Decentralized Stochastic Compositional Gradient Descent for AUPRC Maximization

Hongchang Gao* Yubin Duan* Yihan Zhang* Jie Wu*

Abstract

In this paper, we consider the large-scale Area Under the Precision-Recall Curve (AUPRC) maximization problem for the imbalanced data classification task. Existing optimization methods for AUPRC maximization only focus on the single-machine setting, which are not applicable to the distributed data. To address this problem, we propose a novel decentralized stochastic compositional gradient descent method for large-scale AUPRC maximization. Our theoretical analysis shows that it can achieve a better sample complexity $\mathcal{O}(1/\epsilon^4)$ than $\mathcal{O}(1/\epsilon^6)$ of existing decentralized methods, but has the same communication complexity $\mathcal{O}(1/\epsilon^4)$. To further reduce the communication cost, we developed a novel communication-efficient decentralized stochastic compositional gradient descent method, whose communication complexity is improved to $\mathcal{O}(1/\epsilon^{4-4\alpha})$ (where $\alpha \in (0, 1/4)$). To the best of our knowledge, this is the first work achieving such favorable sample and communication complexities. Finally, we conduct extensive experiments for imbalanced data classification and the empirical results confirm the superior performance of our proposed methods.

1 Introduction

In many real-world machine learning and data mining applications, the data is highly imbalanced, i.e., the distribution of samples across different classes is highly skewed. For instance, in the click-through rate (CTR) prediction task of online advertisement, the number of positive samples is much smaller than the number of negative samples. Meanwhile, the real-world data is usually very large and often distributed on different devices. Thus, it is necessary to develop distributed optimization algorithms for this kind of large-scale imbalanced data classification problem.

The imbalanced data distribution can degenerate the classifier's performance significantly. To address this problem, a feasible way is to learn a classifier by directly optimizing the metric designed for measuring the performance of a classifier on imbalanced data, such as area under the curve (AUC). AUC includes Area Under the ROC Curve (AUROC) and Area Under the Precision-Recall Curve (AUPRC). In the past few years, a lot of efforts [19, 6, 14, 16] have been made to improving the classification performance for the imbalanced data via maximizing AUC. For instance, [19] formulated the AUROC maximization task as a minimax optimization problem and employed the stochastic gradient descent ascent (SGDA) method to optimize it. In addition, [14] formulated the AUPRC maximization problem as a stochastic compositional optimization problem, and then employed the stochastic compositional gradient descent (SCGD) method to optimize it. More recently, [16] developed a stochastic compositional gradient descent method with momentum (SCGDM) to improve the convergence rate. In this paper, we will focus on the AUPRC maximization problem.

However, all aforementioned methods only consider the single-machine case, which are not applicable to the large-scale imbalanced data. In fact, in real-world applications, the data is typically distributed on different devices, and then maximizing AUPRC in a distributed manner becomes more and more necessary. Recently, a wide variety of distributed optimization methods [11, 13, 9, 15, 20] for large-scale machine learning models have been proposed. Among them, the decentralized training method, where the devices conduct peer-to-peer communication with their neighboring devices, has attracted increasing attention, since there exists no communication bottleneck in the central server. For instance, [11] developed the decentralized stochastic gradient descent (DSGD) method and studied its convergence rate for nonconvex problems. [10] developed the communication efficient DSGD by employing the periodic communication strategy. However, these standard DSGD methods cannot be applied to AUPRC maximization, as it is a stochastic *compositional* optimization problem.

Recently, [4] developed the first decentralized stochastic compositional gradient descent (DSCGD) method for stochastic compositional optimization problems. However, this method has several limitations. First, to achieve the ϵ-accuracy stationary point, the

*Department of Computer and Information Sciences, Temple University. hongchang.gao@temple.edu, yubin.duan@temple.edu, yihan.zhang0002@temple.edu, jiewu@temple.edu

batch size should be as large as $\mathcal{O}(1/\epsilon^2)$, which is not feasible for practical applications. Second, DSCGD employed standard SCGD on each device so that the sample complexity in [4] is $\mathcal{O}(1/\epsilon^6)$, which is worse than $\mathcal{O}(1/\epsilon^4)$ of the single-machine method in [5]. More importantly, standard SCGD used in [4] cannot be directly used for AUPRC maximization. Specifically, standard SCGD employs a common moving average estimator for all inner-level functions. However, the inner-level function in the reformulated AUPRC problem is an approximation for the rank of each sample. Thus, one cannot use a common estimator to estimate all inner-level functions as standard SCGD.

To address aforementioned problems, in this paper, we developed a novel sample-efficient decentralized stochastic compositional gradient descent with momentum (SE-DSCGDM) method for AUPRC maximization, which enjoys a smaller batch size and better sample complexity. In particular, the batch size is improved to $\mathcal{O}(1)$ and the sample complexity is improved to $\mathcal{O}(1/\epsilon^4)$, which are much better than those of DSCGD [4]. To the best of our knowledge, this is the first work achieving such a favorable sample complexity under the decentralized setting. However, like existing methods [4], SE-DSCGDM needs to conduct communication at each iteration, resulting in the $\mathcal{O}(1/\epsilon^4)$ communication complexity. To reduce the communication cost, we further developed a communication-efficient method CE-DSCGDM by employing the periodic communication strategy, which can achieve the $\mathcal{O}(1/\epsilon^{(4-4\alpha)})$ communication complexity where $\alpha \in (0, 1/4)$. The comparison between our methods and existing methods can be found in Table 1. Finally, extensive experimental results confirmed the superior performance of our proposed two methods. The contributions of this work is summerazied below:

- We developed a sample-efficient decentralized stochastic compositional gradient descent method for AUPRC maximization. It enjoys a better sample complexity $\mathcal{O}(1/\epsilon^4)$ than existing decentralized SCGD methods.

- We proposed a communication-efficient decentralized stochastic compositional gradient descent method, which achieves a better communication complexity $\mathcal{O}(1/\epsilon^{4-4\alpha})$ where $\alpha \in (0, 1/4)$ than existing decentralized SCGD methods and our first method.

- We conduct extensive experiments on the imbalanced classification problem and the extensive experimental results confirm the effectiveness of our proposed methods.

2 Related Works

2.1 AUC Maximization Traditional classification methods, such as logistic regression, aim to minimize the classification error. They typically do not perform well on the imbalanced data. On the contrary, the method based on AUC maximization directly maximizes the AUC score, which can benefit the imbalanced classification problem. As a result, AUC maximization has attracted increasing attention, and a wide variety of methods have been proposed in recent years. Typically, these methods can be categorized into two classes: AUROC maximization and AUPRC maximization.

