Communication-Efficient Frank-Wolfe Algorithm for Nonconvex Decentralized Distributed Learning

Wenhan Xian [1], **Feihu Huang** [1], **Heng Huang** [1,2]
[1] Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, USA
[2] JD Finance America Corporation, Mountain View, CA, USA
wex37@pitt.edu, huangfeihu2018@gmail.com, heng.huang@pitt.edu

Abstract

Recently decentralized optimization attracts much attention in machine learning because it is more communication-efficient than the centralized fashion. Quantization is a promising method to reduce the communication cost via cutting down the budget of each single communication using the gradient compression. To further improve the communication efficiency, more recently, some quantized decentralized algorithms have been studied. However, the quantized decentralized algorithm for nonconvex constrained machine learning problems is still limited. Frank-Wolfe (a.k.a., conditional gradient or projection-free) method is very efficient to solve many constrained optimization tasks, such as low-rank or sparsity-constrained models training. In this paper, to fill the gap of decentralized quantized constrained optimization, we propose a novel communication-efficient Decentralized Quantized Stochastic Frank-Wolfe (DQSFW) algorithm for non-convex constrained learning models. We first design a new counterexample to show that the vanilla decentralized quantized stochastic Frank-Wolfe algorithm usually diverges. Thus, we propose DQSFW algorithm with the gradient tracking technique to guarantee the method will converge to the stationary point of non-convex optimization safely. In our theoretical analysis, we prove that to achieve the stationary point our DQSFW algorithm achieves the same gradient complexity as the standard stochastic Frank-Wolfe and centralized Frank-Wolfe algorithms, but has much less communication cost. Experiments on matrix completion and model compression applications demonstrate the efficiency of our new algorithm.

Introduction

Nowadays, many machine learning tasks have been deployed on distributed systems that enable computations in parallel, especially for large-scale models such as deep neural networks (DNNs). Recently, the centralized distribution structure has been used often. However, the centralized learning scheme has a key bottleneck of communication, where the communication burden of the central server becomes larger as the number of nodes grows. For example, when the system has M workers, it will suffer from the communication complexity of $O(M)$. Thus, the decentralized distribution structure recently has attracts much attention in machine learning due to its communication efficiency compared with the centralized fashion. Specifically, decentralized optimization adopts a pattern that each node maintains its own local data and model and only communicates with its neighbors. In fact, the communication complexity of decentralized system at each iteration is dependent on the degree of graph topology (usually not dependent on the number of nodes) and data locality is allowed.

Recently, the decentralized learning becomes a popular research topic in machine learning and has been widely studied. For example, it was first studied to solve problems of computing aggregates among clients or be used for the sake of data locality and privacy (Ram, Nedić, and Veeravalli 2010; Yan et al. 2012), where centralized structure is not allowed. A general decentralized algorithm can be traced back to (Nedić and Ozdaglar 2009) that combines gradient descent method and Gossip-type consensus step. Subsequently, a degenerated case of decentralization that achieves huge success is federated optimization (Konečný et al. 2016), which adopts star topology but enables data to be drawn from non-iid distribution. Besides, many other previous fully decentralized works such as (Lian et al. 2017; Tang et al. 2018) that based on general network topology have shown that decentralized method is able to achieve more efficient communication without sacrificing the training result, indicating that decentralization is becoming competitive and advantageous in distributed learning rather than just an alternate of centralization when centralization is not allowed. Lian et al. (2017) presented an important decentralized optimization work to verify that the decentralized method can outperform its centralized counterpart. Lian et al. (2017) proposed an algorithm named Decentralized Parallel Stochastic Gradient Descent (D-PSGD) to directly compute the averaging value among each node with exact communication, which has the same convergence rate as centralized SGD in nonconvex optimization with non-identical data distribution.

To reduce the communication cost in distributed system, gradient quantization (Seide et al. 2014) is another effective method. Recently, many quantized gradient algorithms, such as QSGD (Alistarh et al. 2017), signSGD (Bernstein et al. 2018a) and its variant (Bernstein et al. 2018b), were developed and showed excellent performance. In these algorithms, the number of bits transmitted in each communica-

tion round is reduced by packing and unpacking gradients. Alistarh et al. (2017) proposed an unbiased quantization scheme and proved it is capable to converge under convex and non-convex conditions. However, for other quantization method like 1-bit quantization or signSGD, the unbiased assumption is not always satisfied. Karimireddy et al. (2019) proved that when applying signSGD with a scalar factor and error-feedback technique, the algorithm is guaranteed to converge in non-convex optimization. More recently, to further achieve communication-efficiency, multiple quantized decentralized algorithms (Doan, Maguluri, and Romberg 2018; Reisizadeh et al. 2019a,b; Tang et al. 2019; Koloskova et al. 2020) have been introduced. However, to the best of our knowledge, the existing quantized decentralized algorithm for constrained problem is still very limited. In fact, the large-scale constrained optimization problems are popular in many machine learning applications, such as matrix completion and deep neural network compression.

To address this challenging issue, in this paper, we focus on studying the quantized decentralized algorithm for solving the following constraint optimization problem:

$$\min_{x\in\Omega} \frac{1}{M}\sum_{i=1}^{M} f_i(x), \tag{1}$$

where $f_i(x)$ is a nonconvex smooth loss function, Ω is a convex and compact constraint set, M is the number of worker nodes. $f_i(x)$ is the objective function on node i and could have the stochastic expectation or finite sum formulations:

$$f_i(x) = \begin{cases} \mathbb{E}_{\xi\sim D_i} F^{(i)}(x;\xi), & \text{stochastic} \\ \frac{1}{n_i}\sum_{j=1}^{n_i} F_j^{(i)}(x), & \text{finite-sum} \end{cases} \tag{2}$$

where D_i is the data distributed on i-th node. Apparently, finite-sum objective function is a particular case of stochastic problem where D_i consists of finite samples. We allow distributions D_i to be **non-identical**, which is more adaptive to general tasks in machine learning and is assumed in many previous decentralized analyses (Lian et al. 2017; Tang et al. 2018; Lian et al. 2018).

