Stochastic Optimization of Areas Under Precision-Recall Curves with Provable Convergence

Anonymous Author(s)
Affiliation
Address
email

Abstract

Areas under ROC (AUROC) and precision-recall curves (AUPRC) are common metrics for evaluating classification performance for imbalanced problems. Compared with AUROC, AUPRC is a more appropriate metric for highly imbalanced datasets. While stochastic optimization of AUROC has been studied extensively, principled stochastic optimization of AUPRC has been rarely explored. In this work, we propose a principled technical method to optimize AUPRC for deep learning. Our approach is based on maximizing the averaged precision (AP), which is an unbiased point estimator of AUPRC. We cast the objective into a sum of *dependent compositional functions* with inner functions dependent on random variables of the outer level. We propose efficient adaptive and non-adaptive stochastic algorithms with *provable convergence guarantee under mild conditions* by leveraging recent advances in stochastic compositional optimization. Extensive experimental results on image and graph datasets demonstrate that our proposed method outperforms prior methods on imbalanced problems in terms of AUPRC. To the best of our knowledge, our work represents the first attempt to optimize AUPRC with provable convergence.

1 Introduction

Although deep learning (DL) has achieved tremendous success in various domains, the standard DL methods have reached a plateau as the traditional objective functions in DL are no longer sufficient to model all requirements in new applications, which slows down the democratization of AI. For instance, in healthcare applications, data is often highly imbalanced, e.g., patients suffering from rare diseases are much less than those suffering from common diseases. In these applications, accuracy (the proportion of correctly predicted examples) is deemed as an inappropriate metric for evaluating the performance of a classifier. Instead, area under the curve (AUC), including area under ROC curve (AUROC) and area under the Precision-Recall curve (AUPRC), is widely used for assessing the performance of a model. However, optimizing accuracy on training data does not necessarily lead to a satisfactory solution to maximizing AUC [12].

To break the bottleneck for further advancement, DL must be empowered with the capability of efficiently handling novel objectives such as AUC. Recent studies have demonstrated great success along this direction by maximizing AUROC [57]. For example, Yuan et al. [57] proposed a robust deep AUROC maximization method with provable convergence and achieved great success for classification of medical image data. However, to the best of our knowledge, novel DL by maximizing AUPRC has not yet been studied thoroughly. Previous studies [14, 20] have found that when dealing with highly skewed datasets, Precision-Recall (PR) curves could give a more informative picture of an algorithm's performance, which entails the development of efficient stochastic optimization algorithms for DL by maximizing AUPRC.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Compared with maximizing AUROC, maximizing AUPRC is more challenging. The challenges for optimization of AUPRC are two-fold. First, the analytical form of AUPRC by definition involves a complicated integral that is not readily estimated from model predictions of training examples. In practice, AUPRC is usually computed based on some point estimators, e.g., trapezoidal estimators and interpolation estimators of empirical curves, non-parametric average precision estimator, and parametric binomial estimator [3]. Among these estimators, non-parametric average precision (AP) is an unbiased estimate in the limit and can be directly computed based on the prediction scores of samples, which lends itself well to the task of optimization of model parameters. Second, a surrogate function for AP is highly complicated and non-convex. In particular, an unbiased stochastic gradient is not readily computed, which makes existing stochastic algorithms such as SGD provide no convergence guarantee. Most existing works for maximizing AP-like function focus on how to compute an (approximate) gradient of the objective function [4, 6, 8, 11, 24, 39, 41, 43, 45, 46], which leave stochastic optimization of AP with provable convergence as an open question.

Can we design direct stochastic optimization algorithms in SGD-style and Adam-style for maximizing AP with provable convergence guarantee?

In this paper, we propose a systematic and principled solution for addressing this question towards maximizing AUPRC for DL. By using a surrogate loss in lieu of the indicator function in the definition of AP, we cast the objective into a sum of non-convex compositional functions, which resembles a two-level stochastic compositional optimization problem studied in the literature [50, 51]. However, different from existing two-level stochastic compositional functions, the inner functions in our problem are dependent on the random variable of the outer level, which requires us developing a tailored stochastic update for computing an error-controlled stochastic gradient estimator. Specifically, a key feature of the proposed method is to maintain and update two scalar quantities associated with each positive example for estimating the stochastic gradient of the individual precision score at the threshold specified by its prediction score. By leveraging recent advances in stochastic compositional optimization, we propose both adaptive (Adam-style) and non-adaptive (SGD-style) algorithms, and establish their convergence under mild conditions. We conduct comprehensive empirical studies on class imbalanced graph and image datasets for learning graph neural networks and deep convolutional neural networks, respectively. We demonstrate that the proposed method can consistently outperform prior approaches in terms of AUPRC. In addition, we show that our method achieves better results when the sample distribution is highly imbalanced between classes and is insensitive to mini-batch size.

2 Related Work

AUROC Optimization. AUROC optimization [1] has attracted significant attention in the literature. Recent success of DL by optimizing AUROC on large-scale medical image data has demonstrated the importance of large-scale stochastic optimization algorithms and the necessity of accurate surrogate function [57]. Earlier papers [25, 28] focus on learning a linear model based on the pairwise surrogate loss and could suffer from a high computational cost, which could be as high as quadratic of the size of training data. To address the computational challenge, online and stochastic optimization algorithms have been proposed [18, 36, 42, 55, 60]. Recently, [21, 22, 37] proposed stochastic deep AUC maximization algorithms by formulating the problem as non-convex strongly-concave min-max optimization problem, and derived fast convergence rate under PL condition, and in federated learning setting as well [21]. More recently, Yuan et al. [57] demonstrated the success of their methods on medical image classification tasks, e.g., X-ray image classification, melanoma classification based on skin images. However, an algorithm that maximizes the AUROC might not necessarily maximize AUPRC, which entails the development of efficient algorithms for DL by maximizing AUPRC.

AUPRC Optimization. AUPRC optimization is much more challenging than AUROC optimization since the objective is even not decomposable over pairs of examples. Although AUPRC optimization has been considered in the literature (cf. [15, 45] and references therein), efficient scalable algorithms for DL with provable convergence guarantee is still lacking. Some earlier works tackled this problem by using traditional optimization techniques, e.g., hill climbing search [38], cutting-plane method [58], dynamic programming [48], and by developing acceleration techniques in the framework of SVM [40]. These approaches are not scalable to big data for DL. There is a long list of studies in information retrieval [5, 11, 39, 45] and computer vision [4, 6, 8, 9, 24, 41, 46, 43], which have made efforts

[1] In the literature, AUROC optimization is simply referred to as AUC optimization.

towards maximizing the AP score. However, most of them focus on how to compute an approximate gradient of the AP function or its smooth approximation, and provide no convergence guarantee for stochastic optimization based on mini-batch averaging. Due to lack of principled design, these previous methods when applied to deep learning are sensitive to the mini-batch size [6, 45, 46] and usually require a large mini-batch size in order to achieve good performance. In contrast, our stochastic algorithms are designed in a principled way to guarantee convergence without requiring a large mini-batch size as confirmed by our studies as well. Recently, [15] formulates the objective function as a constrained optimization problem using a surrogate function, and then casts it into a min-max saddle-point problem, which facilitates the use of stochastic min-max algorithms. However, they do not provide any convergence analysis for AUPRC maximization. In contrast, this is the first work that directly optimizes a surrogate function of AP (an unbaised estimator of AUPRC in the limit) and provides theoretical convergence guarantee for the proposed stochastic algorithms.

