Communication-Efficient Stochastic Gradient Descent Ascent with Momentum Algorithms

Yihan Zhang¹, Meikang Qiu² and Hongchang Gao¹∗

¹Temple University, Philadelphia, PA, USA
²Dakota State University, Madison, SD, USA
yihan.zhang0002@temple.edu, qiumeikang@ieee.org, hongchang.gao@temple.edu

Abstract

Numerous machine learning models can be formulated as a stochastic minimax optimization problem, such as imbalanced data classification with AUC maximization. Developing efficient algorithms to optimize such kinds of problems is of importance and necessity. However, most existing algorithms restrict their focus on the single-machine setting so that they are incapable of dealing with the large communication overhead in a distributed training system. Moreover, most existing communication-efficient optimization algorithms only focus on the traditional minimization problem, failing to handle the minimax optimization problem. To address these challenging issues, in this paper, we develop two novel communication-efficient stochastic gradient descent ascent with momentum algorithms for the distributed minimax optimization problem, which can significantly reduce the communication cost via the two-way compression scheme. However, the compressed momentum makes it considerably challenging to investigate the convergence rate of our algorithms, especially in the presence of the interaction between the minimization and maximization subproblems. In this paper, we successfully addressed these challenges and established the convergence rate of our algorithms for nonconvex-strongly-concave problems. To the best of our knowledge, our algorithms are the first communication-efficient algorithm with theoretical guarantees for the minimax optimization problem. Finally, we apply our algorithm to the distributed AUC maximization problem for the imbalanced data classification task. Extensive experimental results confirm the efficacy of our algorithm in saving communication cost.

1 Introduction

Recently, the stochastic minimax optimization problem has been attracting increasing attention since numerous machine learning models can be formulated as a minimax optimization problem. For instance, the adversarial training paradigm [Goodfellow et al., 2014; Madry et al., 2017] solves the maximization and minimization subproblems alternately to obtain a robust machine learning model. The AUC maximization problem is formulated as a minimax optimization problem in [Ying et al., 2016] to facilitate stochastic training. Meanwhile, with the emergence of distributed data in real-world machine learning applications, efficiently solving large-scale stochastic minimax optimization problems becomes an open challenge. In this paper, we focus on developing efficient optimization algorithms to solve the following stochastic minimax optimization problem:

\[
\min_{x \in \mathbb{R}^d} \max_{y \in \mathbb{R}^d} f(x, y) = \frac{1}{K} \sum_{k=1}^{K} f^{(k)}(x, y),
\]

where \( K \) is the number of workers, \( f^{(k)}(x, y) = \mathbb{E}_{\xi \sim D^{(k)}} [f(x, y; \xi)] \) is the loss function on the \( k \)-th worker and \( D^{(k)} \) denotes the dataset on the \( k \)-th worker. In this paper, we assume \( f^{(k)}(x, y) \) is nonconvex regarding \( x \) and strongly-concave regarding \( y \).

A typical application of Eq. (1) is the imbalanced data classification task. Specifically, the data in many machine learning applications is imbalanced, where the number of positive samples is extraordinarily different from that of negative samples. For instance, in the click-through rate (CTR) prediction task, there are much fewer positive samples than negative samples. It is challenging to learn a well-performing classifier with such kinds of imbalanced data. Recently, to address this issue, a line of research is to directly optimize the Area-Under-the-ROC-Curve (AUC) score, rather than the cross-entropy loss function. Specifically, [Ying et al., 2016] developed the following minimax loss function for the AUC maximization problem:

\[
\min_{w, \hat{w}_1, \hat{w}_2} \max_{\theta, a, b} \mathcal{L}(w, \hat{w}_1, \hat{w}_2, \theta; a, b)
\triangleq (1-p)(f(w; a) - \hat{w}_1)^2I_{[b=1]}
+ p(f(w; a) - \hat{w}_2)^2I_{[b=-1]} - p(1-p)\theta^2
+ 2(1+\theta)(pf(w; a)I_{[b=-1]} - (1-p)f(w; a)I_{[b=1]}),
\]

where \( w \in \mathbb{R}^d \) denotes the model parameter of the classifier \( f \), \( \hat{w}_1 \in \mathbb{R}, \hat{w}_2 \in \mathbb{R}, \theta \in \mathbb{R} \) are the additional parameters.

∗Corresponding author
for computing AUC score, $(a, b)$ represents the sample’s feature and label, $p$ denotes the prior probability of the positive class, and $I$ is an indicator function. Here, when the classifier $f$ is nonconvex, such as a deep neural network, Eq. (2) is a nonconvex-strongly-concave problem.