AUROC Maximization. As for AUROC maximization, traditional methods [7, 8] suffer from large computational cost due to the pairwise loss function. Recently, [19] reformulated AUROC maximization as a convex-concave minimax optimization problem for the linear classifier. Then, it can be efficiently optimized by stochastic gradient descent ascent method. Afterwards, numerous methods have been proposed based on the minimax reformulation. For instance, [12] applied this strategy to deep neural networks and reformulated AUROC maximization as a nonconvex-concave minimax problem. [22] developed an margin-based minmax surrogate loss for robust AUROC maximization. Recently, [6, 25] developed distributed AUROC maximization methods for Federated Learning. However, maximizing AUROC does not maximize AUPRC. Thus, it is necessary to study how to efficiently optimize AUPRC.

AUPRC Maximization. Regarding AUPRC maximization, it is more challenging to optimize compared with AUROC maximization, since it is difficult to obtain sample-wise loss function for stochastic optimization. A lot of efforts have been made to address this challenging problem. For instance, [2] developed an approximated method, which reformulated AUPRC maximization as a constrained minimax problem and then it could be solved in a stochastic manner. Recently, [14] proposed to optimize the average precision (AP) instead of AUPRC, since AP is an unbiased estimation for AUPRC. As a result, maximizing AUPRC boils down to optimize a stochastic compositional optimization problem. Based on this reformulation, [14] developed a stochastic compositional gradient descent with momentum (SCGDM) method , which enjoys $\mathcal{O}(1/\epsilon^5)$ sample complexity. However, this sample complexity is suboptimal. More recently, [16] developed a new SCGDM method for AUPRC maximization, whose sample complexity is improved to $\mathcal{O}(1/\epsilon^4)$. However, all these methods only consider the single-machine case. It is unclear whether these methods can converge under the distributed setting.

	Methods	Sample complexity	Communication complexity	Batch size
Single-machine	SCGD [17]	$\mathcal{O}(1/\epsilon^8)$	-	$\mathcal{O}(1)$
	SOAP [14]	$\mathcal{O}(1/\epsilon^5)$	-	$\mathcal{O}(1)$
	MOAP-V2 [16]	$\mathcal{O}(1/\epsilon^4)$	-	$\mathcal{O}(1)$
Decentralized	GP-DSCGD [4]	$\mathcal{O}(1/\epsilon^6)$	$\mathcal{O}(1/\epsilon^4)$	$\mathcal{O}(1/\epsilon^2)$
	SE-DSCGDM (this work)	$\mathcal{O}(1/\epsilon^4)$	$\mathcal{O}(1/\epsilon^4)$	$\mathcal{O}(1)$
	CE-DSCGDM (this work)	$\mathcal{O}(1/\epsilon^4)$	$\mathcal{O}(1/\epsilon^{(4-4\alpha)})$	$\mathcal{O}(1)$

Table 1: The sample and communication complexity of different stochastic compositional gradient descent methods under the single-machine and decentralized settings for achieving the ϵ-accuracy stationary point, i.e., $\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\bar{\mathbf{x}}_t)\|^2 \leq \epsilon^2$. Here, $\alpha \in (0, 1/4)$.

2.2 Stochastic Compositional Optimization

Stochastic Compositional Optimization. The stochastic compositional optimization has been extensively studied in recent years due to its widespread applications in machine learning. For instance, [17] developed the first stochastic compositional gradient descent method, whose sample complexity is $\mathcal{O}(1/\epsilon^8)$. The major difference between SGD and SCGD lies in that SCGD introduces a variance-reduced estimator for the inner-level function to control the gradient variance. Afterwards, a series of variants have been proposed to improve the sample complexity of SCGD based on acceleration techniques [5, 18] and advanced variance reduction techniques [23, 24, 21, 1]. However, as mentioned earlier, these SCGD methods cannot be directly applied to the reformulated AUPRC maximization problem.

Decentralized Optimization. The decentralized optimization method is widely used for training large-scale machine learning models. The engaging devices compose a communication graph and they only communicate with the connected peer devices. Under this setting, decentralized stochastic gradient descent [11, 3] has been extensively studied in recent years. However, most existing methods cannot be applied to AUPRC maximization since they focus on the non-compositional optimization problem. To make decentralized training feasible for stochastic compositional problems, [4] developed two decentralized stochastic compositional gradient descent methods based on the gossip and gradient tracking communication strategy, respectively. As the standard single-machine SCGD, these two methods cannot be utilized for AUPRC maximization. Meanwhile, [4] didn't consider the momentum technique so that its convergence rate is suboptimal. Moreover, DSCGD in [4] conducts communication at each iteration. When the model size is large, it can incur considerably large communication costs, slowing down the overall convergence performance. In summary, these limitations make DSCGD infeasible for AUPRC maximization. It is necessary to develop sample-efficient and communication-efficient algorithms to maximize AUPRC for imbalanced classification problem.

3 Preliminaries

3.1 Stochastic Compositional Optimization for AUPRC Maximization Given n samples $\{(\mathbf{a}_i, b_i)\}_{i=1}^n$ where $\mathbf{a}_i \in \mathbb{R}^d$ and $b_i \in \{-1, 1\}$, the positive and negative samples are denoted as $\mathcal{D}_+ = \{(\mathbf{a}_i, b_i)\}_{i=1}^m$ and $\mathcal{D}_- = \{(\mathbf{a}_i, b_i)\}_{i=m+1}^n$, respectively. In this paper, it is assumed $m \gg n - m$ so that it is an imbalanced classification problem.

