manuscript received 8 Oct. 2020; revised 22 Apr. 2021; accepted 23 July 2021.

On one hand, for class-imbalanced tasks such as disease prediction [32], [78] and rare event detection [43], [47], [72], the label distribution is often highly skewed in the sense that the proportion of the majority classes instances significantly dominates the others. As for a typical instance, in the credit card fraud detection dataset released in Kaggle, the fraudulent transactions only account for 0.172% of the total records. Accuracy is often not a good choice in this case since it might ignore the performance from the minority classes which are often more crucial than the majority ones. By contrast, the value of AUC does not rest on the label distribution, making it a natural metric under the class-imbalance scenario. On the other hand, AUC has also been adopted as a standard metric for applications such as ads click-through rate prediction and recommender systems [10], [17], [19], [60] where pursuing a correct ranking between positive and negative instances is much more critical than label prediction.

Over the past two decades, the importance of AUC has raised an increasing favor in the machine learning community to explore direct AUC optimization methods, e.g., [3], [8], [23], [35], [36], [54], [57]. However, to our knowledge, the vast majority of related studies merely focus on the binary class scenario. Since real-world pattern recognition problems often involve more than two classes, it is natural to pursue its generalization in the world of multiclass problems. To this end, we present a very early study to the problem of AUC guided machine learning framework under the multiclass setting. Specifically, we propose a universal empirical surrogate risk minimization framework with theoretical guarantees.

1. https://www.kaggle.com/mlg-ulb/creditcardfraud
First of all, we provide a review of a multiclass generalization of AUC known as the M metric [30]. Specifically, we show that M metric is an appropriate extension of binary AUC in the sense that it could efficiently avoid the imbalanced issue across class ranking pairs. Motivated by this result, we propose to minimize the 0-1 mis-ranking loss induced by the M metric denoted as MAUC\(^1\). However, directly minimizing the objective is intractable. This intractability has three sources: (a) the 0-1 mis-ranking loss MAUC\(^1\) is a discrete and non-differentiable function, making it impossible to perform efficient optimizations; (b) the data distribution is not a known priori, making the calculation of expectation unavailable; (c) the complexity to estimate the loss and gradient function could be as high as \(O(N^2N_C^2)\) in the worst case, where \(N\) is the number of samples and \(N_C\) is the number of classes.

Targeting at (a), in Section 4, we investigate how to construct differentiable surrogate risks as replacements for the 0-1 risk. To do this, we derive the set of Bayes optimal score functions under the MAUC\(^1\)-criterion. We then provide a general sufficient condition for the fisher consistency. The condition suggests that a large set of popular surrogate loss functions are fisher consistent under certain assumptions in the sense that optimizing the corresponding surrogate expected risk also leads to the Bayes optimal score functions.

Based on the consistency result, in Section 5, we construct an empirical surrogate risk minimization framework against (b), where we propose an unbiased empirical estimation of the surrogate risks over a training dataset as the objective function to avoid using population-level expectation directly. Moreover, we provide a systematic analysis of its generalization ability. The major challenge here is that the empirical risk function could not be decomposed as a finite sum of independent instance-wise loss terms, making the traditional symmetrization technique not available. To fix this problem, we provide a novel form of Rademacher complexity for MAUC\(^1\). Furthermore, we provide generalization upper bounds for a wide range of model classes, including shallow and deep models. The results consistently enjoy an imbalance-aware property, which pays more attention to the bottleneck samples of minority classes than the traditional result.

In Section 6, (c) is attacked by novel acceleration algorithms to speed up loss and gradient evaluations which are the fundamental calculations for gradient-based optimization methods. The mini-batch/full-batch loss and gradient evaluations could be done, with the proposed algorithms, in a complexity comparable with that for the ordinary instance-wise loss functions.

Finally, in Section 7, we perform extensive experiments on 11 real-world datasets to validate our proposed framework.

To sum up, our contribution is three-fold:

- **New Framework:** We provide a theoretical framework for empirical risk minimization under the guidance of the M metric.
- **Theoretical Guarantees:** From a theoretical perspective, our framework is soundly supported by consistency analysis and generalization analysis.
- **Fast Algorithms:** From a practical perspective, we propose efficient algorithms for three surrogate losses to accelerate loss and gradient evaluations. The experiments show that the acceleration ratio could reach 1,0000+ confronting medium size datasets.

## 2 Related Work

### Binary Class AUC Optimization

As a motivating early study, [15] points out that maximizing AUC should not be replaced with minimizing the error rate, which shows the necessity to study direct AUC optimization methods. After that, a series of algorithms have been designed for optimizing AUC. At the early stage, the majority of studies focus on a full-batch off-line setting, [3], [8] optimize AUC based on a logistic surrogate loss function and ordinary gradient descent method. RankBoost [22] provides an efficient ensemble-based AUC learning method based on a ranking extension of the AdaBoost algorithm. [36], [76] constructs SVM\(^{struct}\)-based frameworks that optimize a direct upper bound of the 0 – 1 loss version AUC metric instead of its surrogates. Later on, to accommodate big data analysis, researchers start to explore online extensions of AUC optimization methods. [77] provides an early trial for this direction based on the reservoir sampling technique. [23] provides a completely one-pass AUC optimization method for streaming data based on the squared surrogate loss. [75] reformulates the squared-loss-based stochastic AUC maximization problem as a stochastic saddle point problem. The new saddle point problem’s objective function only involves summations of instance-wise loss terms, which significantly reduces the burden from the pairwise formulation. [58], [59] further accelerate this framework with tighter convergence rates. On top of the reformulation framework, [74] also provides an acceleration framework for general loss functions where the loss functions are approximated by the Bernstein polynomials. [18] proposes a novel large-scale nonlinear AUC maximization method based on the triply stochastic gradient descent algorithm. [28] proposes a scalable and efficient adaptive doubly stochastic gradient algorithm for generalized regularized pairwise learning problems. Beyond optimization methods, a substantial amount of researches also provide theoretical support for this learning framework from different dimensions, including generalization analysis [2], [12], [63], [69], [70] and consistency analysis [1], [25]. Last but not least, there are also some studies focusing on optimizing the partial area under the ROC curve [4], [55], [56]. This paper takes a further step by providing an early study of the theory and algorithms for AUC-guided machine learning under the more complicated multiclass scenario.

### Multiclass AUC Metrics

There exist two general ideas for how to define a multiclass AUC metric. The first idea is on top of the belief that multiclass counterparts of the ROC curve should be represented as higher-dimensional surfaces. As a result, AUC is generalized naturally to the volume Under some specifically designed ROC Surfaces (VUS) [21], [53]. However, this idea is restricted by its high complexity to calculate the volume of such high-dimensional spaces. According to [38], calculating VUS for \(N\) samples and \(N_C\) classes requires \(O(N\log N + N^{[N_C/2]}\) time complexity and \(O(N^{[N_C/2]}\) space complexity. Another idea then comes out to do it in a much simpler manner, which suggests that one can simply take an average of pairwise binary AUCs [30],
This is based on the intuition that if every pair of classes is well-separated from each other in distribution, one can get reasonable performances. Getting rid of calculations on high dimensional spaces renders this formulation a low complexity. Due to its simplicity, the M metric proposed in the representative work [30] has been adopted by a series of popular machine learning software such as scikit-learn in Python and pROC in R. Most recently, [71] also considers online extensions of multi-class AUC metric to deal with the concept drift issue for streaming data. Unlike this line of research, our main focus in this paper is how to learn valuable models from a proper multi-class extension of the AUC metric.

**AUC Optimization for Multiclass AUC Optimization.** There are few studies that focus on AUC optimization algorithms for multiclass problems, which fall in the following two directions. The first direction of studies focuses on the multipartite ranking problem, which is a natural extension of the bipartite ranking problem where the order is presented with more than two discrete degrees. Hitherto, a substantial amount of efforts have been made to explore the AUC optimization method/theory for the multipartite ranking problem [11], [13], [24], [62], [68]. Moreover, a recent work [65] proposes a novel nonlinear semi-supervised multipartite ranking problem for large-scale datasets. It is noteworthy that multipartite ranking approaches could solve multiclass problems only if the classes are ordinal values for the same semantic concept. For example, the age estimation task could be regarded as a multiclass problem where the class labels are the ages for a given person; the movie rating prediction could be regarded as a multiclass problem where the class labels are the ratings for the same given movie. The major difference here is that we focus on the generic multiclass problems where different labels present different semantic concepts and do not have clear ordinal relations. Thus the studies along this line are not available to our setting in general. Most recently, [44] discusses the possibility of exploring AUC optimization for general multipartite settings. The major differences are as follows. First, [44] only focuses on the square loss function, while our study presents a general framework for multiclass AUC optimization. Second, [44] focuses more on the optimization properties, where the original problem is reformulated as a minimax problem. By contrast, our study focuses on the learning properties for multiclass AUC optimization, such as its generalization ability and the consistency of different loss functions. Moreover, we also present a series of acceleration methods that do not require any reformulation of the optimization problem.

### 3 Learning With Multiclass AUC Metrics: An Overview

#### 3.1 Preliminary

**Basic Notations.** In this paper, we will constantly adopt two sets of events: for pair-wise AUC metrics, $E(i,j)$ denotes the event $y_1 = i, y_2 = j \implies y_1 = y_2 = i$, while $E(i)$ denotes the event $y_1 = i, y_2 \neq i \implies y_1 \neq y_2 = i$. Given an event $A$, $I[A]$ is the indicator function associated with this event, which equals 1 if $A$ holds and equals to 0 otherwise. Given a finite dataset $S$, we denote $N_C$ as the number of classes and $N$ as the total number of sample points in the dataset.

**Settings.** For an $N_C$ class problem, we assume that our samples are drawn from a product space $Z = \mathcal{X} \times \mathcal{Y}$. $\mathcal{X}$ is the input feature where $\mathcal{X} \subseteq \mathbb{R}^d$ and $d$ is the input dimensionality. $\mathcal{Y}$ is the label space $\{N_C\}$. Given a label $y_m = i$, we will use the one-hot vector $y_m = [y_m^0, \ldots, y_m^{N_C}]$ to represent it, with $y_m^0 = 0, k \neq i$, and $y_m^i = 1$ if $k = i$. In this paper, we will adopt the one vs. all decomposition [52, Chap.9.4] of the multiclass problem. Under this context, an $N_C$-class ($N_C > 2$) scoring function refers to a set of $N_C$ functions $f = (f^{(1)}, \ldots, f^{(N_C)})$, where $f^{(i)} : \mathcal{X} \rightarrow \mathbb{R}$ serves as a continuous score function supporting $y = i$.