To solve stochastic minimax optimization problems, a lot of efforts have been made in the past few years. In particular, numerous stochastic gradient descent ascent (SGDA) algorithms [Lin et al., 2020; Zhang et al., 2020; Qiu et al., 2020; Yan et al., 2020; Yang et al., 2020; Chen et al., 2021] have been proposed. For instance, [Lin et al., 2020] developed mini-batch SGDA and established its convergence rate for nonconvex-strongly-concave problems. However, this method requires a large batch size. Thus, it is not practical for real-world machine learning applications. To address this issue, [Qiu et al., 2020] developed a momentum SGDA algorithm, which only needs a small constant batch size. Furthermore, a couple of accelerated algorithms [Huang et al., 2020; Luo et al., 2020; Qiu et al., 2020] have been proposed by incorporating the variance reduction techniques [Cutkosky and Orabona, 2019; Fang et al., 2018]. However, all of these algorithms ignore the distributed setting. They cannot be directly leveraged to solve Eq. (1) due to the unique challenges, such as the communication issue, in the distributed setting.

Different from the single-machine setting, the workers in a distributed training system should communicate frequently with the central server to communicate stochastic gradients. When the model is large, i.e., $x$ and $y$ are with high dimensionality, the incurred communication cost will lead to significant performance bottleneck [Gao et al., 2023; Qiu et al., 2019]. In recent years, to alleviate the large communication cost issue, a large number of methods have been proposed. For instance, [Alistarh et al., 2017; Wen et al., 2017] proposed to compress the stochastic gradient for reducing the communication cost. [Stich et al., 2018; Karimireddy et al., 2019] developed the error-feedback strategy to improve the convergence performance of compressed gradient algorithms. [Tang et al., 2019; Zheng et al., 2019] proposed the two-way compression strategy to compress the uplink and downlink gradient. Recently, [Richtárik et al., 2021] developed a recursive compressor such that the compression error could be shrunk in the course of training.

However, all aforementioned communication-efficient algorithms only focus on the minimization problem. It’s unclear whether those techniques still work for the minimax problem. 1) On the algorithmic design side, each worker in Eq. (1) has to communicate the stochastic gradient regarding $x$ and that regarding $y$ with the central server. How to compress those two stochastic gradients such that the communication cost is reduced and the convergence performance is not impaired has not been explored yet. 2) On the theoretical analysis side, how the compressed gradient algorithm for Eq. (1) affects the convergence has not been investigated. Especially, when the momentum technique and the compression technique are employed simultaneously, how they affect the convergence rate has not been studied. In fact, it is much more challenging to establish the convergence rate compared with the minimization problem due to the interaction between those techniques in both minimization and maximization subproblems. Thus, it is necessary to develop communication-efficient algorithms with theoretical guarantees to solve Eq. (1).

In this paper, to address aforementioned challenges, we developed two novel communication-efficient stochastic gradient descent ascent algorithms with momentum algorithms. Specifically, on the algorithmic design side, our first algorithm compresses the momentum, rather than stochastic gradients, in both worker-to-server and server-to-worker directions by employing the plain error-feedback compression scheme [Karimireddy et al., 2019]. Our second algorithm employs the recursive error-feedback compression mechanism [Richtárik et al., 2021] for compressing the momentum communicated in both directions. As such, the communication cost can be reduced significantly. On the theoretical analysis side, we proposed novel theoretical analysis techniques to establish the convergence rate of our algorithms. Importantly, our theoretical results demonstrate how the compression operator and the number of devices affect the convergence rate. To the best of our knowledge, this is the first work to develop communication-efficient algorithm with theoretical guarantees for solving the distributed minimax optimization problem. At last, we apply our algorithm to solve the AUC maximization problem in Eq. (2) for the distributed imbalanced classification task. The extensive experimental results confirm the efficacy of our algorithms in saving communication cost and its effectiveness in preserving the convergence performance. In summary, we made the following contributions in this paper:

- We developed two novel communication-efficient algorithms for optimizing distributed minimax optimization problems. This is the first work studying how to reduce the communication cost for minimax problems.
- We established the convergence rate of our two algorithms, theoretically demonstrating how the compression operator and the number of workers affect convergence rates.
- We conducted extensive experiments on the imbalanced classification task, which confirms the effectiveness of our algorithms.

2 Related Works

2.1 Stochastic Minimax Optimization Algorithms

In machine learning, a large number of models can be formulated as the stochastic minimax optimization problem. A typical example is the adversarial learning model [Goodfellow et al., 2014; Madry et al., 2017], which has been widely applied in a wide variety of data mining and machine learning applications. Due to the extensive application of stochastic minimax optimization problem in machine learning, developing efficient optimization algorithms for this problem has attracted a surge of attention in the past few years. As such, a large number of algorithms have been proposed. For instance, [Lin et al., 2020] leveraged stochastic gradients to solve the maximization and minimization subproblems, and established its convergence rate for nonconvex-strongly-concave problems. This convergence rate is further improved in [Chen et al., 2021] by assuming that the second moment.
of stochastic gradients is bounded. Furthermore, to accelerate the convergence rate of SGDA, [Qiu et al., 2020] developed two momentum-based algorithms by exploiting the moving-average strategy and the STORM strategy [Cutkosky and Orabona, 2019], respectively. Meanwhile, [Luo et al., 2020] exploited the SPIDER variance-reduced gradient [Fang et al., 2018] to accelerate SGDA and achieve a better convergence rate than the standard SGDA for nonconvex-strongly-concave problems.