To solve the above constrained optimization, the Frank-Wolfe (a.k.a, conditional gradient or projection-free) method is one of the most efficient and popular algorithms, because the Frank-Wolfe method only requires to compute a linear oracle instead of the expensive projection operator applied in proximal gradient methods (Ghadimi, Lan, and Zhang 2016) and alternating direction method of multipliers (Huang, Chen, and Huang 2019). In this paper, thus, we focus on designing the communication-efficient quantized decentralized Frank-Wolfe algorithm to solve the above problem (1). It is nontrivial to design such an algorithm. We first provide a counterexample to show that the vanilla quantized decentralized Frank-Wolfe algorithm usually diverges (please see the following Counterexample section). Thus, there exists an important research problems to be addressed:

Can we design a communication-efficient quantized decentralized Frank-Wolfe algorithm with convergence guarantee for non-convex optimization?

In this paper, we answer the above challenging question with positive solution and propose a novel Decentralized Quantized Stochastic Frank-Wolfe (DQSFW) algorithm to solve the problem (1) using the gradient tracking technique to guarantee the DQSFW can safely and fast converge to the stationary point in non-convex optimization. Specifically, our DQSFW algorithm uses 1-bit gradient quantization scheme. In summary, the main **contributions** of this paper are given as follows:

(1) We propose a novel efficient Decentralized Quantized Stochastic Frank-Wolfe (DQSFW) method to solve the problem (1) with less communication cost but still good convergence speed.

(2) We derive the rigorous theoretical analysis for our DQSFW algorithm, and prove that our DQSFW algorithm has the same gradient complexity $O(\epsilon^{-4})$ as the SFW (Reddi et al. 2016) (the sequential algorithm) and QFW (Zhang et al. 2019) (the centralized algorithm), but with much less communication cost.

(3) We provide a new intuitive counterexample to show that the decentralized optimization involving non-linear projection of gradient could lead to a potential divergent problem which also exists in many cases where we generalize other non-Frank-Wolfe methods to decentralized algorithms. To tackle this challenge, we utilize the gradient tracking technique to guarantee the convergence of our decentralized quantized Frank-Wolfe algorithm.

Notations

$\|\cdot\|_1$ denotes one norm of vector. $\|\cdot\|_2$ denotes spectral norm of matrix. $\|\cdot\|_F$ denotes Frobenius norm of matrix. $\|\cdot\|_*$ denotes trace norm of matrix. Let $\mathbf{1}$ be the column vector, each entry of which is one. Given a network with M nodes, we define a mixing matrix $W = (w_{ij}) \in \mathbb{R}^{M\times M}$ that represents the weights of neighbors in the communication round. For example, in D-PSGD (Lian et al. 2017), the consensus step on i-th node is formulated as

$$x_t^{(i)} = \sum_{i=1}^{M} w_{ij} x_t^{(j)}.$$

Generally, W is a symmetric doubly stochastic matrix that satisfies $W^T = W$ and $W\mathbf{1} = \mathbf{1}$. In the experiment section, we will consider a uniformly weighted ring-based network, whose mixing matrix is shown as Eq. (3).

$$W = \begin{bmatrix} 1/3 & 1/3 & 0 & \cdots & 0 & 1/3 \\ 1/3 & 1/3 & 1/3 & 0 & \cdots & 0 \\ 0 & 1/3 & 1/3 & 1/3 & 0 & \cdots \\ \vdots & & \ddots & \ddots & \ddots & \\ 0 & \cdots & 0 & 1/3 & 1/3 & 1/3 \\ 1/3 & 0 & \cdots & 0 & 1/3 & 1/3 \end{bmatrix} \tag{3}$$

Related Works

Decentralized Frank-Wolfe

Decentralized Frank-Wolfe algorithm (DeFW) (Wai et al. 2017) is recently proposed to apply deterministic Frank-Wolfe method in decentralized structure. It is guaranteed to converge in both convex and non-convex problems. The authors compute net averaging on parameters and gradients.

Algorithm	DeFW	QFW	DQSFW
Decentralized	√	×	√
Stochastic	×	√	√
Quantized	×	√	√
Reference	(Wai et al. 2017)	(Zhang et al. 2019)	Ours

Table 1: Comparison of related works.

In previous work about decentralization such as (Lian et al. 2017), they do not have to calculate the averaging of gradients. Compared with DeFW, our DQSFW changes the deterministic algorithm to a stochastic one, which is more adaptive to large scale machine learning tasks. When the number of samples is very large, the full gradient is too expensive to calculate. Besides, we also take advantage of the technique of gradient quantization, which will further reduce the cost of communication.

Quantized Frank-Wolfe

Quantized Frank-Wolfe algorithm (QFW) (Zhang et al. 2019) is recently proposed to solve the centralized distributed problems. It uses the the following momentum scheme as gradient estimator:

$$\bar{g}_t = (1-\rho_t)\bar{g}_{t-1} + \rho_t g_t \tag{4}$$

which is also used in (Locatello et al. 2019) as a way to decrease the noise of gradient. In our algorithm, we combine this momentum scheme with the Gossip method. For gradient compressing, they adopt the s-Partition Encoding Scheme, which encodes the i-th coordinates g_i into an element from set $\{\pm 1, \pm\frac{s-1}{s}, \cdots, \pm\frac{1}{s}, 0\}$. It requires $\log_2(s+1)$ bits to transfer each coordinate of the gradient. A scalar factor $\|g\|_\infty$ is also transmitted thus the total bits of quantized gradient is $32+d\cdot\log_2(s+1)$ where d is the dimension of gradient. When s is large, the variance of this compressor will become small (Zhang et al. 2019), which means the quantized gradient is more precise, but costs more bits. In this paper, we use 1-bit signSGD scheme with a scalar factor (Karimireddy et al. 2019; Koloskova, Stich, and Jaggi 2019) shown as following:

$$C(x) = \frac{\|x\|_1}{d}\text{sign}(x) \tag{5}$$

where d is the dimension of x. **Notice that** here signSGD is only a representative of feasible compressors. We can also use other compressor as long as it satisfies the Compressor Assumption in section 4, which is an important assumption in many theoretical analysis of related work about gradient quantization. For example, we can also use top-k SGD, which is a gradient sparsification method that automatically satisfies the Compressor Assumption. We consider signSGD because it is efficient and convenient to implement.

