Fast Stochastic Recursive Momentum Methods for Imbalanced Data Mining

Xidong Wu[1], Feihu Huang[1,2], Heng Huang[1]
[1] Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA, United States
[2]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
mr.xidong.wu@gmail.com, huangfeihu2018@gmail.com, heng.huang@pitt.edu

***Abstract*—Standard deep learning models have been mainly designed for balanced data mining tasks and use accuracy to evaluate the classifier. However, in many real-world applications, the distribution of data is skewed. If the standard models, which are designed to optimize the accuracy, are applied to the imbalanced data, the prediction performance could be poor because the model bias towards the majority class. To address the imbalanced data mining problem, areas under precision-recall curves (AUPRC) was proposed as a good measure to evaluate the performance of prediction models on imbalanced data sets, and shows excellent capability in identifying the models with high predictive power. To improve the performance of models, researchers recently design methods to directly optimize AUPRC for imbalanced data mining. However, these approaches suffer from a high iteration complexity and efficient methods are desired. In this paper, we propose a faster stochastic method (*i.e.*, ROAP) for maximizing the AURPC based on the momentum-based variance reduced technique. Our new method is based on the maximization of non-parametric averaged precision (AP), which is a popular unbiased point estimator of AUPRC, and the optimization objective in this paper can be converted into a sum of dependent compositional functions, where the inner functions rely on random variables of both inner and outer levels. Compared to previous methods, our ROAP algorithm can achieve a lower iteration complexity of $O\left(\epsilon^{-3}\right)$ for finding an ϵ-stationary solution. Furthermore, we extend our method to an adaptive version (*i.e.*, AROAP) with the same iteration complexity of $O\left(\epsilon^{-3}\right)$. To the best of our knowledge, this paper is the first work showing that the variance reduction method can be incorporated into maximizing the AURPC for efficient data mining on imbalanced datasets. Finally, we conduct extensive experiments on various imbalanced data sets with different models to demonstrate the efficiency of our new algorithms.**

***Index Terms*—AUPRC, imbalanced classification tasks, stochastic compositional optimization**

I. Introduction

Deep Neural Networks (DNN) have achieved remarkable success in various real-world applications, such as computer vision [1], [2] and speech recognition [3]. Although there are many ground-breaking studies to propose different optimization methods [4], [5] to improve the performance of DNN in data mining, most works focus on balanced data classification tasks. The objective function optimized in most of these studies is the error rate. However, in many practical applications, such as activity recognition [6] and healthcare applications [7], [8], the distribution of data is skewed. As an example, for most diseases, the number of patients is far lower than the number of healthy people. In these cases, maximizing training data accuracy does not necessarily lead to an optimal model in the imbalanced data mining as shown in [9]. Instead, areas under the curve (AUC), including ROC (AUROC) and precision-recall curves (AUPRC) attracts wide attention to be used as the metrics to measure the quality of a classification models.

For AUROC, recent works have achieved remarkable progress in maximizing AUROC. Yuan et al. [10] proposed a new deep AUROC maximization method with provable convergence. However, AUROC is not suitable for data with a much larger number of negative examples than positive examples, and AUPRC can address this issue because it doesn't use true negatives. For AUPRC, although there is also a long list of studies [11] which illustrate the superiority of AUPRC for evaluating imbalanced classifiers, how to utilize AUPRC to facilitate algorithm design has not yet been studied thoroughly. More specifically, existing imbalanced data mining algorithms via directly maximizing AUPRC are still computationally expensive and have high iteration complexity. Therefore, it is imperative to develop efficient stochastic optimization algorithms for the maximization of AUPRC in deep learning to meet the challenge of data mining on large-scale high-dimensional imbalanced data sets.

Compared with maximizing AUROC, it is a more challenging question to optimize the AUPRC. The first difficulty lies in the complicated integral definition. Some point estimators of AUPRC have been proposed to address the continuous integral issue [12], [13]. Existing methods include the non-parametric average precision (AP) estimator, parametric binomial estimator, trapezoidal estimators, and interpolation estimators of empirical curves [14]. Among them, non-parametric average precision (AP) is one of the most commonly used estimators. AP can be directly calculated based on the prediction scores of samples and does not suffer from sampling bias at the infinite sample limit. These merits make it very suitable to be applied to optimization problems. Therefore, in this work, we use AP as an optimization objective for AUPRC maximization. For a finite training set $\mathcal{D}$ consists of n number of samples $(\mathbf{x}_i, \mathbf{y}_i)$,

This work was partially supported by NSF IIS 1838627, 1837956, 1956002, 2211492, CNS 2213701, CCF 2217003, DBI 2225775. F.H. Huang was postdoctoral researcher at the University of Pittsburgh.

AP is given by [14]

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

where m denotes the number of positive data points $(\mathbf{x}_i, \mathbf{y}_i = 1)$, r^{+} denotes the point's prediction score rank among all positive data points (*i.e.*, the number of positive data points with no less prediction score than that of x_i including itself) and r denotes its prediction score rank among all data points (*i.e.*, the number of data points with no less prediction score than that of x_i including itself). It can be shown that AP is an unbiased estimator in the limit $n \to \infty$ [14].

The second difficulty of AUPRC maximization is due to the non-differential ranking functions and the non-linear relationship between the AP and sample scores. With the original definition of AP, traditional gradient-based gradient descent methods cannot be directly applied. Most existing works for maximizing AP-like functions are heuristic-driven without provable convergence [15]–[17]. Recently, Qi et al. [18] use a surrogate loss in place of the indicator function in the AP function and shows a provable convergence guarantee for a stochastic method. A critical technique in [18] is to create and update biased estimators of the surrogate ranking functions for each positive data and SOAP reaches the iteration complexity of $O\left(\epsilon^{-5}\right)$. More recently, Wang et al. [19] proposed MOAP and ADAP methods to improve the iteration complexity to $O\left(\epsilon^{-4}\right)$. The biased estimators for tracking individual ranking scores are updated in a randomized coordinate-wise manner and the momentum update is used on both the top and the inner estimator.

Despite the recent advances, it is still not clear whether there exists a provably more efficient way to maximize AUPRC with an imbalanced dataset. We note that in the conventional optimization problem, variance reduction techniques have been shown to reduce the complexity of gradient descent methods, and can greatly reduce the computational cost. Consequently, a natural question is whether one can leverage variance reduction to achieve a faster convergence rate in AUPRC maximization. In this paper, we give an affirmative answer to the above question by proposing new stochastic algorithms for imbalanced data mining by AUPRC maximization. We summarize the main contributions of this paper as follows:

- We propose faster stochastic methods (*i.e.*, ROAP and AROAP which extends ROAP with adaptive step size) based on the momentum-based variance reduced technique to improve the convergence for maximizing AP in the finite-sum stochastic compositional optimization. The methods reach iteration complexity of $O\left(\epsilon^{-3}\right)$. The key idea is to track the stochastic gradient estimator of the objective function, individual ranking scores, and its gradients, respectively.
- We conduct extensive experiments on various datasets compared with previous stochastic algorithms for AUPRC/AP optimization as the benchmark to verify the effectiveness of our methods.

II. Related Work

A. AUROC Optimization

Some studies have investigated the imbalanced data mining with the AUC metric. These studies [10], [20] show the importance of large-scale stochastic optimization algorithms with AUC metric and the necessity of accurate surrogate function. Earlier works about AUROC [7], [16] focused on linear models with pairwise surrogate losses. The issue of these works is their high overheads, which have at most quadratic dependence on the training data size. To mitigate this issue, online and stochastic optimization methods have been proposed [11], [21] for the non-convex strongly-concave min-max optimization and federated learning problems. Recently, Yuan et al. [10] proposed a new deep AUROC maximization method for large-scale medical image classification tasks, e.g., breast cancer screening classification based on the mammogram and microscopic image classification for tumor tissue.

However, an algorithm that maximizes AUROC does not always maximize AUPRC [8]. Furthermore, AUROC is incapable of accurately valuing the performance of models based on data that contains far more negative examples than positive examples. Many real-world data sets fit this case. For example, there are many more healthy people than diseased people ("positive for sickness."). In these cases, a big improvement in the number of false positives only leads to a small change in the false positive rate. Assume model 1 has a lower number of false positives than model 2 (*i.e.* model 1 is better). If the data contains numerous true negatives, the false positive rates of models 1 and 2 will be similar, and their AUROCs may not be that different. Thus, it may be difficult to use AUROC to distinguish the performance of models, which necessitates the development of efficient algorithms for deep learning (DL) by maximizing AUPRC.