As for the AUC maximization problem, traditional methods typically employ a surrogate function, which depends on a pair of training samples. As such, it is not friendly to the stochastic training. To address this problem, [Ying et al., 2016] reformulated it as a minimax loss function, which can be decomposed into a sum of loss functions regarding individual samples. As a result, it can be optimized by stochastic gradient algorithms. Based on this reformulated minimax loss function, a couple of algorithms have been proposed for AUC maximization. For instance, [Ying et al., 2016] developed a stochastic online algorithm and established its convergence rate for convex-concave problems. However, [Ying et al., 2016] assumes that the classifier is a linear function, which is too restrictive to be applied to practical machine learning applications. Later, [Liu et al., 2019] extended it to deep neural networks so that the loss function becomes nonconvex-strongly-concave. Then, they developed the stage-wise primal-dual stochastic gradient algorithm and established its convergence rate based on the Polyak-Łojasiewicz (PL) condition. However, all these algorithms just focus on the single-machine setting so that they are not able to handle the communication challenges in the distributed setting.

2.2 Communication-Efficient Distributed Optimization Algorithms
Under the distributed setting, a major concern is the large communication cost caused by the communication between workers and the central server. In the past few years, much progress has gone towards designing communication-efficient algorithms. The basic idea is to compress the gradient such that fewer bits are demanded in the communication step. Based on this strategy, a large number of communication-efficient stochastic gradient descent (SGD) algorithms [Jiang and Agrawal, 2018; Alistarh et al., 2017; Wen et al., 2017; Gao et al., 2021; Ivkin et al., 2019; Wangni et al., 2018; Gorbunov et al., 2020; Gupta et al., 2021] have been proposed. For instance, [Wen et al., 2017] developed the TernGrad algorithm, which quantizes gradients to ternary levels so that the communication cost can be reduced significantly. [Bernstein et al., 2018] proposed a more aggressive algorithm, which just communicates the sign of gradient entries. However, these compression strategies introduce a large gradient variance, which can impair the convergence performance. To address this problem, the plain error-feedback compression strategy was introduced in [Seide et al., 2014; Stich et al., 2018; Karimireddy et al., 2019]. It aims to reduce the gradient bias by compensating the compression error so that the convergence performance of compressed gradient algorithms can match that of full-precision counterparts. Recently, [Richtárik et al., 2021] developed a new recursive error-feedback compression scheme, which enjoys the contractive compression error property and demonstrates superior performance in practice. However, all these methods only investigate the minimization problem.

Regarding the distributed minimax optimization problem, a few of works have been proposed in recent years. For instance, [Xian et al., 2021; Zhang et al., 2021; Gao, 2022; Zhang et al., 2023b] developed decentralized stochastic variance-reduced gradient descent algorithms where workers perform peer-to-peer communication. On the other hand, [Deng and Mahdavi, 2017; Tarzanaugh et al., 2022; Sharma et al., 2022] developed a federated stochastic gradient descent ascent algorithm for Federated Learning. Moreover, [Guo et al., 2020; Yuan et al., 2021; Zhang et al., 2023a] studied the AUC maximization problem under the federated learning setting. These works are orthogonal to our setting because they reduce the communication cost by skipping the communication round, rather than compressing gradients. In summary, designing communication-efficient algorithms for optimizing Eq. (1) is still an open challenging problem.

3 Methodology
3.1 Problem Setup
The gradient compression technique has been widely studied in recent years. Typically, the compression operator satisfies the following property.

**Definition 1.** A compression operator \( \mathcal{C} : \mathbb{R}^d \rightarrow \mathbb{R}^d \) is \( \alpha \)-contraction if there exists \( \alpha \in (0, 1) \) such that

\[
||x - \mathcal{C}(x)||^2 \leq (1 - \alpha)||x||^2. \tag{3}
\]

The commonly used compression operators that enjoy the \( \alpha \)-contraction property include Top-\( k \) operator [Stich et al., 2018] and the scaled sign operator [Karimireddy et al., 2019].

To investigate the convergence rate of our algorithm, we assume the loss function satisfies the following assumptions, which are also commonly used in existing minimax optimization works [Lin et al., 2020; Luo et al., 2020; Huang et al., 2020; Qiu et al., 2020].

