Momentum Accelerates the Convergence of Stochastic AUPRC Maximization

Guanghui Wang[1], Ming Yang[2], Lijun Zhang[1], Tianbao Yang[3]
[1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
[2]Hefei University of Technology, Hefei 230000, China
[3]Department of Computer Science, the University of Iowa, Iowa City, IA 52242, USA
{wanggh,zhanglj}@lamda.nju.edu.cn, yangming@mail.hfut.edu.cn, tianbao-yang@uiowa.edu

Abstract

In this paper, we study stochastic optimization of areas under precision-recall curves (AUPRC), which is widely used for combating imbalanced classification tasks. Although a few methods have been proposed for maximizing AUPRC, stochastic optimization of AUPRC with convergence guarantee remains an undeveloped territory. A state-of-the-art complexity is $O(1/\epsilon^5)$ for finding an ϵ-stationary solution. In this paper, we further improve the stochastic optimization of AURPC by (i) developing novel stochastic momentum methods with a better iteration complexity of $O(1/\epsilon^4)$ for finding an ϵ-stationary solution; and (ii) designing a novel family of stochastic adaptive methods with the same iteration complexity, which enjoy faster convergence in practice. To this end, we propose two innovative techniques that are critical for improving the convergence: (i) the biased estimators for tracking individual ranking scores are updated in a randomized coordinate-wise manner; and (ii) a momentum update is used on top of the stochastic gradient estimator for tracking the gradient of the objective. The novel analysis of Adam-style updates is also one main contribution. Extensive experiments on various data sets demonstrate the effectiveness of the proposed algorithms. Of independent interest, the proposed stochastic momentum and adaptive algorithms are also applicable to a class of two-level stochastic dependent compositional optimization problems.

1 INTRODUCTION

In supervised machine learning systems, the performance metrics used for model evaluation play a vital role (Ferri et al., 2009). Traditional machine learning algorithms typically employ *accuracy* (proportion of correctly predicted examples) as the measure of performance, which is a natural choice when the classes of data are balanced. However, in many real-world applications, such as activity recognition (Gao et al., 2016), and medical diagnosis (Krawczyk et al., 2016), the distribution of classes is highly skewed, for which the accuracy usually fails to characterize the hardness of the problem. In such cases, a more enhanced metric, named areas under the precision-recall curves (AUPRC), is proposed and commonly used to assess classifiers (Boyd et al., 2013). Over the past decades, the superiority of AUPRC for *evaluating* imbalanced classifiers has been witnessed by a long line of research (Davis and Goadrich, 2006b; Clemencon and Vayatis, 2009; Kok and Domingos, 2010; Boyd et al., 2012; Flach and Kull, 2015).

How to utilize AUPRC to *facilitate algorithm design* remains a challenging question. As observed by Cortes and Mohri (2003), maximizing accuracy on training data does not necessarily lead to a satisfactory solution with maximal AUPRC. On the other hand, directly optimizing AUPRC is generally intractable due to the complicated integral operation. To mitigate this issue, most of the existing works seek to optimize certain estimator of AUPRC (Qin et al., 2010; Brown et al., 2020; Qi et al., 2021). In this paper, we focus on maximizing *average precision* (AP), which is one of the most commonly used estimators in practice for the purpose of maximizing AUPRC. Given a training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, AP is defined as (Boyd et al., 2013):

$$\text{AP} = \frac{1}{m} \sum_{\mathbf{x}_i, y_i = 1} \frac{\text{r}^+(\mathbf{x}_i)}{\text{r}(\mathbf{x}_i)}, \tag{1}$$

where $(\mathbf{x}_i, y_i = 1)$ denotes a positive example, $\mathrm{r}^+(\mathbf{x}_i)$ denotes its rank among all positive examples (i.e., the number of positive examples that are ranked higher than $\mathbf{x}_i$ including itself), $\mathrm{r}(\mathbf{x}_i)$ denotes its rank among all examples (i.e., the number of examples that are ranked higher than $\mathbf{x}_i$ including itself), and m denotes the total number of positive examples. It can be proved that AP is an unbiased estimator of AUPRC as the number of examples n goes infinity.

However, the optimization of AP is challenging due to the non-differential ranking functions $\mathrm{r}^+(\mathbf{x}_i)$ and $\mathrm{r}(\mathbf{x}_i)$ and the complicated form. Although a few studies have tried to optimize AP for AUPRC optimization (Burges et al., 2007; Qin et al., 2010; Henderson and Ferrari, 2016; Mohapatra et al., 2018; Cakir et al., 2019; Rolinek et al., 2020; Brown et al., 2020), most of them are heuristic driven and do not provide any convergence guarantee. Recently, Qi et al. (2021) made a breakthrough towards optimizing a differentiable surrogate loss of AP with provable convergence guarantee. They cast the objective as a sum of non-convex compositional functions, and propose a principled stochastic method named SOAP for solving the special optimization problem. A key in their algorithmic design is to maintain and update biased estimators of the surrogate ranking functions for all positive data. Theoretical analysis show that the iteration complexity of SOAP is on the order of $O(1/\epsilon^5)$.

However, it is still unclear whether faster rates than $O(1/\epsilon^5)$ can be obtained. Moreover, whether more advanced update rules such as momentum and Adam (Kingma and Ba, 2014) are useful to accelerate the convergence also remains an open question. This paper aims to address these problems, and makes the following contributions.

- We propose momentum-based methods to accelerate the convergence for solving the finite-sum stochastic compositional optimization problem of AP maximization. The key idea is to employ a momentum update to compute a stochastic gradient estimator of the objective function.

- We investigate two schemes for updating the biased estimators of the ranking functions, and establish a faster rate in the order of $O(1/\epsilon^4)$ for the iteration complexity. The first is similar to the one proposed by Qi et al. (2021). However, an improved rate of this scheme is difficult to establish unless the sampled positive data include all positive examples due to a subtle randomness issue. To address this limitation, we propose the second scheme by updating them in a randomized coordinate-wise fashion.

- We propose and analyze a family of adaptive al-

gorithms by using different adaptive step sizes including the Adam-style step size. We establish the same order of iteration complexity by employing the second scheme mentioned above for updating the biased estimators of the ranking functions. **To the best our knowledge, this is the first time** the convergence of Adam-style methods for stochastic compositional problems is established in the literature. A comparison between our convergence results and the existing results for maximizing AP is presented in Table 1.

- We conduct extensive experiments on benchmark datasets comparing with previous stochastic algorithms for AUPRC/AP optimization and verify the effectiveness of the proposed algorithms.

2 RELATED WORK

AUPRC/AP Optimization. Many studies in machine learning (Yue et al., 2007; McFee and Lanckriet, 2010; Song et al., 2016), information retrieval (Metzler and Croft, 2005; Burges et al., 2007; Chen et al., 2009; Qin et al., 2010), computer vision (Henderson and Ferrari, 2016; Mohapatra et al., 2018; Cakir et al., 2019; Chen et al., 2019; Oksuz et al., 2020; Rolinek et al., 2020; Brown et al., 2020) have investigated the issue of AUPRC/AP maximization. Traditional machine learning studies are based on classical optimization techniques, such as hill climbing search (Metzler and Croft, 2005), cutting-plane method (Yue et al., 2007; McFee and Lanckriet, 2010), and dynamic programming (Song et al., 2016). However, these methods do not scale well when the training set is large. Many studies have considered various methods for computing an (approximiate) gradient for the original AP score function, e.g., finite difference estimators (Henderson and Ferrari, 2016), linear envelope estimators (Henderson and Ferrari, 2016), soft histogram binning technique (Cakir et al., 2019), a blackbox differentiation of a combinatorial solver (Rolinek et al., 2020), loss-augmented inference for estimating the semi-gradient (Mohapatra et al., 2018), using the gradient of implicit cost functions (Burges et al., 2007), using the gradient of a smooth approximation of AP (Qin et al., 2010; Brown et al., 2020), etc. However, none of these studies provide any convergence guarantee when using these techniques for stochastic optimization of AUPRC/AP.

Recently, Eban et al. (2017) propose a systematic framework for AUPRC optimization, which makes use of a finite set of samples to approximate the integral. They then cast the optimization as a min-max saddle-point problem, and optimize it by SGD-style methods without providing any convergence analysis. To the

Table 1: Comparison with previous results for maximizing AP or its surrogate loss. "-" indicate results not available or applicable.

Method	Provable Convergence	Adaptive Step Size	Iteration Complexity
MOAP (this work)	Yes	No	$O(1/\epsilon^4)$
ADAP (this work)	Yes	Adam, AMSGrad, AdaGrad, Adabound, etc.	$O(1/\epsilon^4)$
SOAP (SGD-style) (Qi et al., 2021)	Yes	No	$O(1/\epsilon^5)$
SOAP (AMSGrad-style) (Qi et al., 2021)	Yes	AMSGrad	$O(1/\epsilon^5)$
SOAP (Adam-style) (Qi et al., 2021)	No	Adam	-

best of our knowledge, (Qi et al., 2021) is the first work that proposes a principled stochastic method for maximizing a smooth approximation of the AP function with provable convergence guarantee.

AUROC Optimization. Another territory related to AUPRC/AP maximiation is AUROC (aka areas under the ROC curves) maximization. Compared to AUPRC, AUROC is easier to optimize, and the problem of AUROC optimization have been investigated by a large number of previous studies (Herschtal and Raskutti, 2004; Joachims, 2005; Zhao et al., 2011; Gao et al., 2013; Liu et al., 2018b; Natole et al., 2018; Ying et al., 2016; Yuan et al., 2020; Guo et al., 2020). However, an algorithm that maximizes AUROC does not necessarily maximize AUPRC (Davis and Goadrich, 2006a). Hence, we will not directly compare with these studies in the present work.

Stochastic Compositional Optimization. The optimization problem considered in this paper for AP maximization is closely related to stochastic compositional optimization (SCO), where the objective function is of the form $E_{\xi_1}[f(E_{\xi_2}[g(\mathbf{w}; \xi_2)]; \xi_1)]$, and ξ_1 and ξ_2 are random variables. SCO has been extensively studied by previous work (Wang et al., 2016, 2017; Lian et al., 2017; Huo et al., 2018; Lin et al., 2018; Liu et al., 2018a; Zhang and Xiao, 2019; Yang et al., 2019; Balasubramanian et al., 2020; Ghadimi et al., 2020; Chen et al., 2021). The major difference is that the inner function in our objective depends on both the random variable of the inner level and that of the outer level, which makes the stochastic algorithm design and theoretical analysis much more involved. We notice that a recent work (Hu et al., 2020) have considered a family of stochastic compositional optimization problems, where the inner function depends on both the random variable of the inner level and that of the outer level. They proposed biased stochastic methods based on mini-batch averaging. However, their algorithms require a large mini-batch size in the order of $O(1/\epsilon^2)$ for finding an ϵ-stationary solution, which is not practical, and have a worse sample complexity in the order of $O(1/\epsilon^6)$ or $O(1/\epsilon^5)$. In contrast, our algo-

rithms do not require a large mini-batch size and enjoy a better sample complexity in the order of $O(m/\epsilon^4)$, where m is the number of positive data points that is usually a moderate number for imbalanced data.

3 MAIN RESULTS

Notation. In this paper, we consider binary classification problems. Let $(\mathbf{x}, y)$ be a feature-label pair where $\mathbf{x} \in \mathbb{R}^d, \mathbf{y} \in \{1, -1\}$, $s_{\mathbf{w}}(\mathbf{x}) : \mathbb{R}^d \mapsto \mathbb{R}$ be the score function of a classifier characterized by a parameter vector $\mathbf{w} \in \mathbb{R}^d$. Let $\mathcal{D}_+ = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ denote a set of positive examples, and let $\mathcal{D}_- = \{(\mathbf{x}_{m+1}, y_{m+1}), \ldots, (\mathbf{x}_n, y_n)\}$ denote a set of negative examples. The whole training set is denoted by $\mathcal{D} = \mathcal{D}_+ \cup \mathcal{D}_-$. We denote by $\|\cdot\|$ the Euclidean norm of a vector. Let $\Pi_\Omega[\mathbf{u}] = \arg\min_{\mathbf{v} \in \Omega} \|\mathbf{v} - \mathbf{u}\|^2$ be the Euclidean projection onto a convex set Ω.