Stochastic Compositional Optimization. Optimization of a two-level compositional function in the form of $\mathbb{E}_\xi[f(\mathbb{E}_\zeta[g(\mathbf{w};\zeta)];\xi)]$ where ξ and ζ are independent random variables, or its finite-sum variant has been studied extensively in the literature [1, 10, 50, 27, 31, 32, 34, 35, 44, 51, 56, 59]. In this paper, we formulate the surrogate function of AP into a similar but more complicated two-level compositional function of the form $\mathbb{E}_\xi[f(\mathbb{E}_\zeta g(\mathbf{w};\zeta,\xi))]$, where ξ and ζ are independent and ξ has a finite support. The key difference between our formulated compositional function and the ones considered in previous work is that the inner function $g(\mathbf{w};\zeta,\xi)$ also depends on the random variable ξ of the outer level. Such subtle difference will complicate the algorithm design and the convergence analysis as well. Nevertheless, the proposed algorithm and its convergence analysis are built on previous studies of stochastic two-level compositional optimization.

3 The Proposed Method

Notations. We consider binary classification problem. Denote by $(\mathbf{x}, y)$ a data pair, where $\mathbf{x} \in \mathbb{R}^d$ denotes the input data and $y \in \{1, -1\}$ denotes its class label. Let $h(\mathbf{x}) = h_\mathbf{w}(\mathbf{x})$ denote the predictive function parameterized by a parameter vector $\mathbf{w} \in \mathbb{R}^D$ (e.g., a deep neural network). Denote by $\mathbf{I}(\cdot)$ an indicator function that outputs 1 if the argument is true and zero otherwise. To facilitate the presentation, denote by X a random data, by Y its label and by $F = h(X)$ its prediction score. Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ denote the set of all training examples and $\mathcal{D}_+ = \{\mathbf{x}_i : y_i = 1\}$ denote the set of all positive examples. Let $n_+ = |\mathcal{D}_+|$ denote the number of positive examples. $\mathbf{x}_i \sim \mathcal{D}$ means that $\mathbf{x}_i$ is randomly sampled from $\mathcal{D}$.

3.1 Background on AUPRC and its estimator AP

Following the work of Bamber [2], AUPRC is an average of the precision weighted by the probability of a given threshold, which can be expressed as

$$A = \int_{-\infty}^{\infty} \Pr(Y = 1 | F \geq c) d \Pr(F \leq c | Y = 1),$$

where $\Pr(Y = 1|F \geq c)$ is the precision at the threshold value of c. The above integral is an importance-sampled Monte Carlo integral, by which we may interpret AUPRC as the fraction of positive examples among those examples whose output values exceed a randomly selected threshold $c \sim F(X)|Y = 1$.

For a finite set of examples $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ with the prediction score for each example $\mathbf{x}_i$ given by $h_\mathbf{w}(\mathbf{x}_i)$, we consider to use AP to approximate AUPRC, which is given by

$$\text{AP} = \frac{1}{n_+} \sum_{i=1}^{n} \mathbf{I}(y_i = 1) \frac{\sum_{s=1}^{n} \mathbf{I}(y_s = 1)\mathbf{I}(h_\mathbf{w}(\mathbf{x}_s) \geq h_\mathbf{w}(\mathbf{x}_i))}{\sum_{s=1}^{n} \mathbf{I}(h_\mathbf{w}(\mathbf{x}_s) \geq h_\mathbf{w}(\mathbf{x}_i))}, \tag{1}$$

where n_+ denotes the number of positive examples. It can be shown that AP is an unbiased estimator in the limit $n \to \infty$ [3].

However, the non-continuous indicator function $\mathbf{I}(h_\mathbf{w}(\mathbf{x}_s) \geq h_\mathbf{w}(\mathbf{x}_i))$ in both numerator and denominator in (1) makes the optimization non-tractable. To tackle this, we use a loss function $\ell(\mathbf{w}; \mathbf{x}_s, \mathbf{x}_i)$ as a surrogate function of $\mathbf{I}(h_\mathbf{w}(\mathbf{x}_s) \geq h_\mathbf{w}(\mathbf{x}_i))$. One can consider different surrogate losses, e.g., hinge loss, squared hinge loss, and smoothed hinge loss, and exponential loss. In this paper, we will consider a smooth surrogate loss function to facilitate the development of an optimization algorithm,

e.g., a squared hinge loss $\ell(\mathbf{w};\mathbf{x}_s;\mathbf{x}_i) = (\max\{m - (h_{\mathbf{w}}(\mathbf{x}_i) - h_{\mathbf{w}}(\mathbf{x}_s)), 0\})^2$, where m is a margin parameter. Note that we do not require ℓ to be a convex function, hence one can also consider non-convex surrogate loss such as ramp loss. As a result, our problem becomes

$$\min_{\mathbf{w}} P(\mathbf{w}) = \frac{1}{n_+} \sum_{\mathbf{x}_i \in \mathcal{D}_+} \frac{-\sum_{s=1}^{n} \mathbf{I}(y_s = 1)\ell(\mathbf{w};\mathbf{x}_s;\mathbf{x}_i)}{\sum_{s=1}^{n} \ell(\mathbf{w};\mathbf{x}_s;\mathbf{x}_i)}. \tag{2}$$

3.2 Stochastic Optimization of AP (SOAP)

We cast the problem into a finite-sum of compositional functions. To this end, let us define a few notations:

$$\begin{aligned} &g(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i) = [g_1(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i), g_2(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)]^\top = [\ell(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)\mathbf{I}(y_j = 1), \ell(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)]^\top \\ &g_{\mathbf{x}_i}(\mathbf{w}) = \mathbb{E}_{\mathbf{x}_j \sim \mathcal{D}}[g(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)], \end{aligned} \tag{3}$$

where $g_{\mathbf{x}_i}(\mathbf{w}) : \mathbb{R}^d \to \mathbb{R}^2$. Let $f(\mathbf{s}) = -\frac{s_1}{s_2} : \mathbb{R}^2 \to \mathbb{R}$. Then, we can write the objective function for maximizing AP as a sum of compositional functions:

$$P(\mathbf{w}) = \frac{1}{n_+} \sum_{\mathbf{x}_i \in \mathcal{D}_+} f(g_{\mathbf{x}_i}(\mathbf{w})) = \mathbb{E}_{\mathbf{x}_i \sim \mathcal{D}_+}[f(g_{\mathbf{x}_i}(\mathbf{w}))]. \tag{4}$$

We refer to the above problem as an instance of **two-level stochastic dependent compositional functions**. It is similar to the two-level stochastic compositional functions considered in literature [50, 51] but with a subtle difference. The difference is that in our formulation the inner function $g_{\mathbf{x}_i}(\mathbf{w}) = \mathbb{E}_{\mathbf{x}_j \sim \mathcal{D}}[g(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)]$ depends on the random variable $\mathbf{x}_i$ of the outer level. This difference makes the proposed algorithm slightly complicated by estimating $g_{\mathbf{x}_i}(\mathbf{w})$ separately for each positive example. It also complicates the analysis of the proposed algorithms. Nevertheless, we can still employ the techniques developed for optimizing stochastic compositional functions to design the algorithms and develop the analysis for optimizing the objective (4).