**Binary Class AUC.** Let $D_2$ denote the joint data distribution and $f$ denote a score function estimating the possibility that an underlying instance belongs to the positive class. In the context of binary class problems, AUC is known to have a clear statistical meaning: it is equivalent to the Wilcoxon Statistics [31], namely the possibility that correct ranking takes place between a random paired samples with distinct labels: $\text{AUC}(f) = \mathbb{E}[\text{I}[\Delta(y)\Delta(f) > 0]\mathbb{E}^{(i,j)}] + \mathbb{E}[\text{I}[\Delta(f) = 0]\mathbb{E}^{(i,j)}] = \mathbb{E}_{Z_1,Z_2,Z_1 \neq Z_2} I[\Delta(y)\Delta(f) > 0] + \frac{1}{2} I[\Delta(f) = 0]\mathbb{E}^{(i,j)}$, where $\Delta(y) = y_1 - y_2$, $\Delta(f) = f(x_1) - f(x_2)$. Note that we adopt the convention [1], [12], [25] to score ties with 0.5. See basic notations in the Section 3.1 for $E^{(i,j)}$. Hereafter, our calculations on AUC follows this definition.

### 3.2 Motivation

First of all, we start our study by finding a proper multiclass metric from the existing literature. As presented in Section 2, the main idea to construct multiclass AUC is to express it as an average of binary class AUCs. Just like the way multiclass classification is transformed into a set of binary classification problems, one can derive multiclass AUC metrics from the following two regimes.

**One versus All Regime (ova).** Given an ova score function $f = (f^{(1)}, \ldots, f^{(N_C)})$, [61] suggests that one can construct a pairwise AUC score for each $f^{(i)}$. Here, the positive instances are drawn from the $i$th class, and the negative instances are drawn from the $j$th distribution conditioned on $j \neq i$. The overall AUC score $\text{AUC}^{\text{ova}}$ is then an average of all $N_C$ pairwise scores, which is defined as follows. Note that here we adopt an equally-weighted average to avoid the imbalance issue.

$$\text{AUC}^{\text{ova}}(f) = \frac{1}{N_C} \sum_{i=1}^{N_C} \text{AUC}_{i \rightarrow a}(f^{(i)}),$$

where $\text{AUC}_{i \rightarrow a}(f^{(i)}) = \mathbb{E}_{Z_1,Z_2} I[\Delta(y^{(i)})\Delta(f^{(i)}) > 0]\mathbb{E}^{(i,j)} + \frac{1}{2} I[\Delta(f^{(i)}) = 0]\mathbb{E}^{(i,j)}$. $\Delta(y^{(i)}) = y_1^{(i)} - y_2^{(i)}$, $\Delta(f^{(i)}) = f^{(i)}(x_1) - f^{(i)}(x_2)$. See basic notations in the Section 3.1 for the definition of $E^{(i,j)}$.

**One versus One Regime (ovo, M metric).** Alternatively, according to [30], we can formulate a multiclass metric as an average of binary AUC scores for every class pair $(i, j)$, which is defined as

$$\text{AUC}^{\text{ovo}}(f) = \sum_{i=1}^{N_C} \sum_{j \neq i} \text{AUC}_{ij}(f^{(i,j)}) / N_C(N_C - 1),$$

where $\text{AUC}_{ij}(f^{(i,j)}) = \mathbb{E}_{Z_1,Z_2} I[\Delta(y^{(i)})\Delta(f^{(i,j)}) > 0]\mathbb{E}^{(i,j)} + \frac{1}{2} I[\Delta(f^{(i,j)}) = 0]\mathbb{E}^{(i,j)}$, note that $\text{AUC}_{ij} \neq \text{AUC}_{ji}$, since they employ...
different score functions. See basic notations in Section 3.1 for the definition of $\mathcal{E}^{(j)}$.

The following theorem reveals that $\text{AUC}^{\text{ovo}}$ is more insensitive toward the imbalanced distribution of the class pairs than $\text{AUC}^{\text{ova}}$.

**Theorem 1 (Comparison Properties)** Given the label distribution as $\mathbb{P}[y = i] = p_i > 0$ and a multiclass scoring function $f$, the following properties hold:

(a) We have that

$$
\text{AUC}^{\text{ova}}(f) = \frac{1}{N_C} \sum_{i=1}^{N_C} \sum_{j \neq i} (p_{j} - p_{i}) \cdot \text{AUC}_{ij}(f^{(i)}).
$$

(b) $\text{AUC}^{\text{ova}}(f) = \text{AUC}^{\text{ovs}}(f)$, when $p_i = \frac{1}{N_C}$, $i = 1, 2, \ldots, N_C$.

(c) We have $\text{AUC}^{\text{ova}}(f) = 1$ if and only if $\text{AUC}^{\text{ovs}}(f) = 1$.

Theorem 1(a) states that $\text{AUC}^{\text{ova}}$ weights different $\text{AUC}_{ij}$ with $p_{j} - p_{i}$, which will overlook the performance of the minority class pairs. On the contrary, $\text{AUC}^{\text{ovs}}$ assigns equal weights for all pair-wise AUCs, which naturally avoids this issue. Theorem 1(b) suggests that $\text{AUC}^{\text{ova}}$ and $\text{AUC}^{\text{ovs}}$ tend to be equivalent when the label distribution is nearly balanced. Theorem 1(c) further shows that $\text{AUC}^{\text{ova}}$ and $\text{AUC}^{\text{ovs}}$ agree with each other when $f$ maximizes the performance. Practically, the following example shows that how the imbalance issue of class pairs affects model selection under different criterions.

**Example 1** Consider a three-class classification dataset with a label distribution $p_1 = 0.5, p_2 = 0.45, p_3 = 0.05$, we assume that there are two scoring functions $f_a, f_b$, where $\text{AUC}_{ij}(f_a)$ are all 1 except that $\text{AUC}_{13}(f_a) = 0.5$, $\text{AUC}_{ij}(f_b)$ are all 1 except that $\text{AUC}_{13}(f_b) = 0.8$. Consequently, we have $\text{AUC}^{\text{ovs}}(f_a) = 98.3$ and $\text{AUC}^{\text{ova}}(f_a) = 93.3$, while $\text{AUC}^{\text{ova}}(f_b) = 91.6$ and $\text{AUC}^{\text{ovs}}(f_b) = 96.6$.

Obviously, $f_a$ in the example should be a bad scoring function since $f_a^{(1)}$ can not tell apart class-1 and class-3, while $f_b$ is a much better choice since it has good performances in terms of all the pairwise AUCs. We see that $\text{AUC}^{\text{ova}}$ supports choosing $f_b$ against $f_a$, which is consistent with our expectations. However, $\text{AUC}^{\text{ovs}}$ chooses $f_a$ against $f_b$ with a significant margin. This is because that the minority class 3 brings an extremely low weight on $\text{AUC}_{13}$ in $\text{AUC}^{\text{ova}}$. This tiny weight makes the fatal disadvantage of $\text{AUC}_{13}(f_a)$ totally ignored.

From the theoretical and practical analysis provided above, we can draw the conclusion that $\text{AUC}^{\text{ova}}$ is a better choice than $\text{AUC}^{\text{ovs}}$.

### 3.3 Objective and the Roadmap

**Objective.** Our goal in this paper is to construct learning algorithms that maximize $\text{AUC}^{\text{ovs}}$. To fit in the standard machine learning paradigm, we follow the widely-adopted convention [11, 12, 25, 55, 56] to cast the maximization problem into an expected-risk-based minimization problem $f \in \text{argmin}_f R(f)$, where

$$
R(f) = \text{MAUC}^1 = \frac{1}{N_C} \sum_{i=1}^{N_C} \sum_{j \neq i} \mathbb{E}_{z_1, z_2} \left[ \ell_{0,j}^{(1,2)} \right] \mathcal{E}^{(j)}(f),
$$

where

$$
\ell_{0,j}^{(1,2)} = \ell_{0-1}(f^{(i)}, x_1, x_2, y_1^{(i)}, y_2^{(i)}),
$$

$$
= E \left[ \Delta(y_1^{(i)}, \Delta(y_2^{(i)}) < 0) \right] \frac{1}{2} E \left[ \Delta(f^{(i)}) = 0 \right],
$$

is the 0-1 mis-ranking loss.

**Roadmap.** The major challenge in this work is that directly minimizing of $R(f)$ is almost impossible in the sense that: (a) the 0-1 mis-ranking loss $\text{MAUC}^1$ is not differentiable; (b) the calculation of population-level expectation is unavailable; (c) the complexity to estimate the ranking loss is $O(N^2 N_C^2)$ in the worst case. In Sections 4, 5, and 6, we will present solutions to address (a)-(c), respectively.

## 4 BAYES OPTIMAL CLASSIFIER, AND CONSISTENCY ANALYSIS FOR SURROGATE RISK MINIMIZATION

Since the 0-1 mis-ranking loss is a discrete and non-differentiable function, directly solving the optimization problem is almost intractable. In this section, we will construct surrogate risks $R_s(f)$ with surrogate losses $\ell$ as differentiable proxies for the 0-1 mis-ranking loss. We start with finding the Bayes optimal scoring function that we should approximate. Then we provide the surrogate risk minimization framework and investigate if it could approximate the true minimization problem well.

**Bayes Optimal Scoring Functions.** First, let us derive what are the best scoring functions that we need to approximate. With a goal to minimize $\text{MAUC}^1$, $f$ reaches the best performance when it realizes the minimum of expected risk under the 0-1 mis-ranking loss. In this paper, since we are dealing with classification problems, we restrict the choice of each $f^{(i)}$ to measurable functions with range $[0,1]$, i.e.,

$$
f \in \mathcal{F}_{\sigma}^{N_C} = \mathcal{F}_{\sigma} \times \mathcal{F}_{\sigma} \times \cdots \times \mathcal{F}_{\sigma},
$$

where $\mathcal{F}_{\sigma} = \{ g : g$ is a measurable function with a range $[0,1] \}$. Given the restriction, the scoring function should be expressed as: $f \in \text{argmin}_f \mathcal{F}_{\sigma}^{N_C} R(f)$.