B. AUPRC Optimization

Previous work about AUPRC optimization mainly uses traditional optimization techniques, such as hill-climbing search [22], cutting-plane method [23], [24], dynamic programming [25] as well as acceleration techniques in the framework of SVM [26]. They do not have good performance when the dataset is large in DL. In addition, many studies in information retrieval [15] and computer vision [12] focus on algorithm design to maximize the AP score and how to approximate the gradient of the AP function. However, these works provide no convergence guarantee for stochastic optimization and are sensitive to the mini-batch size when applied to DL.

Recently, Qi et al. [18] first proposed a method to directly optimize a surrogate function of AP and provides a theoretical convergence guarantee for the proposed stochastic algorithm with iteration complexity of $O\left(\epsilon^{-5}\right)$. Wang et al. [19] develops new stochastic momentum methods. The momentum update is used on the stochastic gradient estimator of the objective and the biased estimators for tracking positive data point ranking scores are updated in a randomized coordinate-wise manner. It has a better iteration complexity of $O\left(\epsilon^{-4}\right)$ for finding an ϵ-stationary solution.

AUPRC is used for binary classification tasks and it is easy to extend this metric to a multi-class setting. For multiple classes tasks, we can suppose that the work is made up of a variety of binary classification tasks and calculate the average precision for each class individually, namely Class A vs. Not Class A, Class B vs. Not Class B, Class C vs. Not Class C, and so on.

C. Stochastic Compositional Optimization

The optimization objective we study resembles a two-level stochastic compositional optimization problem (SCO). It has been studied extensively in the literature [27]–[33]. In these works, the objective function is of the form $\mathrm{E}_{\xi}[f(\mathrm{E}_{\zeta}[g(\mathbf{w};\zeta)];\xi)]$, where ξ and ζ are independent. However, in our work, the inner function in the formulated compositional function depends on both the random variable of the inner level and that of the outer level. This key difference will make the algorithm design and the theoretical analysis more complicated.

D. Variance-Reduction Methods

Stochastic gradient descent has a slow convergence rate due to the inherent variance. Many variance-reduction techniques are proposed to reduce the complexity of finding a stationary point, such as SVRG [34], SAGA [35] and SARAH [36]. Recently, SPIDER [37] and SpiderBoost [38] are proposed to have better converge rate. However, these variance reduction-based methods require a very large mini-batch size at checkpoints, which has a detrimental influence in the large-scale data mining. To address this problem, [39] proposed a new technique called STORM that integrates momentum and the recursive variance reduction technique for solving stochastic smooth non-convex optimization. Most importantly, this algorithm does not need a large mini-batch size and also enjoys the same converge rate for finding an ε-stationary point.

TABLE I: Complexity comparison of the representative methods for finding an ϵ-stationary point of AP maximizition

Methods	Provable Convergence	Complexity
SOAP [18]	Yes	$O\left(1/\epsilon^5\right)$
SOAP (AMSGrad-style) [18]	Yes	$O\left(1/\epsilon^5\right)$
SOAP (Adam-style) [18]	No	-
MOAP [19]	Yes	$O\left(1/\epsilon^4\right)$
ADAP [19]	Yes	$O\left(1/\epsilon^4\right)$
ROAP	Yes	$O\left(1/\epsilon^3\right)$
AROAP	Yes	$O\left(1/\epsilon^3\right)$

III. The Proposed Method

Notation: In this work, we consider binary classification tasks. $(\mathbf{x},\mathbf{y})$ denotes data points, where $\mathbf{x}\in\mathbb{R}^d, y\in\{1,-1\}$. Let $h_w(x)$ denote the score function of a classifier with a parameter vector $\mathbf{w}\in\mathbb{R}^d$. Let $\mathcal{D}_+=\{(\mathbf{x_1},y_1),\cdots,(\mathbf{x_m},y_m)\}$ denotes the set of all positive training data points and $\mathcal{D}=\{(\mathbf{x_1},y_1),\cdots,(\mathbf{x_n},y_n)\}$ denotes the set of all training data points. Let $m=|\mathcal{D}_+|$ denote the number of positive data points and $n=|\mathcal{D}|$ denote the number of data points. $||x||$ denotes the Euclidean norm of vector x. Let $\Pi_\Omega[u]=argmin_{v\in\Omega}||u-v||^2$ be the Euclidean projection onto a convex set Ω. In addition, $\tilde{g}_i(\mathbf{w})$ and $\nabla\tilde{g}_i(\mathbf{w})$ denote independent unbiased stochastic estimators of $g_i(\mathbf{w})$ and $\nabla g_i(\mathbf{w})$, respectively. Specifically, we first sample two sets of data points from $\mathcal{D}$ denoted by $\mathcal{S}_1$ and $\mathcal{S}_2$. Then we calculate $\tilde{g}_i(\mathbf{w})=\frac{n}{|\mathcal{S}_1|}\sum_{\mathbf{x}_j\in\mathcal{S}_1}g(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)$ and $\nabla\tilde{g}_i(\mathbf{w})=\frac{n}{|\mathcal{S}_2|}\sum_{\mathbf{x}_j\in\mathcal{S}_2}\nabla g(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)$, respectively.

A. Preliminaries

As discussed above, we consider using AP to approximate AUPRC. For a finite set of data set $\mathcal{D}=\{(\mathbf{x}_i,y_i),i=1,\ldots,n\}$, it is given by

$$\widehat{\mathrm{AP}}=\frac{1}{m}\sum_{\mathbf{x}_i\in\mathcal{D}_+}\frac{\sum_{j=1}^n\mathbf{I}(h_w(\mathbf{x}_j)\geq h_w(\mathbf{x}_i))\cdot\mathbf{I}\left(y_j=1\right)}{\sum_{j=1}^n\mathbf{I}(h_w(\mathbf{x}_j)\geq h_w(\mathbf{x}_i))}\tag{2}$$

where $h_w(x)$ is prediction score function.

Then, to address the issue arise from the non-continuous ranking functions in both numerator and denominator in (2), the loss function $\ell(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)$ is used as a surrogate function of indicator function $\mathbf{I}(h_w(\mathbf{x}_j)\geq h_w(\mathbf{x}_i))$. In the previous work, [12], [13], various surrogate losses are considered, such as hinge loss, squared hinge loss, smoothed hinge loss, and exponential loss. In this paper, we use a squared hinge loss $\ell(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)=(max\{s-(\mathbf{h}_w(\mathbf{x}_i)-\mathbf{h}_w(\mathbf{x}_j)),0\})^2$, where s is a margin parameter. Then, our optimization objective is given by:

$$\widehat{\mathrm{AP}}=\frac{1}{m}\sum_{\mathbf{x}_i\in\mathcal{D}_+}\frac{\sum_{j=1}^n\ell(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)\,\mathbf{I}\left(y_j=1\right)}{\sum_{j=1}^n\ell(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i)}\tag{3}$$

For convenience, we define the elements in $g_i(w)$ as the surrogates of the two prediction score ranking function $\mathrm{r}^+(\mathbf{x}_i)$ and $\mathrm{r}(\mathbf{x}_i)$ respectively. Define the following equation:

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

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

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

Then, we could reformulate the optimization objective into the following minimization problem:

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

We can see that it is a finite-sum two-level dependent compositional function. It should be mentioned that it is the non-convex even if $g_i(\mathbf{w})$ is convex because the $f(\mathbf{u})$ is non-convex. Our goal in this work is to find an ϵ-stationary solution, such that $\mathbb{E}\|\nabla\Phi(\mathbf{w})\|\leq\epsilon$.