Considering that we have the classifier $h(\mathbf{x}; \mathbf{a}_i)$ parameterized by $\mathbf{x}$, to learn the model parameter $\mathbf{x}$, we optimize the approximated AUPRC as [14], which is defined as follows:

$$\text{AP} = \frac{1}{m}\sum_{\mathbf{a}_i \in \mathcal{D}_+} \frac{\sum_{j=1}^n \ell(\mathbf{x}; \mathbf{a}_j, \mathbf{a}_i) \cdot \mathbb{I}(b_j = 1)}{\sum_{j=1}^n \ell(\mathbf{x}; \mathbf{a}_j, \mathbf{a}_i)}, \tag{3.1}$$

where $\ell(\mathbf{x}; \mathbf{a}_j, \mathbf{a}_i) \in \mathbb{R}$ is the approximation of the indicator function $\mathbb{I}(h(\mathbf{x}; \mathbf{a}_j) \geq h(\mathbf{x}; \mathbf{a}_i))$. Following [14], we use the sigmoid function in our experiments, which is defined as follows:

$$\ell(\mathbf{x}; \mathbf{a}_i, \mathbf{a}_j) = \frac{\exp(\rho h(\mathbf{x}; \mathbf{a}_j) - h(\mathbf{x}; \mathbf{a}_i))}{1 + \exp(\rho h(\mathbf{x}; \mathbf{a}_j) - h(\mathbf{x}; \mathbf{a}_i))}, \tag{3.2}$$

where $\rho > 0$ is a hyperparameter. In fact, AP in Eq. (3.1) computes the average precision score. Specifically, the nominator computes the rank of $\mathbf{a}_i$ in the positive samples, while the denominator gives the rank of $\mathbf{a}_i$ in all samples. It is worth noting that AP is an unbiased estimator of AUPRC. Thus, maximizing AUPRC boils down to maximizing AP.

By introducing the following function

$$\omega(\mathbf{x}; \mathbf{a}_j, \mathbf{a}_i) = \begin{bmatrix} \mathbb{I}(b_j = 1)\,\ell(\mathbf{x}; \mathbf{a}_j, \mathbf{a}_i) \\ \ell(\mathbf{x}; \mathbf{a}_j, \mathbf{a}_i) \end{bmatrix}, \tag{3.3}$$

[14] shows that maximizing AP in Eq. (3.1) can be reformulated as a stochastic compositional minimization

problem as follows:

$$\min_{\mathbf{x}} F(\mathbf{x}) \triangleq \frac{1}{m} \sum_{i \in \mathcal{D}^+} f(g_i(\mathbf{x})) \ , \tag{3.4}$$

where the inner-level function is $g_i(\mathbf{x}) = \sum_{j=1}^{n} \omega(\mathbf{x}; \mathbf{a}_j, \mathbf{a}_i) \in \mathbb{R}^2$ and the outer-level function is $f(\mathbf{y}) = -\frac{y_1}{y_2} \in \mathbb{R}$ for $\mathbf{y} = [y_1, y_2]^T \in \mathbb{R}^2$. As a result, Eq. (3.4) can be optimized via stochastic compositional gradient descent.

3.2 Problem Setup In this paper, we consider to optimize AUPRC in a decentralized manner. Specifically, there are totally K devices in a decentralized training system. Each device connects with a few neighboring devices. The composed communication topology can be denoted by a graph $\mathcal{G} = \{V, W\}$ where $V = \{v_k\}_{k=1}^K$ denotes the device set and $W = [w_{ij}] \in \mathbb{R}^{K \times K}$ denotes the adjacency matrix. Specifically, $w_{ij} > 0$ if the i-th device and the j-th device are connected. Otherwise, $w_{ij} = 0$. In addition, the k-th ($k \in \{1, 2, \cdots, K\}$) device has its own dataset $\mathcal{D}^k = \{\mathcal{D}_+^k, \mathcal{D}_-^k\}$ where $\mathcal{D}_+^k = \{(\mathbf{a}_i^k, b_i^k)\}_{i=1}^m$ denotes the positive samples and $\mathcal{D}_-^k = \{(\mathbf{a}_i^k, b_i^k)\}_{i=m+1}^n$ represents the negative samples on the k-th device, respectively.

Under the aforementioned setting, all devices collaboratively optimize the following loss function:

$$\min_{\mathbf{x}} F(\mathbf{x}) \triangleq \frac{1}{K} \sum_{k=1}^{K} F^k(\mathbf{x}) = \frac{1}{K} \sum_{k=1}^{K} \Big(\frac{1}{m} \sum_{i \in \mathcal{D}_+^k} f(g_i^k(\mathbf{x})) \Big) \ , \tag{3.5}$$

where $g_i^k(\mathbf{x}) = \sum_{j=1}^{n} \omega^k(\mathbf{x}; \mathbf{a}_j^k, \mathbf{a}_i^k)$ denotes the function on the k-th device. Correspondingly, ω^k and ℓ^k are also the functions on the k-th device. Their definitions are the same as Eq. (3.3) and Eq. (3.2) but using the local data $\mathcal{D}^k = \{\mathcal{D}_+^k, \mathcal{D}_-^k\}$ on the k-th device.

To investigate the convergence rate of our proposed methods, we introduce the following assumptions, which are commonly used in existing works [14, 16, 4].

Assumption 1. *For any $k \in \{1, 2, \cdots, K\}$, there are two constant values $0 < C < M$ such that function $\ell^k(\cdot)$ is lower and upper bounded as $C \leq \ell^k(\cdot) \leq M$.*

Assumption 2. *For any $k \in \{1, 2, \cdots, K\}$, function $\ell^k(\cdot)$ is C_ℓ-Lipschitz continuous and $\nabla \ell^k(\cdot)$ is L_ℓ-Lipschitz continuous, where $C_\ell > 0$ and L_ℓ are two constant values.*

Assumption 3. *For any $k \in \{1, 2, \cdots, K\}$, the gradient of function $\omega^k(\cdot)$ is bounded as $\|\nabla \omega^k(\cdot)\| \leq \sigma^2$ where $\sigma > 0$ is a constant value.*

Assumption 4. *The adjacency matrix W satisfies $W^T = W$, $W\mathbf{1} = \mathbf{1}$, and $\mathbf{1}^T W = \mathbf{1}^T$. Additionally, the eigenvalues $\{\lambda_i\}_{i=1}^n$ of W satisfy $|\lambda_n| \leq \cdots \leq |\lambda_2| < |\lambda_1| = 1$.*

Based on Assumption 4, the spectral gap of the adjacency matrix is $1 - \lambda$ where $\lambda = |\lambda_2|$. Additionally, from Assumptions 1-3, we can obtain the following lemma.

LEMMA 3.1. *[16] Given Assumptions 1-3, it can be obtained:*

- *The outer-level function $f(\cdot)$ is C_f-Lipschitz continuous and its gradient is L_f-Lipschitz continuous.*
- *The inner-level function $g_i^k(\cdot)$ is C_g-Lipschitz continuous and its gradient is L_g-Lipschitz continuous.*
- *The compositional function $F^k(\cdot)$ is C_F-Lipschitz continuous and its gradient is L_F-Lipschitz continuous.*
- *The inner-level function value is upper bounded as $\|g_i^k(\cdot)\| \leq G$.*

Here, C_f, C_g, C_F, L_f, L_g, L_F, and G are positive constant values.