The comparisons between DeFW, QFW and our DQSFW are summarized in Table 1. We can see that our DQSFW algorithm is the first work to incorporate stochastic gradient descent and gradient quantization in decentralized Frank-Wolfe type algorithm.

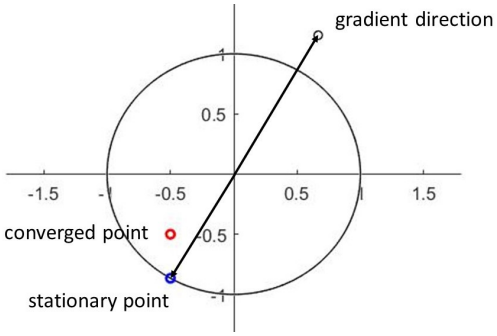


Figure 1: A graph to demonstrate the counter-example.

Counterexample

In this section, we provide an intuitive counterexample that indicates the divergent trap if Frank-Wolfe method is simply generalized to the decentralized algorithm without making consensus on gradient when data at different nodes are drawn from **non-identical distributions**.

Given $f(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x})$, where $\mathbf{x} \in \Omega = \{(x,y)|x^2+y^2 \leq 1\}$, $f_1(\mathbf{x}) = x$ and $f_2(\mathbf{x}) = \sqrt{3}y$ (See Figure 1). We can calculate gradients $\mathbf{v}_1 = (1,0)$, $\mathbf{v}_2 = (0,\sqrt{3})$, $\mathbf{v} = (1,\sqrt{3})$. Since the gradient is never equal to 0, according to the Frank-Wolfe gap (see Eq. (9)), the only stationary point is $(-\frac{1}{2}, -\frac{\sqrt{3}}{2})$ (blue point), where the tangent of unit ball is vertical to direction $(1,\sqrt{3})$. However, if we update $\mathbf{x}$ by Frank-Wolfe algorithm on each node separately, the linear oracle will yield $\mathbf{d}_1 = (-1,0)$ and $\mathbf{d}_2 = (0,-1)$. Then we make consensus on $\mathbf{x}$ and get iteration formula $\mathbf{x}_{i+1} = (1-\gamma)\mathbf{x}_i + \gamma(-\frac{1}{2}, -\frac{1}{2})$. Sequence $\mathbf{x}_i$ eventually converges to point $(-\frac{1}{2}, -\frac{1}{2})$ (red point), which is not a stationary point.

It is reasonable to credit the divergence to the non-commutative relation between linear oracle and addition. For SGD based decentralized learning algorithms, they can converge well because of the commutative property of addition. The above divergence problem is also likely to happen to other variant algorithms of SGD that involves non-linear map of gradients in decentralized system, not just Frank-Wolfe type methods. For example, adaptive gradient method is a family of algorithms that adjust the learning rate according to the magnitude of gradient. The Decentralized ADAM algorithm (DADAM) (Nazari, Tarzanagh, and Michailidis 2019) was proved to converge under the criterion named local regret. Nonetheless, local regret probably leads to a result that each node converges to its own local stationary point, while they do not cooperate well enough as an entire system. This phenomenon reminds us when we generalize an algorithm with steps that are not commutative to addition, similar divergence problem is likely to come out.

In DeFW (Wai et al. 2017), the gradients can be averaged directly by gradient tracking, a technique to accelerate consensus in distributed optimization (Xu et al. 2015; Nedić, Olshevsky, and Shi 2017). DIGing also considers the increment of gradient when averaging the non-quantized full gradient. **However**, in this paper we have to face the variance

of stochastic gradient and the noise of quantization. These issues do not occur in DeFW. Therefore, we have to use a new strategy to let them make consensus gradually.

New Decentralized Quantized Stochastic Frank-Wolfe Algorithm

In this section, we propose a novel efficient Decentralized Quantized Stochastic Frank-Wolfe (DQSFW) algorithm to solve the problem (1) by using the gradient tracking technique (Xu et al. 2015; Wai et al. 2017). The DQSFW algorithm is given in Algorithm 1.

In Algorithm 1, $x_t^{(i)}$ is a column vector that denote the model parameter on i-th node at iteration t. We use upper case X_t to represent the matrix

$$X_t = [x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(M)}]$$

Inspired by Choco-Gossip algorithm (Koloskova, Stich, and Jaggi 2019), we also define a replicated $\hat{x}_t^{(i)}$ of $x_t^{(i)}$ on each node. The reason is when we apply gossip update, the exact value of model parameter on other nodes are unknown since there exists quantization and the communication is inexact. The replica $\hat{x}_t^{(i)}$ is an estimation of $x_t^{(i)}$, which is also updated at each iteration. And the consensus step is formulated as line 8 in Algorithm 1. According to the update of $\hat{x}_t^{(i)}$ (line 9 and line 10 in Algorithm 1), on all neighbors of the i-th node, the replica is added by an indentical transmitted message $z_t^{(i)}$, which implies the values $\hat{x}_t^{(i)}$ on all neighbors of the i-th node are the same. Therefore, replica $\hat{x}_t^{(i)}$ is well-defined. Similar to X_t, we also define matrices

$$\begin{aligned} \hat{X}_t &= [\hat{x}_t^{(1)}, \hat{x}_t^{(2)}, \ldots, \hat{x}_t^{(M)}], \\ \overline{X}_t &= [\bar{x}_t, \bar{x}_t, \ldots, \bar{x}_t] \end{aligned}$$

where $\bar{x}_t$ represent the mean value: $\bar{x}_t = \frac{1}{M}\sum_{i=1}^{M} x_t^{(i)}$.