Preliminaries. To tackle the non-differentiable ranking functions in (1), a differentiable surrogate loss function can be used for approximating the ranking functions. Following Qi et al. (2021), we consider the following approximation:

$$\widehat{\mathrm{AP}} = \frac{1}{m} \sum_{\mathbf{x}_i \in \mathcal{D}_+} \frac{\sum_{j=1}^n \ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) \mathbb{I}(y_j = 1)}{\sum_{j=1}^n \ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)}, \quad (2)$$

where $\ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)$ is an surrogate function of the indicator function $\mathbb{I}(s_{\mathbf{w}}(\mathbf{x}_j) \geq s_{\mathbf{w}}(\mathbf{x}_i))$. In the literature, various surrogate functions $\ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)$ have been considered (Brown et al., 2020; Qi et al., 2021; Qin et al., 2010), including a squared hinge loss $\ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) = \max(0, (s_{\mathbf{w}}(\mathbf{x}_j) - s_{\mathbf{w}}(\mathbf{x}_i) + \gamma))^2$, and a logistic function $\ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) = \frac{\exp(\gamma(s_{\mathbf{w}}(\mathbf{x}_j) - s_{\mathbf{w}}(\mathbf{x}_i)))}{1 + \exp(\gamma(s_{\mathbf{w}}(\mathbf{x}_j) - s_{\mathbf{w}}(\mathbf{x}_i)))}$, where γ is a margin or scaling parameter. Define the following notations:

$$g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) = \begin{bmatrix} g_1(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) \\ g_2(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) \end{bmatrix} = \begin{bmatrix} \mathbb{I}(y_j = 1)\ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) \\ \ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) \end{bmatrix},$$
$$(3)$$

and $g_i(\mathbf{w}) = \sum_{j=1}^n g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) \in \mathbb{R}^2$, and $f(\mathbf{u}) = -\frac{u_1}{u_2} : \mathbb{R}^2 \mapsto \mathbb{R}$ for any $\mathbf{u} = [u_1, u_2]^\top \in \mathbb{R}^2$. We can

see that the two coordinates of $g_i(\mathbf{w})$ are the surrogates of the two ranking functions $r^+(\mathbf{x}_i)$ and $r(\mathbf{x}_i)$, respectively.

Then, the optimization problem for AP maximization based on the surrogate function in (2) is equivalent to the following minimization problem:

$$\min_{\mathbf{w}} F(\mathbf{w}) = \mathrm{E}_{\mathbf{x}_i \sim \mathcal{D}_+}[f(g_i(\mathbf{w}))] = \frac{1}{m} \sum_{\mathbf{x}_i \in \mathcal{D}_+} f(g_i(\mathbf{w})). \tag{4}$$

We refer to the above problem as *finite-sum two-level stochastic compositional optimization*. We emphasize that a standard ℓ_2 norm regularizer $\lambda \|\mathbf{w}\|^2$ can be added to the above objective to control overfitting. Note that the above problem is generally non-convex even if $g_i(\mathbf{w})$ is convex. Hence, our goal for solving (4) is to find a nearly stationary solution. Throughout this paper, we denote by $\tilde{g}_i(\mathbf{w})$ and $\nabla \tilde{g}_i(\mathbf{w})$ independent unbiased stochastic estimators of $g_i(\mathbf{w})$ and $\nabla g_i(\mathbf{w})$, respectively, which are computed based on sampled (mini-batch) data from all data points in $\mathcal{D}$. For example, we can sample two sets of examples from $\mathcal{D}$ denoted by $\mathcal{S}_1$ and $\mathcal{S}_2$ and compute $\tilde{g}_i(\mathbf{w}) = \frac{n}{|\mathcal{S}_1|} \sum_{\mathbf{x}_j \in \mathcal{S}_1} g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)$ and $\nabla \tilde{g}_i(\mathbf{w}) = \frac{n}{|\mathcal{S}_2|} \sum_{\mathbf{x}_j \in \mathcal{S}_1} \nabla g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)$. To this end, we impose the following assumptions (Qi et al., 2021).

Assumption 1. *Assume (i) there exist $C > 0$ such that $\ell(\mathbf{w}; \mathbf{x}_i; \mathbf{x}_i) \geq C$, for any $\mathbf{x}_i \in \mathcal{D}_+$; (ii) $\ell(\mathbf{w}; \mathbf{x}_j; \mathbf{x}_i)$ is C_ℓ-Lipschitz continuous and L_ℓ-smooth with respect to $\mathbf{w}$ for any $\mathbf{x}_i \in \mathcal{D}_+$, $\mathbf{x}_j \in \mathcal{D}$, where $C_\ell, L_\ell > 0$ are constants; (iii) it holds that $0 \leq \ell(\mathbf{w}; \mathbf{x}_j; \mathbf{x}_i) \leq M$ for some $M > 0$ for any $\mathbf{x}_i \in \mathcal{D}_+$, $\mathbf{x}_j \in \mathcal{D}$.*

Assumption 2. *Assume there exists a positive constant σ, such that $\|\nabla g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)\|^2 \leq \sigma^2$ for any $\mathbf{x}_i \in \mathcal{D}_+, \mathbf{x}_j \in \mathcal{D}$.*

Remark 1. *The above assumptions can be easily satisfied for a smooth surrogate loss function $\ell(\cdot; \mathbf{x}_j, \mathbf{x}_i)$ and a bounded score function $s_\mathbf{w}(\mathbf{x})$. For example, for a linear model we can use $s_\mathbf{w}(\mathbf{x}) = 1/(1 + \exp(-\mathbf{w}^\top \mathbf{x})) \in [0, 1]$ as the score function.*

Based on Assumption 1, we can establish the smoothness of the objective function in the optimization problem (4), and show that $g_i(\mathbf{w}) \in \Omega = \{\mathbf{u} \in \mathbb{R}^2, u_1 \leq Mm, C \leq u_2 \leq Mn\}$ (cf. the supplement). Our goal is then to find a solution $\mathbf{w}$ such that $\|\nabla F(\mathbf{w})\| \leq \epsilon$ in expectation, to which we refer as ϵ-stationary solution.

Before ending this section, we summarize the updates of SOAP algorithm (Qi et al., 2021) to facilitate comparisons. An essential component of SOAP is to compute an estimator of the gradient of the objective function, i.e., $\nabla F(\mathbf{w}) = \frac{1}{m} \sum_{i=1}^m \nabla g_i(\mathbf{w}_t)^\top \nabla f(g_i(\mathbf{w}_t))$ at each iteration. Since both $g_i(\mathbf{w}_t)$ and $\nabla g_i(\mathbf{w}_t)$ involves processing all examples, stochastic estimators of these

---

Algorithm 1 MOAP-V1

1: **Input:** η, β_0, β_1, B
2: **Initialize:** $\mathbf{w}_1 \in \mathbb{R}^d, U_1 \in \mathbb{R}^{m \times 2}, \mathbf{m}_1 \in \mathbb{R}^d$
3: **for** $t = 1, \ldots, T$ **do**
4:     Sample B points from $\mathcal{D}_+$, denoted by $\mathcal{B}_t$
5:     Set $[U_{t+1}]_i$ as:
$$\begin{cases} \Pi_\Omega[(1 - \beta_0)[U_t]_i + \beta_0 \tilde{g}_i(\mathbf{w}_t)^\top] & i \in \mathcal{B}_t \\ [U_t]_i & \text{o.w.} \end{cases}$$
6:     $\tilde{\nabla}_t = \frac{1}{B} \sum_{i \in \mathcal{B}_t} \nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f([U_{t+1}]_i^\top)$
7:     $\mathbf{m}_{t+1} = (1 - \beta_1)\mathbf{m}_t + \beta_1 \tilde{\nabla}_t$
8:     $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{m}_{t+1}$
9: **end for**

---

quantities are computed. In addition, due to the fact that $g_i(\mathbf{w}_t)$ is inside ∇f a better technique than a simple mini-batch averaging is used in order to control the variance. To this end, they introduce a sequence $U_t \in \mathbb{R}^{m \times 2}$ for tracking $g(\mathbf{w}) = [g_1(\mathbf{w}), \ldots, g_m(\mathbf{w})]^\top \in \mathbb{R}^{m \times 2}$, and proposes the SOAP algorithm with the following updates:

$$\text{SOAP} \begin{cases} \text{Sample } B \text{ points from } \mathcal{D}_+, \text{ denoted by } \mathcal{B}_t \\ [U_{t+1}]_i = \begin{cases} (1 - \beta)[U_t]_i + \beta \tilde{g}_i(\mathbf{w}_t)^\top & i \in \mathcal{B}_t \\ [U_t]_i & \text{o.w.} \end{cases} \\ \tilde{\nabla}_t = \frac{1}{B} \sum_{i \in \mathcal{B}_t} \nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f([U_{t+1}]_i^\top) \\ \mathbf{w}_{t+1} = \mathbf{w}_t - \eta \tilde{\nabla}_t \end{cases} \tag{5}$$

where $[U_t]_i$ denotes the i-th row of U_t. The U_{t+1} sequence is known as the moving average estimator in the literature of stochastic compositional optimization (Wang et al., 2017). SOAP enjoys an iteration complexity of $O(1/\epsilon^5)$ for using the above update to find an ϵ-stationary solution of the objective when $\mathcal{B}_t = \mathcal{D}_+$ and an iteration complexity of $O(m/\epsilon^5)$ when $|\mathcal{B}_t| = O(1)$. Below, we present novel algorithms to improve these complexities.

3.1 Stochastic Momentum Methods for AP Maximization: MOAP

To improve the convergence of SOAP, we propose to exploit momentum when updating the model parameter $\mathbf{w}_t$. Similar to the stochastic momentum method widely used in stochastic optimization (Yang et al., 2016), we maintain and update the following stochastic gradient estimator:

$$\mathbf{m}_{t+1} = (1 - \beta_1)\mathbf{m}_t + \beta_1 \tilde{\nabla}_t, \tag{6}$$

where $\tilde{\nabla}_t$ is a stochastic estimator of the gradient at $\mathbf{w}_t$, which is computed based on U_{t+1} similarly to SOAP. Then, we update the solution by $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{m}_{t+1}$. We investigate two methods for updating U_{t+1}.

Algorithm 2 MOAP-V2
1: **Input:** η, β_0, β_1, B
2: **Initialize:** $\mathbf{w}_1 \in \mathbb{R}^d, U_1 \in \mathbb{R}^{m\times 2}, \mathbf{m}_1 \in \mathbb{R}^d$
3: **for** $t = 1, \ldots, T$ **do**
4:     Sample B points from $\mathcal{D}_+$, denoted by $\mathcal{B}_t$
5:     Set $[U_{t+1}]_i$ as:
$$\begin{cases} \Pi_\Omega[(1-\beta_0)[U_t]_i + \beta_0 \frac{m}{B}\tilde{g}_i(\mathbf{w}_t)^\top] & i \in \mathcal{B}_t \\ \Pi_\Omega[(1-\beta_0)[U_t]_i] & \text{o.w.} \end{cases}$$
6:     $\tilde{\nabla}_t = \frac{1}{B}\sum_{i\in\mathcal{B}_t}\nabla\tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f([U_{t+1}]_i^\top)$
7:     $\mathbf{m}_{t+1} = (1-\beta_1)\mathbf{m}_t + \beta_1\tilde{\nabla}_t$
8:     $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\mathbf{m}_{t+1}$
9: **end for**

MOAP-V1. The first method for updating U_{t+1} is similar to that in (Qi et al., 2021), which is presented in Algorithm 1. It is worth mentioning that we conduct a projection $\Pi_\Omega[\cdot]$ in Step 5 to ensure the two components of each row in U is appropriately bounded such that $\nabla f(\cdot)$ is Lipschitz continuous with respect to its input. Regarding the convergence of Algorithm 1, we first present the following result by setting $\mathcal{B}_t = \mathcal{D}_+$ at every iteration.

Theorem 1. *Suppose Assumptions 1 and 2 hold. Then, Algorithm 1 with $\mathcal{B}_t = \mathcal{D}_+$, $\beta_0 = O(\epsilon^2), \beta_1 = O(\epsilon^2)$, $\eta = O(\epsilon^2)$, and $T = O(1/\epsilon^4)$ ensures*

$$\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^T \|\nabla F(\mathbf{w}_t)\|^2\right] \le \epsilon^2.$$

Remark 2. Compare with Theorem 2 in Qi et al. (2021), the MOAP has a better iteration complexity, i.e., $T = O(1/\epsilon^4)$ vs $T = O(1/\epsilon^5)$ of SOAP. The total sample complexity of MOAP in this case is $O(m/\epsilon^4)$. We include all omitted proofs in the supplementary materials and exhibit the constants in the big O in the proof.