Following the convention of machine learning terminologies, we refer to such functions as Bayes optimal scoring functions. The following theorem gives the solution for Bayes optimal scoring functions when the data distribution is a known priori.

**Theorem 2 (Bayes Optimal Scoring Functions)** Given $\eta_i(\cdot) = \mathbb{P}[y = i|x]$, $p_i = \mathbb{P}[y = i]$, we have the following consequences:

(a) $f = \{ f(i) \}_{i=1 \ldots N_C} \in \mathcal{F}_{\sigma}^{N_C}$ is a Bayes optimal scoring function under the $\text{MAUC}^1$ criterion, if

$$
\Delta(f(i)) \cdot \Delta(\pi) > 0,
\forall x_1, x_2, \text{ s.t. } f^{(i)}(x_1, x_2) \neq f^{(i)}(x_2, x_1),
$$

where

$$
\Delta(f(i)) = f(i) - \mathbb{E}_{\pi}(f(i)),
$$

and

$$
\Delta(\pi) = \mathbb{E}_{\pi}(f(i)) - \frac{1}{N_C} \sum_{i=1}^{N_C} \mathbb{E}_{\pi}(f(i)).
$$
where
\[
\Delta(f^{(i)}) = f^{(i)}(x_1) - f^{(i)}(x_2)
\]
\[
\Delta(\pi) = \pi^{(i)}(x_1, x_2) - \pi^{(i)}(x_2, x_1)
\]
\[
\pi^{(i)}(x_1, x_2) = \sum_{j \neq i} \eta_j(x_1) \eta_j(x_2),
\]
\[
\pi^{(i)}(x_2, x_1) = \sum_{j \neq i} \eta_j(x_1) \eta_j(x_2).
\]

(b) Define \( \sigma(\cdot) \) as the sigmoid function, \( s_i(x) = \eta_i(x)/p_i \) and \( s_i(x) = \sum_{j \neq i} \eta_i(x) \), then a Bayes optimal scoring function could be given by
\[
f^{(i)}(x) = \begin{cases} 
\sigma\left( \frac{\eta_i(x)}{\sum_{j \neq i} \eta_j(x)} \right), & 1 > \eta_i(x) \geq 0, \\
1, & 1 = \eta_i(x).
\end{cases} \tag{3}
\]

On top of providing a solution for the optimal scoring function, this theorem also sheds light upon what the optimal scoring functions are really after. Theorem 2 shows that the optimal scoring function provides a consistent ranking with \( (\pi^{(i)}(x_1, x_2)) \), which could be regarded as a generalized likelihood ratio of \( y = i \) versus \( y \neq i \). Here, the posterior distribution \( P(y = i|x) \) is weighted by the factor \( 1/p_i \), which eliminates the dependence on the label distribution since \( P(y = i|x)/p_i \) is in proportion to the label-distribution-independent class-conditional distribution \( P(x|y = i) \). This result suggests that the optimal scoring function induced by MAUC is insensitive toward skewed label distribution.

Surrogate Risk Minimization. Unfortunately, since the data distribution is not available and the 0-1 loss is not differentiable, the Bayesian scoring functions are intractable even with the solution shown in Theorem 2. Practically, we need to replace the 0-1 loss with a convex differentiable loss function \( \ell \) to find tractable approximations. Once \( \ell \) is fixed, we can naturally minimize the much simpler surrogate risk formulated as \( f^{(i)} \in \text{argmin}_f R_\ell(f) \), where
\[
R_\ell(f) = \sum_i R^{(i)}_\ell(f^{(i)})
\]
\[
R^{(i)}_\ell(f^{(i)}) = \sum_{j \neq i} E_{z_1, z_2} \left[ \ell(\Delta(y^{(i)}(1), \Delta(f^{(i)}))) \right].
\]

But how could we find out such surrogate losses at all? A potential candidate must be consistent with the 0-1 loss. In other words, one should make sure that Bayes optimal scoring functions be recovered by minimizing the chosen surrogate risk, at least in an asymptotic way. This leads to the following definition of consistency in a limiting sense.

**Definition 1 (MAUC Consistency)** \( 2\ell \) is consistent with MAUC if for every function sequence \( \{ f_i \}_{i=1,2,...} \), we have
\[
R_\ell(f_i) \to \inf_{f \in \mathcal{F}^N} R_\ell(f) \implies R(f_i) \to \inf_{f \in \mathcal{F}^N} R(f).
\]

**Remark 1** Note that since the infimum of \( f \) is taken over all possible measurable functions with a proper range \([0,1]\), the result is thus irrelevant with the choice of the hypothesis space (linear model, Neural Networks). Practically, to reach the theoretical infimum, it is better to consider complicated hypothesis spaces such as deep neural networks.

Based on this definition, we provide a sufficient condition for MAUC consistency, which is shown in the following theorem. See Appendix B.2 for the proof, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2021.3101125.

**Theorem 3 (MAUC Consistency)** The surrogate loss \( \ell \) is consistent with MAUC for all \( f \in \mathcal{F}^N \), if it is differentiable, convex, and nonincreasing within \([-1,1]\) and \( \ell(0) < 0 \).

From the sufficient condition above, we can show that the popular loss functions are all consistent with MAUC, which is summarized as the following corollary.

**Corollary 1** The following statements hold according to Theorem 3:

1. Logit loss \( \ell_{\text{logit}}(z) = \log(1 + \exp(-z)) \) is consistent with MAUC.
2. Exp loss \( \ell_{\exp}(z) = \exp(-z) \) is consistent with MAUC.
3. Square loss \( \ell_{\text{sq}}(z) = (1 - z)^2 \) is consistent with MAUC.
4. The q-norm hinge loss \( \ell_q(z) = (\max(1 - z, 0))^q \) is consistent with MAUC, if \( q > 1 \).
5. The generalized hinge loss given by
   \[
   \ell_{\epsilon}(z) = \begin{cases} 
   m - t, & t \leq 1 - \epsilon \\
   (t - 1 - \epsilon)^2/4\epsilon, & 1 - \epsilon \leq t < 1, \\
   0, & \text{otherwise}
   \end{cases}
   \]
   is consistent with MAUC, if \( 1/2 > \epsilon > 0 \).
6. The distance-weighted loss given by
   \[
   \ell_{d}(z) = \begin{cases} 
   1/t, & t > \epsilon \\
   1/\epsilon \cdot (2 - t/\epsilon), & \text{otherwise}
   \end{cases}
   \]
   is consistent with MAUC, if \( 1 > \epsilon > 0 \).

Note that \( \ell_{\text{hinge}}(t) = \max(1 - t, 0) = \lim_{\epsilon \to 0} \ell_{\epsilon}(t) \). This shows that hinge loss is at least a limit point of the set of all consistent losses.

### 5 Empirical Risk Minimization and Imbalance Aware Generalization Analysis

Thus far, we have defined the suitable replacements for the 0-1 loss and found some popular examples for them. However, the arguments are based on the population version of the risk. Calculating the expectation is hardly available since the data distribution is most likely unknown to us. In this section, we take a step further to explore how to find empirical estimations of the risks given a specific training data set. On top of this, we show theoretical guarantees for such estimations.

#### 5.1 Empirical Surrogate Risk Minimization

Given a finite training data \( S \) collected from the data distribution, we start with an unbiased estimation of the expected surrogate risk \( R(f) \) based on \( S = \{ x_i, y_i \}_{i=1}^N \). To represent the label frequencies in \( S \), we denote \( \mathcal{N}_i = \{ x_k : y_k = i, z_k \in \mathcal{S} \} \).
$S$ as the set of samples having a label $i$. We define $n_i = |N_i|$ as the number of instances belonging to the $i$th class in $S$.

**Proposition 1 (Unbiased Estimation)** Define $\hat{R}_{E,S}(f)$ as

$$\hat{R}_{E,S}(f) = \sum_{i=1}^{N_i} \sum_{j \neq i} \sum_{x_m \in N_i} \sum_{x_n \in N_j} \frac{1}{n_i n_j} \hat{e}_{i,j,m,n},$$

where $\hat{e}_{i,j,m,n}$ is a shorthand for $\ell(f^{(i)}(x_m) - f^{(i)}(x_n))$. Then $\hat{R}_{E,S}(f)$ is an unbiased estimation of $R_{E}(f)$, in the sense that: $R_{E}(f) = \mathbb{E}_{S}(\hat{R}_{E,S}(f))$.

According to the Empirical Risk Minimization (ERM) paradigm, we can turn to minimize $\hat{R}_{E,S}(f)$ based on a finite training dataset $S$ facing the unknown data distribution. Practically, the function $f$ here is often parameterized by a parameter $\theta$. In this way, we can express $f$ as $f_\theta$. For example, linear models could be defined as $f_\theta(x) = \theta^T x$. Moreover, to prevent the over-fitting issue, we restrict our choice of $f$ on a specific hypothesis class $\mathcal{H}$. Again, $\mathcal{H}$ is essentially a subset of parameterized functions $f_\theta$ where $\theta$ is restricted in a set $\Theta$. For example, $\mathcal{H}$ could be defined as all the linear models with $||\theta|| \leq \gamma$. With this hypothesis set, we can construct the following optimization problem to learn $f$ within $\mathcal{H}$

$$\min_{f \in \mathcal{H}} \hat{R}_{E,S}(f).$$

This allows the standard machine learning technologies to come into play. In this section, we will adopt the notations for mathematical convenience: $f$ and $\mathcal{H}$. Instead of explicitly defining $\theta$, we provide the parameterization in the definition of different hypothesis classes.