Algorithm 1 ROAP Algorithm

```
1:  Input: $T$, tuning parameters $\{\alpha,\beta,\gamma,\eta,b\}$ and initial mini-batch size $b_0$;
2:  initialize: Initialize: $\mathbf{w}_1 \in \mathbb{R}^d, U_1 \in \mathbb{R}^{m\times 2}, V_1 \in \mathbb{R}^{m\times d\times 2}$, $\mathbf{F}_1 \in \mathbb{R}^d$. Draw a mini-batch i.i.d. positive samples $\mathcal{B}_1 = \{\xi_i^1\}_{i=1}^{b_0}$, and then compute $U_1 = \nabla\widehat{g}(\mathbf{w}_t;\mathcal{B}_1)$, $V_1 = \widehat{g}(\mathbf{w}_t;\mathcal{B}_1)$ and $F_1 = \frac{1}{b_0}\sum_{\mathcal{B}_1}[[V_1]_i^\top \nabla f([U_1]_i)$
3:  for $t = 1,2,\ldots,T$ do
4:      $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t F_t$;
5:      Draw i.i.d. positive samples $\mathcal{B}_t = \{\xi_i^t\}_{i=1}^b$ from $D_+$
6:      Set $[U_{t+1}]_i$ as (7)
7:      Set $[V_{t+1}]_i$ as (8)
8:      $F_{t+1} = (1-\gamma)F_t + \frac{1}{b}\sum_{\mathcal{B}_t}[[V_{t+1}]_i^\top \nabla f([U_{t+1}]_i) - (1-\gamma)[V_t]_i \nabla f([U_t]_i)]$
9:  end for
10: Output: $w$ chosen uniformly random from $\{w_t\}_{t=1}^T$.
```

B. Stochastic Recursive Momentum Optimization of AP: ROAP

In order to design the method, we first consider how to compute the gradient of $\Phi(w)$. Let

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

[18], [19] maintain U_t sequence as the moving average estimator for estimating individual ranking scores $g(\mathbf{w})$, a common technique from the literature of stochastic compositional optimization. Following the [18], [19] and previous studies of stochastic compositional optimization, we use three estimators to track the stochastic gradient estimator of the objective function, individual ranking scores, and its gradients, respectively. Firstly, we consider the estimators to track individual ranking scores and their gradients. We also maintain a matrix $[U_t] = [U_t^1, U_t^2] \in \mathbb{R}^{m\times 2}$ with each column correspond to each positive example. We use U^1 and U^2 to track the $[g(\mathbf{w})]_1$ and $[g(\mathbf{w})]_2$, respectively. Similarly, we also maintain the $[V_t] = [V_t^1, V_t^2]$ and use V^1 and V^2 to track the $[\nabla g(\mathbf{w})]_1 \in \mathbb{R}^{m\times d}$ and $[\nabla g(\mathbf{w})]_2 \in \mathbb{R}^{m\times d}$, respectively. We use momentum-based variance reduced technique to update the $[U_t]$ and $[V_t]$ and we update all coordinates of $[U_t]$ and $[V_t]$, following the [19]. Algorithm 1 shows the algorithmic framework of the ROAP.

Take $[U_t]$ as an example. We first construct

$$\widehat{g}(\mathbf{w}_t) = \frac{1}{b}\sum_{i\in\mathcal{B}_t}\begin{pmatrix}0\\ \cdot\\ m\tilde{g}_i(\mathbf{w}_t)^\top\\ \cdot\\ 0\end{pmatrix} \tag{6}$$

When we take expectation over randomness, we get $\mathrm{E}_{\mathcal{B}_t}[\widehat{g}(\mathbf{w}_t)] = g(\mathbf{w}_t)$. It means both $\widehat{g}(\mathbf{w}_t)$ and $\tilde{g}(\mathbf{w}_t)$ are unbiased estimator of $g(\mathbf{w}_t)$. Then we write the update of U_{t+1} as $U_{t+1} = \Pi_{\Omega_1}[(1-\beta)U_t + \widehat{g}(\mathbf{w}_{t+1}) - (1-\beta)\widehat{g}(\mathbf{w}_t)]$

It is equivalent to set $[U_{t+1}]_i$ as:

$$\begin{cases}\Pi_{\Omega_1}\left[c_0\tilde{g}_i(\mathbf{w}_{t+1})^\top + (1-\beta)\left([U_t]_i - c_0\tilde{g}_i(\mathbf{w}_t)^\top\right)\right] & i\in\mathcal{B}_t\\ \Pi_{\Omega_1}\left[(1-\beta)[U_t]_i\right] & \text{o.w.}\end{cases} \tag{7}$$

where $c_0 = \frac{m}{b}$. Similarly, we set $[V_{t+1}]_i$ as:

$$\begin{cases}\Pi_{\Omega_2}[c_0\nabla\tilde{g}_i(\mathbf{w}_{t+1})^\top + (1-\alpha)([V_t]_i - c_0\nabla\tilde{g}_i(\mathbf{w}_t)^\top)] & i\in\mathcal{B}_t\\ \Pi_{\Omega_2}\left[(1-\alpha)[V_t]_i\right] & \text{o.w.}\end{cases} \tag{8}$$

Given that updating the entire table induces a heavy computational cost, we could only update the coordinate of U_{t+1} corresponding to the sampled positive data in each iteration. Updates of U_{t+1} coordinates corresponding to the unsampled positive data points are delayed until they are sampled, since in the current iteration these coordinates are not used to update the model parameter.

We can see that in Step 6 of Algorithm 1, we project the results into a convex set and clip them by a lower bound. This operation can ensure the division in computing the stochastic gradient estimator in (5) is always valid and is consistent with the assumption. For the update of $[V_t]$, we use a similar operation in Step 7 of Algorithm 1.

Finally, we maintain and update an additional stochastic gradient estimator based on the U_t and V_t in steps 8 and 9 of the ROAP algorithm. Then we could update the model parameters by

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t F_t \tag{9}$$

C. Stochastic Adaptive Algorithms for AP Maximization: AROAP

In this section, we extend ROAP into a stochastic adaptive algorithm by incorporating adaptive step sizes. In previous works, adaptive optimization methods have achieved great success in practice and various adaptive step sizes have been studied for standard stochastic optimization, such as AdaGrad [40], Adam [41] and AMSGrad [17].

$$\begin{aligned}\text{Adam-style:}\mathbf{v}_{t+1} &= (1-\lambda)\mathbf{v}_t + \lambda d_t^2\\ \text{AdaGrad-style:}\mathbf{v}_{t+1} &= \frac{1}{t+1}\sum_{i=1}^{t} d_t^2\\ \text{AMSGrad-style:}\mathbf{v}'_{t+1} &= (1-\lambda)\mathbf{v}'_t + \lambda d_t^2,\\ \mathbf{v}'_{t+1} &= \max\left(\mathbf{v}_t, \mathbf{v}'_{t+1}\right)\\ \text{AdaBound-style:}\mathbf{v}'_{t+1} &= (1-\lambda)\mathbf{v}'_t + \lambda d_t^2\\ \mathbf{v}_{t+1} &= \Pi_{[1/c_u^2, 1/c_l^2]}\left[\mathbf{v}'_{t+1}\right]\end{aligned} \tag{10}$$

Based on existing adaptive methods, we design the stochastic adaptive method for the AP maximization in AROAP algorithm. Compared with ROAP, we need to add a vector $\mathbf{v}_t \in \mathbb{R}^d$ and update $\mathbf{v}_t$ as in (10). After we have $\mathbf{v}_t$, we could update the model parameter by

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta}{\sqrt{\mathbf{v}_t} + \delta} F_{t+1} \tag{11}$$

where $\delta > 0$ is a parameter to make sure the denominator is larger than zero. Note that based on the assumptions 1, 2 and the operation in AROAP algorithm, $\|d_t\|$ are bounded [42]. According to (11), we could get that $\eta_t^s = \eta / \left(\sqrt{\mathbf{v}_t} + \delta\right)$ is upper bounded by c_u and lower bounded by c_l . It is important for convergence analysis. Algorithm 2 shows the algorithmic framework of the AROAP.

D. Convergence Analysis

Assumption 1: [18] (i) There exist $C > 0$ and $D > 0$ such that $\ell(\mathbf{w}; \mathbf{x}_i; \mathbf{x}_i) \geq C$, and $\ell(\mathbf{w}; \mathbf{x}_i; \mathbf{x}_i) \leq M$ 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.

Assumption 2: [19] There exists a positive constant σ and G, such that $\|g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)\|^2 \leq \sigma^2$, $\|\nabla g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)\|^2 \leq \sigma^2$, $\|g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) - g_{\mathbf{x}_i}(\mathbf{w})\|^2 \leq G^2$ and $\|\nabla g(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i) - \nabla g_{\mathbf{x}_i}(\mathbf{w})\|^2 \leq G^2$ for any $\mathbf{x}_i \in \mathcal{D}_+, \mathbf{x}_j \in \mathcal{D}$.

Remark 1: With a bounded smooth surrogate loss function $\ell(\mathbf{w}; \mathbf{x}_j, \mathbf{x}_i)$ and prediction score function $h_w(x)$, the above assumptions can be easily satisfied. In binary classification problems, the output of prediction score function is in $[0, 1]$.