**Assumption 1.** The loss function \( f^{(k)} \) on the \( k \)-th worker is \( L \)-smooth, i.e., there exists a constant value \( L > 0 \) such that

\[
\|\nabla_x f^{(k)}(z_1) - \nabla_x f^{(k)}(z_2)\| \leq L \|z_1 - z_2\|, \tag{4}
\]

\[
\|\nabla_y f^{(k)}(z_1) - \nabla_y f^{(k)}(z_2)\| \leq L \|z_1 - z_2\|,
\]

for all \( z_1 = (x_1, y_1) \in \mathbb{R}^d \times \mathbb{R}^d \), \( z_2 = (x_2, y_2) \in \mathbb{R}^d \times \mathbb{R}^d \).

**Assumption 2.** The stochastic gradient \( \nabla_x f^{(k)}(x, y; \xi) \) and \( \nabla_y f^{(k)}(x, y; \xi) \) have bounded variances, i.e., there exist constant values \( \sigma_x > 0 \) and \( \sigma_y > 0 \) such that

\[
\mathbb{E}[\|\nabla_x f^{(k)}(x, y; \xi) - \nabla_x f^{(k)}(x, y)\|^2] \leq \sigma_x^2,
\]

\[
\mathbb{E}[\|\nabla_y f^{(k)}(x, y; \xi) - \nabla_y f^{(k)}(x, y)\|^2] \leq \sigma_y^2,
\]

for all \((x, y) \in \mathbb{R}^d \times \mathbb{R}^d \).

**Assumption 3.** The loss function \( f^{(k)} \) is \( \mu \)-strongly-concave with respect to \( y \), i.e., there exists a constant value \( \mu > 0 \) such
that
\[ f^{(k)}(x, y_1) \leq f^{(k)}(x, y_2) + \langle \nabla_y f^{(k)}(x, y_2), y_1 - y_2 \rangle + \frac{\lambda}{2} \| y_1 - y_2 \|^2 , \] (6)

for all \( y_i \in \mathbb{R}^d \times \mathbb{R}^d \), \( \forall (x, y_1) \in \mathbb{R}^d \times \mathbb{R}^d \).

Algorithm 1 SGDAM-PEF

**Input:** \( \eta > 0 \), \( \gamma > 0 \), \( \lambda > 0 \), \( \rho_1 > 0 \), \( \rho_2 > 0 \), \( r_0 = 0 \), \( s_0 = 0 \).

1: for \( t = 0, \cdots, T - 1 \) do
2: \hspace{1em} Worker-\( k \):
3: \hspace{2em} if \( t = 0 \), then
4: \hspace{3em} \( m_0^{(k)} = \nabla_x f^{(k)}(x_0, y_0; s_0^{(k)}) \), \( \phi_0^{(k)} = 0 \),
5: \hspace{3em} else
6: \hspace{4em} \( m_t^{(k)} = (1 - \rho_1 \eta)m_{t-1}^{(k)} + \rho_1 \eta \nabla_x f^{(k)}(x_t, y_t; \xi_t^{(k)}) \),
7: \hspace{4em} end if
8: \hspace{3em} \( p_t^{(k)} = m_t^{(k)} + \phi_t^{(k)}\), \( u_{t+1}^{(k)} = p_t^{(k)} - C(p_t^{(k)}) \),
9: \hspace{3em} Upload \( C(p_t^{(k)}) \) to the central server.
10: \hspace{2em} end if
11: \hspace{2em} receive \( x_t \) and \( y_t \) from the server
12: \hspace{1em} end for
13: \hspace{2em} if \( t = 0 \), then
14: \hspace{3em} \( h_0^{(k)} = \nabla_y f^{(k)}(x_0, y_0; \xi_0^{(k)}) \), \( v_0^{(k)} = 0 \),
15: \hspace{3em} else
16: \hspace{4em} \( h_t^{(k)} = (1 - \rho_2 \eta)h_{t-1}^{(k)} + \rho_2 \eta \nabla_y f^{(k)}(x_t, y_t; \xi_t^{(k)}) \),
17: \hspace{4em} end if
18: \hspace{3em} \( q_t^{(k)} = h_t^{(k)} + v_t^{(k)}\), \( v_{t+1}^{(k)} = q_t^{(k)} \),
19: \hspace{3em} end if
20: \hspace{2em} end for

\[ m_t^{(k)} = (1 - \rho_1 \eta)m_{t-1}^{(k)} + \rho_1 \eta \nabla_x f^{(k)}(x_t, y_t; \xi_t^{(k)}) , \] (7)

**Algorithm 2 SGDAM-REF**

**Input:** \( \eta > 0 \), \( \gamma > 0 \), \( \lambda > 0 \), \( \rho_1 > 0 \), \( \rho_2 > 0 \), \( \omega_0 = 0 \), \( \omega_0 = 0 \), \( \delta_0 = 0 \).