### 5.2 MAUC\(^1\) Rademacher Complexity and Its Properties

Our next step is then to investigate how well such approximations will perform. From one point, choosing a consistent surrogate loss $\ell$ ensures that minimizing the surrogate risk suffices to find the Bayes optimal scoring function. From another point, based on a small empirical risk, we also need to guarantee that the training performance generalizes well to the unseen samples such that the expectation $R_{E}(f)$ is small. To ensure the second point, given the assumption that $f$ is chosen from a hypothesis class $\mathcal{H}$, we will provide a Rademacher-Complexity-based worst-case analysis for the generalization ability in the remainder of this section. The key here is to ensure $R_{E}(f) \leq \hat{R}_{E,S}(f) + \delta$ with high probability, where $\delta$ goes to zero when $N$ goes to infinity. This way, minimizing $\hat{R}_{E,S}(f)$ suffices to minimize $R_{E}(f)$.

Unlike traditional machine learning problems that only involve independent instance-wise losses, AUC’s pair-wise formulation makes the generalization analysis difficult to be carried out. Specifically, the terms in MAUC formulation exert certain degrees of interdependency. This way, the standard symmetrization scheme [5], [52] is not available for our problem. For instance, the terms $\ell(f^{(1)}(x_1) - f^{(1)}(x_2))$ and $\ell(f^{(1)}(x'_1) - f^{(1)}(x'_2))$ are interdependent as long as $x_1 = x'_1$ or $x_2 = x'_2$. To address this issue, we provide an extended form of Rademacher complexity for MAUC\(^1\) losses, which is defined as follows.

**Definition 2 (MAUC\(^1\) Rademacher Complexity)** The Empirical MAUC\(^1\) Rademacher Complexity over a dataset $S = \{(x_i, y_i)\}_{i=1}^m$, and a hypothesis space $\mathcal{H}$ is defined as

$$\hat{R}_{\text{MAUC}^1,S}(\ell \circ \mathcal{H}) = \mathbb{E}_{S}\left[\sup_{f \in \mathcal{H}} \sum_{i=1}^{N_i} \sum_{j \neq i} \sum_{x_m \in N_i} \sum_{x_n \in N_j} \frac{T_{i,j,m,n}}{n_i n_j}\right],$$

where

$$T_{i,j,m,n} = \frac{\sigma^{(i)}_1 + \sigma^{(i)}_2}{2} \ell(f^{(i)}(x_m) - f^{(i)}(x_n)),$$

for $i = 1, 2, \ldots, N$, $\sigma^{(i)}_1, \ldots, \sigma^{(i)}_2$ are i.i.d Rademacher random variables. The population version of the MAUC\(^1\) Rademacher Complexity is defined as $R_{\text{MAUC}^1}(\ell \circ \mathcal{H}) = \mathbb{E}_{S}\[R_{\text{MAUC}^1,S}(\ell \circ \mathcal{H})].$

The most important property of the MAUC\(^1\) Rademacher complexity is that its magnitude is directly related to the generalization upper bounds, which is shown in the following theorem. The proofs are shown in Appendix E in the supplementary materials, available online.

**Theorem 4 (Abstract Generalization Bounds)** Given dataset $S = \{(x_i, y_i)\}_{i=1}^m$, where the instances are sampled independently, for all multiclass scoring functions $f \in \mathcal{H}$, if $\text{Range}(\ell) \subseteq [0, B]$, then for any $\delta \in (0, 1)$, the following inequality holds with probability at least $1 - \delta$

$$R_{E}(f) \leq \hat{R}_{S}(f) + C_1 \cdot \frac{\hat{R}_{\text{MAUC}^1,S}(\ell \circ \mathcal{H})}{N} + C_2 \cdot \frac{B}{NC} \cdot \xi(Y) \cdot \frac{\log \left(\frac{N}{\delta}\right)}{N},$$

where $C_1, C_2$ are universal constants, $\xi(Y) = \sqrt{\sum_{i=1}^{N_y} \gamma_i^2} / \gamma_0; \gamma_0 = \frac{N}{2}$.

According to Theorem 4, we can obtain a generalization bound as soon as we can obtain a proper upper bound on the empirical Rademacher complexity $\hat{R}_{\text{MAUC}^1,S}(\ell \circ \mathcal{H})$. In the remainder of this subsection, we will develop general techniques to derive the upper bound of $R_{\text{MAUC}^1,S}(\ell \circ \mathcal{H})$ based on the notion of covering number and chaining. With the help of the next few theorems, we can convert the upper bound of $\hat{R}_{\text{MAUC}^1,S}(\ell \circ \mathcal{H})$ to upper bounds of Rademacher complexities for much simpler model classes. These techniques are foundations for the practical results developed in the next subsection. Specifically, Theorem 5(a) is used to prove Theorem 6; Theorem 6 is employed in the second half of the proof of Theorem 8; and Theorem 5(b) is employed in the proof of Theorem 9.

We are now ready to present the corresponding results. With the sub-Gaussian property proved in Lemma 9 in Appendix E, available online, we can derive chaining upper bounds for the MAUC\(^1\) Rademacher complexity.
foundation here is the notion of covering number, which is elaborated in Definitions 3 and 4.

Definition 3 (ε-covering) [39] Let (H, d) be a (pseudo)metric space, and Θ ∈ H, \{h_1, \ldots, h_K\} is said to be an ε-covering of Θ if Θ ⊆ \bigcup_{i=1}^{K} B(h_i, ε), i.e., ∀θ ∈ Θ, \exists! s.t. d(θ, h_i) ≤ ε.

Definition 4 (Covering Number) [39] Based on the notations in Definition 3, the covering number of Θ with radius ε is defined as

\[ \mathcal{C}(ε, Θ, d) = \min\{n : \exists ε - \text{covering over} Θ \text{ with size } n\}. \]

Based on the definitions above, we can reach the following results. The proof is shown in Appendix E in the supplementary materials, available online. Note that in the following arguments, the covering number is defined on the metric d_{∞, S}. Specifically, given two vector valued functions \( s = (s^{(1)}, \ldots, s^{(N_C)}) \), \( \tilde{s} = (\tilde{s}^{(1)}, \ldots, \tilde{s}^{(N_C)}) \), and the training data \( S, d_{∞, S}(s, \tilde{s}) \) is defined as

\[ d_{∞, S}(s, \tilde{s}) = \max_{x_i \in S, y_i \in [N_C]} |s^{(j)}(x_i) - \tilde{s}^{(j)}(x_i)|. \]

Theorem 5 (Chaining Bounds for MAUC^↓ Rademacher Complexity) Suppose that the score function \( s^{(j)} \) maps \( \mathcal{X} \) onto a bounded interval \([-R_s, R_s]\), the following properties hold for \( \mathcal{R}_{MAUC, S}^{↓}(\ell \circ \mathcal{H}) \):

(a) For a decreasing precision sequence \( \{ε_k\}_{k=1}^{K} \), with \( ε_{k+1} = \frac{1}{2} ε_k \), \( k = 1, 2, \ldots, K - 1 \) and \( ε_0 ≥ R_s \), we have

\[ \mathcal{R}_{MAUC, S}^{↓}(\ell \circ \mathcal{H}) ≤ N_C \cdot (N_C - 1) \cdot \phi_ε \cdot \epsilon_K + 6 \cdot \sum_{k=1}^{K} ε_k \phi_ε \cdot (N_C - 1) \cdot \xi(Y) \sqrt{\log \left( \mathcal{C}(ε, \mathcal{F}, d_{∞, S}) \right)} / N. \]

(b) There exists a universal constant C, such that

\[ \mathcal{R}_{MAUC, S}^{↓}(\ell \circ \mathcal{H}) ≤ C \phi_ε \inf_{R_s ≥ 0} \left( N_C \cdot (N_C - 1) \cdot \xi(Y) \cdot \int_{R_s}^{\infty} \sqrt{\log \left( \mathcal{C}(ε, \mathcal{F}, d_{∞, S}) \right)} / dε \right). \]

According to Theorem 5, we have the following theorem as an extension of a new minorization technique which appears in a recent work [64].

Theorem 6 (Transformation Upper Bound) Given the Hypothesis class

\[ \mathcal{F} \circ g = \left\{ g(x) = \text{soft}(s(x)) : s \in \mathcal{F} \right\}, \]

where \text{soft}(\cdot) is the softmax function. Suppose that \( s(x) ∈ [-R_s, R_s]^{N_C} \) and \( \ell \) is \( \phi_ε \)-Lipschitz continuous, the following inequality holds:

\[ \mathcal{R}_{MAUC, S}(\ell \circ \mathcal{F}) \frac{1}{N_C(N_C - 1)} ≤ \phi_ε \left( \frac{2^g - 1}{N_C} \cdot \sqrt{\xi(Y)} \cdot \log \left( e \cdot R_s \cdot N \cdot N_C \right) + \sqrt{\frac{1}{N}} \right). \]

where the Rademacher complexity \( \mathcal{R}_{N, N_C}(\Pi \circ \mathcal{F}) \) is defined as

\[ \mathcal{R}_{N, N_C}(\Pi \circ \mathcal{F}) = \mathbb{E}_a \left[ \sup_{f = f(x^{(1)}, \ldots, x^{(N_C)}) \in \mathcal{F}} \frac{1}{N \cdot N_C} \sum_{j=1}^{N_C} \sum_{i=1}^{N} \sigma_j^{(i)} \cdot f^{(j)}(x_i) \right], \]

where \( \{σ_j^{(i)}\}_{(i,j)} \) is a sequence of independent Rademacher random variables.

The result in Theorem 6 directly relates the complicated pairwise Rademacher complexity \( \mathcal{R}_{MAUC, S}(\ell \circ \mathcal{F}) \) to the instance-wise Rademacher complexity \( \mathcal{R}_{N, N_C}(\Pi \circ \mathcal{F}) \). This makes the derivation of the generalization upper bound much easier. Once a bound on the ordinary Rademacher complexity over the functional class \( \mathcal{F} \) is available, we can directly plug it into this theorem and find a resulting bound over the MAUC^↓ complexity.

5.3 Generalization Bounds for Deep and Shallow Model Families

Next, we derive the generalization bounds for three hypothesis classes: (1) the \( ℓ_p \) penalized linear models, (2) deep neural networks with fully-connected layers, and (3) the deep convolutional neural networks.