$g_t^{(i)}$ is a stochastic gradient on i-th node calculated by selected samples and $v_t^{(i)}$ is our key estimation of the gradient on i-th node which is defined as Eq. (6) with a kind of momentum scheme. Here $\hat{v}_t^{(i)}$ is the replica of $v_t^{(i)}$ (see similar concept of $\hat{x}_t^{(i)}$). For initialization, we set $\hat{v}_{-1}^{(i)} = \mathbf{0}$ and $v_{-1}^{(i)} = g_0^{(i)}$. The definition and role of β_t will be discussed later in Remark 2. Our convergence analysis shows that though our gradient estimator in line 4 is biased, the gradients on all nodes are getting close to the full gradient uniformly. To make consensus on gradient and parameter, we adopt the gossip update (Koloskova, Stich, and Jaggi 2019) in line 4 and line 8 respectively.

$$v_t^{(i)} = (1-\beta_t)v_{t-1}^{(i)} + \beta_t g_t^{(i)} + \alpha_t \sum_j w_{ij}(\hat{v}_{t-1}^{(j)} - \hat{v}_{t-1}^{(i)}) \quad (6)$$

In line 5 and line 9 of Algorithm 1, we apply a gradient quantization method that satisfies the Compressor Assumption. As mentioned previously, the quantization scheme is not limited to the signSGD used in this paper. Line 7 is the typical linear oracle in Frank-Wolfe method to get a direction $d_t^{(i)}$. In vanilla Frank-Wolfe algorithm, the update of

Algorithm 1 Decentralized Quantized Stochastic Frank-Wolfe (DQSFW)

Input: restricted domain Ω, matrix W, initial point $\hat{X}_0 = X_0 \in \Omega$
Parameter: η_t, γ_t, β_t, α_t, T
Output: $\bar{x}_{\hat{t}}$, where $\hat{t}$ is chosen uniformly from $\{0, 1, \ldots, T\}$

1: On i-th node:
2: **for** $t = 0, 1, \ldots, T-1$ **do**
3: Compute an estimation of the gradient $g_t^{(i)}$
4: Update $v_t^{(i)} = (1-\beta_t)v_{t-1}^{(i)} + \beta_t g_t^{(i)} + \alpha_t \sum_j w_{ij}(\hat{v}_{t-1}^{(j)} - \hat{v}_{t-1}^{(i)})$
5: Compute $q_t^{(i)} = C(v_t^{(i)} - \hat{v}_{t-1}^{(i)})$ and communicate with neighbors
6: Update replica $\hat{v}_t^{(j)} = \hat{v}_{t-1}^{(j)} + q_t^{(j)}$ for neighbor j
7: Calculate linear oracle $d_t^{(i)}$ such that $d_t^{(i)} = \arg\max_{d\in\Omega}\langle d, -v_t^{(i)}\rangle$
8: Update $x_{t+1}^{(i)} = (1-\eta_t)x_t^{(i)} + \eta_t d_t^{(i)} + \gamma_t \sum_j w_{ij}(\hat{x}_t^{(j)} - \hat{x}_t^{(i)})$
9: Compute $z_t^{(i)} = C(x_{t+1}^{(i)} - \hat{x}_t^{(i)})$ and communicate with neighbors
10: Update replica $\hat{x}_{t+1}^{(j)} = \hat{x}_t^{(j)} + z_t^{(j)}$ for neighbor j
11: **end for**

$x_t^{(i)}$ should be $x_{t+1}^{(i)} = x_t^{(i)} + \eta_t(d_t^{(i)} - x_t^{(i)})$. For convenience, we define matrices

$$\begin{aligned} V_t &= [v_t^{(1)}, v_t^{(2)}, \ldots, v_t^{(M)}], \\ \hat{V}_t &= [\hat{v}_t^{(1)}, \hat{v}_t^{(2)}, \ldots, \hat{v}_t^{(M)}], \\ \overline{V}_t &= [\bar{v}_t, \bar{v}_t, \ldots, \bar{v}_t], \\ D_t &= [d_t^{(1)}, d_t^{(2)}, \ldots, d_t^{(M)}], \\ \overline{D}_t &= [\bar{d}_t, \bar{d}_t, \ldots, \bar{d}_t] \end{aligned}$$

where $\bar{v}_t$ and $\bar{d}_t$ are mean values

$$\bar{v}_t = \frac{1}{M}\sum_{i=1}^{M} v_t^{(i)}, \quad \bar{d}_t = \frac{1}{M}\sum_{i=1}^{M} d_t^{(i)}$$

By the doubly stochastic property of W, we have

$$\overline{X}_{t+1} = (1-\eta_t)\overline{X}_t + \eta_t \overline{D}_t \quad (7)$$

It is easy to verify that when $x_0 \in \Omega$, $\bar{x}_t \in \Omega$ for $\forall t$. Hence the constraint is always satisfied. Here we should notice that we do not have to store all the replica in practice.

We can regard $\sum_j w_{ij}(\hat{x}_t^{(j)} - \hat{x}_t^{(i)})$ as a term, and obtain iteration formula

$$\begin{aligned} \sum_j w_{ij}(\hat{x}_{t+1}^{(j)} - \hat{x}_{t+1}^{(i)}) = &\sum_j w_{ij}(\hat{x}_t^{(j)} - \hat{x}_t^{(i)}) \\ &+ \sum_j w_{ij}(z_t^{(j)} - z_t^{(i)}). \end{aligned} \quad (8)$$

Therefore, we only need one buffer with the size of x_t to compute this term. So it is with $\sum_j w_{ij}(\hat{v}_t^{(j)} - \hat{v}_t^{(i)})$. We will use Eq. (8) to save memory in our experiments.