Next, we consider using random samples $\mathcal{B}_t \sim \mathcal{D}$. Without loss of generality, we assume $|\mathcal{B}_t| = 1$ and the sample is randomly chosen from $\mathcal{D}$ with replacement. Then, for Algorithm 1, we provide the following convergence rate.

Theorem 2. *Suppose Assumptions 1 and 2 hold. Then, Algorithm 1 with $B = 1$, $\beta_0 = O(\epsilon^2), \beta_1 = O(\epsilon^2)$, $\eta = O(\epsilon^3/m)$, $T = O(m/\epsilon^5)$ satisfies*

$$\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^T \|\nabla F(\mathbf{w}_t)\|^2\right] \le \epsilon^2.$$

Remark 3. Theorem 2 implies that, for a stochastic $\mathcal{B}_t$, Algorithm 1 suffers a worst case iteration complexity in the order of $O(m/\epsilon^5)$, which is the same as SOAP for using a stochastic $\mathcal{B}_t$. This is mainly due to a subtle dependent issue caused by the *infrequent updates* of the moving average estimator U_t, which makes

the momentum fail to accelerate the convergence. The detailed discussion is presented in the supplementary.

MOAP-V2. To address the limitation of Algorithm 1 for using a stochastic $\mathcal{B}_t$, we propose a second method for updating U_{t+1}. The procedure is presented in Algorithm 2. Different from MOAP-V1, MOAP-V2 updates all coordinates of U_{t+1} according to

$$[U_{t+1}]_i = \begin{cases} \Pi_\Omega[(1-\beta_0)[U_t]_i + \beta_0\frac{m}{B}\tilde{g}_i(\mathbf{w}_t)^\top] & i\in\mathcal{B}_t \\ \Pi_\Omega[(1-\beta_0)[U_t]_i] & \text{o.w.} \end{cases},$$

i.e, the coordinate of U_{t+1} corresponding to a sampled positive data $i \in \mathcal{B}_t$ is updated similarly to that in MOAP-V1 except the unbiased estimator $\tilde{g}_i(\mathbf{w}_t)$ is scaled by m/B, and the coordinate of U_{t+1} corresponding to a non-sampled data $i \in \mathcal{B}_t$ is updated trivially by multiplying with a scaling factor $1 - \beta_0$. It is notable that we can delay updating these coordinates until they are sampled because at the current iteration, these coordinates are not used for updating the model parameter $\mathbf{w}_{t+1}$.

In order to understand why this method can help improve the convergence for using a stochastic $\mathcal{B}_t$. We can write the update of U_{t+1} equivalently as $U_{t+1} = \Pi_{\Omega^m}[(1-\beta_0)U_t + \beta_0\hat{g}(\mathbf{w}_t)]$, where $\hat{g}(\mathbf{w}_t)$ is defined as

$$\hat{g}(\mathbf{w}_t) = \frac{1}{B}\sum_{i\in\mathcal{B}_T}\begin{pmatrix} 0 \\ \cdot \\ m\tilde{g}_i(\mathbf{w}_t)^\top \\ \cdot \\ 0 \end{pmatrix}. \qquad (7)$$

It is not difficult to show that $\mathrm{E}[\hat{g}(\mathbf{w}_t)] = g(\mathbf{w}_t)$, where the expectation is taken over the randomness in $\mathcal{B}_t$ and $\tilde{g}_i(\mathbf{w}_t)$. We refer to this update of U_{t+1} as **the randomized coordinate update**. This update is similar to Step 5 in MOAP-V1 when using $\mathcal{B}_t = \mathcal{D}_+$ to update the model, in which case all coordinates of U_{t+1} are updated by $U_{t+1} = \Pi_{\Omega^m}[(1-\beta_0)U_t + \beta_0\tilde{g}(\mathbf{w}_t)]$, where $\tilde{g}(\mathbf{w}_t) = (\tilde{g}_1(\mathbf{w}_t), \ldots, \tilde{g}_m(\mathbf{w}_t))^\top$. Both $\hat{g}(\mathbf{w}_t)$ and $\tilde{g}(\mathbf{w}_t)$ are unbiased estimators of $g(\mathbf{w}_t)$. The difference is that the variance of $\hat{g}(\mathbf{w}_t)$ is scaled up by a factor of $\frac{m}{B}$. We emphasize it is the combination of the momentum update (6) and the randomized coordinate update (7) that yields an improved convergence rate of Algorithm 2 presented below.

Theorem 3. *Suppose Assumptions 1 and 2 hold. Then, Algorithm 2 with $\beta_0 = O(\epsilon^2 B/m)$, $\beta_1 = O(\epsilon^2 B/m)$, $\eta = O(\beta)$, $T = O(m/[B\epsilon^4])$ ensures that $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^T \|\nabla F(\mathbf{w}_t)\|^2\right] \le \epsilon^2$.*

Remark 4. The above theorem implies that, for a stochastic $\mathcal{B}_t$, Algorithm 2 enjoys a better iteration complexity in the order of $O(m/\epsilon^4)$, which is the better than SOAP for using a stochastic $\mathcal{B}_t$.

Algorithm 3 ADAP

1: **Input:** η, β_0, β_1, δ, B
2: **Initialize:** $\mathbf{w}_1 \in \mathbb{R}^d, U_1 \in \mathbb{R}^{m \times 2}, \mathbf{m}_1 \in \mathbb{R}^d$
3: **for** $t = 1, \ldots, T$ **do**
4:      Sample B points from $\mathcal{D}_+$, denoted by $\mathcal{B}_t$
5:      Set $[U_{t+1}]_i$ as:
     $\begin{cases} \Pi_\Omega[(1 - \beta_0)[U_t]_i + \beta_0 \frac{m}{B} \tilde{g}_i(\mathbf{w}_t)^\top] & i \in \mathcal{B}_t \\ \Pi_\Omega[(1 - \beta_0)[U_t]_i] & \text{o.w.} \end{cases}$
6:      $\tilde{\nabla}_t = \frac{1}{B} \sum_{i \in \mathcal{B}_t} \nabla_\mathbf{w} \tilde{g}_{i,t}(\mathbf{w}_t)^\top \cdot \nabla f([U_{t+1}]_i^\top)$
7:      $\mathbf{m}_{t+1} = (1 - \beta)\mathbf{m}_t + \beta \tilde{\nabla}_t$
8:      $\mathbf{v}_{t+1} = h_t(\mathbf{v}_t, \tilde{\nabla}_t^2)$ $\diamond h_t$ can be implemented by any of the methods in (8)
9:      $\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta}{\sqrt{\mathbf{v}_{t+1}} + \delta} \mathbf{m}_{t+1}$
10: **end for**

3.2 Stochastic Adaptive Algorithms for AP Maximization: ADAP

In this section, we extend our technique to stochastic adaptive algorithms, which use adaptive step sizes. For standard stochastic optimization, various adaptive step sizes have been investigated, including AdaGrad (Duchi et al., 2011), Adam (Kingma and Ba, 2014), AMSGrad (Reddi et al., 2018), AdaBound (Luo et al., 2019), etc.

Motivated by these existing adaptive methods, we propose stochastic adaptive algorithms for AP maximization in Algorithm 3, which is referred to as ADAP. The difference from Algorithm 2 is that we need to maintain and update another sequence $\mathbf{v}_t \in \mathbb{R}^d$ in order to compute an adaptive step size in Step 8, where $h_t(\cdot)$ denotes a generic updating function. The $\mathbf{v}_t$ is usually computed from the second-order moment (i.e., coordinate-wise square of the stochastic gradient estimator). For our problem, we compute $\mathbf{v}_t$ from $\tilde{\nabla}_t^2$, and $h_t(\cdot)$ can be implemented by the following methods for having different adaptive step sizes:

$$\text{Adam-style: } \mathbf{v}_{t+1} = (1 - \beta_t')\mathbf{v}_t + \beta_t' \tilde{\nabla}_t^2$$

$$\text{AdaGrad-style: } \mathbf{v}_{t+1} = \frac{1}{t+1} \sum_{i=1}^{t} \tilde{\nabla}_i^2$$

$$\text{AMSGrad-style: } \mathbf{v}_{t+1}' = (1 - \beta_t')\mathbf{v}_t' + \beta_t' \tilde{\nabla}_t^2, \quad (8)$$
$$\mathbf{v}_{t+1} = \max(\mathbf{v}_t, \mathbf{v}_{t+1}')$$

$$\text{AdaBound-style: } \mathbf{v}_{t+1}' = (1 - \beta_t')\mathbf{v}_t' + \beta_t' \tilde{\nabla}_t^2,$$
$$\mathbf{v}_{t+1} = \Pi_{[1/c_u^2, 1/c_l^2]}[\mathbf{v}_{t+1}']$$

where $c_l < c_u$ are two parameters of the AdaBound method and $\Pi_{[a,b]}$ denotes a clipping operator that clips the input argument into the range $[a, b]$. Given $\mathbf{v}_{t+1}$, we update the model parameter by

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta}{\sqrt{\mathbf{v}_{t+1}} + \delta} \mathbf{m}_{t+1},$$

where $\delta > 0$ is a parameter. The convergence analysis of ADAP relies on the boundness of the step size scaling factor $\eta_t^s = 1/(\sqrt{\mathbf{v}_{t+1}} + \delta)$ following similarly a recent study on the convergence of the Adam-style algorithms for standard stochastic non-convex optimization (Guo et al., 2021a). Specifically, under Assumptions 1 and 2, we could show that $\|\tilde{\nabla}_t\|_\infty$ are bounded by a constant, which further implies that the step size scaling factor $\eta_t^s = 1/(\sqrt{\mathbf{v}_{t+1}} + \delta)$ is upper bounded and lower bounded by some constants c_u and c_l. Finally, we present the convergence of Algorithm 3 below.

Theorem 4. *Suppose Assumptions 1 and 2 hold. Then, Algorithm 3 with $\beta_0 = O(B\epsilon^2/m)$, $\beta_1 = O(B\epsilon^2/m)$, $\eta = O(\beta)$, $T = O(m/[B\epsilon^4])$ and any of the methods in (8) for computing $\mathbf{v}_{t+1}$ ensures that $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\nabla F(\mathbf{w}_t)\|^2\right] \leq \epsilon^2$.*

Remark 5. We note that previous work (Qi et al., 2021) also proposed a Adam-style algorithm, but their analysis only covers the AMSGrad-style adaptive step size and has a worse iteration complexity in the order of $O(m/\epsilon^5)$ for using a stochastic $\mathcal{B}_t$.

Finally, it is worth mentioning that both Theorem 3 and Theorem 4 can be extended to using a decreasing step size η and parameters β_0, β_1 in the order of $1/\sqrt{t}$, where t denotes the current round. We refer the readers to the supplement for the proofs.

4 EXPERIMENTS

In this section, we conduct experiments on benchmark datasets to demonstrate the effectiveness of the proposed algorithms. More results can be found in the Appendix.

4.1 Optimizing Linear Models

We first consider learning a linear model for prediction. For baselines, we choose three state-of-the-art methods for stochastic optimization of AP, namely SOAP (SGD) (Qi et al., 2021), SOAP (Adam) (Qi et al., 2021) and SmoothAP (Brown et al., 2020). For our algorithms, we implement MOAP (v2) and ADAP (Adam-style).

Datasets. We use six imbalanced benchmark datasets from LIBSVM data (Chang and Lin, 2011), whose statistics are summarized in the appendix. For all data sets, we scale the feature vectors into $[0, 1]$. For *mushrooms* and *phishing* dataset, since the original class distribution is relatively balanced, and no testing set is given, we randomly drop a set of positive data from the training examples, and divide them into training and testing according to 2:1 ratio.

Table 2: Statistics of Datasets.