Lemma 1: [18] Suppose Assumption 1 and 2 holds, then there exist G, C_g, L_g, $\|g_i(\mathbf{w})\|^2 \leq G^2, g_i(\mathbf{w})$ is C_g-Lipschitz and L_g-smooth, and there exist C_f, L_f, $\forall \mathbf{u} \in \Omega, f(\mathbf{u})$ is C_f-Lipschitz and L_f-smooth. There exist $C_F, L_\Phi > 0$ $\forall \mathbf{w}, \Phi(\mathbf{w})$ is C_Φ-Lipschiz and L_Φ-smooth.

Lemma 2: [43] Consider a sequence $m_{t+1} = (1-\alpha)m_t + (1-\alpha)[f(\mathbf{w}_{t+1}; \mathcal{B}_{t+1}) - f(\mathbf{w}_t; \mathcal{B}_{t+1})] + \alpha f(\mathbf{w}_{t+1}; \mathcal{B}_{t+1})$ for tracking $f(\mathbf{w}_t)$, where $\mathbb{E}[f(\mathbf{w}_{t+1}; \mathcal{B}_{t+1}] = f(\mathbf{w}_{t+1})$ and $\mathbb{E}[f(\mathbf{w}_t; \mathcal{B}_{t+1})] = f(\mathbf{w}_t)$ and $f(\mathbf{w})$ is a C-Lipchitz continuous and bounded variance σ. The batch size is b. Then we have

$$\begin{aligned} \mathbb{E}\|m_{t+1} - f(w_{t+1})\|^2 \leq (1-\alpha)^2 \mathbb{E}\|m_t - f(w_t)\|^2 \\ + \frac{2(1-\alpha)^2 C^2}{b}\|w_{t+1} - w_t\|^2 + \frac{2\alpha^2\sigma^2}{b} \end{aligned} \tag{12}$$

$$\begin{aligned} \sum_{t=1}^{T} \mathbb{E}\|m_{t+1} - f(w_{t+1})\|^2 \leq & \frac{\sigma^2}{b_0 \alpha} + \frac{2C^2 \sum_{t=1}^T \mathbb{E}\|w_{t+1} - w_t\|^2}{\alpha b} \\ & + \frac{2\alpha\sigma^2 T}{b} \end{aligned} \tag{13}$$

Theorem 1: Suppose Assumptions 1 and 2 hold. Assume that the sequence $\{w_t\}_{t=1}^T$ be generated from the algorithm 1 using ROAP algorithm. Let $\alpha = O(1/T^{2/3})$, $\beta = O(1/T^{2/3}), \gamma = O(1/T^{2/3})$, $\eta = O(1/T^{1/3})$ and initial batch-size $b_0 = O(T^{1/3})$. Then we have $T = O(\epsilon^{-3})$ to ensure that $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^T \|\nabla\Phi(\mathbf{w}_t)\|^2\right] \leq \epsilon^2$.

Theorem 2: Suppose Assumptions 1 and 2 hold. Assume that the sequence $\{w_t\}_{t=1}^T$ is generated from the algorithm 2 using AROAP algorithm. Let $\alpha = O(1/T^{2/3})$, $\beta = O(1/T^{2/3}), \gamma = O(1/T^{2/3})$, $\eta = O(1/T^{1/3})$ and initial batch-size $b_0 = O(T^{1/3})$. And use any of methods in eq. (11) to compute v_t. Then we have $T = O(\epsilon^{-3})$ to ensure that $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^T \|\nabla\Phi(\mathbf{w}_t)\|^2\right] \leq \epsilon^2$.

Algorithm 2 Our AROAP Algorithm

```
1:  **Input:** $T$, tuning parameters $\{\alpha, \beta, \gamma, \eta, b\}$ and initial mini-batch size $b_0$;
2:  **initialize:** Initialize: $\mathbf{w}_1 \in \mathbb{R}^d, U_1 \in \mathbb{R}^{m\times 2}, V_1 \in \mathbb{R}^{m\times d\times 2}$, $\mathbf{F}_1 \in \mathbb{R}^d$, $\mathbf{v}_1 \in \mathbb{R}^d$. Draw a mini-batch i.i.d. positive samples $\mathcal{B}_1 = \{\xi_i^1\}_{i=1}^{b_0}$, and then compute $U_1 = \nabla\widehat{g}(\mathbf{w}_t; \mathcal{B}_1)$, $V_1 = \widehat{g}(\mathbf{w}_t; \mathcal{B}_t)$, and $F_1 = \frac{1}{b_0}\sum_{\mathcal{B}_1}[[V_1]_i^\top \nabla f([U_1]_i)$
3:  **for** $t = 1, 2, \ldots, T$ **do**
4:      $\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta}{\sqrt{\mathbf{v}_t}+\delta} F_t$;
5:      Draw i.i.d. positive samples $\mathcal{B}_t = \{\xi_i^t\}_{i=1}^b$ from $D_+$
6:      Set $[U_{t+1}]_i$ as (7)
7:      Set $[V_{t+1}]_i$ as (8)
8:      $F_{t+1} = (1-\gamma)F_t + \frac{1}{B}\sum_{\mathcal{B}_t}[[V_{t+1}]_i^\top \nabla f([U_{t+1}]_i) - (1-\gamma)[V_t]_i \nabla f([U_t]_i)]$
9:      Update parameters $\mathbf{v}_{t+1}$ as in (10) and $d_t = \frac{1}{B}\sum_{\mathcal{B}_t}[V_{t+1}]_i^\top \nabla f([U_{t+1}]_i)$
10: **end for**
11: **Output:** $w$ chosen uniformly random from $\{w_t\}_{t=1}^T$.
```

TABLE II: Statistics of benchmark datasets used in the linear model task

Dataset	training data	testing data	Positive data Proportion
mushrooms	3170	1587	17.66%
a6a	11,220	21,341	23.99%
a7a	16,100	16,461	23.85%
w7a	24,692	25,057	2.99%
w8a	49,749	14,951	2.97 %
covtype	238,484	119,244	20.81%

IV. Experiments

In this section, we empirically evaluate proposed algorithms by conducting comprehensive experiments on imbalanced benchmark datasets. The goal of our experiments is two-fold: (1) to illustrate that directly optimizing the AUPRC would improve the model performance compared with traditional loss optimization, and (2) to show the advantage of our methods compared with four existing state-of-the-art methods in table I, including SOAP (SGD) [18], SOAP (Adam-style) [18], MOAP [19] and ADAP (Adam-style) [19]. For our algorithms, we implement ROAP and AROAP. The results illustrate that our methods can outperform prior methods for imbalanced classification tasks. All experiments are run over a machine with Intel Xeon E5-2683 CPU and 4 Nvidia Tesla P40 GPU. The code is available online [1].

A. Linear Models

We first consider experiments on the linear model to illustrate that direct optimization of AP would improve the model performance compared with the traditional optimization method based on the AUPRC metric. In addition to the methods

[1]https://github.com/Wxdlele/auprc

TABLE III: Final averaged AP scores on the testing data in the linear model task

Method	mushrooms	a6a	a7a
SOAP	0.9703	0.5695	0.6203
SOAP (Adam)	0.9980	0.5981	0.6215
MOAP	0.9606	0.6298	0.6943
ADAP	0.9971	0.6695	0.7111
ROAP	0.9845	0.6860	0.7214
AROAP	**1**	**0.6959**	**0.7319**
Method	w7a	w8a	covtype
SOAP	0.4953	0.6254	0.5076
SOAP (Adam)	0.4916	0.5628	0.5215
MOAP	0.5005	0.6032	0.5334
ADAP	0.5445	0.6563	0.5490
ROAP	0.5565	0.6686	0.5421
AROAP	**0.6092**	**0.6839**	**0.5611**

mentioned above, we also use stochastic gradient descent (sgd) to optimize the error rate. In this section, we use the sigmoid function $1/\left(\exp\left(-\mathbf{w}^\top\mathbf{x}\right)+1\right)$ as prediction scores function to compute AP. As mentioned in section III-A, it is a non-convex task because the $f(\mathbf{u})$ is non-convex.

Datasets: We conduct experiments on six imbalanced benchmark data sets from LIBSVM data [2] and the sizes of data sets range from 3,170 to 238,484. Table II summarizes the statistics of datasets. For all datasets, we scale features to [0, 1]. Because the original distributions of datasets mushrooms and covtype are balanced and only training data sets are provided, we randomly eliminate 80% of positive data points and split them into training and testing data sets with a 2:1 ratio.