In order to motivate the proposed method, let us consider how to compute the gradient of $P(\mathbf{w})$. Let the gradient of $g_{\mathbf{x}_i}(\mathbf{w})$ be denoted by $\nabla_{\mathbf{w}} g_{\mathbf{x}_i}(\mathbf{w})^\top = (\nabla_{\mathbf{w}}[g_{\mathbf{x}_i}(\mathbf{w})]_1, \nabla_{\mathbf{w}}[g_{\mathbf{x}_i}(\mathbf{w})]_2)$. Then we have

$$\begin{aligned} \nabla_{\mathbf{w}} P(\mathbf{w}) &= \frac{1}{n_+} \sum_{\mathbf{x}_i \in \mathcal{D}_+} \nabla_{\mathbf{w}} g_{\mathbf{x}_i}(\mathbf{w})^\top \nabla f(g_{\mathbf{x}_i}(\mathbf{w})) \\ &= \frac{1}{n_+} \sum_{\mathbf{x}_i \in \mathcal{D}_+} \nabla_{\mathbf{w}} g_{\mathbf{x}_i}(\mathbf{w})^\top \left(\frac{-1}{[g_{\mathbf{x}_i}(\mathbf{w})]_2}, \frac{[g_{\mathbf{x}_i}(\mathbf{w})]_1}{([g_{\mathbf{x}_i}(\mathbf{w})]_2)^2} \right)^\top. \end{aligned} \tag{5}$$

The major cost for computing $\nabla_{\mathbf{w}} P(\mathbf{w})$ lies at evaluating $g_{\mathbf{x}_i}(\mathbf{w})$ and its gradient $\nabla_{\mathbf{w}} g_{\mathbf{x}_i}(\mathbf{w})$, which involves passing through all examples in $\mathcal{D}$.

To this end, we will approximate these quantities by stochastic samples. The gradient $\nabla_{\mathbf{w}} g_{\mathbf{x}_i}(\mathbf{w})$ can be simply approximated by the stochastic gradient, i.e.,

$$\widehat{\nabla}_{\mathbf{w}} g_{\mathbf{x}_i}(\mathbf{w}) = \begin{pmatrix} \frac{1}{B} \sum_{\mathbf{x}_j \in \mathcal{B}} \mathbf{I}(y_j = 1) \nabla \ell(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i) \\ \frac{1}{B} \sum_{\mathbf{x}_j \in \mathcal{B}} \nabla \ell(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i) \end{pmatrix}, \tag{6}$$

where $\mathcal{B}$ denote a set of B random samples from $\mathcal{D}$. For estimating $g_{\mathbf{x}_i}(\mathbf{w}) = \mathbb{E}_{\mathbf{x}_j \sim \mathcal{D}} g(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)$, however, we need to ensure its approximation error is controllable due to the compositional structure such that the convergence can be guaranteed. We borrow a technique from the literature of stochastic compositional optimization [50] by using moving average estimator for estimating $g_{\mathbf{x}_i}(\mathbf{w})$ for all positive examples. To this end, we will maintain a matrix $\mathbf{u} = [\mathbf{u}^1, \mathbf{u}^2]$ with each column indexable by any positive example, i.e., $\mathbf{u}^1_{\mathbf{x}_i}, \mathbf{u}^2_{\mathbf{x}_i}$ correspond to the moving average estimator of $[g_{\mathbf{x}_i}(\mathbf{w})]_1$ and $[g_{\mathbf{x}_i}(\mathbf{w})]_2$, respectively. The matrix $\mathbf{u}$ is updated by the subroutine UG in Algorithm 2, where $\gamma \in (0, 1)$ is a parameter. It is notable that in Step 3 of Algorithm 2, we clip the moving average update of $\mathbf{u}^2_{\mathbf{x}_i}$ by a lower bound u_0, which is a given parameter. This step can ensure the division in computing the stochastic gradient estimator in (7) always valid and is also important for convergence analysis. With these stochastic estimators, we can compute an estimate of $\nabla P(\mathbf{w})$ by equation (7), where $\mathcal{B}_+$ includes a batch of sampled positive data. With this stochastic gradient estimator, we can employ SGD-style method and Adam-style shown in Algorithm 3 to update the model parameter $\mathbf{w}$. The final algorithm named as SOAP is presented in Algorithm 1.

Algorithm 1: SOAP

Input: γ, α, u_0, and other parameters for SGD-stype update or Adam-stype update.
Initialize $\mathbf{w}_0 \in \mathbb{R}^d$, $\mathbf{u} \in \mathbb{R}^{|n_+|\times 2}$
for $t = 1, \ldots, T$ **do**
  Draw a batch of B_+ positive samples denoted by $\mathcal{B}_+$.
  Draw a batch of B samples denoted by $\mathcal{B}$.
  $\mathbf{u} = \text{UG}(\mathcal{B}, \mathcal{B}_+, \mathbf{u}, \mathbf{w}_t, \gamma, u_0)$
  Compute (biased) Stochastic Gradient Estimator

$$G(\mathbf{w}_t) = \frac{1}{B_+} \sum_{\mathbf{x}_i \in \mathcal{B}_+} \sum_{\mathbf{x}_j \in \mathcal{B}} \frac{(\mathbf{u}^1_{\mathbf{x}_i} - \mathbf{u}^2_{\mathbf{x}_i} \mathbf{I}(\mathbf{y}_j = 1)) \nabla \ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)}{B(\mathbf{u}^2_{\mathbf{x}_i})^2} \tag{7}$$

  Update $\mathbf{w}_{t+1}$ by a SGD-style method or by a Adam-style method

$$\mathbf{w}_{t+1} = \text{UW}(\mathbf{w}_t, G(\mathbf{w}_t))$$

end for
Return: last solution.