Algorithm 1 SE-DSCGDM

Input: $\mathbf{x}_0^k = \mathbf{x}_0$, $\beta \in (0, 1)$, $\gamma \in (0, 1)$, $\eta > 0$.

1: **for** $t = 0, \cdots, T-1$, each device k **do**
2: Select a minibatch of samples $\mathcal{B}_t^k$ from $\mathcal{D}_+^k$ and a minibatch of samples $\mathcal{S}_t^k$ from $\mathcal{D}^k$ to compute:
3: $\mathbf{U}_{i,t+1}^k = \begin{cases} (1-\beta)\,\mathbf{U}_{i,t}^k + \beta \frac{m}{|\mathcal{B}_t^k|} \tilde{g}_i^k\left(\mathbf{x}_t^k\right) & i \in \mathcal{B}_t^k \\ (1-\beta)\,\mathbf{U}_{i,t}^k & \text{o.w.} \end{cases}$
4: $\mathbf{v}_t^k = \frac{1}{|\mathcal{B}_t^k|} \sum_{i \in \mathcal{B}_t^k} \nabla \tilde{g}_i^k(\mathbf{x}_t^k)^T \nabla f(\mathbf{U}_{i,t+1}^k)$
5: $\mathbf{m}_{t+1}^k = (1-\gamma)\mathbf{m}_t^k + \gamma \mathbf{v}_t^k$
6: $\tilde{\mathbf{x}}_{t+1}^k = \mathbf{x}_t^k - \eta \mathbf{m}_{t+1}^k$
7: $\mathbf{x}_{t+1}^k = \sum_{j \in \mathcal{N}_{v_k}} w_{kj} \tilde{\mathbf{x}}_{t+1}^j$
8: **end for**

4 Decentralized Stochastic Compositional Gradient Descent with Momentum

4.1 Sample-Efficient Decentralized Stochastic Compositional Gradient Descent with Momentum In Algorithm 1, we developed the sample-efficient decentralized stochastic compositional gradient descent with momentum (SE-DSCGDM) method. In detail, at the t-th iteration, each device k selects a minibatch of samples $\mathcal{B}_t^k$ from positive samples $\mathcal{D}_+^k$ and a minibatch of samples $\mathcal{S}_t^k$ from all samples $\mathcal{D}^k$. Then, for $i \in \mathcal{B}_t^k$, Algorithm 1 uses the samples $\mathcal{S}_t^k$ to compute the stochastic

inner-level function value and its stochastic gradient as follows:

$$\begin{aligned}\tilde{g}_i^k\left(\mathbf{x}_t^k\right) &= \frac{n}{|\mathcal{S}_t^k|}\sum_{j\in\mathcal{S}_t^k}\omega^k(\mathbf{x}_t^k;\mathbf{a}_j^k,\mathbf{a}_i^k)\ ,\\ \nabla\tilde{g}_i^k\left(\mathbf{x}_t^k\right) &= \frac{n}{|\mathcal{S}_t^k|}\sum_{j\in\mathcal{S}_t^k}\nabla\omega^k(\mathbf{x}_t^k;\mathbf{a}_j^k,\mathbf{a}_i^k)\ ,\end{aligned} \tag{4.6}$$

where x_t^k denotes the model parameter of the k-th device at the t-th iteration, $|\mathcal{S}_t^k|$ denotes the size of the minibatch $\mathcal{S}_t^k$.

To compute the stochastic compositional gradient, we should use the moving average estimator to estimate the inner-level function value for controlling the gradient variance. The standard SCGD method uses a common estimator for all samples to estimate the inner-level function. However, from the definition of $g_i^k(\mathbf{x}_t^k)$, it can be observed that the inner-level function involves the rank of the i-th sample. Thus, it is not reasonable to use a common estimator for all samples. In this paper, we employ the sample-wise moving average estimator proposed in [16] to estimate the inner-level function $g_i^k(\mathbf{x}_t^k)$:

$$\mathbf{U}_{i,t+1}^k = \begin{cases}(1-\beta)\,\mathbf{U}_{i,t}^k + \beta\frac{m}{|\mathcal{B}_t^k|}\tilde{g}_i^k\left(\mathbf{x}_t^k\right), & i\in\mathcal{B}_t^k\\ (1-\beta)\,\mathbf{U}_{i,t}^k\ , & \text{o.w.}\end{cases}, \tag{4.7}$$

where $\beta\in(0,1)$ is a hyperparameter, $\mathbf{U}_{i,t}^k\in\mathbb{R}^2$ is the moving average estimation of the inner-level function for the i-th positive sample on the k-th device. Then, the k-th device computes the stochastic compositional gradient $\mathbf{v}_t^k$ as shown in Step 4 and the momentum $\mathbf{m}_{t+1}^k$ as shown in Step 5 in Algorithm 1. Afterwards, the k-th device updates its local model parameter $\mathbf{x}_t^k$ with its local momentum $\mathbf{m}_{t+1}^k$ and communicates the updated model parameter $\tilde{\mathbf{x}}_{t+1}^k$ with its neighboring devices $\mathcal{N}_k$ as shown in Steps 6-7, where $\eta>0$ denotes the learning rate. All devices repeat this procedure until it converges.

In what follows, we establish the convergence rate of Algorithm 1 for nonconvex problems.