Configurations: In the linear model experiments, the batch size of positive data points is 50 and the batch size of total data points is 350. The squared hinge loss is used as the surrogate function following Qi, *et al.* [18], namely, $\ell\left(\mathbf{w};\mathbf{x}_j,\mathbf{x}_i\right) = (max\{s-[\mathbf{h}_w(\mathbf{x}_i)-\mathbf{h}_w(\mathbf{x}_j)],0\})^2$, where s is a margin parameter. We tune the initial value of the step size η in the set $\{10^{-2},10^{-2.5},10^{-3}\}$, margin parameter s as 0.6, β in the set $\{0.1,0.5,0.9\}$ and c_0 in range [1,2]. For a fair comparison, we assign the same initial step size in each experiment. Furthermore, for MOAP and ADAP, we decrease the stepsize η on the order of $O(1/\sqrt{t})$ according to the theoretical analysis in [19]. For SOAP (SGD), SOAP (Adam-style), we tune decrease the stepsize η on the order of $O(1/t^{2/5})$ [18]. Similarly, we tune our methods ROAP and AROAP on the order of $O(1/t^{1/3})$. We repeat each experiment 5 times on each data and report the averaged results in table III. Moreover, we use Xavier initialization to initialize models.

Results: To present the advantage of optimization of AP, we plot the Precision-Recall curves of the final models on testing data after the same training epochs in Figure 1. Results show that our method has a significant improvement compared with SGD in terms of AUPRC. Then we illustrate the convergence curve of AP on training data sets in Figure 2. From Figure 2, we can see that our method AROAP converges faster than other methods. These results demonstrate the superiority of our methods, especially the adaptive method AROAP. Table III presents the achieved AP on the test sets by all methods.

[2]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

TABLE IV: Final averaged AP scores on the testing data Deep Neural Networks

Method	MNIST	Fashion-MNIST	Tiny-ImageNet
SOAP	0.9178	0.9258	0.5476
SOAP (Adam)	0.9738	0.9350	0.5509
MOAP	0.9299	0.9292	0.5469
ADAP	0.9828	0.9464	0.5508
ROAP	0.9777	0.9501	0.5487
AROAP	**0.9828**	**0.9510**	**0.5511**

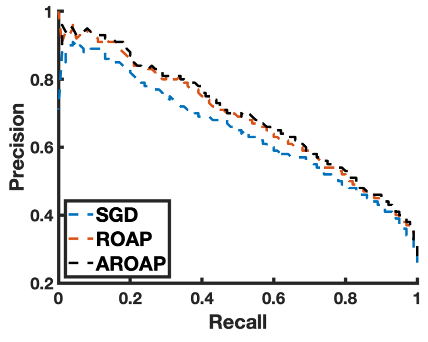
B. Deep Neural Networks

Next, we use the deep neural networks to evaluate the different methods. A lot of architectures have been proposed in DL. To illustrate the efficiency of our methods, in this part, we will use various data sets and various model structures to evaluate methods.

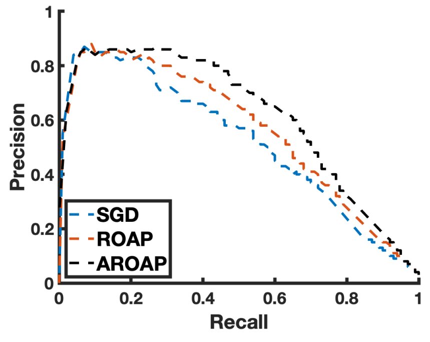
Datasets: We conduct experiments on three typical image data sets: Fashion-MNIST dataset, MNIST dataset, and Tiny-ImageNet dataset. Fashion-MNIST dataset and MNIST dataset includes $60,000$ training images and $10,000$ testing images. Each image consists of 28×28 grayscale pixel and there are total 10 classes. Tiny-ImageNet consists of 100,000 colored images (64×64) of 200 classes (500 for each class) to images. Each class has 500 training images, 50 validation images, and 50 test images. Following [18], we convert them into imbalanced binary-class versions, which are constructed as follows: Firstly, the first half of the classes (0 - 4) in the original MNIST, Fashion-MNIST, and (0 - 99) in Tiny-ImageNet datasets are designated to be the negative class, and the rest half of classes are considered to be a positive class. Then, we randomly drop 98% of the positive examples in the training set to make it imbalanced, while test sets keep unchanged.

Configurations.: For MNIST and Fashion MNIST data sets, we choose model architectures from [44] for imbalanced binary image classification tasks. For Tiny-ImageNet, we choose ResNet-18 [1] as the neural network.

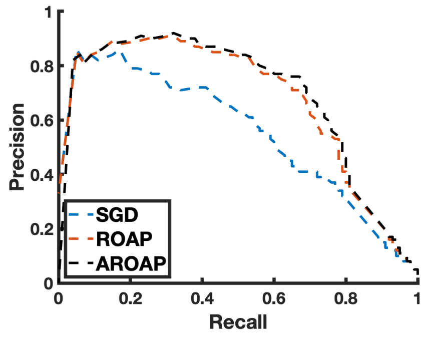
For all models, we modify the last layer output from 10 to 1 since we consider binary classification tasks. Then, in each of the experiments, models are trained by different AP maximization methods, with the same initial step size as 0.001 for a fair comparison. The margin parameter s is 0.6, β is in the set $\{0.1,0.5,0.9\}$ and c_0 is in range [1,2]. We decrease the step size following the theoretical analysis. As in the linear model experiment, for MOAP and ADAP, we decrease the stepsize η on the order of $O(1/\sqrt{t})$ according to the theoretical analysis in [19]. Similarly, we tune our methods ROAP and AROAP on the order of $O(1/t^{1/3})$. Moreover, for ROAP and AROAP, given that in the DNN models, the large model needs large storage, we set the α and γ to be 1 to reduce the storage cost. We tune the batch size of positive data points in set $\{10,100\}$. We use Xavier normal initialization to DL models. In addition, we set l_2 regulation parameter as 0.0001.



(a) a6a dataset

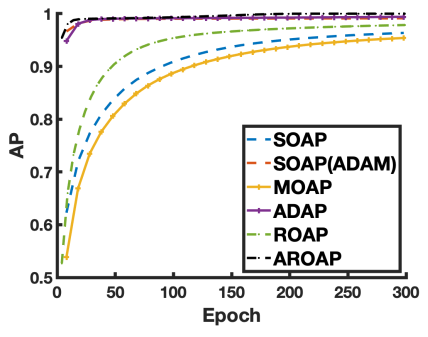


(b) w7a dataset

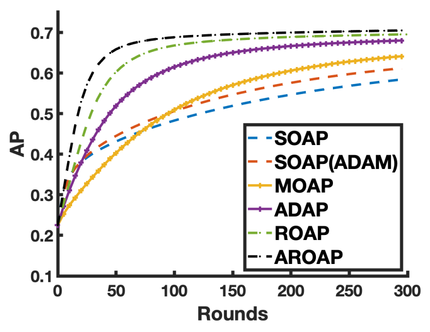


(c) w8a dataset

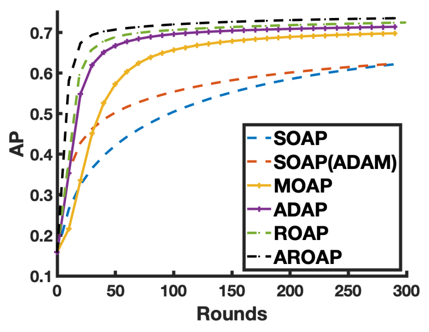
Fig. 1: Precision-Recall curves of the Final models on the testing set on the Linear Model task