Data Set	#training examples	#testing examples	Proportion of positive data
mushrooms	2920	1504	5.27%
phishing	4987	2568	35.65%
w6a	17188	32561	3.05%
a9a	32561	16281	24.08%
w8a	49749	14951	2.97%
ijcnn1	49990	91701	9.71%

(a) mushrooms dataset   (b) phishing dataset   (c) w6a dataset

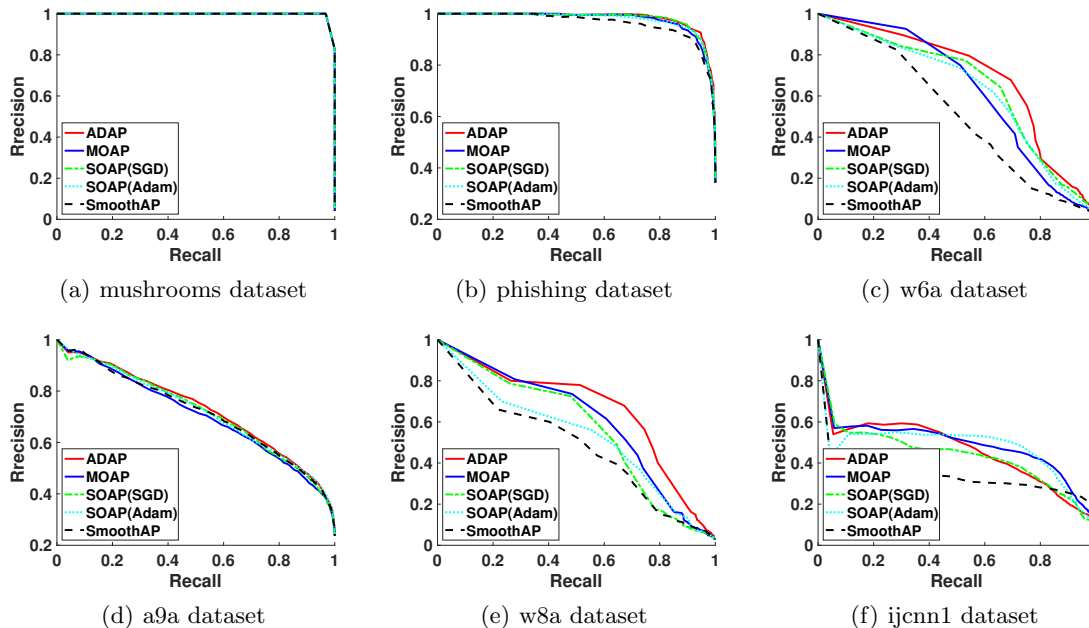(d) a9a dataset   (e) w8a dataset   (f) ijcnn1 dataset

Figure 1: Precision-Recall curves of the Final models on the testing set

Configurations. In all experiments, we use the sigmoid function $1/(\exp(-\mathbf{w}^\top \mathbf{x}) + 1)$ to generate a prediction score for computing the AP. We set the ℓ_2-regulation parameter as 10^{-4}, the mini-batch size as 20, and run a total of $T = 500$ iterations. For MOAP, ADAP, SOAP, we choose the squared hinge ls as the surrogate function following Qi et al. (2021). For SmoothAP, we apply the sigmoid function to approximate the indicator function following Brown et al. (2020). Other involved parameters for each algorithm are tuned on the training data. For MOAP and ADAP, we decrease η and $\beta_0 = \beta_1$ on the order of $O(1/\sqrt{t})$ according to the theoretical analysis, and tune the initial value of the step size in the set $\{20, 10, 1, 10^{-1}, 10^{-2}\}$, and $\beta_0 = \beta_1$ in the set $\{0.9, 0.5, 0.1\}$. For other algorithms, we observe poor performance when using a decreasing step size and β parameters, and thus report their results by using fixed parameters tuned in the same range as our algorithms. We repeat each algorithm 5 times on each data and report the averaged results.

Results. The convergence curves of AP on training examples are reported in Figure 2, and the final AP scores on the testing data are shown in Table 3. We also plot the Precision-Recall curves of the final models on testing data in Figure 1. From Figure 2, we can see that the proposed ADAP converges faster than other methods. MOAP converges faster than SOAP on mushrooms, phishing, and w8a, and has similar performance as SOAP on other three datasets. From Table 3, we can see that ADAP has the best performance on all data sets except ijcnn1, on which ADAP is similar to SOAP (Adam). These results verify the effectiveness of the proposed methods, in particular the adaptive method ADAP.

4.2 Training Deep Neural Networks

Next, we present empirical results on optimizing deep neural networks. We mainly focus on comparing the performance of our proposed ADAP (Adam-style) algorithm with the two state-of-the-art algo-

Table 3: Final averaged AP scores on the testing data.

Method	mushrooms	phishing	w6a
MOAP	$0.999 \pm 2E\text{-}6$	$0.978 \pm 2E\text{-}6$	$0.596 \pm 2E\text{-}3$
ADAP	**1**	**$0.981 \pm 2E\text{-}7$**	**$0.675 \pm 1E\text{-}4$**
SOAP (SGD)	$0.997 \pm 1E\text{-}5$	$0.978 \pm 4E\text{-}6$	$0.612 \pm 4E\text{-}5$
SOAP (Adam)	1	$0.977 \pm 1E\text{-}6$	$0.592 \pm 3E\text{-}4$
SmoothAP	$0.962 \pm 1E\text{-}3$	$0.967 \pm 8E\text{-}6$	$0.473 \pm 1E\text{-}4$

Method	a9a	w8a	ijcnn1
MOAP	$0.715 \pm 3E\text{-}5$	$0.584 \pm 4E\text{-}4$	$0.471 \pm 6E\text{-}4$
ADAP	**$0.730 \pm 1E\text{-}6$**	**$0.659 \pm 5E\text{-}4$**	$0.536 \pm 1E\text{-}3$
SOAP (SGD)	$0.714 \pm 3E\text{-}5$	$0.561 \pm 1E\text{-}3$	$0.476 \pm 2E\text{-}4$
SOAP (Adam)	$0.721 \pm 1E\text{-}5$	$0.475 \pm 2E\text{-}3$	**$0.547 \pm 1E\text{-}3$**
SmoothAP	$0.713 \pm 3E\text{-}5$	$0.459 \pm 3E\text{-}4$	$0.374 \pm 3E\text{-}4$

(a) mushrooms dataset   (b) phishing dataset   (c) w6a dataset
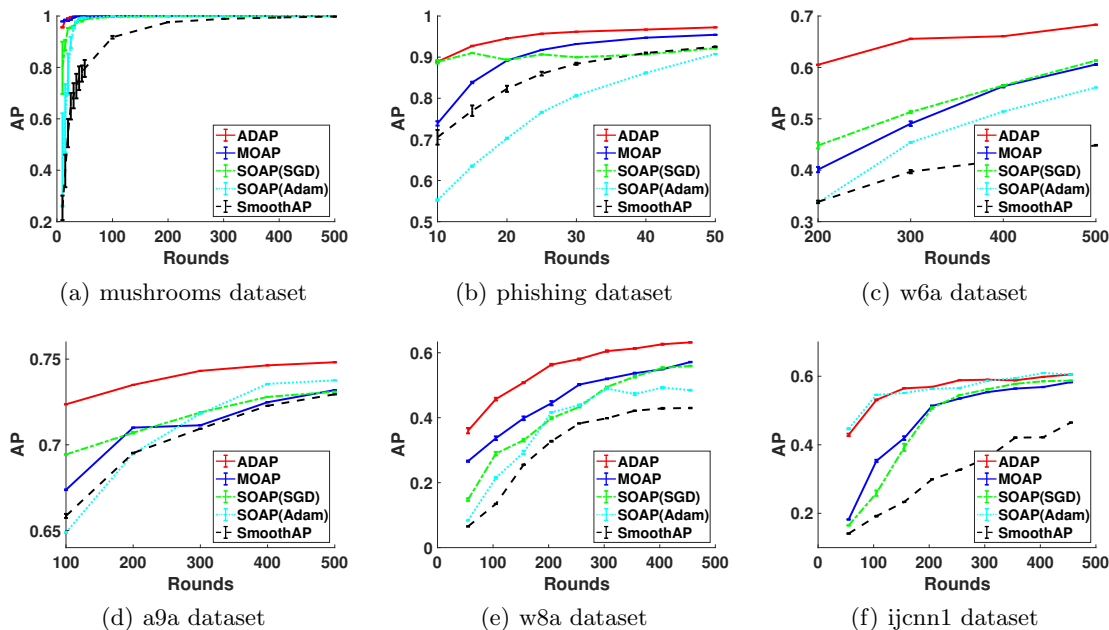
(d) a9a dataset   (e) w8a dataset   (f) ijcnn1 dataset

Figure 2: AP vs # of rounds on the training set

Table 4: Test AUPRC on CIFAR10 and CIFAR100.

Method	CIFAR10	CIFAR100
ADAP	**0.7667 ± 0.0015**	**0.6371 ± 0.0041**
SOAP (Adam)	0.7629 ± 0.0014	0.6251 ± 0.0053
SmoothAP	0.7365 ± 0.0088	0.6071 ± 0.0064

rithms, i.e., SmoothAP (Brown et al., 2020) and SOAP (Adam) (Qi et al., 2021).

Datasets. Following Qi et al. (2021), we conduct experiments on the imbalanced binary-class version of CIFAR10 and CIFAR100 datasets, which are constructed as follows: Firstly, half of the classes in the original CIFAR10 and CIFAR100 datasets are designated to be the positive class, and the rest half of classes are considered to be the negative class. Then, we remove 98% of the positive examples in the training set to make it imbalanced, while keeping the test set unchanged. Finally, the training data set is splited into train/validation sets as a $4 : 1$ ratio.

Configurations. We choose ResNet-18 (He et al., 2016) to be the neural network for our imbalanced binary image classification task. Before the training process, similar to (Qi et al., 2021), the ResNet-18 model is initialized with a model pretrained by the Adam optimizer for optimizing the cross entropy loss, whose learning rate and weight decay parameter are searched in $\{10^{-6}, 10^{-5}, 10^{-4}\}$. Then, the last layer of the neural network is re-initialized and the network is trained by different AP maximization methods, with

hyper-parameters individually tuned for fair comparison. For SOAP and ADAP, we tune the β_0 parameter in range $\{0.001, 0.1, 0.5, 0.9, 0.99, 0.999\}$, the margin parameter m in $\{0.5, 1, 2, 5, 10\}$, and learning rate η in $\{10^{-6}, 10^{-5}, 10^{-4}\}$.

Results We conduct each experiment for 5 times, and report the mean test AUPRC as well as the standard variation in Table 4. We could observe that our proposed ADAP enjoys the best results on both datasets.

5   CONCLUSION AND FUTURE WORK

In this paper, we investigate the stochastic optimization of AUPRC by maximizing a surrogate loss of AP, which is one of the most important performance metrics for imbalanced classification problems. Previous study has shown that an $O(1/\epsilon^5)$ convergence rate can be achieved. In this work, we further improve the theoretical guarantee by proposing a stochastic momentum method as well as a family of adaptive methods, which enjoy a better convergence rate of $O(1/\epsilon^4)$. Our essential ideas the are two-folds: (i) updating the biased estimator for individual ranking-scores-tracking in a randomized coordinate-wise manner; (ii) applying the momentum step on top of the stochastic gradient estimator for tracking the gradient of the objective. Empirical studies on optimizing linear models and deep networks show the effectiveness of the proposed adaptive methods. For future work, we consider applying the proposed algorithms to other deep learning tasks.

Acknowledgements

References

Krishnakumar Balasubramanian, Saeed Ghadimi, and Anthony Nguyen. Stochastic multi-level composition optimization algorithms with level-independent convergence rates. *arXiv preprint arXiv:2008.10526*, 2020.

Kendrick Boyd, Vitor Santos Costa, Jesse Davis, and David Page. Unachievable region in precision-recall space and its effect on empirical evaluation. In *Proceedings of 29th International Conference on Machine Learning*, pages 639–646, 2012.

Kendrick Boyd, Kevin H Eng, and C David Page. Area under the precision-recall curve: point estimates and confidence intervals. In *Proceedings of the 13rd Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 451–466, 2013.

Andrew Brown, Weidi Xie, Vicky Kalogeiton, and Andrew Zisserman. Smooth-ap: Smoothing the path towards large-scale image retrieval. In *Proceedings of the 6th European Conference on Computer Vision*, pages 677–694, 2020.