THEOREM 4.1. *Given Assumptions 1-4, by setting* $|\mathcal{S}_t^k| = |\mathcal{B}_t^k| = B$, $\eta\le\frac{\gamma}{2\sqrt{2L_F^2+10C_g^4}}$, $\beta\in(0,1)$, $\gamma\in(0,1)$, *Algorithm 1 has the convergence rate:*

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\bar{\mathbf{x}}_t)\|^2 &\le \frac{2(F(\mathbf{x}_0)-F(\mathbf{x}_*))}{\eta T}+\frac{8\eta^2C_g^2C_f^2L_F^2}{(1-\lambda)^2}\\ &+\frac{2(C_g^2L_f^2+5C_g^2L_f^2G^2)}{\gamma T}+2\gamma(10C_g^2L_f^2G^2\frac{m}{B}+2C_g^2C_f^2)\ ,\end{aligned} \tag{4.8}$$

where $\bar{\mathbf{x}}_t=\frac{1}{K}\sum_{k=1}^K\mathbf{x}_t^k$ *and* $\mathbf{x}_*$ *is the optimal solution.*

Remark 1. *For Theorem 4.1, by setting* $\eta=\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$, $\gamma=\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$, *and* $B=\mathcal{O}(1)$, *we can get*

$$\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\bar{\mathbf{x}}_t)\|^2\le\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)+\mathcal{O}\left(\frac{1}{(1-\lambda)^2T}\right)\ . \tag{4.9}$$

From this convergence rate, we have two observations. First, the spectral gap only affects the high-order term of the convergence rate, which is consistent with existing decentralized optimization methods, such as DSGD [11]. Second, to achieve the ϵ*-accuracy stationary point, i.e.,* $\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\bar{\mathbf{x}}_t)\|^2\le\epsilon^2$, *the communication complexity (i.e., the number of iterations) is* $\mathcal{O}(1/\epsilon^4)$ *and the sample complexity is* $T\times B=\mathcal{O}(1/\epsilon^4)$. *Obviously, SE-DSCGDM has the same communication complexity as [4] but enjoys a better sample complexity than [4]. The reason for this improvement is that our batch size is* $\mathcal{O}(1)$, *while the batch size in [4] is* $\mathcal{O}(1/\epsilon^2)$.

Algorithm 2 CE-DSCGDM

Input: $\mathbf{x}_0^k=\mathbf{x}_0$, $\beta\in(0,1)$, $\gamma\in(0,1)$, $\eta>0$, $p>1$.

1: **for** $t=0,\cdots,T-1$, each device k **do**
2: Select a minibatch of samples $\mathcal{B}_t^k$ from $\mathcal{D}_+^k$ and a minibatch of samples $\mathcal{S}_t^k$ from $\mathcal{D}^k$ to compute:
3: $\mathbf{U}_{i,t+1}^k=\begin{cases}(1-\beta)\,\mathbf{U}_{i,t}^k+\beta\frac{m}{|\mathcal{B}_t^k|}\tilde{g}_i^k\left(\mathbf{x}_t^k\right), & i\in\mathcal{B}_t^k\\ (1-\beta)\,\mathbf{U}_{i,t}^k, & \text{o.w.}\end{cases}$
4: $\mathbf{v}_t^k=\frac{1}{|\mathcal{B}_t^k|}\sum_{i\in\mathcal{B}_t^k}\nabla\tilde{g}_i^k(\mathbf{x}_t^k)^T\nabla f(\mathbf{U}_{i,t+1}^k)$
5: $\mathbf{m}_{t+1}^k=(1-\gamma)\mathbf{m}_t^k+\gamma\mathbf{v}_t^k$
6: $\tilde{\mathbf{x}}_{t+1}^k=\mathbf{x}_t^k-\eta\mathbf{m}_{t+1}^k$
7: $\mathbf{x}_{t+1}^k=\begin{cases}\sum_{j\in\mathcal{N}_k}w_{kj}\tilde{\mathbf{x}}_{t+1}^j, & \text{mod(t+1,p)}=0\\ \tilde{\mathbf{x}}_{t+1}^k, & \text{o.w.}\end{cases}$
8: **end for**

4.2 Communication-Efficient Decentralized Stochastic Compositional Gradient Descent with Momentum From Algorithm 1, it can be observed that devices conduct communication at every iteration. This operation can incur large communication cost. To reduce the communication cost, we developed a communication-efficient decentralized stochastic compositional gradient descent with momentum (CE-DSCGDM) method in Algorithm 2. In detail, same as Algorithm 1, we compute the momentum $\mathbf{m}_{t+1}^k$ and then use it to update the local model parameter. Different from Algorithm 1, each device communicates with its neighbors at every p ($p>1$) iterations:

$$\mathbf{x}_{t+1}^k=\begin{cases}\sum_{j\in\mathcal{N}_k}w_{kj}\tilde{\mathbf{x}}_{t+1}^j, & \text{mod(t+1,p)}=0\\ \tilde{\mathbf{x}}_{t+1}^k, & \text{o.w.}\end{cases}\ . \tag{4.10}$$

In this way, the number of communication rounds is reduced to T/p, which is smaller than T in Algorithm 1. Thus, Algorithm 2 is more communication-efficient than Algorithm 1. It is worth noting that Algorithm 2 is identical to Algorithm 1 if $p = 1$.

In Theorem 4.2, we establish the convergence rate of Algorithm 2 for nonconvex problems.

THEOREM 4.2. *Given Assumptions 1-4, by setting* $|\mathcal{S}_t^k| = |\mathcal{B}_t^k| = B$, $\eta \leq \frac{\gamma}{2\sqrt{2L_F^2+10C_g^4}}$, $\beta \in (0,1)$, $\gamma \in (0,1)$, *Algorithm 2 has the convergence rate:*

(4.11)
$$\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\bar{\mathbf{x}}_t)\|^2 \leq \frac{2(F(\mathbf{x}_0)-F(\mathbf{x}_*))}{\eta T} + \frac{2(C_g^2L_f^2+5C_g^2L_f^2G^2)}{\gamma T} + 2\gamma(10C_g^2L_f^2G^2\frac{m}{B}+2C_g^2C_f^2) + 16p^2\eta^2C_g^2C_f^2L_F^2\Big(1+\frac{1}{(1-\lambda)^2}\Big) ,$$

where $\bar{\mathbf{x}}_t = \frac{1}{K}\sum_{k=1}^K \mathbf{x}_t^k$ *and* $\mathbf{x}_*$ *is the optimal solution.*

Compared with Theorem 4.1, it can be observed that there is an additional term $16p^2\eta^2C_g^2C_f^2L_F^2\Big(1+\frac{1}{(1-\lambda)^2}\Big)$ in Theorem 4.2, which is caused by the periodic communication. In what follows, we demonstrate how the communication period p affects the convergence rate.