Convergence Analysis

In this section, we study the convergence properties of our DQSFW algorithm. All proofs can be found in Supplementary Material. We begin with introducing some mild assumptions and the definition of Frank-Wolfe gap (Jaggi 2013):

$$\mathcal{G}(x) = \max_{v \in \Omega} \langle v - x, -\nabla f(x) \rangle \tag{9}$$

The convergence criteria is $\mathbb{E}[\mathcal{G}(x)] \leq \epsilon$.

Assumption 1. *(Lipschitz Gradient) There is a constant L such that for $\forall i \in \{1, 2, \ldots, M\}$, we have*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|. \tag{10}$$

Assumption 2. *(Compact Domain) There is a diameter D of domain Ω.*

Assumption 3. *(Lower Bound) Function $f(x)$ has the lower bound $\inf_{x \in \Omega} f(x) = f^- > -\infty$.*

Assumption 4. *(Spectral Gap) Given the symmetric doubly stochastic matrix W, we define $\lambda_1, \lambda_2, \ldots, \lambda_M$ to be its eigenvalues in descending order. Then $\max\{|\lambda_2|, |\lambda_M|\} < 1$. Let $\rho = \max\{|\lambda_2|, |\lambda_M|\}$ and $\zeta = 1 - \lambda_M$.*

Assumption 5. *(Compressor Assumption) Compressor $C(\cdot)$ satisfies $\|C(x) - x\|^2 \leq (1 - \delta)\|x\|^2$, where $0 < \delta \leq 1$.*

Assumption 6. *(Bounded Gradient and Bounded Variance) The generated gradient estimator $g_t^{(i)}$ satisfies $\mathbb{E}[g_t^{(i)}] = \nabla f_i(x_t^{(i)})$, $\mathbb{E}\|g_t^{(i)} - \nabla f_i(x_t^{(i)})\|^2 \leq \sigma^2$, $\|\nabla F^{(i)}(x_t^{(i)}; \xi)\| \leq G$.*

Based on above assumptions, we demonstrate three lemmas of our DQSFW algorithm. Lemma 1 and lemma 2 estimate the consensus between model parameter X_t and gradient V_t respectively. Lemma 3 implies that our gradient estimator Eq. (6) on different node approaches the value of full gradient uniformlly and gradually. The detailed proof of lemmas can be found in supplementary material **section A**.

Lemma 1. *Let $\delta_0 = 1 - \sqrt{1 - \delta^2}$, $\delta_1 = 1 - (1 - \delta_0^2)^2$. $\alpha_t = \gamma_t = \gamma = min\{1, \frac{\delta_0}{\zeta}, \frac{(1-\rho)\delta_1}{2\zeta^2}\}$. $c_1 = (1-\rho)\gamma$, $c_2 = \delta$, $c_3 = min\{\frac{(1-\rho)\gamma}{2}, \frac{\delta_1}{2}\}$. $A = (1+c_1)(1-(1-\rho)\gamma)^2 + (1-\delta)(1+\frac{1}{c_2})\gamma^2\zeta^2$, $B = (1+\frac{1}{c_1})\gamma^2\zeta^2 + (1-\delta)(1+c_2)(1+\gamma\zeta)^2$. Let $Q = 8(1+\frac{1}{c_3})(A+B)$. Set $\eta_0 = \frac{c_3^3}{16(1+c_3)(A+B)}$ and $\eta_t = \frac{\eta_0}{(t+1)^\theta}$, $0 < \theta < 1$. Then there exists a constant R_1 satisfying*

$$\|X_t - \overline{X_t}\|_F^2 + \|X_t - \hat{X}_t\|_F^2 \leq \frac{QR_1MD^2}{(t+1)^{2\theta}}$$

Lemma 2. *Let $c_4 = c_3^2$, $\beta_0 = \frac{B(1+c_4)}{Ac_4}$ and $\beta_t = \frac{\beta_0}{(t+1)^{2\theta/3}}$. Then there exists constant R_2 such that*

$$\|V_t - \overline{V_t}\|_F^2 + \|V_t - \hat{V}_t\|_F^2 \leq \frac{QR_2MG^2}{(t+1)^{2\theta/3}}$$

Lemma 3. *Denote $\bar{v}_t = \frac{1}{M}\sum_{i=1}^M v_t^{(i)}$. There exists constant S such that*

$$\mathbb{E}\|\bar{v}_t - \frac{1}{M}\sum_{i=1}^M \nabla f_i(x_t^{(i)})\|^2 \leq \frac{S}{(t+1)^{2\theta/3}}$$

Next, we will propose the main theorem of our convergence analysis. Please check the detailed proof in supplementary material **section B**.

Theorem 1. *Let Q, R_1, R_2 and S be the constants defined in Lemma 1 to Lemma 3. Step size η_t is set as Lemma 1. Then we have*

$$\mathbb{E}[\mathcal{G}(\bar{x}_t)] \leq \mathbb{E}\big[\frac{f(\bar{x}_t - f(\bar{x}_{t+1})}{\eta_t})\big] + \frac{D\sqrt{2(S + QR_1L^2D^2)}}{(t+1)^{\theta/3}} + \frac{D\sqrt{QR_2}G}{(t+1)^{\theta/3}} + \frac{\eta_t L^2 D^2}{2}.$$