Christopher Burges, Robert Ragno, and Quoc Le. Learning to rank with nonsmooth cost functions. In *Advances in Neural Information Processing Systems 19*, 2007.

Fatih Cakir, Kun He, Xide Xia, Brian Kulis, and Stan Sclaroff. Deep metric learning to rank. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:1–27, 2011.

Kean Chen, Jianguo Li, Weiyao Lin, John See, Ji Wang, Lingyu Duan, Zhibo Chen, Changwei He, and Junni Zou. Towards accurate one-stage object detection with ap-loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Tianyi Chen, Yuejiao Sun, and Wotao Yin. Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization. *IEEE Transactions on Signal Processing*, 69:4937–4948, 2021.

Wei Chen, Tie-Yan Liu, Yanyan Lan, Zhiming Ma, and Hang Li. Ranking measures and loss functions in learning to rank. In *Advances of International Conference on Neural Information Processing Systems 22*, page 315–323, 2009.

Stephan Clemencon and Nicolas Vayatis. Nonparametric estimation of the precision-recall curve. In *Proceedings of the 26th International Conference on Machine Learning*, pages 185–192, 2009.

Corinna Cortes and Mehryar Mohri. Auc optimization vs. error rate minimization. *Advances in Neural Information Processing Systems 16*, pages 313–320, 2003.

Jesse Davis and Mark Goadrich. The Relationship Between Precision-Recall and ROC Curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006a.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 233–240, 2006b.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

Elad Eban, Mariano Schain, Alan Mackey, Ariel Gordon, Ryan Rifkin, and Gal Elidan. Scalable learning of non-decomposable objectives. In *the 20th International Conference on Artificial Intelligence and Statistics*, pages 832–840, 2017.

César Ferri, José Hernández-Orallo, and R Modroiu. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1):27–38, 2009.

Peter A Flach and Meelis Kull. Precision-recall-gain curves: Pr analysis done right. In *Advances in Neural Information Processing Systems 28*, pages 838–846, 2015.

Wei Gao, Rong Jin, Shenghuo Zhu, and Zhi-Hua Zhou. One-pass auc optimization. In *Proceedings of the 30th International Conference on Machine learning*, pages 906–914, 2013.

Xingyu Gao, Zhenyu Chen, Sheng Tang, Yongdong Zhang, and Jintao Li. Adaptive weighted imbalance learning with application to abnormal activity recognition. *Neurocomputing*, 173:1927–1935, 2016.

Saeed Ghadimi, Andrzej Ruszczynski, and Mengdi Wang. A single timescale stochastic approximation method for nested stochastic optimization. *SIAM Journal on Optimization*, 30(1):960–979, 2020.

Zhishuai Guo, Mingrui Liu, Zhuoning Yuan, Li Shen, Wei Liu, and Tianbao Yang. Communication-efficient distributed stochastic auc maximization with deep neural networks. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3864–3874, 2020.

Zhishuai Guo, Yi Xu, Wotao Yin, Rong Jin, and Tianbao Yang. On stochastic moving-average estimators for non-convex optimization. *CoRR*, abs/2104.14840, 2021a.

Zhishuai Guo, Yi Xu, Wotao Yin, Rong Jin, and Tianbao Yang. On stochastic moving-average estimators for non-convex optimization. *arXiv preprint arXiv:2104.14840*, 2021b.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Paul Henderson and Vittorio Ferrari. End-to-end training of object class detectors for mean average precision. In *Proceedings of the 13th Asian Conference on Computer Vision*, pages 198–213. Springer, 2016.

Alan Herschtal and Bhavani Raskutti. Optimising area under the roc curve using gradient descent. In *Proceedings of the 21st international conference on Machine learning*, pages 49–57, 2004.

Yifan Hu, Siqi Zhang, Xin Chen, and Niao He. Biased stochastic first-order methods for conditional stochastic optimization and applications in meta learning. *Advances in Neural Information Processing Systems 33*, 2020.

Zhouyuan Huo, Bin Gu, Ji Liu, and Heng Huang. Accelerated method for stochastic composition optimization with nonsmooth regularization. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 3287–3294, 2018.

Thorsten Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine learning*, pages 377–384, 2005.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Stanley Kok and Pedro M Domingos. Learning markov logic networks using structural motifs. In *Proceedings of the 21st international conference on Machine learning*, 2010.

Bartosz Krawczyk, Mikel Galar, Lukasz Jelen, and Francisco Herrera. Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy. *Applied Soft Computing*, 38:714–726, 2016.

Xiangru Lian, Mengdi Wang, and Ji Liu. Finite-sum composition optimization via variance reduced gradient descent. In *The 20th International Conference on Artificial Intelligence and Statistics*, pages 1159–1167, 2017.

Tianyi Lin, Chenyou Fan, Mengdi Wang, and Michael I Jordan. Improved oracle complexity for stochastic compositional variance reduced gradient. *arXiv preprint arXiv:1806.00458*, 2018.

Liu Liu, Ji Liu, and Dacheng Tao. Dualityfree methods for stochastic composition optimization. *Advances in neural information processing systems 31*, pages 1205–1217, 2018a.

Mingrui Liu, Xiaoxuan Zhang, Zaiyi Chen, Xiaoyu Wang, and Tianbao Yang. Fast stochastic auc maximization with $o(1/n)$-convergence rate. In *Proceedings of the 35th International Conference on Machine Learning*, pages 3189–3197, 2018b.

Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. In *the 7th International Conference on Learning Representations*, 2019.

Brian McFee and Gert Lanckriet. Metric learning to rank. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, page 775–782, 2010. ISBN 9781605589077.

Donald Metzler and W Bruce Croft. 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*, pages 472–479, 2005.

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

Michael Natole, Yiming Ying, and Siwei Lyu. Stochastic proximal algorithms for auc maximization. In *Proceedings of the 35th International Conference on Machine Learning*, pages 3710–3719. PMLR, 2018.

Kemal Oksuz, Baris Can Cam, Emre Akbas, and Sinan Kalkan. A ranking-based, balanced loss function unifying classification and localisation in object detection. In *Advances in Neural Information Processing Systems 33*, pages 15534–15545, 2020.

Qi Qi, Youzhi Luo, Zhao Xu, Shuiwang Ji, and Tianbao Yang. Stochastic optimization of area under precision-recall curve for deep learning with provable convergence. *Advances in Neural Information Processing Systems 34*, 2021.

Tao Qin, Tie-Yan Liu, and Hang Li. A general approximation framework for direct optimization of information retrieval measures. *Information retrieval*, 13 (4):375–397, 2010.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *the 6th International Conference on Learning Representations*, 2018.

Michal Rolinek, Vit Musil, Anselm Paulus, Marin Vlastelica, Claudio Michaelis, and Georg Martius. Optimizing rank-based metrics with blackbox differentiation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Yang Song, Alexander Schwing, Raquel Urtasun, et al. Training deep neural networks via direct loss minimization. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2169–2177, 2016.

Mengdi Wang, Ji Liu, and Ethan X Fang. Accelerating stochastic composition optimization. *arXiv preprint arXiv:1607.07329*, 2016.

Mengdi Wang, Ethan X Fang, and Han Liu. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Mathematical Programming*, 161(1-2):419–449, 2017.

Shuoguang Yang, Mengdi Wang, and Ethan X Fang. Multilevel stochastic gradient methods for nested composition optimization. *SIAM Journal on Optimization*, 29(1):616–659, 2019.

Tianbao Yang, Qihang Lin, and Zhe Li. Unified convergence analysis of stochastic momentum methods for convex and non-convex optimization. *arXiv preprint arXiv:1604.03257*, 2016.

Yiming Ying, Longyin Wen, and Siwei Lyu. Stochastic online auc maximization. In *Advances in neural information processing systems 30*, pages 451–459, 2016.

Zhuoning Yuan, Yan Yan, Milan Sonka, and Tianbao Yang. Robust deep auc maximization: A new surrogate loss and empirical studies on medical image classification. *arXiv preprint arXiv:2012.03173*, 2020.

Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. 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*, pages 271–278, 2007.

Junyu Zhang and Lin Xiao. A stochastic composite gradient method with incremental variance reduction. *arXiv preprint arXiv:1906.10186*, 2019.

Peilin Zhao, Steven CH Hoi, Rong Jin, and Tianbao Yang. Online auc maximization. In *Proceedings of the 28th International Conference on Machine Learning*, pages 233–240, 2011.

Supplementary Material: Momentum Accelerates the Convergence of Stochastic AUPRC Maximization

A NOTATIONS

We first introduce the following lemma, which demonstrates the basic properties of the functions in our optimization problem (4).

Lemma 1. *Suppose Assumptions 1 and 2 hold, then there exist $G, C_g, L_g, C_f, L_f, C_F, L_F > 0$, such that $\forall \mathbf{w}$, $\|g_i(\mathbf{w})\|^2 \leq G^2$, $g_i(\mathbf{w})$ is C_g-Lipschitz and L_g-smooth, and $\forall \mathbf{u} \in \Omega$, $f(\mathbf{u})$ is C_f-Lipschitz and L_f-smooth. Finally, $\forall \mathbf{w}$, $F(\mathbf{w})$ is C_F-Lipschiz and L_F-smooth.*

Proof. For $g_i(\mathbf{w})$, we have $\|g_i(\mathbf{w})\|^2 \leq m^2 M^2 + n^2 M^2 = G^2$, $\|\nabla g_i(\mathbf{w})\|^2 \leq m^2\sigma^2 + n^2\sigma^2 = C_g^2$, and $\|\nabla g_i(\mathbf{w}) - \nabla g_i(\mathbf{w}')\|^2 \leq m^2 L_l^2 \|\mathbf{w} - \mathbf{w}'\|^2 + n^2 L_l^2 \|\mathbf{w} - \mathbf{w}'\|^2 = L_g^2 \|\mathbf{w} - \mathbf{w}'\|^2$. Similarly, for $f(\mathbf{u})$, we have $\|\nabla f(\mathbf{u})\|_2^2 = \|(\frac{-1}{u_2}, \frac{u_1}{u_2^2})\|^2 \leq 1/C^2 + M^2 m^2/C^2 = C_f^2$ and $\|\nabla f(\mathbf{u}) - \nabla f(\mathbf{u}')\|_2^2 = \|(\frac{-1}{u_2} - \frac{-1}{u_2'}, \frac{u_1}{u_2^2} - \frac{u_1'}{u_2'^2})\|^2 \leq \frac{(u_2 - u_2')^2}{C^2} + \frac{2(u_1 - u_1')^2}{C^4} + \frac{8(Mm)^2(Mn)^2(u_2 - u_2')^2}{C^8} \leq L_f^2(\|\mathbf{u} - \mathbf{u}'\|^2)$ for any $\mathbf{u}, \mathbf{u}' \in \Omega$. Hence, $f(g_i(\mathbf{w}))$ is $C_F = (C_g C_f)$-Lipschitz, and $L_F = (C_g^2 L_f + L_g C_f)$-smooth. Since $F(\mathbf{w})$ is the average of $f(g_i(\mathbf{w}))$, it is also C_F-Lipschitz and L_F-smooth. $\square$

B PROOF OF THEOREM 1

Let $\Delta_t = \mathbf{m}_{t+1} - \nabla F(\mathbf{w}_t)$. According to the update rule in Algorithm 1, and we have

$$
\begin{aligned}
\|\Delta_t\|^2 = &\left\| (1 - \beta_1)\mathbf{m}_t + \beta_1 \frac{1}{m} \sum_{i=1}^m \nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_i^\top\right) - \nabla F(\mathbf{w}_t) \right\|^2 \\
= &\left\| \underbrace{(1 - \beta_1)[\mathbf{m}_t - \nabla F(\mathbf{w}_{t-1})]}_{A_1} + \underbrace{(1 - \beta_1)[\nabla F(\mathbf{w}_{t-1}) - \nabla F(\mathbf{w}_t)]}_{A_2} \right. \\
&+ \beta_1 \underbrace{\left[\frac{1}{m} \sum_{i=1}^m \nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_i^\top\right) - \frac{1}{m} \sum_{i=1}^m \nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left(g_i(\mathbf{w}_t)\right) \right]}_{A_3} \\
&\left. + \beta_1 \underbrace{\left[\frac{1}{m} \sum_{i=1}^m \nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left(g_i(\mathbf{w}_t)\right) - \nabla F(\mathbf{w}_t) \right]}_{A_4} \right\|^2 .
\end{aligned}
\tag{9}
$$