Remark 2. *For Theorem 4.2, by setting* $\eta = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$, $\gamma = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$, $B = \mathcal{O}(1)$, *and* $p = \mathcal{O}(T^\alpha)$ *where* $\alpha \in (0,1/2)$, *we can get*

(4.12)
$$\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\bar{\mathbf{x}}_t)\|^2 \leq \mathcal{O}\left(\frac{1}{\sqrt{T}}\right) + \mathcal{O}\left(\frac{1}{(1-\lambda)^2T^{1-2\alpha}}\right).$$

Here, we have two observations. First, when $\alpha \in (0,1/4)$, *the first term on the right hand side is dominant. Then, Algorithm 2 has the same convergence rate with Algorithm 1, which means that the periodic communication strategy does not worsen the convergence rate. When* $\alpha \in (1/4,1/2)$, *the second term on the right hand side is dominant, resulting in a slower convergence rate than Algorithm 1. Second, by setting* $\alpha \in (0,\frac{1}{4})$, *the sample complexity of Algorithm 2 is* $\mathcal{O}(1/\epsilon^4)$ *and the communication complexity is* $T/p = \mathcal{O}(1/\epsilon^{(4-4\alpha)})$. *Thus, Algorithm 2 can achieve a better communication complexity than Algorithm 1 when the communication period* $p \in (1, T^{1/4})$.

5 Proof Sketch

In this section, we provide the proof sketch of Theorem 4.1 and Theorem 4.2

5.1 Proof of Theorem 4.1

To prove Theorem 4.1, we introduce the following key lemmas, whose proof can be found in Appendix A.1.

LEMMA 5.1. *Given Assumptions 1-3, we can get*

$$\|\mathbf{v}_t^k - \bar{\mathbf{v}}_t\|^2 \leq 4C_g^2C_f^2 . \tag{5.13}$$

LEMMA 5.2. *Given Assumptions 1-3, we can get*

$$\|\mathbf{m}_t^k - \bar{\mathbf{m}}_t\|^2 \leq 4C_g^2C_f^2 . \tag{5.14}$$

LEMMA 5.3. *Given Assumptions 1-4, we can get*

$$\sum_{k=1}^K\|\mathbf{x}_{t+1}^k - \bar{\mathbf{x}}_{t+1}\|^2 \leq \frac{4K\eta^2C_g^2C_f^2}{(1-\lambda)^2} . \tag{5.15}$$

LEMMA 5.4. *[16] Given Assumptions 1-4, by setting* $\beta = \gamma$, *we can get*

(5.16)
$$\sum_{t=0}^{T-1}\|\nabla F^k(\mathbf{x}_t^k) - \mathbf{m}_{t+1}^k\|^2 \leq \frac{C_g^2L_f^2+5C_g^2L_f^2G^2}{\gamma} + \frac{\eta^2(2L_F^2+10C_g^4)}{\gamma^2}\sum_{t=0}^{T-1}\|\mathbf{m}_{t+1}^k\|^2 + 10\gamma C_g^2L_f^2G^2\frac{m}{B}T + 2\gamma C_g^2C_f^2T .$$

Based on these lemmas, we provide the proof sketch of Theorem 4.1. The details can be found in Appendix A.1.

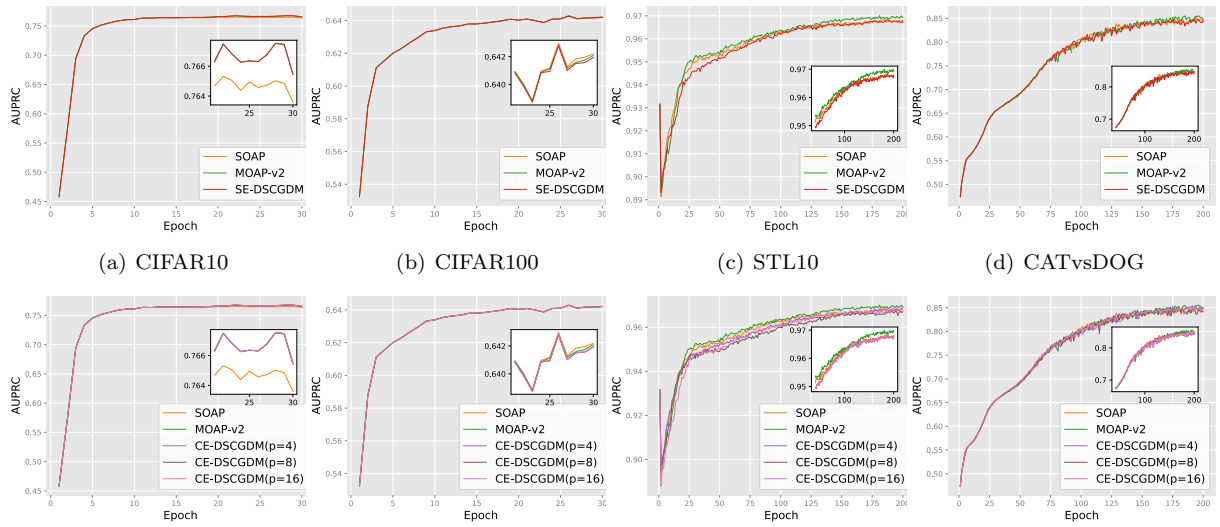
Proof. At first, since $F(x)$ is L_F-smooth, we can get

(5.17)
$$F(\bar{\mathbf{x}}_{t+1}) \leq F(\bar{\mathbf{x}}_t) - \frac{\eta}{2}\|\nabla F(\bar{\mathbf{x}}_t)\|^2 - \frac{\eta}{4}\frac{1}{K}\sum_{k=1}^K\|\mathbf{m}_{t+1}^k\|^2 + \frac{\eta L_F^2}{K}\sum_{k=1}^K\|\bar{\mathbf{x}}_t - \mathbf{x}_t^k\|^2 + \frac{\eta}{K}\sum_{k=1}^K\|\nabla F^k(\mathbf{x}_t^k) - \mathbf{m}_{t+1}^k\|^2 ,$$

By combining it with Lemmas 5.3- 5.4, we can get

(5.18)
$$\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\bar{\mathbf{x}}_t)\|^2 \leq \frac{2(F(\mathbf{x}_0)-F(\mathbf{x}_*))}{\eta T} + 20\gamma C_g^2L_f^2G^2\frac{m}{B} + \left(\frac{2\eta^2(2L_F^2+10C_g^4)}{\gamma^2TK} - \frac{1}{2TK}\right)\sum_{t=0}^{T-1}\sum_{k=1}^K\|\mathbf{m}_{t+1}^k\|^2 + \frac{8\eta^2C_g^2C_f^2L_F^2}{(1-\lambda)^2} + \frac{2(C_g^2L_f^2+5C_g^2L_f^2G^2)}{\gamma T} + 4\gamma C_g^2C_f^2 .$$

By setting $\eta \leq \frac{\gamma}{2\sqrt{2L_F^2+10C_g^4}}$, we complete the proof. ∎



(a) CIFAR10 (b) CIFAR100 (c) STL10 (d) CATvsDOG

(e) CIFAR10 (f) CIFAR100 (g) STL10 (h) CATvsDOG

Figure 1: The comparison between our two methods and baseline methods. The first row shows SE-DSCGDM and the second row shows CE-SCGDM.