1: for \( t = 0, \cdots, T - 1 \) do
2: \hspace{1em} Worker-\( k \):
3: \hspace{2em} receive \( x_t \) and \( y_t \) from the server
4: \hspace{2em} if \( t = 0 \), then
5: \hspace{3em} \( m_0^{(k)} = \nabla_x f^{(k)}(x_0, y_0; s_0^{(k)}) \), \( u_0^{(k)} = 0 \),
6: \hspace{3em} else
7: \hspace{4em} \( m_t^{(k)} = (1 - \rho_1 \eta)m_{t-1}^{(k)} + \rho_1 \eta \nabla_x f^{(k)}(x_t, y_t; \xi_t^{(k)}) \),
8: \hspace{4em} end if
9: \hspace{3em} \( p_t^{(k)} = C(m_t^{(k)} - u_t^{(k)})\), \( u_{t+1}^{(k)} = u_t^{(k)} + p_t^{(k)} \),
10: \hspace{3em} Upload \( p_t^{(k)} \) to the central server.
11: \hspace{2em} end if
12: \hspace{2em} if \( t = 0 \), then
13: \hspace{3em} \( h_0^{(k)} = \nabla_y f^{(k)}(x_0, y_0; \xi_0^{(k)}) \), \( v_0^{(k)} = 0 \),
14: \hspace{3em} else
15: \hspace{4em} \( h_t^{(k)} = (1 - \rho_2 \eta)h_{t-1}^{(k)} + \rho_2 \eta \nabla_y f^{(k)}(x_t, y_t; \xi_t^{(k)}) \),
16: \hspace{4em} end if
17: \hspace{3em} \( q_t^{(k)} = h_t^{(k)} - v_t^{(k)}\), \( v_{t+1}^{(k)} = q_t^{(k)} + q_t^{(k)} \),
18: \hspace{3em} Upload \( q_t^{(k)} \) to the central server.
19: \hspace{2em} end if
20: \hspace{2em} Server:
21: \hspace{3em} \( u_{t+1}^{(k)} = u_t^{(k)} + \frac{1}{K} \sum_{k = 1}^{K} p_t^{(k)} \),
22: \hspace{3em} \( v_{t+1}^{(k)} = v_t^{(k)} + \frac{1}{K} \sum_{k = 1}^{K} q_t^{(k)} \),
23: \hspace{3em} \( r_{t+1}^{(k)} = C(u_{t+1}^{(k)} - \hat{u}_t)\), \( u_{t+1}^{(k)} = \hat{u}_t + r_{t+1}^{(k)} \),
24: \hspace{3em} \( s_{t+1}^{(k)} = C(v_{t+1}^{(k)} - \hat{v}_t)\), \( v_{t+1}^{(k)} = \hat{v}_t + s_{t+1}^{(k)} \),
25: \hspace{3em} Broadcast \( r_{t+1}^{(k)} \) and \( s_{t+1}^{(k)} \) to all workers.
26: \hspace{2em} Worker-\( k \):
27: \hspace{3em} \( x_{t+1}^{(k)} = x_t + \gamma \eta C(u_t^{(k)})\), \( y_{t+1}^{(k)} = y_t + \lambda \eta C(v_t^{(k)})\).
28: end for

3.2 Communication-Efficient Stochastic Gradient Descent Ascent with Momentum Algorithms

In this paper, we focus on the stochastic gradient descent ascent with momentum algorithm, where the momentum stochastic gradient is employed to update the minimization and maximization subproblems. To reduce the communication cost, we proposed two communication-efficient stochastic gradient descent ascent with momentum algorithms. In particular, in Algorithm 1, we developed the communication-efficient stochastic gradient descent ascent with momentum algorithm, i.e., SGDAM-PEF, which employs the plain error-feedback technique to compress the momentum in two directions. In Algorithm 2, we proposed the SGDAM-REF algorithm, which leverages the recursive error-feedback technique to compress the momentum in two directions.

Algorithm 1. SGDAM-PEF

Both algorithms exploits the momentum stochastic gradient to update model parameters. For instance, for the minimization subproblem with respect to \( x \), at the \( t \)-th iteration, each worker \( k \) computes the momentum \( m_t^{(k)} \) based on the stochastic gradient as follows:

\[ m_t^{(k)} = (1 - \rho_1 \eta)m_{t-1}^{(k)} + \rho_1 \eta \nabla_x f^{(k)}(x_t, y_t; \xi_t^{(k)}) , \] (7)

where \( \rho_1 \) and \( \eta \) are two positive hyperparameters such that \( \rho_1 \eta < 1 \). \( \xi_t^{(k)} \) denotes the randomly selected samples from the local dataset on the \( k \)-th worker. Note that the model parameter \( x_t^{(k)} \) on the \( k \)-th worker is the same with other workers due to the synchronization across all workers. Thus, we omit the superscript of \( x_t^{(k)} \) and \( y_t^{(k)} \) throughout this paper.