Algorithm 2: UG($\mathcal{B}, \mathcal{B}_+, \mathbf{u}, \mathbf{w}_t, \gamma, u_0$)

for each positive $\mathbf{x}_i \in \mathcal{B}_+$ **do**
  Compute

$$[\tilde{g}_{\mathbf{x}_i}(\mathbf{w}_t)]_1 = \frac{1}{|\mathcal{B}|} \sum_{\substack{x_j \in \mathcal{B} \\ y_j = 1}} \ell(\mathbf{w}_t; \mathbf{x}_j, \mathbf{x}_i)$$

$$[\tilde{g}_{\mathbf{x}_i}(\mathbf{w}_t)]_2 = \frac{1}{|\mathcal{B}|} \sum_{\mathbf{x}_j \in \mathcal{B}} \ell(\mathbf{w}_t; \mathbf{x}_j, \mathbf{x}_i)$$

  Compute
  $\mathbf{u}^1_{\mathbf{x}_i} = (1-\gamma)\mathbf{u}^1_{\mathbf{x}_i} + \gamma[\tilde{g}_{\mathbf{x}_i}(\mathbf{w}_t)]_1$
  $\mathbf{u}^2_{\mathbf{x}_i} = \max((1-\gamma)\mathbf{u}^2_{\mathbf{x}_i} + \gamma[\tilde{g}_{\mathbf{x}_i}(\mathbf{w}_t)]_2, u_0)$
end for
Return $\mathbf{u}$

Algorithm 3: UW($\mathbf{w}_t, G(\mathbf{w}_t)$)

Option 1: SGD-style update (paras: α)

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha G(\mathbf{w}_t)$$

Option 2: Adam-style update (paras: $\alpha, \epsilon, \eta_1, \eta_2$)

$$h_{t+1} = \eta_1 h_t + (1-\eta_1) G(\mathbf{w}_t)$$

$$v_{t+1} = \eta_2 \hat{v}_t + (1-\eta_2)(G(\mathbf{w}_t))^2$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \frac{h_{t+1}}{\sqrt{\epsilon + \hat{v}_{t+1}}}$$

where $\hat{v}_t = v_t$ (Adam) or
$\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$ (AMSGrad)
Return: $\mathbf{w}_{t+1}$

3.3 Convergence Analysis

In this subsection, we present the convergence results of SOAP and also highlight its convergence analysis. To this end, we first present the following assumption.

Assumption 1. *Assume that (a) there exists Δ_0 such that $P(\mathbf{w}_0) - \min_{\mathbf{w}} P(\mathbf{w}) \leq \Delta_0$; (b) there exist $C, M > 0$ such that $\ell(\mathbf{w}; \mathbf{x}_i, \mathbf{x}_i) \geq C$ for any $\mathbf{x}_i \in \mathcal{D}_+$, $\ell(\mathbf{w}; \mathbf{x}_i, \mathbf{x}_j) \leq M$, and $\ell(\mathbf{w}; \mathbf{x}_i, \mathbf{x}_j)$ is Lipscthiz continuous and smooth with respect to $\mathbf{w}$ for any $\mathbf{x}_i \in \mathcal{D}_+, \mathbf{x}_j \in \mathcal{D}$; (c) there exists $V > 0$ such that $\mathbb{E}_{\mathbf{x}_j \sim \mathcal{D}}[\|g(\mathbf{w}; \mathbf{x}_i, \mathbf{x}_j) - g_{\mathbf{x}_i}(\mathbf{w})\|^2] \leq V$, and $\mathbb{E}_{\mathbf{x}_j \sim \mathcal{D}}[\|\nabla g(\mathbf{w}; \mathbf{x}_i, \mathbf{x}_j) - \nabla g_{\mathbf{x}_i}(\mathbf{w})\|^2] \leq V$ for any $\mathbf{x}_i$.*

With a bounded score function $h_{\mathbf{w}}(\mathbf{x})$ the above assumption can be easily satisfied. Based on the above assumption, we can prove that the objective function $P(\mathbf{w})$ is smooth.

Lemma 1. *Suppose Assumption 1 holds, then there exists $L > 0$ such that $P(\cdot)$ is L-smooth. In addition, there exists $u_0 \geq C/n$ such that $g_{\mathbf{x}_i}(\mathbf{w}) \in \Omega = \{\mathbf{u} \in \mathbb{R}^2, 0 \leq [\mathbf{u}]_1 \leq M, u_0 \leq [\mathbf{u}]_2 \leq M\}$, $\forall \mathbf{x}_i \in \mathcal{D}_+$.*

Next, we highlight the convergence analysis of SOAP employing the SGD-stype update and include that for employing Adam-style update in the supplement. Without loss of generality, we assume $|\mathcal{B}_+| = 1$ and the positive sample in $\mathcal{B}_+$ is randomly selected from $\mathcal{D}_+$ with replacement. When the context is clear, we abuse the notations $g_i(\mathbf{w})$ and $\mathbf{u}_i$ to denote $g_{\mathbf{x}_i}(\mathbf{w})$ and $\mathbf{u}_{\mathbf{x}_i}$ below, respectively. We first establish the following lemma following the analysis of non-convex optimization.

Lemma 2. *With $\alpha \leq 1/2$, running T iterations of SOAP (SGD-style) updates, we have*

$$\frac{\alpha}{2}\mathbb{E}[\sum_{t=1}^{T} \|\nabla P(\mathbf{w}_t)\|^2] \leq \mathbb{E}[\sum_{t}(P(\mathbf{w}_t) - P(\mathbf{w}_{t+1}))] + \frac{\alpha C_1}{2}\mathbb{E}[\sum_{t=1}^{T} \|g_{i_t}(\mathbf{w}_t) - \mathbf{u}_{i_t}\|^2] + \alpha^2 T C_2,$$

where i_t denotes the index of the sampled positive data at iteration t, C_1 and C_2 are proper constants.

Our key contribution is the following lemma that bounds the second term in the above upper bound.

Lemma 3. *Suppose Assumption 1 holds, with* $\mathbf{u}$ *initialized by (6) for every* $\mathbf{x}_i \in \mathcal{D}_+$ *we have*

$$\mathbb{E}[\sum_{t=1}^{T} \|g_{i_t}(\mathbf{w}_t) - \mathbf{u}_{i_t}\|^2] \leq \frac{n_+ V}{\gamma} + 2\gamma V T + 2\frac{n_+^2 \alpha^2 T C_3}{\gamma^2}, \tag{8}$$

where C_3 *is a proper constant.*

Remark: The innovation of proving the above lemma is by grouping $\mathbf{u}_{i_t}, t = 1, \ldots, T$ into n_+ groups corresponding to the n_+ positive examples, and then establishing the recursion of the error $\|g_{i_t}(\mathbf{w}_t) - \mathbf{u}_{i_t}\|^2$ within each group, and then summing up these recursions together.

Based on the two lemmas above, we establish the following convergence of SOAP with a SGD-style update.

Theorem 1. *Suppose Assumption 1 holds, let the parameters be* $\alpha = \frac{1}{n_+^{2/5} T^{3/5}}, \gamma = \frac{n_+^{2/5}}{T^{2/5}}$, $\forall\, t \in 1, \cdots, T$, *and* $T > n_+$. *Then after running* T *iterations, SOAP with a SGD-style update satisfies* $\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} \|\nabla P(\mathbf{w}_t)\|^2\right] \leq O(\frac{n_+^{2/5}}{T^{2/5}})$, *where O suppresses constant numbers.*

Remark: To the best of our knowledge, this is the first time a stochastic algorithm was proved to converge for AP maximization.

Similarly, we can establish the following convergence of SOAP by employing an Adam-style update, specifically the AMSGrad update.