Let $\mathrm{E}_t[\cdot] = \mathrm{E}[\cdot | \mathcal{F}_{t-1}]$, where $\mathcal{F}_{t-1}$ is the σ-algebra of MOAP-V1, which contains the learning history from round 1 to round $t-1$. Since $\tilde{g}_t(\mathbf{w})$ is an unbiased estimator of $g_i(\mathbf{w})$, we have $\mathrm{E}_t[A_4] = 0$. Moreover, A_1 and A_2 are

fixed given $\mathcal{F}_{t-1}$. Thus

$$
\begin{aligned}
&\mathrm{E}_t[\|\Delta_t\|^2] \\
&\leq \mathrm{E}_t[\|A_1 + A_2 + A_3\|^2] + \mathrm{E}_t[\|A_4\|^2] + 2\mathrm{E}_t[\|A_3\|\|A_4\|] + \underbrace{2(A_1 + A_2)^\top \mathrm{E}_t[A_4]}_{=0} \\
&\leq \mathrm{E}_t[\|A_1 + A_2 + A_3\|^2] + 2\mathrm{E}_t[\|A_4\|^2] + \mathrm{E}_t[\|A_3\|^2] \\
&\leq (1+\beta_1)\mathrm{E}_t[\|A_1\|^2] + \left(1 + \frac{1}{\beta_1}\right)\mathrm{E}_t[\|A_2 + A_3\|^2] + 2\mathrm{E}_t[\|A_4\|^2] + \mathrm{E}_t[\|A_3\|^2] \\
&\leq (1+\beta_1^2)(1-\beta_1)\|\Delta_{t-1}\|^2 + \frac{2(1+\beta_1)}{\beta_1}\|A_2\|^2 + \frac{2+3\beta_1}{\beta_1}\mathrm{E}_t[\|A_3\|^2] + 2\mathrm{E}_t[\|A_4\|^2],
\end{aligned}
\tag{10}
$$

where the third inequality is based on Young's inequality. To proceed, according to Lemma 1, we have

$$
\|A_2\|^2 = (1-\beta_1)^2\|\nabla F(\mathbf{w}_{t-1}) - \nabla F(\mathbf{w}_t)\|^2 \leq (1-\beta_1)^2 L_F^2\|\mathbf{w}_t - \mathbf{w}_{t-1}\|^2,
\tag{11}
$$

$$
\begin{aligned}
\|A_3\|^2 &\leq \beta_1^2 \frac{1}{m} \sum_{i=1}^m \|\nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_i^\top\right) - \nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left(g_i(\mathbf{w}_t)\right)\|^2 \\
&\leq \beta_1^2 \frac{C_g^2}{m} \sum_{i=1}^m \|\nabla f([U_{t+1}]_i^\top) - \nabla f(g_i(\mathbf{w}_t))\|^2 \\
&\leq \beta_1^2 \frac{C_g^2 L_f^2}{m} \sum_{i=1}^m \|[U_{t+1}]_i^\top - g_i(\mathbf{w}_t)\|^2 = \frac{\beta_1^2 C_g^2 L_f^2}{m}\|U_{t+1} - g(\mathbf{w}_t)\|^2,
\end{aligned}
\tag{12}
$$

where $g(\mathbf{w}_t) = [g_1(\mathbf{w}_t), \dots, g_m(\mathbf{w}_t)]^\top$, and

$$
\mathrm{E}_t[\|A_4\|^2] \leq \beta_1^2 \mathrm{E}_t\left[\left\|\frac{1}{m}\sum_{i=1}^m \nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f([U_{t+1}]_i^\top)\right\|^2\right] \leq \beta_1^2 C_g^2 C_f^2.
\tag{13}
$$

Combining (10), (11), (12), (13), we get

$$
\begin{aligned}
\mathrm{E}_t[\|\Delta_t\|^2] \leq{}& (1-\beta_1)\|\Delta_{t-1}\|^2 + \frac{2L_F^2\|\mathbf{w}_t - \mathbf{w}_{t-1}\|^2}{\beta_1} + \frac{5\beta_1 C_g^2 L_f^2 \mathrm{E}_t[\|U_{t+1} - g(\mathbf{w}_t)\|^2]}{m} \\
&+ 2\beta_1^2 C_g^2 C_f^2.
\end{aligned}
\tag{14}
$$

Summing over from 1 to T, we get

$$
\begin{aligned}
\mathrm{E}\left[\sum_{i=1}^T \|\Delta_i\|^2\right] \leq{}& \frac{\|\Delta_0\|^2}{\beta_1} + \frac{2L_F^2 \sum_{i=1}^T \|\mathbf{w}_i - \mathbf{w}_{i-1}\|^2}{\beta_1^2} + \frac{5C_g^2 L_f^2 \sum_{i=1}^T \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2]}{m} \\
&+ 2\beta_1 C_g^2 C_f^2 T \\
={}& \frac{2C_F^2}{\beta_1} + \frac{2L_F^2 \eta^2 \sum_{i=1}^T \|\mathbf{m}_i\|^2}{\beta^2} + \frac{5C_g^2 L_f^2 \sum_{i=1}^T \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2]}{m} \\
&+ 2\beta_1 C_g^2 C_f^2 T
\end{aligned}
\tag{15}
$$

Note that here $\mathbf{w}_0$ can be considered as a psudo-decision which follows the same update procedure as $\mathbf{w}_t$. Next, we turn to bound the third term of the R.H.S. of the inequality, and introduce the following lemma.

Lemma 2 (Variance Recursion Wang et al. (2017))**.** *Consider a sequence $\mathbf{m}_{t+1} = (1-\beta)\mathbf{m}_t + \beta \tilde{h}(\mathbf{w}_t)$ for tracking $h(\mathbf{w}_t)$, where $\mathrm{E}[\tilde{h}(\mathbf{w}_t)] = h(\mathbf{w}_t)$ and h is a C-Lipchitz continuous mapping. Then we have*

$$
\mathrm{E}_t[\|\mathbf{m}_{t+1} - h(\mathbf{w}_t)\|^2] \leq (1-\beta)\|\mathbf{m}_t - h(\mathbf{w}_{t-1})\|^2 + \beta^2 \mathrm{E}_t[\|\tilde{h}(\mathbf{w}_t) - h(\mathbf{w}_t)\|^2] + \frac{2C^2\|\mathbf{w}_t - \mathbf{w}_{t-1}\|^2}{\beta}.
\tag{16}
$$

With Lemma 2, we have

$$
\sum_{i=1}^T \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2] \leq \frac{mG^2}{\beta_0} + 2\beta_0 T m G^2 + \frac{2m C_g^2 \eta^2 \sum_{i=1}^T \|\mathbf{m}_i\|^2}{\beta_0^2}.
\tag{17}
$$

Combining all the about results together, and set $\beta_0 = \beta_1 = \beta$, we have

$$\mathrm{E}\left[\sum_{i=1}^{T}\|\Delta_i\|^2\right] \leq \frac{2C_F^2 + 5C_g^2 L_f^2 G^2}{\beta} + \frac{(2L_F^2 + 10C_g^4 L_f^2)\eta^2}{\beta^2} \sum_{i=1}^{T}\|\mathbf{m}_i\|^2 \tag{18}$$
$$+ \beta(10C_g^2 L_f^2 G^2 + 2C_g^2 C_f^2)T.$$

We introduce the following lemma Guo et al. (2021b), which can be obtained following the definition of the smoothness.

Lemma 3. *Consider a sequence* $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\mathbf{m}_{t+1}$ *for a* L_F*-smooth function* F*, with* $\eta L_F \leq 1/2$ *we have*

$$F(\mathbf{w}_{t+1}) \leq F(\mathbf{w}_t) + \frac{\eta}{2}\|\nabla F(\mathbf{w}_t) - \mathbf{m}_{t+1}\|^2 - \frac{\eta}{2}\|\nabla F(\mathbf{w}_t)\|^2 - \frac{\eta}{4}\|\mathbf{m}_{t+1}\|^2.$$

Based on Lemma 3, with $(2L_F^2 + 10C_g^4 L_f^2)\eta^2/\beta^2 \leq 1/2$, we get

$$\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\nabla F_{\mathbf{w}}(\mathbf{w}_t)\|^2\right] \leq \frac{2(F(\mathbf{w}_{T+1}) - F(\mathbf{w}_1))}{\eta T} + \frac{C_F^2 + 5C_g^2 L_f^2 G^2}{\beta T} \tag{19}$$
$$+ \beta(10C_g^2 L_f^2 G^2 + 2C_g^2 C_f^2).$$

Thus, by setting $\beta = O(\epsilon^2)$ and $\eta = O(\beta)$, we have $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\nabla F_{\mathbf{w}}(\mathbf{w}_t)\|^2\right] = O(\epsilon^2)$ for

$$T \geq \max\left\{\frac{2(F(\mathbf{w}_1) - F(\mathbf{w}_{T+1}))}{\eta\epsilon^2}, \frac{C_F^2 + 10C_g^2 L_f^2 G^2}{\beta\epsilon^2}\right\}.$$

C   PROOF OF THEOREM 2

Let i_t be the positive data chosen in round t, and for convenience we set $\beta_0 = \beta_1 = \beta$. According to the update rule in Algorithm 1, and we have

$$\|\Delta_t\|^2 = \left\|(1-\beta)\mathbf{m}_t + \beta\nabla\tilde{g}_{i_t}(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_{i_t}^\top\right) - \nabla F(\mathbf{w}_t)\right\|^2$$
$$= \left\|\underbrace{(1-\beta)[\mathbf{m}_t - \nabla F(\mathbf{w}_{t-1})]}_{A_1} + \underbrace{(1-\beta)[\nabla F(\mathbf{w}_{t-1}) - \nabla F(\mathbf{w}_t)]}_{A_2}\right.$$
$$+ \underbrace{\beta\left[\nabla\tilde{g}_{i_t}(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_{i_t}^\top\right) - \nabla\tilde{g}_{i_t}(\mathbf{w}_t)^\top \cdot \nabla f\left(g_{i_t}(\mathbf{w}_t)\right)\right]}_{A_3} \tag{20}$$
$$+ \left.\underbrace{\beta\left[\nabla\tilde{g}_{i_t}(\mathbf{w}_t)^\top \cdot \nabla f\left(g_{i_t}(\mathbf{w}_t)\right) - \nabla F(\mathbf{w}_t)\right]}_{A_4}\right\|^2.$$

Thus

$$\mathrm{E}_t[\|\Delta_t\|^2] \leq (1-\beta)\|\Delta_{t-1}\|^2 + \frac{2(1+\beta)}{\beta}\|A_2\|^2 + \frac{2+3\beta}{\beta}\mathrm{E}_t[\|A_3\|^2] + 2\mathrm{E}_t[\|A_4\|^2], \tag{21}$$

where

$$\|A_2\|^2 = (1-\beta_1)^2\|\nabla F(\mathbf{w}_{t-1}) - \nabla F(\mathbf{w}_t)\|^2 \leq (1-\beta_1)^2 L_F^2\|\mathbf{w}_t - \mathbf{w}_{t-1}\|^2, \tag{22}$$

$$\|A_3\|^2 \leq \beta^2\|\nabla\tilde{g}_{i_t}(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_{i_t}^\top\right) - \nabla\tilde{g}_{i_t}(\mathbf{w}_t)^\top \cdot \nabla f\left(g_{i_t}(\mathbf{w}_t)\right)\|^2$$
$$\leq \beta^2 C_g^2\|\nabla f([U_{t+1}]_{i_t}^\top) - \nabla f(g_{i_t}(\mathbf{w}_t))\|^2 \tag{23}$$
$$\leq \beta^2 C_g^2 L_f^2\|[U_{t+1}]_{i_t}^\top - g_{i_t}(\mathbf{w}_t)\|^2,$$

and

$$\mathrm{E}[\|A_4\|^2] \leq \beta^2\mathrm{E}[\|\nabla\tilde{g}_{i_t}(\mathbf{w}_t)^\top \cdot \nabla f([U_{t+1}]_{i_t}^\top)\|^2] \leq \beta_1^2 C_g^2 C_f^2. \tag{24}$$