Theorem 2. *Suppose T iterations have been completed. Let $\hat{t}$ is chosen randomly with identical probability from $\{0, 1, \ldots, T\}$. Set $\theta = \frac{3}{4}$. Then by* **Theorem 1** *we can obtain*

$$\mathbb{E}[\mathcal{G}(\bar{x}_{\hat{t}})] = O(\frac{1}{T^{1/4}}).$$

Remark 1. *Theorem 2 shows that our DQSFW algorithm reach a gradient complexity of $O(\epsilon^{-4})$ to achieve ϵ-accuracy stationary point. And the Frank-Wolfe gap is asymptotic to 0, rather than a neighborhood of which the size is dependent on ϵ. This is because all parameters in our algorithm are independent of ϵ, while in SFW, step size and number of iterations are functions of ϵ.*

Remark 2. *$\theta = \frac{3}{4}$ is the best trade-off between consensus and stepsize. If β_t is too large, the noise of quantization and the variance of stochastic gradient will cause bad consensus and then affect the convergence. If β_t is too small, the stepsize should also be small. Otherwise the averaged gradient cannot catch up with the changing of x, which will cause slow convergence. This trade-off is the challenge and intuition to define our gradient estimator as Eq. (6).*

Remark 3. *From the proof in supplementary material we can see the theoretical framework does not only work for signSGD, but also all compressors that satisfy Assumption 5.*

Experimental Results

To validate the efficiency of our new DQSFW algorithm, we conduct the experiments on two constrained machine learning applications: matrix completion and model compression.

Decentralized Low-Rank Matrix Completion

Low-rank matrix completion is a model to solve a broad range of learning tasks, such as collaborative filtering (Koren, Bell, and Volinsky 2009) and multi-label learning (Xu, Jin, and Zhou 2013). The loss function of low-rank matrix completion problem has the following form:

$$\min_{X \in \mathbb{R}^{M \times N}} \sum_{(i,j) \in \Omega} \phi(X_{ij} - Y_{ij}), \quad \text{s.t. } \|X\|_* \leq C$$

where $\phi : \mathbb{R} \to \mathbb{R}$ is the potential non-convex empirical loss function. Y is the target matrix and Ω is the set of observed entries. (Wai et al. 2017) also conducts this experiment with MSE loss function and robust Gaussian loss function. In our

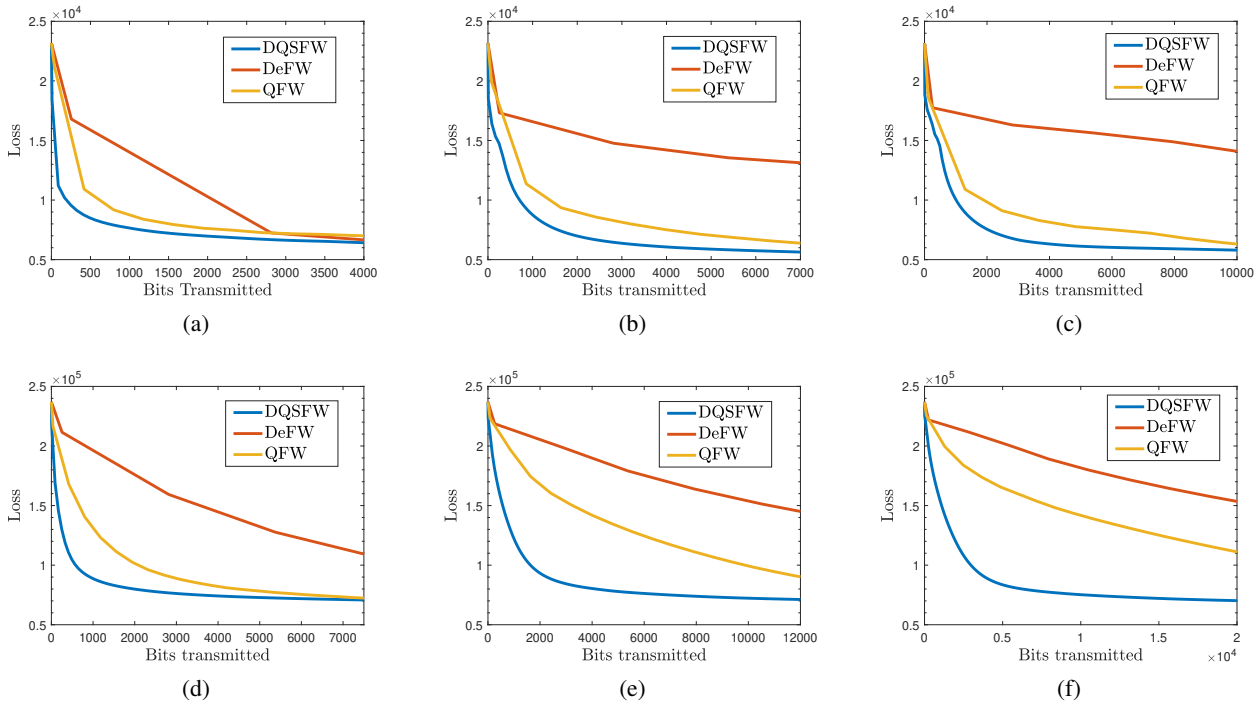


Figure 2: The experimental results of decentralized low-rank matrix completion for dataset MovieLens 100k and MovieLens 1m. Figure (a), (b) and (c) show the training loss with respect to bits transferred on MovieLens 100k with 20, 40 and 60 workers respectively. Figure (d), (e) and (f) show the training loss with respect to bits transferred on MovieLens 1m with 20, 40 and 60 workers respectively.