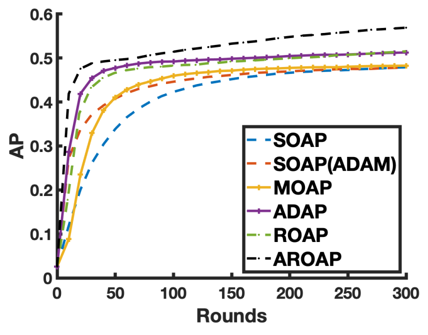
(a) mushroom dataset



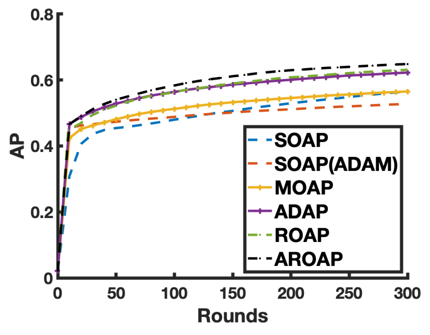
(b) a6a dataset



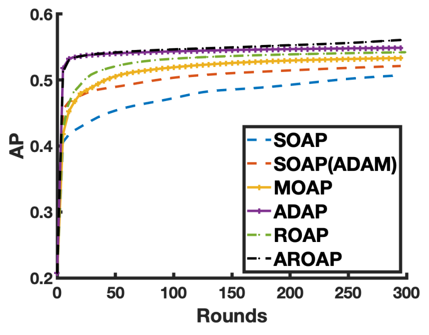
(c) a7a dataset



(d) w7a dataset

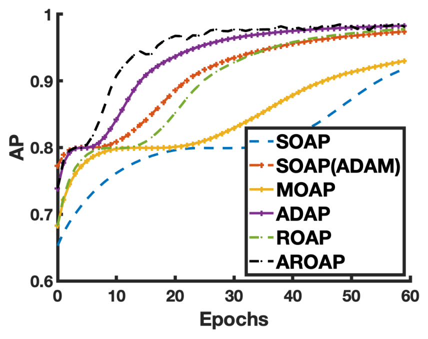


(e) w8a dataset

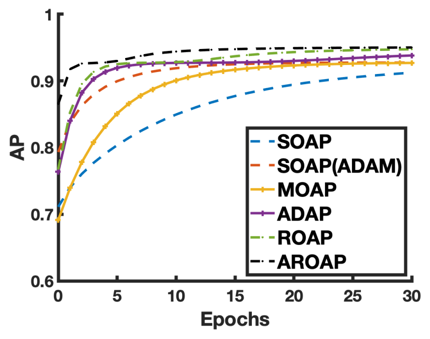


(f) covtype dataset

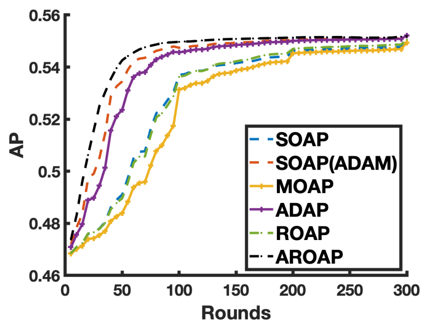
Fig. 2: AP vs number of epochs on the training set on the Linear Model task



(a) MNIST



(b) Fashion MNIST



(c) Tiny ImageNet

Fig. 3: AP vs number of epochs on the training set on the Deep Neural Networks task

Results.: Figure 3 shows that AP on test datasets for different optimizers. Results show that AROAP can outperform all baselines in terms of AUPRC, regardless of which model structure is used. We conduct each experiment 5 times, and the mean of test results are summarized in Table IV.

V. Conclusion

In this paper, we explore data mining on imbalanced data sets. We propose novel stochastic Recursive Momentum methods to maximize AP, a popular unbiased point estimator of AUPRC. Our methods improve the convergence rate to $O(1/\epsilon^3)$. We also design a family of stochastic adaptive methods with the same iteration complexity of $O(1/\epsilon^3)$. The experiments to train linear models and Deep Neural Networks demonstrate the effectiveness of our methods. Particularly, AROAP outperforms other related methods in all experiments. In addition, it is easy to extend this optimization to a multi-class setting as discussed in section II-B.

References

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

[2] H. Wang, J. Pang, M. A. Lodhi, Y. Tian, and D. Tian, "Festa: Flow estimation via spatial-temporal attention for scene point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 14 173–14 182.

[3] Y. Zhou, X. Liang, Y. Gu, Y. Yin, and L. Yao, "Multi-classifier interactive learning for ambiguous speech emotion recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 695–705, 2022.

[4] Y. Wei, R. Sheth, and R. Khardon, "Direct loss minimization for sparse gaussian processes," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2566–2574.

[5] R. Bao, X. Wu, W. Xian, and H. Huang, "Doubly sparse asynchronous learning for stochastic composite optimization," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI*, 2022, pp. 1916–1922.

[6] X. Gao, Z. Chen, S. Tang, Y. Zhang, and J. Li, "Adaptive weighted imbalance learning with application to abnormal activity recognition," *Neurocomputing*, vol. 173, pp. 1927–1935, 2016.

[7] T. Joachims, "A support vector method for multivariate performance measures," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 377–384.

[8] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 233–240.

[9] C. Cortes and M. Mohri, "Auc optimization vs. error rate minimization," *Advances in neural information processing systems*, vol. 16, 2003.

[10] Z. Yuan, Y. Yan, M. Sonka, and T. Yang, "Large-scale robust deep auc maximization: A new surrogate loss and empirical studies on medical image classification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3040–3049.

[11] W. Gao, R. Jin, S. Zhu, and Z.-H. Zhou, "One-pass auc optimization," in *International conference on machine learning*. PMLR, 2013, pp. 906–914.

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

[13] T. Qin, T.-Y. Liu, and H. Li, "A general approximation framework for direct optimization of information retrieval measures," *Information retrieval*, vol. 13, no. 4, pp. 375–397, 2010.

[14] K. Boyd, K. H. Eng, and C. D. Page, "Area under the precision-recall curve: point estimates and confidence intervals," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2013, pp. 451–466.

[15] C. Burges, R. Ragno, and Q. Le, "Learning to rank with nonsmooth cost functions," *Advances in neural information processing systems*, vol. 19, 2006.

[16] A. Herschtal and B. Raskutti, "Optimising area under the roc curve using gradient descent," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 49.

[17] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *arXiv preprint arXiv:1904.09237*, 2019.

[18] Q. Qi, Y. Luo, Z. Xu, S. Ji, and T. Yang, "Stochastic optimization of areas under precision-recall curves with provable convergence," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[19] G. Wang, M. Yang, L. Zhang, and T. Yang, "Momentum accelerates the convergence of stochastic auprc maximization," *arXiv preprint arXiv:2107.01173*, 2021.

[20] Z. Guo, M. Liu, Z. Yuan, L. Shen, W. Liu, and T. Yang, "Communication-efficient distributed stochastic auc maximization with deep neural networks," in *International Conference on Machine Learning*. PMLR, 2020, pp. 3864–3874.

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

[22] D. Metzler and W. B. 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*, 2005, pp. 472–479.

[23] B. McFee and G. R. Lanckriet, "Metric learning to rank," in *ICML*, 2010.

[24] Y. Yue, T. Finley, F. Radlinski, and T. 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*, 2007, pp. 271–278.

[25] Y. Song, A. Schwing, R. Urtasun *et al.*, "Training deep neural networks via direct loss minimization," in *International conference on machine learning*. PMLR, 2016, pp. 2169–2177.

[26] P. Mohapatra, C. Jawahar, and M. P. Kumar, "Efficient optimization for average precision svm," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[27] M. Wang, E. X. Fang, and H. Liu, "Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions," *Mathematical Programming*, vol. 161, no. 1, pp. 419–449, 2017.

[28] M. Wang, J. Liu, and E. Fang, "Accelerating stochastic composition optimization," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[29] S. Ghadimi, A. Ruszczynski, and M. Wang, "A single timescale stochastic approximation method for nested stochastic optimization," *SIAM Journal on Optimization*, vol. 30, no. 1, pp. 960–979, 2020.

[30] Z. Huo, B. Gu, J. Liu, and H. Huang, "Accelerated method for stochastic composition optimization with nonsmooth regularization," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[31] J. Zhang and L. Xiao, "A stochastic composite gradient method with incremental variance reduction," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[32] X. Lian, M. Wang, and J. Liu, "Finite-sum composition optimization via variance reduced gradient descent," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1159–1167.

[33] L. Liu, J. Liu, C.-J. Hsieh, and D. Tao, "Stochastically controlled stochastic gradient for the convex and non-convex composition problem," *arXiv preprint arXiv:1809.02505*, 2018.

[34] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," *Advances in neural information processing systems*, vol. 26, 2013.

[35] A. Defazio, F. Bach, and S. Lacoste-Julien, "Saga: A fast incremental gradient method with support for non-strongly convex composite objectives," *Advances in neural information processing systems*, vol. 27, 2014.

[36] L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč, "Sarah: A novel method for machine learning problems using stochastic recursive gradient," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2613–2621.

[37] C. Fang, C. J. Li, Z. Lin, and T. Zhang, "Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[38] Z. Wang, K. Ji, Y. Zhou, Y. Liang, and V. Tarokh, "Spiderboost and momentum: Faster variance reduction algorithms," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[39] A. Cutkosky and F. Orabona, "Momentum-based variance reduction in non-convex sgd," *Advances in neural information processing systems*, vol. 32, 2019.