Combining the results above, we get

$$
\begin{aligned}
\mathrm{E}[\|\Delta_t\|^2] \leq{} & (1-\beta)\|\Delta_{t-1}\|^2 + \frac{2L_F^2\|\mathbf{w}_t - \mathbf{w}_{t-1}\|^2}{\beta_1} + 5\beta C_g^2 L_f^2 \mathrm{E}[\|[U_{t+1}]_{i_t} - g_{i_t}(\mathbf{w}_t)\|^2] \\
& + 2\beta^2 C_g^2 C_f^2.
\end{aligned}
\tag{25}
$$

Next, we turn to bound the third term on the R.H.S. of (25). Following Qi et al. (2021), we divide the whole interval into m groups, with the i-th group $\mathcal{T}_i = \{t_1^i, \ldots, t_k^i, \ldots\}$, where t_k^i denotes the k-th time that the i-th positive data is chosen to update $[U_{t_k^i}]_i$ and obtain $[U_{t_k^i+1}]_i$. Note that since we only update the chosen data, we have $[U_{t_k^i+1}]_i = \cdots = [U_{t_{k+1}^i}]_i$. Without loss of generality, assume i_t is picked for the k-th time at round t. Thus

$$
\begin{aligned}
& \mathrm{E}[\|[U_{t+1}]_{i_t} - g_{i_t}(\mathbf{w}_t)\|^2] \\
={} & \mathrm{E}[\|[U_{t_k^{i_t}+1}]_{i_t} - g_{i_t}(\mathbf{w}_{t_k^{i_t}})\|^2] \\
\leq{} & \mathrm{E}[\|(1-\beta)[U_{t_{k-1}^{i_t}+1}]_{i_t} + \beta\tilde{g}_i(\mathbf{w}_{t_k^{i_t}}) - g_{i_t}(\mathbf{w}_{t_k^{i_t}})\|^2] \\
={} & \mathrm{E}[\|(1-\beta)([U_{t_{k-1}^{i_t}+1}]_{i_t} - g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}})) + \beta(\tilde{g}_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}}) - g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}})) + g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}}) - g_{i_t}(\mathbf{w}_{t_k^{i_t}})\|^2] \\
\leq{} & (1+\beta)(1-\beta)^2 \mathrm{E}[\|[U_{t_{k-1}^{i_t}+1}]_{i_t} - g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}})\|^2] + 2(1+1/\beta)\beta^2 \cdot 2G^2 \\
& + 2(1+1/\beta)\mathrm{E}[\|g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}}) - g_{i_t}(\mathbf{w}_{t_k^{i_t}})\|^2] \\
\leq{} & (1-\beta)\mathrm{E}[\|[U_{t_{k-1}^{i_t}+1}]_{i_t} - g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}})\|^2] + 8\beta G^2 + \frac{4C_g^2}{\beta}\mathrm{E}[\|\mathbf{w}_{t_{k-1}^{i_t}} - \mathbf{w}_{t_k^{i_t}}\|^2] \\
\leq{} & (1-\beta)\mathrm{E}[\|[U_{t_{k-1}^{i_t}+1}]_{i_t} - g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}})\|^2] + 8\beta G^2 + \frac{4C_g^2\eta^2}{\beta}\mathrm{E}\left[\left\|\sum_{j=t_{k-1}^{i_t}+1}^{t_k^{i_t}} \mathbf{m}_t\right\|^2\right] \\
\leq{} & (1-\beta)\mathrm{E}[\|[U_{t_{k-1}^{i_t}+1}]_{i_t} - g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}})\|^2] + 8\beta G^2 + \frac{4C_g^2\eta^2}{\beta}\mathrm{E}\left[(t_k^{i_t} - t_{k-1}^{i_t})\sum_{j=t_{k-1}^{i_t}+1}^{t_k^{i_t}} \|\mathbf{m}_j\|^2\right]
\end{aligned}
\tag{26}
$$

where the second and last inequalities are based on Young's inequality. Note that, based on Lemma 1 of Qi et al. (2021), $t_k^{i_t} - t_{k-1}^{i_t}$ is a random variable with conditional distribution given by a geometric distribution with $p = 1/m$, i.e., $\mathrm{E}[t_k^{i_t} - t_{k-1}^{i_t}|t_{k-1}^{i_t}] \leq m$. However, since $t_k^{i_t} - t_{k-1}^{i_t}$ and $\sum_{j=t_{k-1}^{i_t}+1}^{t_k^{i_t}} \|\mathbf{m}_j\|^2$ are *dependent*, this conclusion can not be applied to bound the last term, which further making the advantages of the momentum can not be used. Because of this issue, we have to rewrite (26) and bound the last term as

$$
\mathrm{E}\left[\left\|\sum_{j=t_{k-1}^{i_t}+1}^{t_k^{i_t}} \mathbf{m}_t\right\|^2\right] \leq C_F^2 \mathrm{E}[(t_k^{i_t} - t_{k-1}^{i_t})^2] \leq 2m^2 C_F^2.
\tag{27}
$$

Thus

$$
\mathrm{E}[\|[U_{t_k^{i_t}+1}]_{i_t} - g_{i_t}(\mathbf{w}_{t_k^{i_t}})\|^2] \leq (1-\beta)\mathrm{E}[\|[U_{t_{k-1}^{i_t}}]_{i_t} - g_{i_t}(\mathbf{w}_{t_{k-1}^{i_t}})\|^2] + 8\beta G^2 + \frac{8m^2 C_F^2 C_g^2 \eta^2}{\beta}.
$$

Summing over for all intervals, we have

$$
\sum_{i=1}^{m}\mathrm{E}[\sum_{k=1}^{K_i} \|[U_{t_{k-1}^i}]_i - g_i(\mathbf{w}_{t_{k-1}^i})\|^2] \leq \frac{mG^2}{\beta} + 8G^2 T + \frac{8m^2 C_F^2 C_g^2 \eta^2 T}{\beta^2}
$$

where K_i denotes that the i-th positive data is chosen for K_i times. Combining with (25), we get

$$
\mathrm{E}[\sum_{i=1}^{T} \|\Delta_i\|^2] \leq \frac{2L_f^2 \eta^2 C_F^2 T}{\beta^2} + \frac{40 C_g^4 L_f^2 m^2 C_F^2 \eta^2 T}{\beta} + \frac{C_F^2}{\beta} + 5C_g^2 L_f^2 mG^2 + \beta T(40 C_G^2 L_f^2 G^2 + 8C_g^2 C_f^2).
\tag{28}
$$

With Lemma 3, we obtain

$$\mathrm{E}\left[\frac{1}{T}\sum_{i=1}^{T}\|\nabla F(\mathbf{w})\|^2\right] \leq \frac{2(F(\mathbf{w}_1)-F(\mathbf{w}_{T+1}))}{\eta T} + \frac{2L_f^2\eta^2 C_F^2}{\beta^2} + \frac{40C_g^4 L_f^2 m^2 C_F^2 \eta^2}{\beta}$$
$$+ \frac{C_F^2}{\beta T} + \frac{5C_g^2 L_f^2 m G^2}{T} + \beta(40C_G^2 L_f^2 G^2 + 8C_g^2 C_f^2). \tag{29}$$

Finally, we can obtain a ϵ-stationary solution by setting $\beta = O(\epsilon^2)$, $\eta = O(\epsilon^3/m)$, and

$$T \geq \left\{\frac{2(F(\mathbf{w}_1)-F(\mathbf{w}_{T+1}))}{\eta\epsilon^2}, \frac{C_F^2}{\beta\epsilon^2}, \frac{5C_g^2 L_f^2 m G^2}{\epsilon^2}\right\}.$$

D PROOF OF THEOREM 3

Let $\Delta_t = \mathbf{m}_{t+1} - \nabla F(\mathbf{w}_t)$. According to the update rule in Algorithm 2, and we have

$$\|\Delta_t\|^2 = \left\|(1-\beta_1)\mathbf{m}_t + \beta_1 \frac{1}{B}\sum_{\mathbf{x}_i\in\mathcal{B}_t}\nabla\tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_i^\top\right) - \nabla F(\mathbf{w}_t)\right\|^2$$

$$= \left\|\underbrace{(1-\beta_1)[\mathbf{m}_t - \nabla F(\mathbf{w}_{t-1})]}_{A_1} + \underbrace{(1-\beta_1)[\nabla F(\mathbf{w}_{t-1}) - \nabla F(\mathbf{w}_t)]}_{A_2}\right.$$

$$+ \beta_1\underbrace{\left[\frac{1}{B}\sum_{\mathbf{x}_i\in\mathcal{B}_t}\nabla\tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_i^\top\right) - \frac{1}{B}\sum_{\mathbf{x}_i\in\mathcal{B}_t}\nabla\tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left(g_i(\mathbf{w}_t)\right)\right]}_{A_3} \tag{30}$$

$$\left.+ \beta_1\underbrace{\left[\frac{1}{B}\sum_{\mathbf{x}_i\in\mathcal{B}_t}\nabla\tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left(g_i(\mathbf{w}_t)\right) - \nabla F(\mathbf{w}_t)\right]}_{A_4}\right\|^2.$$

Following similar procedure as in the proof of Theorem 1, we have

$$\mathrm{E}_t[\|\Delta_t\|^2] \leq (1-\beta_1)\|\Delta_{t-1}\|^2 + \frac{2(1+\beta_1)}{\beta_1}\|A_2\|^2 + \frac{2+3\beta_1}{\beta_1}\mathrm{E}_t[\|A_3\|^2] + 2\mathrm{E}_t[\|A_4\|^2], \tag{31}$$

where

$$\|A_2\|^2 = (1-\beta_1)^2\|\nabla F(\mathbf{w}_{t-1}) - \nabla F(\mathbf{w}_t)\|^2 \leq (1-\beta_1)^2 L_F^2\|\mathbf{w}_t - \mathbf{w}_{t-1}\|^2 \tag{32}$$

$$\|A_3\|^2 \leq \beta_1^2\frac{1}{B}\sum_{\mathbf{x}_i\in\mathcal{B}_t}\|\nabla\tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_i^\top\right) - \nabla\tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left(g_i(\mathbf{w}_t)\right)\|^2$$

$$\leq \beta_1^2\frac{C_g^2}{B}\sum_{\mathbf{x}_i\in\mathcal{B}_t}\|\nabla f([U_{t+1}]_i^\top) - \nabla f(g_i(\mathbf{w}_t))\|^2 \tag{33}$$

$$\leq \beta_1^2\frac{C_g^2 L_f^2}{B}\sum_{\mathbf{x}_i\in\mathcal{B}_t}\|[U_{t+1}]_i^\top - g_i(\mathbf{w}_t)\|^2$$

and

$$\mathrm{E}_t[\|A_4\|^2] \leq \beta_1^2\mathrm{E}_t\left[\left\|\frac{1}{B}\sum_{\mathbf{x}_i\in\mathcal{B}_t}\nabla\tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f([U_{t+1}]_i^\top)\right\|^2\right] \leq \beta_1^2 C_g^2 C_f^2. \tag{34}$$

Combining (31), (32), (33), (34), and taking expectation over the randomness of $\mathcal{B}_t$, we get

$$\mathrm{E}_t[\|\Delta_t\|^2] \leq (1-\beta_1)\|\Delta_{t-1}\|^2 + \frac{2L_F^2\|\mathbf{w}_t - \mathbf{w}_{t-1}\|^2}{\beta_1} + \frac{5\beta_1 C_g^2 L_f^2\mathrm{E}_t[\|U_{t+1} - g(\mathbf{w}_t)\|^2]}{m}$$
$$+ 2\beta_1^2 C_g^2 C_f^2. \tag{35}$$

Summing over from 1 to T and taking the expectation with respect to all the randomness, we get

$$
\begin{aligned}
\mathrm{E}\left[\sum_{i=1}^{T}\|\Delta_i\|^2\right] &\leq \frac{\|\Delta_0\|^2}{\beta_1} + \frac{2L_F^2 \sum_{i=1}^{T}\|\mathbf{w}_i - \mathbf{w}_{i-1}\|^2}{\beta_1^2} + \frac{5C_g^2 L_f^2 \sum_{i=1}^{T} \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2]}{m} \\
&\quad + 2\beta_1 C_g^2 C_f^2 T \\
&= \frac{C_F^2}{\beta_1} + \frac{2L_F^2 \eta^2 \sum_{i=1}^{T}\|\mathbf{m}_i\|^2}{\beta^2} + \frac{5C_g^2 L_f^2 \sum_{i=1}^{T} \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2]}{m} \\
&\quad + 2\beta_1 C_g^2 C_f^2 T.
\end{aligned} \tag{36}
$$