Algorithm 1 employs the following error-feedback scheme to compress the momentum \( m_t^{(k)} \):

\[ p_t^{(k)} = m_t^{(k)} + \phi_t^{(k)}\), \( \phi_{t+1}^{(k)} = p_t^{(k)} - C(p_t^{(k)}) , \] (8)

where \( \phi_{t+1}^{(k)} \) denotes the residual error between the full-precision momentum \( p_t^{(k)} \), which is the original momentum \( m_t^{(k)} \) corrected by the residual error \( \phi_t^{(k)} \) in the prior iteration, and the compressed momentum \( C(p_t^{(k)}) \). Then, \( C(p_t^{(k)}) \) is uploaded to the central server and thus the communication cost is reduced. With such an error-feedback mechanism, we can control the bias caused by the compression operation to improve the convergence.

To reduce the communication cost when broadcasting the global momentum to all workers, we also compress the global
momentum with the same error-feedback compression mechanism to get the compressed global momentum $C(u_t)$, which is shown in Line 15 of Algorithm 1. Then, each worker exploits this global momentum to update its model parameters. As for the model parameter $y$, our algorithm leverages the same compression technique to reduce communication cost.

It can be observed that our algorithm compresses the momentum of the minimization and maximization subproblems, rather than the stochastic gradient, in both worker-to-server and server-to-worker directions to reduce the communication overhead. To the best of our knowledge, our work is the first one applying this technique to the minimax optimization algorithm, especially the compression of the momentum, which is much more challenging. Specifically, the minimax structure and momentum cause significant challenges to investigate the convergence rate. We addressed these challenges and established the convergence rate in Section 4.

Algorithm 2. SGDAM-REF

However, the plain error-feedback mechanism in Algorithm 1 cannot guarantee the residual error converges to zero [Richtárik et al., 2021]. Therefore, in Algorithm 2, we resort to the recursive error-feedback compression strategy, which is first proposed for the minimization problem in [Richtárik et al., 2021], to compress momentum $m_t^{(k)}$ as follows:

$$p_t^{(k)} = C(m_t^{(k)} - u_t^{(k)}) , u_{t+1}^{(k)} = u_t^{(k)} + p_t^{(k)},$$

(9)

where $u_{t+1}^{(k)}$ can be viewed as an approximation to momentum $m_t^{(k)}$. It can be observed that this compression mechanism compresses the difference between the original momentum $m_t^{(k)}$ and the approximated one $u_t^{(k)}$ in the prior iteration. With this compression strategy, $u_{t+1}^{(k)}$ will converge to $m_t^{(k)}$, which will be shown in our theoretical analysis. Here, $p_t^{(k)}$ is uploaded to the central server and the communication cost is reduced. Similarly, the maximization subproblem with respect to $y$ also follows the same strategy to compute the local momentum $h_t^{(k)}$ and compress it with the recursive compression strategy to obtain $v_{t+1}^{(k)}$.

As for the central server, when it receives $p_t^{(k)}$ from all workers, it computes the average $\hat{u}_{t+1}^{(k)} = \hat{u}_t^{(k)} + \frac{1}{K} \sum_{k=1}^{K} p_t^{(k)}$ and compresses it with the same recursive compression strategy, which is shown in Line 16 of Algorithm 2. Then, the server broadcasts $r_{t+1}$ to all workers. Similarly, the server applies the same procedure to the maximization subproblem. As such, the communication cost in both worker-to-server and server-to-worker directions are reduced significantly.

After receiving the global $r_{t+1}$ from the central server, each worker $k$ updates its local $\hat{u}_{t+1}^{(k)}$ and exploits it to update the local model parameter as follows:

$$\bar{u}_{t+1}^{(k)} = \hat{u}_t^{(k)} + r_{t+1}^{(k)} , \quad x_{t+1} = x_t - \gamma \eta \bar{u}_{t+1}^{(k)},$$

(10)

where $\gamma > 0$ is a hyperparameter. It is worth noting that all workers and the central server maintain an identical sequence $\{\hat{u}_t\}$, since $u_0 = 0$ and $r_t$ is shared by all workers and the central server. In addition, due to the employed recursive compression strategy, $\bar{u}_{t+1}$ will converge to $\bar{u}_{t+1}$, while $\bar{u}_{t+1}$ will approach to the global momentum $\frac{1}{K} \sum_{k=1}^{K} m_t^{(k)}$. Thus, $\bar{u}_{t+1}$ is an approximation to the global momentum. As such, each worker leverages the compressed global momentum to update its model parameters. Regarding the maximization subproblem with respect to $y$, our algorithm exploits the same strategy to update it.