Theorem 2. *Suppose Assumption 1 holds, let the parameters* $\eta_1 \leq \sqrt{\eta_2} \leq 1$, $\alpha = \frac{1}{n_+^{2/5} T^{3/5}}, \gamma = \frac{n_+^{2/5}}{T^{2/5}}$, $\forall\, t \in 1, \cdots, T$, *and* $T > n_+$. *Then after running T iterations, SOAP with an AMSGRAD update satisfies* $\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} \|\nabla P(\mathbf{w}_t)\|^2\right] \leq O(\frac{n_+^{2/5}}{T^{2/5}})$, *where O suppresses constant numbers.*

4 Experiments

In this section, we evaluate the proposed method through comprehensive experiments on imbalanced datasets. We show that the proposed method can outperform prior state-of-the-art methods for imbalanced classification problems. In addition, we conduct experiments on (i) the effects of imbalance ratio; (ii) the insensitivity to batch size and (iii) the convergence speed on testing data; and observe that our method (i) is more advantageous when data is more imbalanced, (ii) is not sensitive to batch size, and (iii) converges faster than baseline methods.

Our proposed loss function and optimization are independent of specific datasets and tasks. Therefore, we perform experiments on both graph and image prediction tasks. In particular, the graph prediction tasks in the contexts of molecular property prediction and drug discovery suffer from very severe imbalance problems as positive labels are very rare while negative samples are abundantly available. Thus, we choose to use graph data intensively in our experiments. Additionally, the graph data we use allow us to vary the imbalance ratio to observe the performance change of different methods.

In all experiments, we compare our method with the regular cross entropy loss (**CE**), two surrogate AP optimization methods and several other state-of-the-art approaches for imbalanced classification problems. For the imbalanced classification approaches, **CB-CE** refers to a method using a class-balanced weighed cross entropy loss function, in which the weights for positive and negative samples are adjusted with the strategy proposed by Cui et al. [13]. **Focal** is to up-weight the penalty on hard examples using focal loss [33]. **LDAM** refers to training with label-distribution-aware margin loss [7]. **AUC-M** [57] is a surrogate loss proposed to optimize AUROC. In addition, we compare with three methods for optimizing AUPRC or AP, namely, the **MinMax method** [15] - a method for optimizing a discrete approximation of AUPRC, **SmoothAP** [4] - a method to optimize a smoothed approximation of AP, and **FastAP** - a method that uses soft histogram binning to approximate the gradient of AP [6]. For all of these methods, we use the Adam-style optimization algorithms unless specified otherwise. We refer to **imbalance ratio** as the number of positive samples over the total number of examples of a considered set. The hyper-parameters of all methods are fine tuned using cross-validation with training/validation splits mentioned below. For AP maximization methods, we use a sigmoid function to produce the prediction score. For simplicity, we set $u_0 = 0$ for SOAP and encounter no numerical problems in experiments.

Table 1: The test AUPRC on the image datasets with two ResNet models. We report the average AUPRC and standard deviation (within brackets) over 5 runs.

Datasets	CIFAR-10		CIFAR-100	
Networks	ResNet18	ResNet34	ResNet18	ResNet34
CE	0.7298 ($\pm$ 0.0058)	0.7201($\pm$ 0.0031)	0.6035 ($\pm$ 0.0031)	0.6047 ($\pm$ 0.0028)
CB-CE	0.7528 ($\pm$ 0.0038)	0.7428($\pm$0.0028)	0.6066 ($\pm$ 0.0023)	0.6108($\pm$ 0.0028)
Focal	0.7554($\pm$ 0.0023)	0.7383($\pm$ 0.0031)	0.6104($\pm$ 0.0024)	0.6198($\pm$ 0.0019)
LDAM	0.7501 ($\pm$ 0.0019)	0.7466($\pm$ 0.0026)	0.6012 ($\pm$ 0.0024)	0.6207($\pm$ 0.0015)
AUC-M	0.7529($\pm$ 0.0029)	0.7510($\pm$ 0.0033)	0.6061 ($\pm$ 0.0031)	0.6124($\pm$ 0.0026)
SmoothAP	0.7542 ($\pm$ 0.0034)	0.7459 ($\pm$ 0.0032)	0.6132($\pm$ 0.0015)	0.6256 ($\pm$ 0.0015)
FastAP	0.5949 ($\pm$ 0.0035)	0.6011 ($\pm$ 0.0023)	0.5446($\pm$ 0.0043)	0.5432($\pm$ 0.0051)
MinMax	0.7226 ($\pm$ 0.0025)	0.7183($\pm$ 0.0043)	0.6077($\pm$ 0.0032)	0.6164($\pm$ 0.0030)
SOAP	**0.7644**($\pm$ 0.0032)	**0.7572**($\pm$ 0.0025)	**0.6237** ($\pm$ 0.0015)	**0.6260** ($\pm$ 0.0016)

4.1 Image Classification

Data. We first conduct experiments on three image datasets: CIFAR10, CIFAR100 and Melanoma dataset [47]. We construct imbalanced version of CIFAR10 and CIFAR100 for binary classification. In particular, for each dataset we manually take the first half of classes as positive class and last half of classes as negative class. To construct highly imbalanced data, we remove 98% of the positive images from the training data and keep the test data unchanged (i.e., the testing data is still balanced). And we split the training dataset into train/validation set at 80%/20% ratio. The Melanoma dataset is from a medical image Kaggle competition, which serves as a natural real imbalanced image dataset. It contains 33,126 labeled medical images, among which 584 images are related to malignant melanoma and labelled as positive samples. Since the test set used by Kaggle organization is not available, we manually split the training data into train/validation/test set at 80%/10%/10% ratio and report the achieved AUPRC on the test set by our method and baselines. The images of Melanoma dataset are always resized to have a resolution of 384×384 in our experiments.

Setup. We use two ResNet [23] models, *i.e.*, ResNet18 and ResNet34, as the backbone networks for image classification. For all methods except for CE, the ResNet models are initialized with a model pre-trained by CE with an Adam optimizer for running 60 epochs. Momentum parameters of Adam are set as their default value in [29]. We tune the learning rate in a range $\{$1e-5, 1e-4, 1e-3, 1e-2$\}$ and the weight decay parameter in a range $\{$1e-6, 1e-5, 1e-4$\}$. Then the last fully connected layer is randomly re-initialized and the network is trained by different methods with the same weight decay parameter but other hyper-parameters individually tuned for fair comparison, e.g., we tune γ of SOAP in a range $\{0.9, 0.99, 0.999\}$, and tune m in $\{0.5, 1, 2, 5, 10\}$. We refer to this scheme as two-stage training, which is widely used for imbalanced data [57]. We consistently observe that this strategy can bring the model to a good initialization state and improve the final performance of our method and baselines.

Results. Table 1 shows the AUPRC on testing sets of CIFAR-10 and CIFAR-100. We report the results on Melanoma in Table 3. We can observe that the proposed method SOAP outperforms all baselines. It is also striking to see that on Melanoma dataset, our proposed SOAP can outperform all baselines by a large margin, and all other methods have very poor performance. The reason is that the testing set of Melanoma is also imbalanced (imbalanced ratio=1.72%), while the testing sets of CIFAR-10 and CIFAR-100 are balanced. We also observe that the AUROC maximization (AUC-M) does not necessarily optimize AUPRC. We also plot the final PR curves in Figure 2 in the supplement.