Assumption 1 (Common Assumptions) In this subsection, we require some common assumptions listed as follows:

- The sample points in the training dataset \( S = \{(x_i, y_i)\}_{i=1}^{m} \) are sampled independently
- \( \text{Range} (\ell) ⊆ [0, B] \)
- The loss function \( ℓ \) is \( \phi_ε \)-Lipschitz continuous
- The input features are sampled from \( \mathcal{X} ⊂ \mathbb{R}^d \), and for all \( x ∈ \mathcal{X} \), we have \( ||x||_{2} ≤ R_X \).

5.3.1 Generalization Bound for \( ℓ_p \) Penalized Linear Models

We begin with the generalization bound for \( ℓ_p \) norm penalized linear models. The proof is shown in Appendix F in the supplementary materials, available online.

Theorem 7 (Practical Generalization Bounds for Linear Models) Define the \( ℓ_p \) norm penalized linear model as

\[ \mathcal{H}_{p, y}^{Lin} = \{ f = (f^{(1)}, \ldots, f^{(N_C)}) : f^{(i)}(x) = W^{(i)} x, ||W^{(i)}||_{p} ≤ y \}, \]

with \( 0 < p < ∞, \frac{1}{p} + \frac{1}{q} = 1 \). Based on Assumption 1, for all \( f ∈ \mathcal{H}_{p, y}^{Lin} \), we have the following inequality holds with probability at least \( 1 - δ \)

\[ \mathcal{R}_{\ell}(f) ≤ \tilde{R}_{\ell, S}(f) + \mathcal{I}_{Lin}(\mathcal{X}(Y), \xi(Y), δ, \frac{1}{N}). \]
where 
\[
I_{\text{Lin}}(\chi(Y), \xi(Y), \delta) = \frac{4R_x \phi_2}{N_C - 1} \sqrt{\frac{2(p - 1)}{N_C}} \cdot \chi(Y) + \frac{5B}{N_C} \cdot \sqrt{2\log\left(\frac{2}{\delta}\right)} \cdot \xi(Y).
\]

### 5.3.2 Generalization Bound for Deep Fully-Connected Neural Networks

Next, we take a further step to explore the generalization ability of a specific type of deep neural networks where only fully-connected layers and activation functions exist. The detailed here is as follows.

**Settings.** We denote an \( L \)-layer deep fully-connected neural network with \( N_C \)-way output as 
\[
f(x) = W f_{\omega, L}(x) = W s(\omega_{L-2} \cdots s(\omega_1 x)),
\]
The notations are as follows: \( s(\cdot) \) is the activation function; \( n_{h_i} \) is the number of hidden neurons for the \( i \)-th layer; \( \omega_j \in \mathbb{R}^{n_{h_{j-1}} \times n_{h_j}}, j = 1, 2, \ldots, L - 2 \) are the weights for the first \( L - 1 \) layers; \( W \in \mathbb{R}^{n_{h_{L-1}} \times N_C} \) is the weight for the output layer. Moreover, the output from the \( i \)-th layer of the network is defined as \( f(i)(x) = W(i) f_{\omega, L}(x) \), where \( W(i) \) is the \( i \)-th row of \( W \). In the next theorem, we focus on a specific hypothesis class for such networks where the product of weight norms \( \Pi_{W, \omega} = \|W\|_F \cdot \prod_{i=1}^{L-2} \|\omega_i\|_F \) are no more than \( \gamma \). We denote such a hypothesis class as \( \mathcal{H}_{Y, R_s, \omega}^{DNN} \)
\[
\mathcal{H}_{Y, R_s, \omega}^{DNN} = \left\{ f : f(i)(x) = W(i)^T f_{\omega, L}(x), \|f(i)\|_F \leq R_s, i = 1, \ldots, N_C, \Pi_{W, \omega} \leq \gamma \right\}.
\]
To obtain the final output, we perform a softmax operation over \( f(x) \). Thus, the valid model under this setting could be chosen from the following hypothesis class:
\[
\text{soft} \circ \mathcal{H}_{Y, R_s, \omega}^{DNN} = \left\{ g : g(i) = \frac{\exp(f(i)(x))}{\sum_{j=1}^{N_C} \exp(f(j)(x))}, f \in \mathcal{H}_{Y, R_s, \omega}^{DNN} \right\}.
\]
We have the following bound for this type of models. The result here is a merge of two independent results we proposed in Appendix E and Appendix F in the supplementary material, available online, which is based on the Talagrand contraction properties and the chaining technology shown in Theorems 5 and 6.

**Theorem 8 (Practical Generalization Bounds for Deep Models)** On top of Assumption 1, if we further assume that \( s(\cdot) \) is a 1-Lipschitz and positive homogeneous activation function, then for all \( f \in \text{soft} \circ \mathcal{H}_{Y, R_s, \omega}^{DNN} \), we have the following inequality holds with probability at least \( 1 - \delta \):
\[
R_l(f) \leq R_S(f) + \min(\mathcal{I}_{DNN,1}, \mathcal{I}_{DNN,2}) \cdot \sqrt{\frac{1}{N}}.
\]
where \( \mathcal{I}(Y) = \sqrt{\sum_{i=1}^{N_C} \frac{\gamma^2}{N_C - 1} \cdot \chi(Y) + \frac{5B}{N_C} \cdot \sqrt{2\log\left(\frac{2}{\delta}\right)} \cdot \xi(Y)} \)
\[
\mathcal{I}_{DNN,1} = C_1 \phi_1 \cdot \chi(Y) + \left( \sqrt{2C_1 R_x \phi_1} \cdot \frac{C_3}{N_C} \cdot \sqrt{2\log\left(\frac{2}{\delta}\right)} \cdot \xi(Y) \right),
\]
\[
\mathcal{I}_{DNN,2} = C_1 \phi_1 \left( \frac{2^g}{N_C} \cdot \chi(Y) \cdot \log^{3/2}(K \cdot N \cdot N_C) \cdot \gamma \cdot R_x \cdot \left( \sqrt{2\log\left(\frac{2}{\delta}\right)} L + 1 \right) \right) + C_2 \cdot \sqrt{2\log\left(\frac{2}{\delta}\right)} \cdot \xi(Y),
\]
\( C_1, C_2 \) are universal constants as Theorem 4, \( K = e \cdot R_s, C_3 = \sqrt{2\log^{2+1}/N_C} \).

### 5.3.3 Generalization Bound for Deep Convolutional Neural Networks

Now we use the result in Theorem 5 to derive a generalization bound for a class of deep neural networks where fully-connected layers and convolutional layers coexist. In a nutshell, the result is essentially an application of Theorem 5 to a recent idea appeared in [48].

**Settings.** Now we are ready to introduce the setting of the deep neural networks employed in the forthcoming theoretical analysis, which is adopted from [48]. We focus on the deep neural networks with \( N_{\text{conn}} \) fully-connected layers and \( N_{\text{conv}} \) convolutional layers. The \( i \)-th convolutional layer has a kernel \( K(i) \in \mathbb{R}^{K(i)h_k \times K(i)w_k \times N_C} \). Recall that convolution is a linear operator. For a given kernel \( K \), we denote its associated matrix as \( \text{op}(K) \), such that \( K(x) = \text{op}(K)x \). Moreover, we assume that, at each time, the convolution layer is followed by a componentwise non-linear activation function and an optional pooling operation. We assume that the activation functions and the pooling operations are all 1-Lipschitz. For the \( i \)-th fully-connected layer, we denote its weight as \( V(i) \).

Above all, the complete parameter set of a given deep neural network could be represented as \( P = \{K(1), \ldots, K(N_{\text{conv}}), V(1), \ldots, V(N_{\text{conv}})\} \). Again, we also assume that the loss function is \( \phi_1 \)-Lipschitz and \( \text{Range}(\ell) \subseteq [0, B] \). Finally, given two deep neural networks with parameters \( P \) and \( \bar{P} \), we adopt a metric \( d_{\text{NN}}(\cdot, \cdot) \) to measure their distance
\[
d_{\text{NN}}(P, \bar{P}) = \sum_{i=1}^{N_{\text{conv}}} \|\text{op}(K(i)) - \text{op}(\bar{K}(i))\|_2 + \sum_{i=1}^{N_{\text{conv}}} \|V(i) - V(i)\|_2.
\]

### Constraints Over the Parameters

First, we define \( \mathcal{P}_{\text{NN}}^{(0)} \) as the class for initialization of the parameters
\[
\mathcal{P}_{\text{NN}}^{(0)} = \left\{ P : \left( \max_{i \in [1 \cdots N_{\text{conn}}]} \|\text{op}(K(i))\|_2 \right) \leq 1 + v, \left( \max_{i \in [1 \cdots N_{\text{conv}}]} \|V(i)\|_2 \right) \leq 1 + v \},
\]
Now we further assume that the learned parameters should be chosen from a class denoted by \( \mathcal{P}_{\phi, g} \), where the distance
between the learned parameter and the fixed initialization residing in $P^{(0)}_v$ is no bigger than $\beta$

$$P_{\beta,v} = \left\{ P: d_N(P, \hat{P}_0) \leq \beta, \hat{P}_0 \in P^{(0)}_v \right\}.$$ 

We have the following result for this class of deep neural networks, which is an application of Theorem 5. The proof is shown in Appendix F in the supplementary materials, available online.

**Theorem 9** Denote the hypothesis class,

$$\text{soft} \circ \mathcal{F}_{\beta,v} = \left\{ g(x) = \text{soft}(s_P(x)) : s_P \in \mathcal{F}_{\beta,v} \right\},$$

$$\mathcal{F}_{\beta,v} = \left\{ s_P : \mathbb{R}^{N_{L-1}} \rightarrow \mathbb{R}^C | P \in P_{\beta,v}, \right. \right. \left. \left. \text{Range}(s_P) \subseteq [-R_s, R_s]^C \right\}. $$

On top of Assumption 1, if we further assume that

$$R_s > 1/\min \left\{ \sqrt{N}, \frac{\xi(Y)}{N_C} \cdot \sqrt{N_{par}}(vN_L + \beta + \log(3R_s \beta N)) \right\},$$

then for all multiclass scoring functions $f \in \text{soft} \circ \mathcal{F}_{\beta,v}, \forall \delta \in (0, 1)$, the following inequalities hold with probability at least $1 - \delta$

$$R_{\ell}(f) \leq \hat{R}_{\ell}(f) + C_1 \alpha_1 \alpha_2$$

$$+ C_2 \cdot \frac{B(Y)}{N_C} \cdot \sqrt{\frac{\log(\frac{17}{\delta})}{N}},$$

where

$$\alpha_1 = \tilde{C} \cdot \phi_{\ell} \cdot R_s \cdot \frac{\xi(Y)}{N_C},$$

$$\alpha_2 = \sqrt{N_{par} (vN_L + \beta + \log(3\beta R_s \beta N))} \cdot \frac{1}{N}.$$ 

$\tilde{C}, C_1, C_2$ are universal constants, $N_L = N_{conv} + N_{conv}, N_{par}$ is total number of parameters in the neural network.