Name	User	Movie	Record
MovieLens 100k	943	1682	100000
MovieLens 1m	6040	3952	1000209

Table 2: Descriptions of MovieLens Datasets.

experiment, to verify that our algorithm works well for non-convex objective functions, we adopt the robust Gaussian loss function

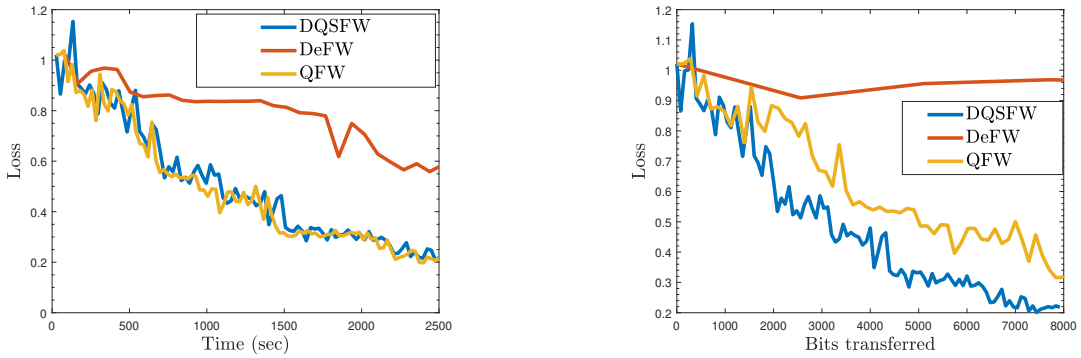
$$\phi(z) = \frac{\sigma_0}{2}(1 - \exp(-\frac{z^2}{\sigma_0})) \tag{11}$$

In our experiment, parameter σ_0 is fixed to be 2. We run our experiment on two benchmark datasets, MovieLens 100k and MovieLens 1m (Harper and Konstan 2015). Both of two datasets are records of movie ratings from plenty of users, and are usually used to train recommendation systems. The descriptions of these two datasets are shown in Table 2. MovieLens 100k has 943 users, 1682 movies, and 100000 rating records. MovieLens 1M has 6040 users, 3952 movies, and 1000209 rating records. All ratings vary from 0 to 5. We scale them to interval $[0, 1]$. The rating records can be converted into matrix, where row represents user id and column represents movie id. Each record serves as an observation. As our purpose is to verify the performance of optimization algorithm, we take all data for training.

For both datasets, we deploy our experiment on $M = 20, 40, 60$ MPI worker nodes respectively by mpi4py. Each node is an Intel Xeon E5-2660 machine within an infiniband network. We assign $1/M$ of the rating records to each worker. For MovieLens 100k, we set $C = 2000$ while for MovieLens 1m we set $C = 5000$.

In this task, the linear oracle can be obtained by singular value decomposition (SVD). Let the SVD of $v_t^{(i)}$ be $U \cdot S \cdot V^T$. Then the linear oracle $d = -C \cdot u \cdot v^T$ where u and v are the singular vectors corresponding to the largest singular value (also named leading vectors of SVD). In practice, we only need to compute the leading vectors, while in projected algorithms we have to do the completed SVD.

We choose two other projection-free methods DeFW (Wai et al. 2017) and QFW (Zhang et al. 2019) as baseline methods. For decentralized algorithms, we use a ring-based topology as the communication network because it is convenient to implement and achieves linear speedup in communication (Lian et al. 2017). For QFW, s of the s-partition encoding is set to be 1. For all of the three algorithms, we set step size $\eta_t = t^{-0.75}$. The results of low-rank matrix completion on MovieLens 100k and MovieLens 1m are shown in Figure 2. As many previous quantization work did (including QFW), we analyze the experimental results with respect to bits transferred, which means the number of bits sent or received on the busiest node. For decentralized algorithms it can be any node and for centralized algorithm it is the mas-



(a) (b)

Figure 3: The experimental result of decentralized model compression for VGG11 neural network on dataset CIFAR 10. Figure (a) shows the training loss against time. Figure (b) shows the training loss against the number of bits transferred

ter node. The number is divided by the size of x since it is always proportional to the size of x.

From the experimental results we can see that our DQSFW algorithm achieves the best performance on both datasets. Moreover, we can see that our algorithm becomes more competitive as the number of workers becomes larger, which verifies the scalability of our method. With more workers, more sample gradients can be computed, but the number of gradients computed by DeFW keeps the same.

Model Compression

Deep neural networks (DNNs) have achieved remarkable performance in many fields at recent years. But one of its shortages is the high cost of a large number of parameters. Thus, many attempts have been made to reduce the number of parameters in DNNs, such as dropout and pruning. Among these methods, one solution is to add constraint on parameters to make them sparse compulsively (Liu et al. 2015; Bietti et al. 2019). In (Bietti et al. 2019), one popular method was proposed via adding the spectral norm constraint as follows:

$$\|W_l\|_2 \leq \tau \tag{12}$$

for each layer l. Because the model compression is attracting increasing attentions in both machine learning research and applications, in this experiment, we solve the model compression problem using the decentralized learning setting, and validate the performance of different decentralized quantized algorithms on this task. Here the linear oracle is $d = -\tau \cdot U \cdot V^T$, where $U \cdot S \cdot V^T$ is the SVD of $v_t^{(i)}$. This result can be achieved easily by the fact that trace norm and spectral norm are dual norms.

In our experiment, we run this task to compress VGG11 network on CIFAR 10 dataset, which has 50000 training samples and 10 labels, with constraint (12). We conduct this task on decentralized settings where data are distributed on different nodes to verify our algorithms. Following (Bietti et al. 2019), we use cross-entropy loss function as the criterion and set $\tau = 0.8$. The experiment is implemented on 8 GTX1080 GPUs by Pytorch. Each GPU is treated as a single worker. The communication is based on NVIDIA NCCL.

We consider DeFW and QFW as baseline methods and ring-based topology as communication network. For DeFW and our DQSFW, the decentralized system is uniform weight ring network. For QFW, s is set to be 1. For all three algorithms, step size is chosen as $\eta_t = \frac{1}{2}t^{-0.75}$. Because of the limitation of CUDA memory, we cannot compute the full gradient for DeFW. We calculate $1/5$ of the full gradient instead. This issue also indicates the limitation of DeFW algorithm.