Next, note that $U_{t+1} = \Pi_{\Omega^m}[(1 - \beta_0)U_t + \beta_0 \widehat{g}(\mathbf{w}_t)]$, where $\widehat{g}(\mathbf{w}_t)$ is an unbiased estimator of $g(\mathbf{w}_t)$, which is defined in (7). Thus, with Lemma 2 and the fact that

$$
\mathrm{E}[\|\widehat{g}(\mathbf{w}_t) - g(\mathbf{w}_t)\|^2] \leq \mathrm{E}[\|\widehat{g}(\mathbf{w}_t)\|^2] = \mathrm{E}\left[\sum_{\mathbf{x}_i \in \mathcal{B}_t}\left\|\frac{m}{B}\tilde{g}_i(\mathbf{w})\right\|^2\right] \leq \frac{m^2}{B}G^2.
$$

we have

$$
\sum_{i=1}^{T} \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2] \leq \frac{mG^2}{\beta_0} + 2\beta_0 T \frac{m^2}{B}G^2 + \frac{2mC_g^2 \eta^2 \sum_{i=1}^{T}\|\mathbf{m}_i\|^2}{\beta_0^2}. \tag{37}
$$

Combining all the about results together, and set $\beta_0 = \beta_1 = \beta$, we have

$$
\begin{aligned}
\mathrm{E}\left[\sum_{i=1}^{T}\|\Delta_i\|^2\right] &\leq \frac{C_F^2 + 5C_g^2 L_f^2 G^2}{\beta} + \frac{(2L_F^2 + 10C_g^4 L_f^2)\eta^2}{\beta^2}\sum_{i=1}^{T}\|\mathbf{m}_i\|^2 \\
&\quad + \beta(10C_g^2 L_f^2 G^2 \frac{m}{B} + 2C_g^2 C_f^2)T.
\end{aligned} \tag{38}
$$

Based on Lemma 3, with $(2L_F^2 + 10C_g^4 L_f^2)\eta^2/\beta^2 \leq 1/2$, we get

$$
\begin{aligned}
\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\nabla F(\mathbf{w}_t)\|^2\right] &\leq \frac{2(F(\mathbf{w}_1) - F(\mathbf{w}_T))}{\eta T} + \frac{C_F^2 + 5C_g^2 L_f^2 G^2}{\beta T} \\
&\quad + \beta(5C_g^2 L_f^2 G^2 \frac{m}{B} + 2C_g^2 C_f^2).
\end{aligned} \tag{39}
$$

Thus, by setting $\beta = O(\epsilon^2 B/m)$ and $\eta = O(\beta)$, we have $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\nabla F_{\mathbf{w}}(\mathbf{w}_t)\|^2\right] = O(\epsilon^2)$ for

$$
T \geq \max\left\{\frac{2(F(\mathbf{w}_1) - F(\mathbf{w}_{T+1}))}{\eta \epsilon^2}, \frac{C_F^2 + 5C_g^2 L_f^2 G^2}{\beta \epsilon^2}\right\}.
$$

E   PROOF OF THEOREM 4

Let $\Delta_t = \mathbf{m}_{t+1} - \nabla F(\mathbf{w}_t)$, and $\eta_t^s = 1/(\sqrt{\mathbf{v}_{t+1}} + \delta)$. Based on the boundedness of $\|tilde\nabla\|$, we know that $\eta c_l \leq [\eta_t^s]_i \leq \eta c_u$ for all $i \in [d]$. According to the update rule in Algorithm 3, we have

$$
\|\Delta_t\|^2 = \left\|(1 - \beta_1)\mathbf{m}_t + \beta_1 \frac{1}{B}\sum_{\mathbf{x}_i \in \mathcal{B}_t}\nabla \tilde{g}_i(\mathbf{w}_t)^\top \cdot \nabla f\left([U_{t+1}]_i^\top\right) - \nabla F(\mathbf{w}_t)\right\|^2 \tag{40}
$$

Following similar procedure as in Appendix D, we can get

$$
\begin{aligned}
\mathrm{E}\left[\sum_{i=1}^{T}\|\Delta_i\|^2\right] &\leq \frac{C_F^2 + 5C_g^2 L_f^2 G^2}{\beta} + \frac{(2L_F^2 + 10C_g^4 L_f^2)\eta^2 c_u^2}{\beta^2}\sum_{i=1}^{T}\|\mathbf{m}_i\|^2 \\
&\quad + \beta(10C_g^2 L_f^2 G^2 \frac{m}{B} + 2C_g^2 C_f^2)T.
\end{aligned} \tag{41}
$$

Next, we introduce the following lemma, which is the counterpart of Lemma 3 for Adam-style algorithms Guo et al. (2021b).

Table 5: Final averaged AP scores on the testing data.

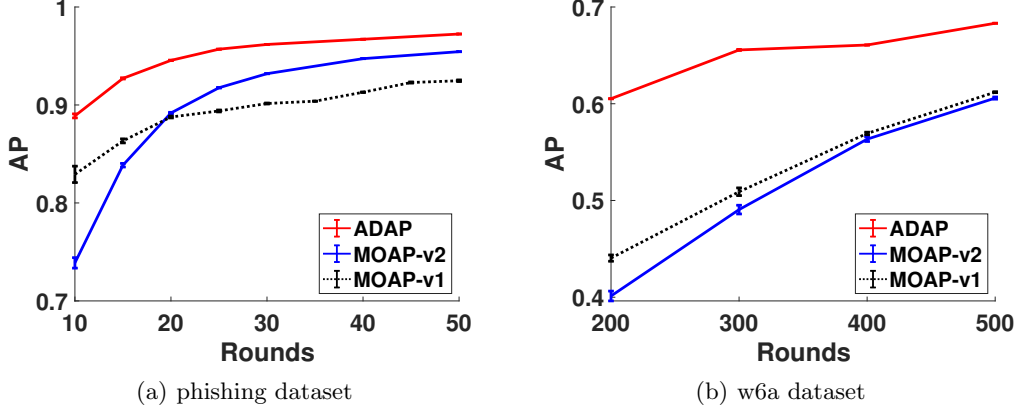| Method	phishing	w6a
ADAP	$0.981 \pm$ 2E-7	$0.675 \pm$ 1E-4
MOAP-V2	$0.978 \pm$ 2E-6	$0.596 \pm$ 2E-3
MOAP-V1	$0.972 \pm$ 4E-6	$0.608 \pm$ 4E-4

(a) phishing dataset   (b) w6a dataset

Figure 3: AP vs # of rounds on the training set

Lemma 4. *For $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t^s \cdot \mathbf{m}_{t+1}$ with $\eta c_l \leq [\eta_t]_i \leq \eta c_u$ and $\eta L_F \leq c_l/(2c_u^2)$, we have*

$$F(\mathbf{w}_{t+1}) \leq F(\mathbf{w}_t) + \frac{\eta c_u}{2}\|\nabla F(\mathbf{w}_t) - \mathbf{m}_{t+1}\|^2 - \frac{\eta c_l}{2}\|\nabla F(\mathbf{w}_t)\|^2 - \frac{\eta c_l}{4}\|\mathbf{m}_{t+1}\|^2.$$

Combining (41) and Lemma 4, and with $(2L_F^2 + 10C_g^4 L_f^2)\eta^2 c_u^3/(c_l\beta^2) \leq 1/2$, we get we get

$$\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^T \|\nabla F(\mathbf{w}_t)\|^2\right] \leq \frac{2(F(\mathbf{w}_1) - F(\mathbf{w}_{T+1}))/c_l}{\eta T} + \frac{c_u(C_F^2 + 5C_g^2 L_f^2 G^2)/c_l}{\beta T} \tag{42}$$
$$+ \beta c_u(5C_g^2 L_f^2 G^2 \frac{m}{B} + 2C_g^2 C_f^2)/c_l.$$

Thus, by setting $\beta = O(\epsilon^2 B/m)$ and $\eta = O(\beta)$, we have $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^T \|\nabla F_\mathbf{w}(\mathbf{w}_t)\|^2\right] = O(\epsilon^2)$ for

$$T \geq \max\left\{ \frac{2(F(\mathbf{w}_1) - F(\mathbf{w}_{T+1}))}{c_l\eta\epsilon^2}, \frac{c_u(C_F^2 + 5C_g^2 L_f^2 G^2)}{c_l\beta\epsilon^2} \right\}.$$

F COMPARISONS BETWEEN MOAP-V1 AND MOAP-V2

In this section, we first discuss the differences between the analysing techniques of MOAP-V1 and MOAP-V2, and then compare their empirical performances on real-world datasets.

Comparison of proofs We note that, when $B = m$, that is, when $\mathcal{B}_t$ includes all positive data, the proofs of the two algorithms are the same. For $B < m$, the major difference lies in the techniques used for bounding $\sum_{i=1}^T \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2]$, i.e., the variance term of $g(\mathbf{w})$ estimation. In MOAP-V2, U_{t+1} is the moving average of $\widehat{g}(\mathbf{w}_t)$ (defined in (7)), which is an unbiased estimator of $g(\mathbf{w}_t)$. Thus, we can directly use the variance recursion property (Lemma 2) on U_{t+1} to bound $\sum_{i=1}^T \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2]$ (Eq. (37)), and then further combine the property of momentum step (Lemma 3) to obtain a tight bound. In contrast, the U_{t+1} in MOAP-V1 is only

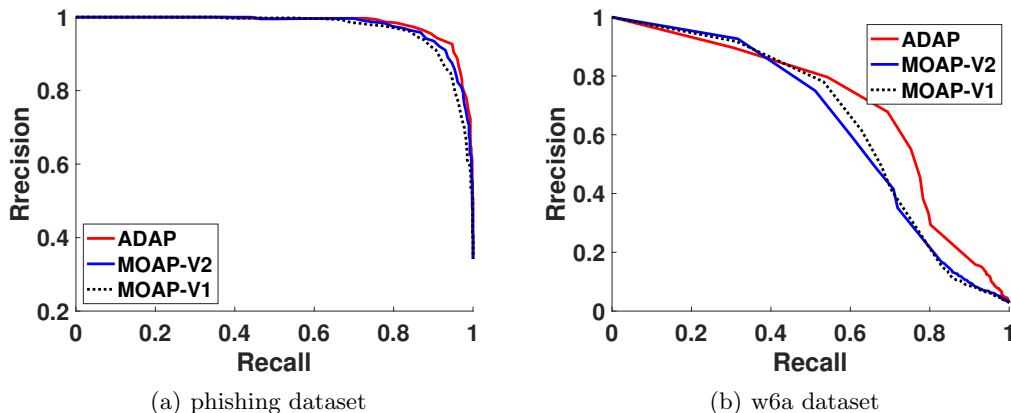(a) phishing dataset   (b) w6a dataset

Figure 4: Precision-Recall curves of the Final models on the testing set

updated for the sampled data, and thus tracking a biased estimator of $g(\mathbf{w}_t)$. Because of this problem, Lemma 2 can not be directly applied on U_{t+1}, and we have to bound $\sum_{i=1}^{T} \mathrm{E}_i[\|U_{i+1} - g(\mathbf{w}_i)\|^2]$ coordinate-wisely as in (26), using the fact that $\tilde{g}_i(\mathbf{w}_t)$ is an biased estimator of $g_i(\mathbf{w}_t)$. However, the dependent issue makes the advantages of the momentum step in Lemma 3 can not be fully exploited, which leads to a worse convergence rate. Whether we can overcome the dependant problem and directly improve the convergence rate of MOAP-V1 is still an open question.

Experiments Next, we compare the performances of MOAP-V1 and MOAP-V2 on two real-world datasets, that is, *w6a* and *phishing*. The parameter configurations are the same as the experiments in Section 4. The convergence curves of AP on training examples are reported in Figure 3, and the final AP scores on the testing data are shown in Table 3. We also plot the Precision-Recall curves of the final models on testing data in Figure 4. From these results, it can be seen that, although MOAP-V1 currently suffers worse theoretical guarantees, it achieves comparable empirical results as MOAP-V2.