Note that the recursive compression strategy is first proposed in [Richtárik et al., 2021]. However, our algorithm is extraordinarily different from it. First, [Richtárik et al., 2021] studies the full gradient descent algorithm. Our algorithm focuses on stochastic gradients. Additionally, [Richtárik et al., 2021] compresses gradients, while our algorithm compresses the momentum, which is much more challenging. Second, [Richtárik et al., 2021] just compresses the gradient sent from workers to the central server. On the contrary, our algorithm performs compression on both directions. Thus, our algorithm is able to save much more communication cost. Meanwhile, the two-way compression makes it much more challenging to establish the convergence rate of our algorithm. At last, [Richtárik et al., 2021] established the convergence rate for minimization problem. As such, their convergence analysis does not hold for our minimax optimization problem. All in all, our algorithm is significantly different from [Richtárik et al., 2021] and it is much more challenging to establish the convergence rate of our algorithm due to the employed two-way compression and momentum techniques.

4 Theoretical Analysis

To investigate the convergence rate of our algorithm, we first introduce two auxiliary functions: $\Phi(x) = \max y f(x, y)$ and $y^*(x) = \arg \max_y f(x, y)$. Then, based on Assumption 1, it is easy to get that $\Phi(x)$ is $L_F$-smooth where $L_F = 2L^2/\mu$ [Lin et al., 2020]. Moreover, we introduce an additional assumption about the compression operator in the following, which has been commonly used in existing works [Alistarh et al., 2018; Li and Li, 2022; Haddadpour et al., 2021], e.g., Assumption 1 of [Alistarh et al., 2018] and Assumption 3 of [Li and Li, 2022].

Assumption 4. The compression operator $C : \mathbb{R}^d \rightarrow \mathbb{R}^d$ satisfies the following condition:

$$\|\frac{1}{K} \sum_{k=1}^{K} a^{(k)} - \frac{1}{K} \sum_{k=1}^{K} C(a^{(k)})\|^2 \leq (1 - \alpha) \|\frac{1}{K} \sum_{k=1}^{K} a^{(k)}\|^2,$$

(11)

where $\alpha \in (0, 1)$ and $a \in \mathbb{R}^d$.

Then, based on the aforementioned assumptions and auxiliary functions, we establish the convergence rate of our algorithm for nonconvex-strongly-concave problems.

Theorem 1. Given Assumptions 1-4, by setting $\rho_1 > 0, \rho_2 > 0, \eta < \min\{\frac{1}{2L^2 \mu^2}, \frac{1}{\rho_1}, \frac{1}{\rho_2}, 1\}$, and

$$\gamma \leq \min\{\frac{\lambda \mu^2}{12L^2}, \frac{\alpha^2 \mu}{4L^2 \sqrt{128/\rho_1^2 + 3240 + 2135/\rho_2^2}}\} ,$$

$$\lambda \leq \min\{\frac{1}{6L}, \frac{3 \mu^4}{L^2 (128/\rho_1 + 2134/\rho_2 + 2544)}\}.$$
Algorithm 1 has the following convergence rate

\[
\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}[\|\nabla \Phi(x_t)\|^2] \leq \frac{4(\Phi(x_0) - \Phi(x_\ast))}{(\ln T \gamma_\ast) T} + \frac{16L^2}{\lambda \mu T} \|y_0 - y^\ast(x_0)\|^2 + \frac{400L^2 \sigma_0^2}{3 \mu^2 \rho_0 \gamma_\ast TK} + \frac{400 \rho_0 \gamma_\ast k \mu^2 L^2}{3 \mu^2 K} + \frac{8 \sigma_z^2}{\rho_1 \eta T K} + \frac{8 \rho_1 \eta \sigma_x^2}{K},
\]

where \(x_\ast\) represents the optimal solution.

**Remark 1.** From Theorem 1, it can be observed that \(\gamma = O(\alpha^4), \lambda = O(\alpha^4), \rho_1 = O(1), \) and \(\rho_2 = O(1).\)

**Remark 2.** In terms of Theorem 1, by setting \(\eta = O(Kc^2), T = O(\frac{\alpha^{-4}}{K^{3/2}}),\) Algorithm 1 can achieve the \(c\)-accuracy solution, i.e., \(\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}[\|\nabla \Phi(x_t)\|^2] \leq c^2.\) The dependence on \(\alpha\) indicates that the compression operation increases the number of iterations. To the best of our knowledge, this is the first algorithm disclosing how the compression operation affects the convergence rate of minimax optimization algorithms. When there is no compression operation, i.e., \(\alpha = 1,\) we can get the iteration complexity \(O(\frac{1}{\alpha T})\), which indicates that our algorithm can achieve linear speedup with respect to the number of devices. Additionally, compared with Theorem 1, the learning rate has an additional dependence on \(\alpha,\) resulting a larger iteration complexity \(O(\frac{\alpha^{-4}}{K^{3/2}})\) than \(O(\frac{\alpha^{-4}}{K^{3/2}})\) of Theorem 1.