[40] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization." *Journal of machine learning research*, vol. 12, no. 7, 2011.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[42] H. Yuan and W. Hu, "Stochastic recursive momentum method for non-convex compositional optimization," *arXiv preprint arXiv:2006.01688*, 2020.

[43] R. Xin, U. Khan, and S. Kar, "A hybrid variance-reduced method for decentralized stochastic non-convex optimization," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 459–11 469.

[44] M. Nouiehed, M. Sanjabi, T. Huang, J. D. Lee, and M. Razaviyayn, "Solving a class of non-convex min-max games using iterative first order methods," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

APPENDIX

A. Proof of Theorem 1

According to the above lemma 1, the function $\Phi(w)$ has L_Φ-Lipschitz continuous gradient. Then we have

$$\begin{aligned}\Phi(w_{t+1}) \leq&\Phi(w_t) + \langle \nabla\Phi(w_t), w_{t+1}-w_t\rangle + \frac{L_\Phi}{2}\|w_{t+1}-w_t\|^2\\ =&\Phi(w_t) - \frac{\eta}{2}\|\nabla\Phi(w_t)\|^2 + \frac{\eta}{2}\|\nabla\Phi(w_t)-F_t\|^2\\ &-(\frac{\eta}{2}-\frac{L_\Phi\eta^2}{2})\|F_t\|^2 \end{aligned}\tag{14}$$

where the first equality holds by $w_{t+1}=w_t-\eta F_t$. Then we decompose the third term in (14) and obtain

$$\begin{aligned}&\|\nabla\Phi(w_t)-F_t\|^2\\ =&\|\nabla\Phi(w_t)-\frac{1}{m}\sum_{i=1}^m \nabla g_i(w_t)^\top \nabla f([U_t]_i)\\ &+\frac{1}{m}\sum_{i=1}^m \nabla g_i(w_t)^\top \nabla f([U_t]_i) - \frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i)\\ &+\frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i) - F_t\|^2\\ \leq&3\|\nabla\Phi(w_t)-\frac{1}{m}\sum_{i=1}^m \nabla g_i(w_t)^\top \nabla f([U_t]_i)\|^2\\ &+3\|\frac{1}{m}\sum_{i=1}^m \nabla g_i(w_t)^\top \nabla f([U_t]_i) - \frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i)\|^2\\ &+3\|\frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i) - F_t\|^2 \end{aligned}\tag{15}$$

For the first term in (15), according to lemma 1, we have

$$\begin{aligned}&\mathbb{E}\|\nabla\Phi(w_t)-\frac{1}{m}\sum_{i=1}^m \nabla g_i(w_t)^\top \nabla f([U_t]_i)\|^2\\ \leq&\frac{C_g^2}{m}\sum_{n=1}^m \mathbb{E}\|\nabla f(g_i(w_t)) - \nabla f([U_t]_i)\|^2\\ \leq&\frac{C_g^2L_f^2}{m}\sum_{i=1}^m \mathbb{E}\|g_i(w_t)-[U_t]_i\|^2 \end{aligned}\tag{16}$$

Based on lemma 2 and (16), taking average over $t=1,2,\cdots,T$ on both sides, we have

$$\begin{aligned}&\frac{1}{T}\sum_{t=1}^T \mathbb{E}\|\nabla\Phi(w_t)-\frac{1}{m}\sum_{i=1}^m \nabla g_i(w_t)^\top \nabla f([U_t]_i)\|^2\\ \leq&\frac{C_g^2L_f^2}{Tb_0\beta}\sigma^2 + \frac{2C_g^4L_f^2}{\beta bT}\sum_{t=1}^T\mathbb{E}\|w_{t+1}-w_t\|^2 + \frac{2C_g^2L_f^2\beta\sigma^2}{b} \end{aligned}\tag{17}$$

Similarly, for the second term in (15), we have

$$\begin{aligned}&\mathbb{E}\|\frac{1}{m}\sum_{i=1}^m \nabla g_i(w_t)^\top \nabla f([U_t]_i) - \frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i)\|^2\\ \leq&\frac{C_f^2}{m}\sum_{i=1}^m \mathbb{E}\|\nabla g_i(w_t)-[V_t]_i\|^2 \end{aligned}\tag{18}$$

Based on lemma 2, and (18), taking average over $t=1,2,\cdots,T$ on both sides, we have

$$\begin{aligned}&\frac{1}{T}\sum_{n=1}^T \mathbb{E}\|\frac{1}{m}\sum_{i=1}^m \nabla g_i(w_t)^\top \nabla f([U_t]_i) - \frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i)\|^2\\ \leq&\frac{C_f^2}{Tb_0\alpha}\sigma^2 + \frac{2C_f^2L_g^2\sum_{t=1}^T\mathbb{E}\|w_t-w_t\|^2}{\alpha bT} + \frac{2C_f^2\alpha\sigma^2}{b} \end{aligned}\tag{19}$$

For the third term in (18), taking expectation over the randomness of B_t, we get

$$\begin{aligned}&\mathbb{E}_t\|F_t - \frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i)\|\\ =&\mathbb{E}_t\|(1-\gamma)F_{t-1} + \frac{1}{b}\sum_{B_t}\Big[[V_t]_i^\top \nabla f([U_t]_i)\\ &-(1-\gamma)[V_{t-1}]_i^\top \nabla f([U_{t-1}]_i)\Big] - \frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i)\|^2\\ \leq&\mathbb{E}_t\|(1-\gamma)(F_{t-1}-\frac{1}{m}\sum_{i=1}^m [V_{t-1}]_i^\top \nabla f(U_{t-1})_i)\|^2\\ &+\mathbb{E}_t\|\gamma[\frac{1}{b}\sum_{B_t}[V_t]_i^\top \nabla f([U_t]_i) - \frac{1}{m}\sum_{i=1}^m [V_t]_i^\top \nabla f([U_t]_i)]\\ &+(1-\gamma)[\frac{1}{b}\sum_{B_t}[V_t]_i\nabla f([U_t]_i) - \frac{1}{b}\sum_{B_t}[V_{t-1}]_i\nabla f([U_{t-1}]_i)\|^2\\ &-(1-\gamma)\frac{1}{m}\sum_{i=1}^m[[V_t]_i^\top\nabla f([U_t]_i) - [V_{t-1}]_i^\top\nabla f([U_{t-1}]_i)]\|^2\\ =&(1-\gamma)^2\mathbb{E}_t\|F_{t-1}-\frac{1}{m}\sum_{i=1}^m[V_{t-1}]_i^\top\nabla f([U_{t-1}]_i)\|^2\\ &+2\gamma^2\mathbb{E}_t\|\frac{1}{b}\sum_{B_t}[[V_t]_i\nabla f([U_t]_i)] - \frac{1}{m}\sum_{i=1}^m[V_t]_i^\top\nabla f([U_t]_i)\|^2\\ &+2(1-\gamma)^2\mathbb{E}_t\|\frac{1}{b}\sum[[V_t]_i^\top\nabla f([U_t]_i) - [V_{t-1}]_i\nabla f([U_{t-1}]_i)]\\ &-\frac{1}{m}\sum_{i=1}^m[[V_t]_i\nabla f([U_t]_i) - [V_{t-1}]_i\nabla f([U_{t-1}]_i)]\|^2 \end{aligned}\tag{20}$$

Then based on assumption 2, we have

$$\mathbb{E}_t\|F_t - \frac{1}{m}\sum_{i=1}^{m}[V_t]_i^\top \nabla f([U_t]_i)\|$$
$$=(1-\gamma)^2\mathbb{E}_t\|F_{t-1} - \frac{1}{m}\sum[V_{t-1}]_i\nabla f(U_{t-1}]_i)\|^2 + 2\gamma^2\sigma^2$$
$$+2(1-\gamma)^2\frac{1}{b}\sum_{B_t}\mathbb{E}_t\|[V_t]_i\nabla f([U_t]_i) - [V_{t-1}]_i\nabla f([U_{t-1}]_i)\|^2 \tag{21}$$