5.2 Proof of Theorem 4.2 Similarly, we introduce the following key lemmas to prove Theorem 4.2. Their proof can be found in Appendix A.2.

LEMMA 5.5. *Given Assumptions 1-4, for any t, for $\forall s_t \in \{0, 1, 2, \cdots, \lfloor T/p \rfloor\}$, we can get*

$$\sum_{k=1}^{K} \|\mathbf{x}_{s_t p}^k - \bar{\mathbf{x}}_{s_t p}\|^2 \leq \frac{4Kp^2\eta^2 C_g^2 C_f^2}{(1-\lambda)^2} . \tag{5.19}$$

LEMMA 5.6. *Given Assumptions 1-4, we can get*

$$\sum_{k=1}^{K} \|\mathbf{x}_{t+1}^k - \bar{\mathbf{x}}_{t+1}\|^2 \leq 8Kp^2\eta^2 C_g^2 C_f^2 \Big(1 + \frac{1}{(1-\lambda)^2}\Big) . \tag{5.20}$$

Then, we provide the proof sketch of Theorem 4.2.

Proof. Same as Theorem 4.1, we can get Eq. (5.17). Then, by plugging Lemma 5.4 and Lemma 5.6 into Eq. (5.17), we can get

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1} \|\nabla F(\bar{\mathbf{x}}_t)\|^2 \leq{}& \frac{2(F(\mathbf{x}_0) - F(\mathbf{x}_*))}{\eta T} \\ &+ 4\gamma C_g^2 C_f^2 + 16p^2\eta^2 C_g^2 C_f^2 L_F^2 \Big(1 + \frac{1}{(1-\lambda)^2}\Big) \\ &+ \frac{2(C_g^2 L_f^2 + 5C_g^2 L_f^2 G^2)}{\gamma T} + 20\gamma C_g^2 L_f^2 G^2 \frac{m}{B} \\ &+ \left(\frac{2\eta^2(2L_F^2 + 10C_g^4)}{\gamma^2 TK} - \frac{1}{2TK}\right) \sum_{t=0}^{T-1}\sum_{k=1}^{K} \|\mathbf{m}_{t+1}^k\|^2 .\end{aligned} \tag{5.21}$$

By setting $\eta \leq \frac{\gamma}{2\sqrt{2L_F^2 + 10C_g^4}}$, we complete the proof. □

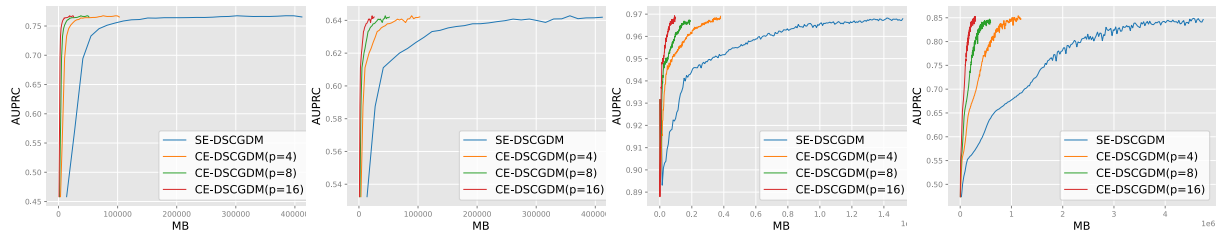
6 Experiment

6.1 Experimental Settings

Dataset Our experiment is conducted with four image classification datasets: CIFAR10, CIFAR100[1], STL10[2], and CATvsDOG[3]. For CIFAR10 and CIFAR100, 10% samples are randomly selected as the testing set. For CATvsDOG and STL10, 20% samples are randomly selected as the testing set. Then, following [14], we convert the training set of these four datasets to imbalanced binary classification datasets. Specifically, the first half classes in CIFAR10, CIFAR100 and STL10 are marked as negative and the other half classes are treated as positive samples. Then, for these three training sets, 98% of positive samples are randomly selected and removed to construct imbalanced datasets.

Setup Our testbed is set on an HPC server which has 8× NVIDIA Tesla Volta V100 GPUs connected with NVlink2 and 512GB of RAM. We implement our algorithms with PyTorch and OpenMPI backends. In particular, we write new loss and optimizer functions with PyTorch distributed training framework to integrate our decentralized momentum stochastic compositional gradient descent methods. In our experiments, we use ResNet18 as the classifier, and set $\beta = 0.99, \gamma =$

[1] https://www.cs.toronto.edu/~kriz/cifar.html
[2] https://cs.stanford.edu/~acoates/stl10/
[3] https://www.kaggle.com/c/dogs-vs-cats



(a) CIFAR10 (b) CIFAR100 (c) STL10 (d) CATvsDOG

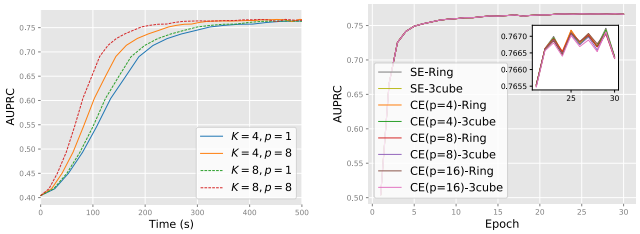
Figure 2: The communication cost of SE-DSCGDM and CE-DSCGDM.

$0.1, \eta = 0.0001, \rho = 5$. The batch size of CATvsDOG on each worker is set to 64 and that of other datasets is set to 16. In addition, we compare our methods with SOAP [14] and MOAP-V2 [16]. To make a fair comparison, we parallelize these two baseline methods by using the all-reduce communication.

6.2 Results and Analysis In Figure 1(a)-1(d), we plot the testing AUPRC score of SE-DSCGDM and baseline methods versus the number of epochs. In this experiment, we use four devices (GPU) and the ring topology. From them, it can be observed that our SE-DSCGDM method can converge to almost the same AUPRC score as the baseline methods for all datasets. Similarly, we plot the testing AUPRC score of CE-DSCGDM in Figure 1(e)-1(f) with different communication periods p. We can still observe that CE-DSCGDM with different p converges to almost the same AUPRC score with SE-DSCGDM, indicating p does not impair the convergence performance. All of these observations confirm the correctness of our proposed two methods.