4.2 Graph Classification for Molecular Property Prediction

Data. To further demonstrate the advantages of our method, we conduct experiments on two graph classification datasets. We use the datasets HIV and MUV from the MoleculeNet [53], which is a benchmark for molecular property prediction. The HIV dataset has 41,913 molecules from the Drug Therapeutics Program (DTP), and the positive samples are molecules tested to have inhibition ability to HIV. The MUV dataset has 93,127 molecules from the PubChem library, and molecules are labelled by whether a bioassay property exists or not. Note that the MUV dataset provides labels of 17 properties in total and we only conduct experiments to predict the third property as this property is more imbalanced. The percentage of positive samples in HIV and MUV datasets are 3.51% and 0.20%, respectively. We use the split of train/validation/test set provided by MoleculeNet. Molecules are treated as 2D graphs in our experiments, and we use the feature extraction procedure

Table 2: The test AUPRC values on the HIV and MUV datasets with three graph neural network models. We report the average AUPRC and standard deviation (within brackets) over 3 runs.

Dataset	Method	GINE	MPNN	ML-MPNN
HIV	CE	0.2774 (± 0.0101)	0.3197 (± 0.0050)	0.2988 (± 0.0076)
	CB-CE	0.2901 (± 0.0078)	0.3056 (± 0.0018)	0.3291 (± 0.0189)
	Focal	0.2976 (± 0.0008)	0.3136 (± 0.0197)	0.3279 (± 0.0173)
	LDAM	0.2986 (± 0.0095)	0.2994 (± 0.0128)	0.3044 (± 0.0116)
	AUC-M	0.3057 (± 0.0035)	0.2786 (± 0.0456)	0.3305 (± 0.0165)
	SmothAP	0.0274 (± 0.0007)	0.3276 (± 0.0063)	0.3235 (± 0.0092)
	FastAP	0.0270 (± 0.0002)	0.0826 (± 0.0112)	0.0202 (± 0.0002)
	MinMax	0.2824 (± 0.0073)	0.3119 (± 0.0075)	0.3098 (± 0.0167)
	SOAP	**0.3145 (± 0.0073)**	**0.3401 (± 0.0045)**	**0.3547 (± 0.0077)**
MUV	CE	0.0017 (± 0.0001)	0.0021 (± 0.0002)	0.0025 (± 0.0004)
	CB-CE	0.0016 (± 0.0000)	0.0049 (± 0.0031)	0.0033 (± 0.0018)
	Focal	0.0019 (± 0.0003)	0.0060 (± 0.0029)	0.0028 (± 0.0001)
	LDAM	0.0073 (± 0.0018)	0.0060 (± 0.0029)	0.0051 (± 0.0021)
	AUC-M	0.0020 (± 0.0002)	0.0049 (± 0.0011)	0.0054 (± 0.0024)
	SmothAP	0.0015 (± 0.0000)	0.0022 (± 0.0002)	0.0015 (± 0.0000)
	FastAP	0.0014 (± 0.0000)	0.0021 (± 0.0001)	0.0011 (± 0.0000)
	MinMax	0.0015 (± 0.0000)	0.0025 (± 0.0013)	0.0067 (± 0.0016)
	SOAP	**0.0354 (± 0.0025)**	**0.3365 (± 0.0008)**	**0.0121 (± 0.0014)**

of MoleculeKit [52] to obtain node features of graphs. The same data preprocessing is used for all of our experiments on graph data.

Setup. Many recent studies have shown that graph neural networks (GNNs) are powerful models for graph data analysis [30, 17, 16]. Hence, we use three different GNNs as the backbone network for graph classification, including the message passing neural network (**MPNN**) [19], an invariant of graph isomorphism network [54] named by **GINE** [26], and the multi-level message passing neural network (**ML-MPNN**) proposed by Wang et al. [52]. We use the same two-stage training scheme with a similar hyper-parameter tuning. We pre-train the networks by Adam with 100 epochs and a tuned initial learning rate 0.0005, which is decayed by half after 50 epochs.

Results. The achieved AUPRC on the test set by all methods are presented in Table 2. Results show that our method can outperform all baselines by a large margin in terms of AUPRC, regardless of which model structure is used. These results clearly demonstrate that our method is effective for classification problems in which the sample distribution is highly imbalanced between classes.

4.3 Graph Classification for Drug Discovery

Data. In addition to molecular property prediction, we explore applying our method to drug discovery. Recent studies have shown that GNNs are effective in drug discovery through predicting the antibacterial property of chemical compounds [49]. Such application scenarios involves training a GNN model on labeled datasets and making predictions on a large library of chemical compounds so as to discover new antibiotic. However, because the positive samples in the training data, *i.e.*, compounds known to have antibacterial property, are very rare, there exists very severe class imbalance.

We show that our method can serve as a useful solution to the above problem. We conduct experiments on the MIT AICURES dataset from an open challenge (`https://www.aicures.mit.edu/tasks`) in drug discovery. The dataset consists of 2097 molecules. There are 48 positive samples that have antibacterial activity to *Pseudomonas aeruginosa*, which is the pathogen leading to secondary lungs infections of COVID-19 patients. We conduct experiments on three random train/validation/test splits at 80%/10%/10% ratio, and report the average AUPRC on the test set over three splits.

Setup. Following the setup in Sec. 4.2, we use three GNNs: MPNN, GINE and ML-MPNN. We use the same two-stage training scheme with a similar hyper-parameter tuning. We pre-train GNNs by the Adam method for 100 epochs with a batch size of 64 and a tuned learning rate of 0.0005, which is decayed by half at the 50th epoch. Due to the limit of space, Table 3 only reports GINE and MPNN results. Please refer to Table 5 in the supplement for the full results of all three GNNs.

Results. The average test AUPRC from three independent runs over three splits are summarized in Table 3, Table 5. We can see that our SOAP can consistently outperform all baselines on all three GNN models. Our proposed optimization method can significantly improve the achieved AUPRC

Table 3: The test AUPRC values on the MIT AICURES dataset with two graph neural networks, and on the Kaggle Melanoma dataset with two CNN models. We report the average AUPRC and standard deviation (within brackets) from 3 independent runs over 3 different train/validation/test splits.

Data	MIT AICURES		Kaggle Melanoma	
Networks	GINE	MPNN	ResNet18	ResNet34
CE	0.6571 ($\pm$ 0.0619)	0.7289 ($\pm$ 0.0489)	0.0706 ($\pm$ 0.0027)	0.0592 ($\pm$ 0.0016)
CB-CE	0.7448 ($\pm$ 0.0615)	0.7111 ($\pm$ 0.1567)	0.0758 ($\pm$ 0.0076)	0.0701 ($\pm$ 0.0016)
Focal	0.7247 ($\pm$ 0.0735)	0.7211 ($\pm$ 0.1669)	0.0671 ($\pm$ 0.0038)	0.0612 ($\pm$ 0.0011)
LDAM	0.7267 ($\pm$ 0.0708)	0.7386 ($\pm$ 0.1879)	0.0743 ($\pm$ 0.0036)	0.0677 ($\pm$ 0.0015)
AUC-M	0.7220 ($\pm$ 0.0339)	0.7316 ($\pm$ 0.0520)	0.1077 ($\pm$ 0.0038)	0.1900 ($\pm$ 0.0089)
SmothAP	0.1582 ($\pm$ 0.0733)	0.5649 ($\pm$ 0.0744)	0.0377 ($\pm$ 0.0022)	0.0384 ($\pm$ 0.0013)
FastAP	0.1275 ($\pm$ 0.0628)	0.3866 ($\pm$ 0.0364)	0.0343 ($\pm$ 0.0022)	0.0415 ($\pm$ 0.0024)
MinMax	0.6964 ($\pm$ 0.0386)	0.7103 ($\pm$ 0.2089)	0.0555 ($\pm$ 0.0035)	0.0561 ($\pm$ 0.0019)
SOAP	**0.7573 ($\pm$ 0.0552)**	**0.7942 ($\pm$ 0.1336)**	**0.1582 ($\pm$ 0.0139)**	**0.2346 ($\pm$ 0.0337)**