For the last term, we have

$$\mathbb{E}_t[\frac{1}{B}\sum_{B_t}\|[V_t]_i^\top\nabla f([U_t]_i) - [V_{t-1}]_i^\top\nabla f([U_{t-1}]_i)\|]$$
$$\leq 2\mathbb{E}_t[\frac{1}{B}\sum_{B_t}\|[V_t]_i^\top\nabla f([U_t]_i) - [V_{t-1}]_i^\top\nabla f([U_t]_i)\|^2$$
$$+\|[V_{t-1}]_i^\top\nabla f([U_t]_i) - [V_{t-1}]_i^\top\nabla f([U_{t-1}]_i)\|^2]$$
$$\leq 2\frac{C_f^2}{m}\mathbb{E}_t\|V_t - V_{t-1}\|^2 + 2\frac{C_g^2L_f^2}{m}\mathbb{E}_t\|U_t - U_{t-1}\|^2 \tag{22}$$

where the last inequality is due to lemma 1 and lemma A.5 in [42]. Then according to the update step in algorithm 1, we have

$$\mathbb{E}_t\|U_t - U_{t-1}\|^2$$
$$=\mathbb{E}_t\|\hat{g}(w_t) - \hat{g}(w_{t-1}) - \beta(U_{t-1} - \hat{g}(w_{t-1})\|^2$$
$$\leq 2\mathbb{E}_t\|\hat{g}(w_t) - \hat{g}(w_{t-1})\|^2 + 2\mathbb{E}_t\|\beta(U_{t-1} - \hat{g}(w_{t-1})\|^2$$
$$\leq 2mC_g^2\mathbb{E}_t\|w_t - w_{t-1}\|^2 + 2\beta^2 mG_g^2 \tag{23}$$

Similarly,

$$\mathbb{E}_t\|V_t - V_{t-1}\|^2$$
$$=\mathbb{E}_t\|\nabla\hat{g}(w_t) - \nabla\hat{g}(w_{t-1}) - \alpha(V_{t-1} - \nabla\hat{g}(w_{t-1})\|^2$$
$$\leq 2mL_g^2\mathbb{E}_t\|w_t - w_{t-1}\|^2 + 2\alpha^2 mC_g^2. \tag{24}$$

Plugging the (22) (23) and (24) into (20) and we have

$$\mathbb{E}_t\|F_t - \frac{1}{m}\sum[V_t]_i\nabla f([U_t]_i)\|^2$$
$$\leq(1-\gamma)^2\mathbb{E}_t\|F_{t-1} - \frac{1}{m}\sum[V_{t-1}]_i\nabla f([U_{t-1}]_i)\|^2 + 2\gamma^2\sigma^2$$
$$+8(1-\gamma)^2[(C_g^2 + L_g^2)\mathbb{E}_t\|w_t - w_{t-1}\|^2 + (\alpha^2C_g^2 + \beta^2G_g^2)] \tag{25}$$

Taking average over $t = 1, 2, \cdots, T$ and the recursive expansion on both sides, we have

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_t\|F_t - \frac{1}{m}\sum_{i=1}^{m}[V_t]_i\nabla f([U_t]_i)\|^2 \leq \frac{G}{\gamma b_0 T} + 2\gamma\sigma^2$$
$$+\frac{8}{\gamma T}(C_g^2 + L_g^2)\sum_{t=1}^{T}\mathbb{E}_t\|w_t - w_{t-1}\|^2 + 8\frac{\alpha^2C_g^2 + \beta^2G_g^2}{\gamma} \tag{26}$$

From (14), we have

$$\mathbb{E}[\Phi(w_{t+1})] \leq \mathbb{E}[\Phi(w_t)] - \frac{\eta}{2}\mathbb{E}\|\nabla\Phi(w_t)\|^2$$
$$+\frac{\eta}{2}\mathbb{E}\|\nabla\Phi(w_t) - F_t\|^2 - \left(\frac{\eta}{2} - \frac{L_\Phi\eta^2}{2}\right)\mathbb{E}\|F_t\|^2 \tag{27}$$

Taking average over $t = 1, 2, \cdots, T$ on both sides, we have

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla\Phi(w_t)\|^2 \leq \frac{2[\Phi(w_0) - \Phi(w_{t+1})]}{\eta T}$$
$$+\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla\Phi(w_t) - F_t\|^2 - \frac{1}{T}\sum_{t=1}^{T}(1 - L_\Phi\eta)\mathbb{E}\|F_t\|^2 \tag{28}$$

Then by plugging (15) (17), (19) and (26) into (28), we have

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla\Phi(w_t)\|^2 \leq \frac{2(\Phi(w_0) - \Phi(w^\star))}{\eta T}$$
$$-\frac{1}{T}\sum_{t=1}^{T}(1 - L_\Phi\eta)\mathbb{E}\|F_t\|^2 + (\frac{C_g^2L_f^2}{\beta} + \frac{C_f^2}{\alpha} + \frac{G}{\gamma})\frac{3\sigma}{Tb_0}$$
$$+[\frac{C_g^4L_f^2}{\beta b} + \frac{C_f^2L_g^2}{\alpha b} + \frac{4(C_g^2 + L_g^2)}{\gamma}]\frac{6\sum_{t=1}^{T}\eta^2\mathbb{E}\|F_t\|^2}{T}$$
$$+\frac{6C_g^2L_f^2\beta\sigma^2}{b} + \frac{6C_f^2\alpha\sigma^2}{b} + 6\gamma\sigma^2 + 24\frac{\alpha^2C_g^2 + \beta^2G_g^2}{\gamma} \tag{29}$$

Then setting $O(\alpha) = O(\beta) = O(\gamma) = O(T^{-2/3})$, $\eta = O(T^{-1/3})$, and $b_0 = O(T^{1/3})$, $b = O(1)$ and $1 - L_\Phi\eta - 6(\frac{C_g^4L_f^2}{\beta} + \frac{C_f^2L_g^2}{\alpha} + \frac{4(C_g^2+L_g^2)}{\gamma})\eta^2 > 0$, then we have $T = O(\epsilon^{-3})$ to ensure that $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\nabla\Phi(\mathbf{w}_t)\|^2\right] \leq \epsilon^2$.

B. Proof of Theorem 2

The main proof of Theorem 2 is similar to the Theorem 1, and we just change the update method of model parameters.

Let $\eta_t^s = 1/\left(\sqrt{\mathbf{v}_t} + \delta\right)$. Based on the boundedness of estimator $[U]$ and $[V]$ with project operator, the assumptions 1, 2, $\|d_t\|$ are bounded [42]. It is clear that $\eta c_l \leq [\eta_t^s]_i \leq \eta c_u$ for all $i \in [d]$. Given that (14), we have

$$\Phi(w_{t+1}) = \Phi(w_t) - \frac{c_l\eta}{2}\|\nabla\Phi(w_t)\|^2$$
$$+\frac{c_u\eta}{2}\|\nabla\Phi(w_t) - F_t\|^2 - \left(\frac{\eta}{2} - \frac{L_\Phi c_u\eta^2}{2}\right)\|F_t\|^2 \tag{30}$$

Following the proof in theorem 1, we can get

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla\Phi(w_t)\|^2 \leq \frac{2(\Phi(w_0) - \Phi(w^\star))}{c_l\eta T}$$
$$-\frac{1}{Tc_l}\sum_{t=1}^{T}(1 - L_\Phi c_u\eta)\mathbb{E}\|F_t\|^2 + (\frac{C_g^2L_f^2}{\beta} + \frac{C_f^2}{\alpha} + \frac{G}{\gamma})\frac{3\sigma c_u}{Tb_0c_l}$$
$$+(\frac{C_g^4L_f^2}{\beta b} + \frac{C_f^2L_g^2}{\alpha b} + \frac{4(C_g^2 + L_g^2)}{\gamma})\frac{6\sum_{t=1}^{T}c_u\eta^2\mathbb{E}_t\|F_t\|^2}{Tc_l}$$
$$+\frac{c_u}{c_l}[\frac{6C_g^2L_f^2\beta\sigma^2}{b} + \frac{6C_f^2\alpha\sigma^2}{b} + 6\gamma\sigma^2 + 24\frac{(\alpha^2C_g^2 + \beta^2G_g^2)}{\gamma}] \tag{31}$$

Then setting $O(\alpha) = O(\beta) = O(\gamma) = O(T^{-2/3})$, $\eta = O(T^{-1/3})$, $b_0 = O(T^{1/3})$, $b = O(1)$ and $1 - L_\Phi c_u\eta - 6(\frac{C_g^4L_f^2}{\beta} + \frac{C_f^2L_g^2}{\alpha} + \frac{4(C_g^2+L_g^2)}{\gamma})c_u\eta^2 > 0$, then we have $T = O(\epsilon^{-3})$ to ensure that $\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\nabla\Phi(\mathbf{w}_t)\|^2\right] \leq \epsilon^2$.