### 5.4 Summary

Note that there are two imbalance-aware factors that appear in the upper bounds above. The first factor $\xi(Y)$ captures the imbalance of class label distribution in $S$. With an analogous spirit, $\chi(Y)$ captures the degree of imbalance of the class pair distribution in $S$. To improve generalization, the training data $S$ must simultaneously have a large sample size $N$ and a small degree of imbalance captured by $\chi(Y)$. In other words, our bounds suggest that blindly increasing the sample size of the training dataset will not improve generalization. To really improve generalization, one should increase data points of the minority classes which are the sources for performance bottleneck. Consequently, compared with the ordinary $O(\sqrt{1/N})$ result, our bound is much more aware of the imbalance issue hidden behind the training data.

### 6 Efficient Computations in Optimization

Till now, we have conducted a series of theoretical analyses for our framework. In this section, we turn our focus to the practical issues we must face during optimization. As shown in the previous sections, the complicated formulations of MAUC surrogate losses bring a great burden to the fundamental operations of the downstream optimization algorithm. Specifically, given $T_i$ as the time complexity for loss evaluation of a single sample pair and given $T_{grad}$ as that for the gradient evaluation, we can find that even a single full(mini)-batch loss and gradient evaluation takes $O(\sum_{i=1}^{N_C} \sum_{j \neq i} n_i n_j T_i)$ and $O(\sum_{i=1}^{N_C} \sum_{j \neq i} n_i n_j T_{grad})$, which scales almost quadratically to the sample (batch) size. In general, such complexities are unaffordable facing medium and large-scale datasets. However, we find that for some of the popular surrogate losses, the pairwise computation could be largely simplified. Consequently, we propose acceleration algorithms for loss and gradient evaluations for three well-known losses: exponential loss, squared loss, and hinge loss. The time complexity with/without acceleration is shown in Table 1, which shows the effectiveness of our proposed algorithms.

| Algorithms            | loss                                | gradient                            | requirement       |
|-----------------------|-------------------------------------|-------------------------------------|-------------------|
| exp + acceleration    | $O(N_C \cdot N \cdot T_i)$         | $O(N_C \cdot N \cdot T_{grad})$     | $\min n_i \gg 2$ |
| squared + acceleration| $O(N_C \cdot N \cdot T_i)$         | $O(N_C \cdot N \cdot T_{grad})$     | $\epsilon_2^{(N-n_i)} n_i \gg N - \epsilon_2^{(N-n_i)}$ |
| hinge + acceleration  | $O(N_C \cdot N \cdot T_i)$         | $O(N_C \cdot N \cdot T_{grad})$     | $\min n_i \gg 2$ |
| w/o acceleration      | $O(\sum_{i=1}^{N_C} \sum_{j \neq i} n_i n_j \cdot T_i)$ | $O(\sum_{i=1}^{N_C} \sum_{j \neq i} n_i n_j \cdot T_{grad})$ | \-

We adopt this general form since it covers a lot of popular models. For example, when $g_i(\cdot)$ is defined as the activation function of the last layer of a neural network (say a softmax function), $w^{(i)}$ are the weights of the last layer, and $h\theta(\cdot)$ is the neural net where the last layer is excluded,}

TABLE 1 Acceleration for Three Losses, Where $N = \sum_{i=1}^{N_C} n_i \log n_i + (N - n_i) \log (N - n_i)$
becomes a deep neural network architecture with the last layer designed as a fully-connected layer. As an another instance, if both \( g_i(x) = x \) and \( h_\theta(x) = x \), then we reach a simple linear multiclass scoring function. Note that the scalability with respect to sample size only depends on the choice of \( \ell \). The choice of \( g_i(\cdot) \) and \( h_\theta(\cdot) \) only affects the instance-wise chain rule. This allows us to provide a general acceleration framework once the surrogate loss is fixed.

Due to the limited space of this paper, we only provide loss evaluation acceleration methods in the main paper. The readers are referred to Appendix G for more details, available online, where we also present a discussion to deal with the acceleration for general loss functions.

### 6.1 Exponential Loss

For the exponential loss, we can simplify the computations by a factorization scheme

\[
\hat{R}_{\text{exp}} = \sum_{i=1}^{N_C} \sum_{x_m \in N_i} \sum_{j \neq i} \sum_{x_n \in N_j} \frac{1}{n_i n_j} \cdot \exp \left( \alpha \cdot (f^{i,j,m,n}) \right),
\]

\[
= \sum_{i=1}^{N_C} \left( \sum_{x_m \in N_i} \exp \left( \alpha \cdot f^{i}(x_m) \right) \right) \cdot \left( \sum_{j \neq i} \frac{1}{n_i n_j} \cdot \exp \left( -\alpha \cdot f^{i}(x_n) \right) \right),
\]

where \( f^{i,j,m,n} = f^{i}(x_m) - f^{i}(x_n) \). From the derivation above, the loss evaluation could be done by first calculating \((a_i), (b_i)\) separately and then performing the multiplication, which only takes \( O(NN_C T_i) \). This is a significant improvement compared with the original \( O(\sum_{i=1}^{N_C} \sum_{j \neq i} n_i n_j N C T_i) \) result.

### 6.2 Hinge Loss

First we put down the hinge surrogate loss as

\[
\hat{R}_{\text{hinge}} = \sum_{i=1}^{N_C} \sum_{x_m \in N_i} \sum_{j \neq i} \sum_{x_n \in N_j} \frac{1}{n_i n_j} \cdot \left( \alpha - (f^{i,j,m,n}) \right)_+.
\]

The key of our acceleration is to notice that the terms are non-zero only if \( (f^{i}(x_m) - f^{i}(x_n)) \geq \alpha \). Moreover, for these non-zero terms \( \max(x, 0) = x \). This means that the hinge loss degenerates to an identity function for the activated nonzero terms, which enjoys efficient computation. So the key step in our algorithm is to find out the non-zero terms in an efficient manner.

Given a fixed class \( i \) and an instance \( x_m \in N_i \), we denote

\[
A^{(i)}(x_m) = \{ x_n \notin N_i, \alpha > (f^{i}(x_m) - f^{i}(x_n)) \}.
\]

With \( A^{(i)}(x_m) \), one can reformulate \( \hat{R}_{\text{hinge}} \) as

\[
\hat{R}_{\text{hinge}} = \sum_{i=1}^{N_C} \sum_{x_m \in N_i} \left[ \left( \sum_{j \neq i} \sum_{x_n \in N_j \setminus A^{(i)}(x_m)} \frac{1}{n_i n_j} \cdot (\alpha - f^{i}(x_m)) \right) \right].
\]

For each fixed class \( i \), we construct an affinity matrix \( Aff^{(i)} \).
\[ A_{m,n}^{(i)} = \begin{cases} \frac{1}{n_{n,m}}, & n \in N_i, m \notin N_i, \\ \frac{1}{n_{m,m}}, & m \in N_i, n \notin N_i, \\ 0, & \text{Otherwise,} \end{cases} \]

which could be written in a matrix form,

\[ A^{(i)} = D^{(i)}(1 - Y^{(i)})Y^{(i)\top} + Y^{(i)}(1 - Y^{(i)})\top D^{(i)}. \]

Then the empirical risk function under the squared surrogate loss could be reformulated as

\[ \hat{R}_{sq} = \sum_{i=1}^{N_C} \Delta_{sq}^{(i)} \mathcal{L}^{(i)} \Delta_{sq}^{(i)}, \]

where

\[ \Delta_{sq}^{(i)} = Y^{(i)} - f^{(i)}(X), \]
\[ f^{(i)}(X) = [f^{(i)}(x_1), \ldots, f^{(i)}(x_N)]\top, \]  

\[ \mathcal{L}^{(i)} = \text{diag}(A^{(i)}) - A^{(i)}. \]

\[ \Delta_1^{(i)} = D^{(i)}(1 - Y^{(i)}) \]
\[ \Delta_2^{(i)} = Y^{(i)\top} \Delta_{sq}^{(i)} \]
\[ \mathbf{k}^{(i)} = n_iD^{(i)}(1 - Y^{(i)}) + \frac{N_C - 1}{n_i} Y^{(i)}. \]

Taking all together, we see that the accelerated evaluation requires only \(O(N \cdot N_C \cdot T_i)\) time.

7 Experiments

7.1 Datasets

Now we start the empirical analyses on the real-world datasets. First, we describe all the datasets involved in the forthcoming discussions. Generally speaking, the datasets come from three types of sources: (a) LIBSVM website, (b) KEEL website, and (c) others. Note that for all the datasets prefixed with Imb, we use an imbalanced subset sampled from the original dataset to perform our experiments. Here we only present a brief summary. More detailed contents could be found in Appendix H.2, available online.

(a) LIBSVM Datasets,\(^3\) which includes: Shuttle, Svmguide-2, SegmentImb.\(^4\)
(b) KEEL Datasets,\(^5\) which includes: Balance, Dermatology, Ecoli, New Thyroid, Page Blocks, Yeast.