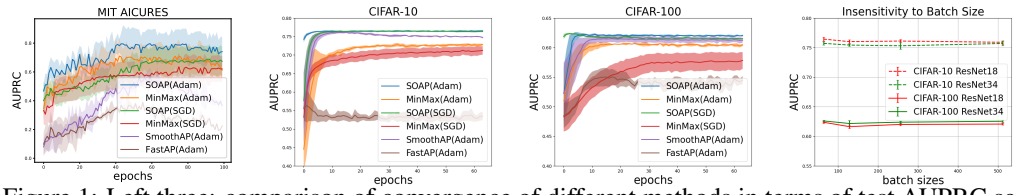

Figure 1: Left three: comparison of convergence of different methods in terms of test AUPRC scores on MIT AICURES data, CIFAR-10 and CIFAR100; Right most: Insensitivity to batch size of SOAP.

of GNN models, indicating that models tend to assign higher confidence scores to molecules with antibacterial activity. This can help identify a larger number of candidate drugs.

4.4 More Studies

Effects of Imbalance Ratio. We now study the effects of imbalance ratio on the performance improvements of our method. We use two datasets Tox21 and ToxCast from the MoleculeNet [53]. The Tox21 and ToxCast contain 8014 and 8589 molecules, respectively. There are 12 property prediction tasks in Tox21, and we conduct experiments on Task 0 and Task 2. Similarly, we select Task 12 and Task 8 of ToxCast for experiments. We use the split of train/validation/test set provided by MoleculeNet. The imbalanced ratios on the training sets are 4.14% for Task 0 of Tox21, 12.00% for Task 2 of Tox21, 2.97% for Task 12 of ToxCast, 8.67% for Task 8 of ToxCast.

Following Sec. 4.2, we test three neural network models MPNN, GINE and ML-MPNN. The hyper-parameters for training models are also the same as those in Sec. 4.2. We present the results of Tox21 and ToxCast in Table 4 in the supplement. Our SOAP can consistently achieve improved performance when the data is extremely imbalanced. However, it sometimes fails to do so if the imbalance ratio is not too low. Clearly, the improvements from our method are higher when the imbalance ratio of labels is lower. In other words, our method is more advantageous for data with extreme class imbalance.

Insensitivity to Batch Size. We conduct experiments on CIFAR-10 and CIFAR-100 data by varying the mini-batch size for the SOAP algorithm and report results in Figure 1 (right most). We can see that SOAP is not sensitive to the mini-batch size. This is consistent with our theory. In contrast, many previous methods for AP maximization are sensitive to the mini-batch size [45, 46, 6].

Convergence Speed. Finally, we report the convergence curves of different methods for maximizing AUPRC or AP in Figure 1 (left three) on different datasets. We can see that the proposed SOAP algorithms converge much faster than other baseline methods.

5 Conclusions and Outlook

In this work, we have proposed a stochastic method to optimize AUPRC that can be used in deep learning for tackling highly imbalanced data. Our approach is based on maximizing the averaged precision, and we cast the objective into a sum of dependent compositional functions. We proposed efficient adaptive and non-adaptive stochastic algorithms with provable convergence guarantee to compute the solutions. Extensive experimental results on graph and image datasets demonstrate that our proposed method can achieve promising results, especially when the class distribution is highly imbalanced. One limitation of SOAP is its convergence rate is still slow. In the future, we will consider to improve the convergence rate to address the limitation of the present work.

References

[1] Balasubramanian, K., Ghadimi, S., and Nguyen, A. Stochastic multi-level composition optimization algorithms with level-independent convergence rates. *CoRR*, abs/2008.10526, 2020.

[2] Bamber, D. The area above the ordinal dominance graph and the area below the receiver operating characteristic graph. *Journal of Mathematical Psychology*, 12:387–415, 1975.

[3] Boyd, K., Eng, K. H., and Page, C. D. Area under the precision-recall curve: Point estimates and confidence intervals. In Blockeel, H., Kersting, K., Nijssen, S., and Zelezny, F. (eds.), *Machine Learning and Knowledge Discovery in Databases*, pp. 451–466, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[4] Brown, A., Xie, W., Kalogeiton, V., and Zisserman, A. Smooth-ap: Smoothing the path towards large-scale image retrieval. In *European Conference on Computer Vision*, pp. 677–694. Springer, 2020.

[5] Burges, C., Ragno, R., and Le, Q. Learning to rank with nonsmooth cost functions. In Schölkopf, B., Platt, J., and Hoffman, T. (eds.), *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2007.

[6] Cakir, F., He, K., Xia, X., Kulis, B., and Sclaroff, S. Deep metric learning to rank. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[7] Cao, K., Wei, C., Gaidon, A., Arechiga, N., and Ma, T. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems*, pp. 1567–1578, 2019.

[8] Chen, K., Li, J., Lin, W., See, J., Wang, J., Duan, L., Chen, Z., He, C., and Zou, J. Towards accurate one-stage object detection with ap-loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[9] Chen, K., Lin, W., See, J., Wang, J., Zou, J., et al. Ap-loss for accurate one-stage object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[10] Chen, T., Sun, Y., and Yin, W. Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization. *arXiv preprint arXiv:2008.10847*, 2020.

[11] Chen, W., Liu, T.-Y., Lan, Y., Ma, Z., and Li, H. Ranking measures and loss functions in learning to rank. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, NIPS'09, pp. 315–323, Red Hook, NY, USA, 2009. Curran Associates Inc. ISBN 9781615679119.

[12] Cortes, C. and Mohri, M. Auc optimization vs. error rate minimization. In Thrun, S., Saul, L. K., and Schölkopf, B. (eds.), *Advances in Neural Information Processing Systems 16*, pp. 313–320. 2004.

[13] Cui, Y., Jia, M., Lin, T.-Y., Song, Y., and Belongie, S. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9268–9277, 2019.

[14] Davis, J. and Goadrich, M. The Relationship Between Precision-Recall and ROC Curves. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pp. 233–240, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2.

[15] Eban, E., Schain, M., Mackey, A., Gordon, A., Saurous, R. A., and Elidan, G. Scalable learning of non-decomposable objectives. In *International Conference on Arti cial Intelligence and Statistics (AISTATS)*, 2017.

[16] Gao, H. and Ji, S. Graph u-nets. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2083–2092. PMLR, 09–15 Jun 2019.

[17] Gao, H., Wang, Z., and Ji, S. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pp. 1416–1424, New York, NY, USA, 2018. Association for Computing Machinery.