To verify the communication efficiency of our CE-DSCGDM method, we plot the testing AUPRC score versus the communication cost (MB) under different communication periods $p = 4, 8, 16$ in Figure 2. It can be observed that CE-DSCGDM with a smaller communication period incurs more communication costs, while that with a larger communication period leads to less communication costs. Moreover, those variants can finally achieve almost the same AUPRC score as baseline methods, which confirms the communication efficiency of our proposed CE-DSCGDM method.

Additionally, to demonstrate the speedup of our decentralized training methods, in Figure 3(a), we plot the testing AUPRC score versus the consumed time when using different number of devices. Due to the space limitation, we only report the result of CIFAR10. Here, we use $K = 4$ and $K = 8$ devices, respectively. It can be observed that our methods converge faster when using more devices. For instance, SE-DSCGDM with $p = 1$ and $K = 8$ converges faster than that with $p = 1$ and $K = 4$. CE-DSCGDM with $p = 8$ and $K = 8$ converges faster than that with $p = 8$ and $K = 4$. All these observations confirm the efficiency of our methods.



(a) The consumed time of our methods with different number of devices for CIFAR10. $p = 1$ denotes SE-DSCGDM and $p > 1$ denotes CE-DSCGDM.

(b) The convergence performance of our two methods with the Ring and 3cube communication topology for CIFAR10. SE denotes SE-DSCGDM and CE represents CE-DSCGDM.

Figure 3: Ablation Studies

In Figure 3(b), we demonstrate the performance of our two methods when using different communication topology (different topology has different spectral gap). Specifically, we use eight devices and then build the Ring graph and the 3cube graph. In this experiment, we use CIFAR10 dataset. The batch size on each worker is set to 8. The other hyperparameters are set as those in Figure 1. From Figure 3(b), we can observe that our two methods achieve almost the same testing AUPRC score for different communication topology, which confirms that the spectral gap does not impair the convergence performance of our algorithms.

7 Conclusion

In this paper, we developed two decentralized stochastic compositional gradient descent methods to train the large-scale AUPRC maximization problem. As for the first method, which performs communication in every iteration, it can achieve a better sample complexity than existing decentralized SCGD methods [4]. However, it still has the same communication complexity as existing methods. As for the second method, it enjoys better sample and communication complexities than existing

methods by utilizing the periodic communication strategy to reduce the communication cost. To the best of our knowledge, this is the first work achieving these theoretical results. The extensive experimental results on imbalanced datasets confirm the correctness and effectiveness of our proposed methods.

References

[1] Tianyi Chen, Yuejiao Sun, and Wotao Yin. Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization. *arXiv preprint arXiv:2008.10847*, 2020.

[2] Elad Eban, Mariano Schain, Alan Mackey, Ariel Gordon, Ryan Rifkin, and Gal Elidan. Scalable learning of non-decomposable objectives. In *Artificial intelligence and statistics*, pages 832–840. PMLR, 2017.

[3] Hongchang Gao and Heng Huang. Periodic stochastic gradient descent with momentum for decentralized training. *arXiv preprint arXiv:2008.10435*, 2020.

[4] Hongchang Gao and Heng Huang. Fast training method for stochastic compositional optimization problems. *Advances in Neural Information Processing Systems*, 34, 2021.

[5] 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.

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

[7] Alan Herschtal and Bhavani Raskutti. Optimising area under the roc curve using gradient descent. In *Proceedings of the twenty-first international conference on Machine learning*, page 49, 2004.

[8] Thorsten Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd international conference on Machine learning*, pages 377–384, 2005.

[9] Anastasia Koloskova, Tao Lin, Sebastian U Stich, and Martin Jaggi. Decentralized deep learning with arbitrary communication compression. *arXiv preprint arXiv:1907.09356*, 2019.

[10] Xiang Li, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Communication efficient decentralized training with multiple local updates. *arXiv preprint arXiv:1910.09126*, 2019.

[11] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *arXiv preprint arXiv:1705.09056*, 2017.

[12] Mingrui Liu, Zhuoning Yuan, Yiming Ying, and Tianbao Yang. Stochastic auc maximization with deep neural networks. *arXiv preprint arXiv:1908.10831*, 2019.

[13] Shi Pu and Angelia Nedić. Distributed stochastic gradient tracking methods. *Mathematical Programming*, pages 1–49, 2020.

[14] Qi Qi, Youzhi Luo, Zhao Xu, Shuiwang Ji, and Tianbao Yang. Stochastic optimization of areas under precision-recall curves with provable convergence. *Advances in Neural Information Processing Systems*, 34, 2021.

[15] Hanlin Tang, Xiangru Lian, Shuang Qiu, Lei Yuan, Ce Zhang, Tong Zhang, and Ji Liu. Deepsqueeze: Decentralization meets error-compensated compression. *arXiv preprint arXiv:1907.07346*, 2019.

[16] Guanghui Wang, Ming Yang, Lijun Zhang, and Tianbao Yang. Momentum accelerates the convergence of stochastic auprc maximization. *arXiv preprint arXiv:2107.01173*, 2021.

[17] 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.

[18] Mengdi Wang, Ji Liu, and Ethan X Fang. Accelerating stochastic composition optimization. *The Journal of Machine Learning Research*, 18(1):3721–3743, 2017.

[19] Yiming Ying, Longyin Wen, and Siwei Lyu. Stochastic online auc maximization. *Advances in neural information processing systems*, 29:451–459, 2016.

[20] Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. *arXiv preprint arXiv:1905.03817*, 2019.

[21] Huizhuo Yuan, Xiangru Lian, and Ji Liu. Stochastic recursive variance reduction for efficient smooth non-convex compositional optimization. *arXiv preprint arXiv:1912.13515*, 2019.

[22] 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.

[23] Junyu Zhang and Lin Xiao. A composite randomized incremental gradient method. In *International Conference on Machine Learning*, pages 7454–7462, 2019.

[24] Junyu Zhang and Lin Xiao. A stochastic composite gradient method with incremental variance reduction. In *Advances in Neural Information Processing Systems*, pages 9078–9088, 2019.

[25] Xinwen Zhang, Yihan Zhang, Tianbao Yang, Richard Souvenir, and Hongchang Gao. Federated compositional deep auc maximization. *arXiv preprint arXiv:2304.10101*, 2023.