(c) Other Datasets:

1. CIFAR-100-Imb. We sampled an imbalanced version of the CIFAR-100\(^6\) dataset, which originally contains 100 image classes each with 600 instances. The 100 classes are encoded as 1,..., 100.
2. User-Imb. The original dataset is collected from TalkingData, a famous third-party mobile data platform from China, which predicts mobile users’ demographic characteristics based on their app usage records. The dataset is collected for the Kaggle Competition named Talking Data Mobile User Demographics.\(^7\) The raw features include logged events, app attributes, and device information. There are 12 target classes ‘F23’, ‘F24-26’, ‘F27-28’, ‘F29-32’, ‘F33-42’, ‘F43’, ‘M22’, ‘M23-26’, ‘M27-28’, ‘M29-31’, ‘M32-38’, ‘M39’, which describe the demographics (gender and age) of users. In our experiments, we sample an imbalanced subset of the original dataset to leverage a class-skewed dataset.
3. iNaturalist2017. iNaturalist Challenge 2017 dataset\(^8\) is a large-scale image classification benchmark with 675,170 images covering 5,089 different species of plants and animals. We split the dataset into the training set, validation set, and test set at a ratio of 0.7:0.15:0.15. Since directly training deep models in such a large-scale dataset is time-consuming, we instead generate 2048-d features with a ResNet-50 model pre-trained on ImageNet for each image and train models with three fully-connected layers for all methods. We utilize Adam optimizer to train the models, with an initial learning rate of \(10^{-5}\). To ensure all categories are covered in a mini-batch, the batch size is set to 8,196. Other hyperparameters are the same as those in the CIFAR-100-Imb dataset.

7.2 Competitors

Choice of Competitors. Specifically, our goal is to validate that direct MAUC optimization techniques leverage better MAUC performance than other imbalanced learning methods. Motivated by this, the competitors include: a Baseline that does not deal with the imbalance distribution; Oversampling and Under-sampling Methods that tackle imbalance issues by re-sampling; and Imbalanced Loss Functions that tackle imbalance issues by reformulating the loss function. Moreover, we implement our framework with the squared loss, exponential loss, and hinge loss with the acceleration methods proposed in Section 6. Note that, for the sampling-based methods, we use the code from the python Lib Imbalanced-learn\(^4\) to implement these competitors. The details are listed as follows:

1. Standard Baseline: LR. This is the baseline model where no measures are taken against imbalance issues. For LR, we adopt a multiclass version of the
logistic regression. For traditional datasets, LR is built upon a linear model. For deep learning datasets, LR is built upon a common deep backbone.

2) Over-sampling Methods. For these competitors, we first plug in an oversampling method to generate a more balanced dataset, then we use LR to train a model on the new dataset. Here we adopt the following two oversampling methods as our competitors:
   - BM. (BorderlineSMOTE) [29]: This method is a variant of the SMOTE method (Synthetic Minority Over-sampling Technique), which restricts the oversampling process to the hard minority samples which lie at the borderline of the decision boundary.
   - MM. (MWMOTE) [6]: MWMOTE is another variant of SMOTE. It first identifies the hard-to-learn informative minority class samples and assigns them weights according to their euclidean distance from the nearest majority class samples.

3) Under-Sampling Methods. For these competitors, we first adopt an undersampling method to generate a more balanced dataset, then we use LR to train a model on the new dataset. Here we adopt the following two methods as our competitors:
   - IHT. (InstanceHardnessThreshold) [66]. This method works by filtering out the hard and noisy samples from the majority classes based on the instance hardness measure proposed in [66].
   - NM. (NearMiss) [49]: It adopts an under-sampling idea to make majority class samples surround most of the minority class samples.
   - TL. (TomekLinks) [67]: It adopts the Tomek Links method to remove redundant samples that fail to contribute to the outline of the decision boundary from the majority class.

4) Imbalanced Loss Functions (for deep learning): For CIFAR-100-Imb and User-Imb dataset, we also compare our method with some of the recently proposed imbalanced loss functions for deep learning.
   - Focal Loss [42]: It tackles the imbalance problem by adding a modulating factor to the cross-entropy loss to highlight the hard and minority samples during the training process.
CB-CE: It refers to the loss function that applies the reweighting scheme proposed in [16] on the cross-entropy loss.

CB-Focal: It refers to the loss function that applies the reweighting scheme proposed in [16] on the Focal loss.

LDAM[9]: It proposes a Label-distribution-aware margin loss based on the minimum margin per class. Here we adopt the smooth relaxation for cross-entropy loss proposed therein.

5) Existing Deep AUC optimization methods:
- DeepAUC: [44] is a state-of-the-art stochastic AUC maximization algorithm developed for the deep neural network. It solves the AUC maximization from the saddle point problem. To ensure a fair comparison, we implement the algorithm that extends to the multi-class AUC problem by PyTorch. In addition, there are two optimization configurations in [44], including Proximal Primal-Dual Stochastic Gradient (PPD-SG) and Proximal Primal-Dual AdaGrad (PPD-AdaGrad). PPD-AdaGrad is employed in our experiment because it usually demonstrates better performance than PPD-SG in most cases.

6) Our Methods:
- Ours1: An implementation of our learning framework with the square surrogate loss $\ell_{sq}(\alpha, t) = (\alpha - t)^2$
- Ours2: An implementation of our learning framework with the exponential loss $\ell_{exp}(\alpha, t) = \exp(-\alpha t)$

Fig. 2. Fine-grained Comparison Over the Minority Class Pairs (Traditional Datasets). The $x$-axis gives the frequency rank of the class pairs $(i, j)$, i.e., $p_ip_j$, where a larger rank represents a larger frequency. The $y$-axis represents the AUC$_{ij}$ of the corresponding class pairs. Each line in a plot then captures the minority class pair performance of a given algorithm. To have a clearer look at tendency, we carry out two filtering processes before we visualize the plot: (a) For those datasets which has more than 5 class pairs, we only visualize the bottom-5 pairs in terms of the frequency to turn our focus to the minority pairs in the dataset. (b) To have a clearer look at the difference of the top competitors, we filter out the pairs with a smaller AUC$_{ij}$ than 0.7.
7.3 Implementation Details

Infrastructure. All the experiments are carried out on a ubuntu 16.04.6 server equipped with Intel(R) Xeon(R) CPU E5-2620 v4 cpu and a TITAN RTX GPU. The codes are implemented via python 3.6.7, the basic dependencies are: pytorch (v-1.1.0), sklearn (v-0.21.3), numpy (v-1.16.2). For traditional datasets, we implement our proposed algorithms with the help of the sklearn and numpy. For hinge loss, we use Cython to accelerate the dynamic programming algorithm. For the deep learning datasets, our proposed algorithms are implemented with pytorch.

Evaluation Metric. Given a trained scoring function \( f = (f_1, \ldots, f_{NC}) \), all the forthcoming results are evaluated with the MAUC metric (the larger the better).

7.4 Results and Analysis

Traditional Datasets. The average performances of 15 repetitions for the nine traditional datasets are shown in Fig. 1, where the scatters show the 15 observations over different dataset splits, and the bar plots show the average performance over 15 repetitions. Consequently, we have the following observations: 1) The best performance of our proposed algorithm consistently surpasses all the competitors significantly on all the datasets. More specifically, the best algorithm among Ours1, Our2 and Our3 outperforms the best competitors by a margin of 5.3, 1.5, 5.6, 1.3, 4.4, 2.9, 4.9, 1.4, 1.8 for Balance, Dermatology, Ecoli, New Thyroid, Page Blocks, SegmentImb, Shuttle, Svmguide2, and Yeast, respectively. It turns out that our improvements are significant in most cases. 2) It could always be observed that some of the sampling-based competitors fail to outperform the original LR algorithm. One reason for this phenomenon might be that the sampling-based methods are not directly designed for optimizing AUC. Another possible reason is that most of the sampling methods adopt an ova strategy to perform multiclass sampling. This has a similar negative effect, as shown in Theorem 1 for AUCova, where the imbalance issue across class pairs is not taken into consideration. 3) Based on the fact that an imbalanced dataset’s performance bottleneck comes from its minority classes, we investigate a closer look at the performance comparison on minority class pairs.

| type | method | CIFAR-100 | User | iNaturalist |
|------|--------|-----------|------|-------------|
| Baseline | CE | 59.80 | 55.26 | 66.84 |

| Sampling | | | |
|---------|--------|-----------|------|-------------|
| BM | 59.07 | 58.95 | 78.18 |
| MM | 58.52 | 55.35 | 67.07 |
| IHT | 60.56 | 55.87 | 67.08 |
| NM | 60.24 | 54.52 | 66.74 |
| TL | 60.28 | 52.99 | 66.77 |

| Loss | | | |
|------|--------|-----------|------|-------------|
| FOCAL | 60.03 | 55.34 | 66.97 |
| CBCE | 60.04 | 58.91 | 77.71 |
| CBFOCAL | 60.42 | 58.75 | 71.12 |
| LDAM | 60.26 | 56.47 | 73.18 |

| AUC | | | |
|------|--------|-----------|------|-------------|
| Ours1 | 61.90 | 60.64 | 79.67 |
| Ours2 | 62.08 | 60.94 | 84.87 |
| Ours3 | 59.64 | 59.52 | 79.16 |

Ours3: An implementation of our learning framework with the hinge loss \( \ell_{\text{hinge}}(a, t) = (a - t)_+ \).
Specifically, we visualize the finer-grained comparison for 5 class pairs having the smallest frequency for each data, which is shown in Fig. 2. Note that we only present the results that are greater than 0.7 to have a clearer view of the differences across top algorithms. The results show that our improvement over the minority class pairs is even more significant, especially for the Balance, Ecoli, Page Blocks, SegmentImb, Shuttle, and Yeast.

Deep Learning Datasets. The performance comparisons are shown in the Table 2. Consequently, our proposed algorithm consistently outperforms all the competitors for all the datasets. Note that DeepAUC has a reasonable performance on the first two datasets. However, it fails in the last dataset. The possible reason here is that DeepAUC requires an extra set of parameters with the size of which depending on $N_C$. Hence, for a large dataset like iNaturalist, training DeepAUC becomes significantly more difficult than other methods.

Next, we show the finer-grained comparisons of the minority class pairs in Tables 3 and 4. For CIFAR-100-Imb and User-Imb, we adopt a similar to the traditional datasets where the performance comparisons in terms of $AUC_{i,j}$ on the class pairs $(i,j)$ with bottom-5 pairwise frequency $p_i p_j$. For the CIFAR-100-Imb dataset, the best algorithm among Ours1, Ours2, and Ours3 significantly outperform their best competitor for all the class pairs except the 4th one. Moreover, even though the hinge loss (Ours3) does not provide an improvement of MAUC1, it mitigates the imbalance issue by providing good performances on the first four minority pairs. For User-Imb, the corresponding improvements produced by our proposed methods are not that sharp as the previous results. The possible reasons is that the degree of imbalance for User-Imb is relatively moderate. These two traits automatically alleviate the imbalance issue in this dataset, making the difference between different imbalance-aware methods seem less significant.