[18] Gao, W., Jin, R., Zhu, S., and Zhou, Z. One-pass AUC optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pp. 906–914, 2013.

[19] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272, International Convention Centre, Sydney, Australia, 2017.

[20] Goadrich, M., Oliphant, L., and Shavlik, J. Gleaner: Creating ensembles of firstorder clauses to improve recall-precision curves. In *Machine Learning*, pp. 2006, 2006.

[21] Guo, Z., Liu, M., Yuan, Z., Shen, L., Liu, W., and Yang, T. Communication-efficient distributed stochastic auc maximization with deep neural networks. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pp. 3864–3874, 2020.

[22] Guo, Z., Yuan, Z., Yan, Y., and Yang, T. Fast objective and duality gap convergence for non-convex strongly-concave min-max problems. *arXiv preprint arXiv:2006.06889*, 2020.

[23] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[24] Henderson, P. and Ferrari, V. End-to-end training of object class detectors for mean average precision. In *Computer Vision – ACCV 2016*, pp. 198–213. Springer International Publishing, 2017. doi: 10.1007/978-3-319-54193-8_13. URL `https://doi.org/10.1007%2F978-3-319-54193-8_13`.

[25] Herschtal, A. and Raskutti, B. Optimising area under the ROC curve using gradient descent. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, pp. 49, 2004.

[26] Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., and Leskovec, J. Strategies for pre-training graph neural networks. In *Proceddings of the 7th international conference on learning representations*, 2019.

[27] Huo, Z., Gu, B., Liu, J., and Huang, H. Accelerated method for stochastic composition optimization with nonsmooth regularization. In McIlraith, S. A. and Weinberger, K. Q. (eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, pp. 3287–3294, 2018.

[28] Joachims, T. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine learning*, pp. 377–384, 2005.

[29] Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations (ICLR)*, 2015.

[30] Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations*, 2017.

[31] Lian, X., Wang, M., and Liu, J. Finite-sum composition optimization via variance reduced gradient descent. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1159–1167, 2017.

[32] Lin, T., Fan, C., Wang, M., and Jordan, M. I. Improved oracle complexity for stochastic compositional variance reduced gradient. *CoRR*, abs/1806.00458, 2018.

[33] Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[34] Liu, L., Liu, J., Hsieh, C., and Tao, D. Stochastically controlled stochastic gradient for the convex and non-convex composition problem. *CoRR*, abs/1809.02505, 2018.

[35] Liu, L., Liu, J., and Tao, D. Dualityfree methods for stochastic composition optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 30(4):1205–1217, 2019.

[36] Liu, M., Zhang, X., Chen, Z., Wang, X., and Yang, T. Fast stochastic auc maximization with $o(1/n)$-convergence rate. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 3189–3197. PMLR, 2018.

[37] Liu, M., Yuan, Z., Ying, Y., and Yang, T. Stochastic auc maximization with deep neural networks. In *International Conference on Learning Representations*, 2020.

[38] Metzler, D. and Croft, W. B. A markov random field model for term dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pp. 472–479, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 1595930345.

[39] Metzler, D. and Croft, W. B. A markov random field model for term dependencies. In *Proceedings of the 28th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 472–479, 2005.

[40] Mohapatra, P., Jawahar, C., and Kumar, M. P. Efficient optimization for average precision svm. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems*, volume 27, pp. 2312–2320. Curran Associates, Inc., 2014.

[41] Mohapatra, P., Rolinek, M., Jawahar, C. V., Kolmogorov, V., and Kumar, M. Efficient optimization for rank-based loss functions. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3693–3701, 2018.

[42] Natole, M., Ying, Y., and Lyu, S. Stochastic proximal algorithms for auc maximization. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 3710–3719. PMLR, 2018.

[43] Oksuz, K., Cam, B. C., Akbas, E., and Kalkan, S. A ranking-based, balanced loss function unifying classification and localisation in object detection. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15534–15545. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/b2eeb7362ef83deff5c7813a67e14f0a-Paper.pdf`.

[44] Qi, Q., Guo, Z., Xu, Y., Jin, R., and Yang, T. A practical online method for distributionally deep robust optimization. *arXiv preprint arXiv:2006.10138*, 2020.

[45] Qin, T., Liu, T.-Y., and Li, H. A general approximation framework for direct optimization of information retrieval measures. Technical Report MSR-TR-2008-164, November 2008.

[46] Rolinek, M., Musil, V., Paulus, A., Vlastelica, M., Michaelis, C., and Martius, G. Optimizing rank-based metrics with blackbox differentiation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[47] Rotemberg, V., Kurtansky, N., Betz-Stablein, B., Caffery, L., Chousakos, E., Codella, N., Combalia, M., Dusza, S., Guitera, P., Gutman, D., et al. A patient-centric dataset of images and metadata for identifying melanomas using clinical context. *arXiv preprint arXiv:2008.07360*, 2020.

[48] Song, Y., Schwing, A., Richard, and Urtasun, R. Training deep neural networks via direct loss minimization. In Balcan, M. F. and Weinberger, K. Q. (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 2169–2177, New York, New York, USA, 20–22 Jun 2016. PMLR.

[49] Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, A., Donghia, N. M., MacNair, C. R., French, S., Carfrae, L. A., Bloom-Ackerman, Z., et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

[50] Wang, M., Fang, E. X., and Liu, H. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Mathematical Programming*, 161(1-2): 419–449, 2017.

[51] Wang, M., Liu, J., and Fang, E. X. Accelerating stochastic composition optimization. *Journal Machine Learning Research*, 18:105:1–105:23, 2017.

[52] Wang, Z., Liu, M., Luo, Y., Xu, Z., Xie, Y., Wang, L., Cai, L., and Ji, S. Advanced graph and sequence neural networks for molecular property prediction and drug discovery. *arXiv preprint arXiv:2012.01981*, 2020.

[53] Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. MoleculeNet: a benchmark for molecular machine learning. *Chemical science*, 9 (2):513–530, 2018.

[54] Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *7th International Conference on Learning Representations*, 2019.

[55] Ying, Y., Wen, L., and Lyu, S. Stochastic online auc maximization. In *Advances in neural information processing systems 30*, pp. 451–459, 2016.

[56] Yu, Y. and Huang, L. Fast stochastic variance reduced ADMM for stochastic composition optimization. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3364–3370, 2017.

[57] Yuan, Z., Yan, Y., Sonka, M., and Yang, T. Robust deep auc maximization: A new surrogate loss and empirical studies on medical image classification. *arXiv preprint arXiv:2012.03173*, 2020.

[58] Yue, Y., Finley, T., Radlinski, F., and Joachims, T. A support vector method for optimizing average precision. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pp. 271–278, New York, NY, USA, 2007. Association for Computing Machinery.

[59] Zhang, J. and Xiao, L. A composite randomized incremental gradient method. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97, pp. 7454–7462, 2019.

[60] Zhao, P., Hoi, S. C. H., Jin, R., and Yang, T. Online auc maximization. In *ICML*, pp. 233–240, 2011.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No] This work has not been discovered any negative social impacts yet.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3.3
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]