In summary, we established the convergence rate for our two algorithms, disclosing how the compression operation and the number of devices affect the convergence rate. To the best of our knowledge, this is the first work achieving these theoretical results. In fact, it is challenging to establish these convergence rates. Specifically, our algorithms compress the momentum on both directions so that the interaction among the momentum technique, compression scheme, and two subproblems make it difficult to study how the function value and the compression error evolves across iterations. We developed novel theoretical analysis strategies, e.g., a novel potential function for Algorithm 2, to establish the convergence rate. All in all, establishing the convergence rate of our two algorithms is challenging. Our new theoretical analysis strategies are novel and can benefit other distributed minimax optimization, such as federated minimax optimization.

5 Experiments

5.1 Experimental Setup

In our experiments, we apply our two algorithms to the distributed AUC maximization problem for imbalanced data classification.

Datasets. In our experiments, five benchmark datasets are employed to evaluate the performance of our algorithm. They are CATvsDOG \(^1\), CIFAR10, CIFAR100 \(^2\), STL10 [Coates et al., 2011], Melanoma [Rotemberg et al., 2021]. For the first four datasets, we partition each dataset into two groups according to its classes. Specifically, the first half of classes are viewed as the positive class, while the second half of classes are viewed as the negative class. Then, we randomly drop some samples from the positive class in the training set such that the ratio between positive samples and all samples is 0.1. As such, the first four datasets are imbalanced binary classification datasets. The statistics of all datasets are shown in Appendix. Then, the training set is randomly distributed to all workers, while the testing set are the same for all workers.

Experimental settings. To evaluate the performance of our algorithm, we compare it with the full-precision stochastic gradient descent with momentum algorithm (SGDM), which is to optimize the cross-entropy loss function, the full-precision stochastic gradient descent ascent with momentum algorithm (SGDAM) [Qiu et al., 2020], which is to optimize the AUC loss function, and the compressed SGDAM without the error-feedback technique (SGDAM-NEF). To make a fair comparison, we set \(T = \frac{2}{\eta^2}\) with the function value in the training set, where \(\eta^2\) is the number of devices. We use the following categorical datasets \(\text{CATvsDOG}^1, \text{CIFAR10}, \text{CIFAR100}^2, \text{STL10} [\text{Coates et al.}, 2011], \text{Melanoma} [\text{Rotemberg et al.}, 2021].\)

\(^1\)https://www.kaggle.com/c/dogs-vs-cats
\(^2\)https://www.cs.toronto.edu/~kriz/cifar.html
In this paper, we developed two novel communication-efficient stochastic gradient descent ascent algorithms for distributed minimax optimization problems. This is the first work to demonstrate how to reduce the communication cost of minimax optimization algorithms. Moreover, we established the convergence rate, disclosing how the compression operator and the number of devices affect the convergence rate. Extensive experimental results confirm the effectiveness of our algorithms.

6 Conclusion

In Figure 1, we report the testing AUC score versus the number of epochs on testing sets. Here we use four workers where each worker is a V100-GPU. From Figure 1, we have the following observations. 1) Our two algorithms with the error-feedback technique significantly outperform those without error-feedback. 2) Our SGDAM-PEF and SGDAM-REF can achieve almost the same AUC score, which means the empirical performance of two error-feedback strategies does not have significant difference. 3) Our algorithms with Top-k compressor perform better than the variants with Rand-k compressor. The possible reason for this phenomenon is that the Rand-k compressor discards too much informative gradient components. When using Top-k compressor, our two algorithms can achieve almost the same AUC score as the full-precision SGDAM. In Figure 2, we plot the testing AUC score versus the number of communicated megabytes. The same experimental settings are used as Figure 1. It can be observed that our two algorithms with Top-k compressor achieve almost the same final testing performance as full-precision SGDAM under the condition of greatly reducing the communication cost, which confirms the efficacy of our algorithms in saving the communication cost and preserving the convergence performance.

To further demonstrate the communication efficiency of our algorithm, we use different compression ratios for our algorithms. The testing AUC score versus the consumed megabytes is shown in Figure 3. Here, due to the limitation of space, we only show the result of CIFAR10 and three different compression ratio: Top-20%, Top-10%, Top-5%, and Top-2%. From Figure 3, we can observe our algorithms can achieve almost the same final performance with full-precision SGDAM consistently, which means that they are robust to high compression ratios. For example, our algorithm SGDAM-REF-Top5 with 95% compression ratios still achieve almost the same AUC score compared with full-precision SGDAM on CIFAR10 dataset.
References