For the iNaturalist 2017 Dataset, since the tail classes are insufficient to differentiate the performance from the different algorithms, we turn to report the result for bottom 0-5%, 5-10%, 11-15%, 16-20%, 21-25% class pairs. Again, we can see that our proposed algorithm outperforms the competitors consistently. To validate the effectiveness of our acceleration method, we report the average running time per epoch in Table 5 here. Accelerated refers to our proposed method. For the Selected method, the sample indexes of different classes are cached in different tensors. Naive refers to a straightforward implementation.

8 Conclusion

This paper provides an early study of AUC-guided machine learning for multiclass problems. Specifically, we propose a novel empirical risk minimization framework to learn scoring functions by optimizing the popular M metric. First, we show that a series of surrogate loss functions are Fisher consistent with the 0-1 loss based M metric. Moreover, we provide an empirical surrogate risk minimization framework to minimize the MAUC1 with guaranteed generalization upper bounds. Practically, we propose acceleration methods for three implementations of our proposed framework. Finally, empirical studies on 12 real-world datasets show the efficacy of our proposed algorithms.

Acknowledgments

This work was supported in part by the National Key R&D Program of China under Grant 2018AAA0102003, in part by the National Natural Science Foundation of China under Grants 61620106009, 62025604, 6181166002, 61931008, 61836002, and 61976202, in part by the Fundamental Research Funds for the Central Universities, in part by the National Postdoctoral Program for Innovative Talents under Grant BX2021298, in part by Youth Innovation Promotion Association CAS, and in part by the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDB28000000.

References

[1] S. Agarwal, “Surrogate regret bounds for bipartite ranking via strongly proper losses,” J. Mach. Learn. Res., vol. 15, no. 1, pp. 1653–1674, 2014.
[2] S. Agarwal, T. Graepel, R. Herbrich, S. Har-Peled, and D. Roth, “Generalization bounds for the area under the ROC curve,” J. Mach. Learn. Res., vol. 6, pp. 393–425, 2005.
[3] A. H. Alan and B. Raskutti, “Optimising area under the ROC curve using gradient descent,” in Proc. 21st Int. Conf. Mach. Learn., 2004, pp. 49–56.
[4] Z. Bai, X.-L. Zhang, and J. Chen, “Partial AUC optimization based deep speaker embeddings with class-center learning for text-independent speaker verification,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., 2020, pp. 6819–6823.
[5] F. L. Bartlett and S. Mendelson, “Rademacher and Gaussian complexities: Risk bounds and structural results,” J. Mach. Learn. Res., vol. 3, pp. 463–482, 2002.
[6] S. Barua, M. M. Islam, X. Yao, and K. Murase, “MWMOTE-majority weighted minority oversampling technique for imbalanced data set learning,” IEEE Trans. Knowl. Data Eng., vol. 26, no. 2, pp. 405–425, Feb. 2012.
F. Pan, S. Li, X. Ao, P. Tang, and Q. He, “Warm up cold-start advertisements: Improving CTR predictions via learning to rank ID embeddings,” in Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2019, pp. 695–704.

F. J. Provost and P. M. Domingos, “Tree Induction for probability-based ranking,” Mach. Learn., vol. 52, no. 3, pp. 199–215, 2003.

S. Rajaram and S. Agarwal, “Generalized AUC and G-AUC for k-partite ranking,” in Proc. NIPS Workshop Learn. Rank, 2005, pp. 18–23.

L. Rašialva, M. Szafranski, and G. Stempfel, “Chromatic PAC-Bayes bounds for non-ID data: Applications to ranking and stationary β-mixing processes,” J. Mach. Learn. Res., vol. 11, pp. 1927–1956, 2010.

H. W. Reeve and A. Kaban, “Optimistic bounds for multi-output prediction,” 2020, arXiv:2002.05769.

W. Shi, B. Gu, X. Li, and H. Huang, “Quadruply stochastic gradient method for large scale nonlinear semi-supervised ordinal regression AUC optimization,” in Proc. AAAI Conf. Artif. Intell., pp. 5734–5741, 2020.

M. R. Smith, T. Martinez, and C. Giraud-Carrier, “An instance level analysis of data complexity,” Mach. Learn., vol. 95, no. 2, pp. 225–256, 2019.

I. Tomek, “Two modifications of CNN,” IEEE Trans. Syst., Man, Cybern., vol. SMC-6, no. 11, pp. 769–772, Nov. 1976.

K. Uematsu and Y. Lee, “Statistical optimality in multipartite ranking and ordinal regression,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 5, pp. 1080–1094, May 2015.

N. Usunier, M.-R. Amini, and P. Gallinari, “A data-dependent generalisation error bound for the AUC,” in Proc. ICML Workshop ROC Anal. Mach. Learn., 2005.

N. Usunier, M. R. Amini, and P. Gallinari, “Generalization error bounds for classifiers trained with interdependent data,” in Proc. 18th Int. Conf. Adv. Neural Inf. Process. Syst., 2006, pp. 1369–1376.

S. Wang and L. L. Minku, “AUC estimation and concept drift detection for imbalanced data streams with multiple classes,” in Proc. Int. Joint Conf. Neural Netw., 2020, pp. 1–8.

H. Wu, Z. Hu, J. Jia, Y. Bu, X. He, and T.-S. Chua, “Mining unfollow behavior in large-scale online social networks via spatial-temporal interaction,” in Proc. AAAI Conf. Artif. Intell., 2020, vol. 34, pp. 254–261.

B. Yang, “The extension of the area under the receiver operating characteristic curve to multi-class problems,” in Proc. ISECS Int. Conf. Comput. Commun. Control, Manage., 2009, pp. 463–466.

Z. Yang, X. Shen, Y. Ying, and X. Yuan, “Stochastic AUC optimization with general loss,” Commun. Pure Appl. Anal., vol. 19, no. 8, pp. 4191–4212, 2020.

Y. Ying, L. Wen, and S. Lyu, “Stochastic online AUC maximization,” in Proc. 30th Int. Conf. Adv. Neural Inf. Process. Syst., 2016, pp. 451–459.

X. Zhang, A. Saha, and S. Vishwanathan, “Smoothing multivariate performance measures,” J. Mach. Learn. Res., vol. 13, no. 1, pp. 3623–3680, 2012.

P. Zhao, S. C. Hoi, R. Jin, and T. Yang, “Online AUC maximization,” in Proc. Int. Conf. Mach. Learn., 2011, pp. 233–240.

K. Zhou et al., “Sparse-GAN: Sparsity-constrained generative adversarial network for anomaly detection in retinal OCT image,” in Proc. IEEE Int. Symp. Biomed. Imag., 2020, pp. 1227–1213.

F. Zou, L. Shen, Z. Jie, W. Zhang, and W. Liu, “A sufficient condition for convergences of Adam and RMSProp,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 11127–11135.

Zhiyong Yang received the MSc degree in computer science and technology from the University of Science and Technology Beijing in 2017 and the PhD degree from the University of Chinese Academy of Sciences (UCAS) in 2021. He is currently a postdoctoral research fellow with UCAS. He has authored or coauthored several academic papers in top-tier international conferences and journals including the IEEE Transactions on Pattern Analysis and Machine Intelligence, ICML, NeurIPS, and CVPR. His research interests include machine learning and learning theory, with special focus on AUC optimization, meta-learning/multitask learning, and learning theory for recommender systems. He was a reviewer for several top-tier journals and conferences, including the IEEE Transactions on Pattern Analysis and Machine Intelligence, ICML, NeurIPS, and ICLR.

Qiangqian Xu (Senior Member, IEEE) received the BS degree in computer science from the China University of Mining and Technology in 2007 and the PhD degree in computer science from the University of Chinese Academy of Sciences in 2013. She is currently an associate professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. She has authored or coauthored more than 40 academic papers in prestigious international journals and conferences, including the IEEE Transactions on Pattern Analysis and Machine Intelligence, International Journal of Computer Vision, IEEE Transactions on Image Processing, NeurIPS, ICML, CVPR, and AAAI. Her research interests include statistical machine learning, with applications in multimedia and computer vision. She was a member of the professional committee of CAAI and a member of online program committee of VALSE.

Shilong Bao received the BS degree from the College of Computer Science and Technology, Qingdao University in 2018. He is currently working toward the the MS degree with the University of Chinese Academy of Sciences. His research interests include machine learning and data mining.

Xiaochun Cao (Senior Member, IEEE) received the BS and ME degrees in computer science from Beihang University, China, and the PhD degree in computer science from the University of Central Florida, USA. He is currently a professor with the Institute of Information Engineering, Chinese Academy of Sciences. He was a research scientist with ObjectVideo Inc. for three years. From 2008 to 2012, he was a professor with Tianjin University. He has authored and coauthored more than 100 journal and conference papers. In 2004 and 2010, he was the recipient of the Piero Zamperoni Best Student Paper Award at the International Conference on Pattern Recognition and his dissertation was nominated for the University Level Outstanding Dissertation Award. He is a fellow of IET. He is currently an associate editor for the IEEE Transactions on Image Processing, IEEE Transactions on Circuits and Systems for Video Technology, and IEEE Transactions on Multimedia.

Qingming Huang (Fellow, IEEE) received the bachelor’s degree in computer science and the PhD degree in computer engineering from the Harbin Institute of Technology, China, in 1988 and 1994, respectively. He is currently a chair professor with the University of Chinese Academy of Sciences and an adjunct research professor with the Institute of Computing Technology, Chinese Academy of Sciences. He has authored or coauthored more than 400 academic papers in prestigious international journals and top-level international conferences. His research interests include multimedia computing, image processing, computer vision, and pattern recognition. He is an associate editor for the IEEE Transactions on Circuits and Systems for Video Technology and Acta Automatica Sinica, and the reviewer of various international journals including the IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Image Processing, and IEEE Transactions on Multimedia. He was the general chair, program chair, track chair, and TPC member for various conferences, including ACM Multimedia, CVPR, ICCV, ICME, ICMR, PCM, BigMM, and PSIVT.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.