The experimental results are visualized in Figure 3. To validate the efficiency of our algorithm, we compare the loss with respect to the bits transmitted. Similar to the matrix completion experiment, the number of bits transferred is divided by the size of parameter. For decentralized algorithms, the number is counted on any node, while for centralized algorithm it is counted on master node. According to the results, we can see that DeFW is almost infeasible for this task. From the view of time, QFW and DQSFW have similar performance. From the view of bits transferred, our DQSFW has the best performance among the three algorithms, which verifies the superior performance of our new algorithm.

Conclusion

In this paper, we proposed a new Decentralized Quantized Stochastic Frank-Wolfe (DQSFW) algorithm to solve the non-convex constrained optimization problem. We revealed a potential divergence problem that is likely to occur in the general decentralized training, not just for Frank-Wolfe type methods, and also provided a solution by making consensus on gradient. We derived the new theoretical analysis to prove that our algorithm can achieve the same gradient complexity $O(\epsilon^{-4})$ as the Stochastic Frank-Wolfe (SFW) method with much less communication cost, and the Frank-Wolfe gap is asymptotic to 0. The experimental results on two machine learning applications, matrix completion and deep neural network compression, validate the superior performance of our new algorithm.

Acknowledgements

We thank the anonymous reviewers for their helpful comments. We also thank the IT Help Desk at the University of Pittsburgh. This work was partially supported by NSF IIS 1845666, 1852606, 1838627, 1837956, 1956002, 2040588.

References

Alistarh, D.; Grubic, D.; Li, J. Z.; Tomioka, R.; and Vojnovic, M. 2017. QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding. *Neural Information Processing Systems* .

Bernstein, J.; Wang, Y.; Azizzadenesheli, K.; and Anandkumar, A. 2018a. signSGD: Compressed Optimization for Non-Convex Problems. *arXiv:1802.04434* .

Bernstein, J.; Zhao, J.; Azizzadenesheli, K.; and Anandkumar, A. 2018b. signSGD with Majority Vote is Communication Efficient And Fault Tolerant. *arXiv:1810.05291* .

Bietti, A.; Mialon, G.; Chen, D.; and Mairal, J. 2019. A Kernel Perspective for Regularizing Deep Neural Networks. *International Conference on Machine Learning* .

Doan, T. T.; Maguluri, S. T.; and Romberg, J. 2018. Fast Convergence Rates of Distributed Subgradient Methods with Adaptive Quantization. *arXiv:1810.13245* .

Ghadimi, S.; Lan, G.; and Zhang, H. 2016. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming* 155(1-2): 267–305.

Harper, F. M.; and Konstan, J. A. 2015. The Movielens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems* 5(4): 1–19.

Huang, F.; Chen, S.; and Huang, H. 2019. Faster Stochastic Alternating Direction Method of Multipliers for Nonconvex Optimization. In *International Conference on Machine Learning*, 2839–2848.

Jaggi, M. 2013. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. In *International Conference on Machine Learning*, 427–435.

Karimireddy, S. P.; Rebjock, Q.; Stich, S. U.; and Jaggi, M. 2019. Error Feedback Fixes SignSGD and other Gradient Compression Schemes. *International Conference on Machine Learning* .

Koloskova, A.; Lin, T.; Stich, S. U.; and Jaggi, M. 2020. Decentralized Deep Learning with Arbitrary Communication Compression. *International Conference on Learning Representations* .

Koloskova, A.; Stich, S. U.; and Jaggi, M. 2019. Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication. *International Conference on Machine Learning* .

Konečný, J.; McMahan, H. B.; Ramage, D.; and Richtárik, P. 2016. Federated Optimization: Distributed Machine Learning for On-Device Intelligence. *arXiv:1610.02527* .

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *Computer* 42(8): 30–37.

Lian, X.; Zhang, C.; Zhang, H.; Hsieh, C.-J.; ; Zhang, W.; and Liu, J. 2017. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. *Neural Information Processing Systems* .

Lian, X.; Zhang, W.; Zhang, C.; and Liu, J. 2018. Asynchronous Decentralized Parallel Stochastic Gradient Descent. *International Conference on Machine Learning* .

Liu, B.; Wang, M.; Foroosh, H.; Tappen, M.; and Penksy, M. 2015. Sparse Convolutional Neural Networks. *Conference on Computer Vision and Pattern Recognition* .

Locatello, F.; Yurtsever, A.; Fercoq, O.; and Cevher, V. 2019. Stochastic Frank-Wolfe for Composite Convex Minimization. In *Advances in Neural Information Processing Systems*, 14269–14279.

Nazari, P.; Tarzanagh, D. A.; and Michailidis, G. 2019. DADAM: A Consensus-based Distributed Adaptive Gradient Method for Online Optimization. *arXiv:1901.09109* .

Nedić, A.; Olshevsky, A.; and Shi, W. 2017. Achieving Geometric Convergence for Distributed Optimization over Time-varying Graphs. *SIAM Journal on Optimization* 27(4): 2597–2633.

Nedić, A.; and Ozdaglar, A. 2009. Distributed Subgradient Methods for Multi-Agent Optimization. *IEEE transactions on Automatic Control* 54(1): 48–61.

Ram, S. S.; Nedić, A.; and Veeravalli, V. V. 2010. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of optimization theory and applications* 147(3): 516–545.

Reddi, S. J.; Sra, S.; Póczós, B.; and Smola, A. 2016. Stochastic Frank-Wolfe Methods for Nonconvex Optimization. *arXiv:1607.08254* .

Reisizadeh, A.; Mokhtari, A.; Hassani, H.; and Pedarsani, R. 2019a. An Exact Quantized Decentralized Gradient Descent Algorithm. *IEEE Transactions on Signal Processing* 67(19): 4934–4947.

Reisizadeh, A.; Taheri, H.; Mokhtari, A.; Hassani, H.; and Pedarsani, R. 2019b. Robust and Communication-Efficient Collaborative Learning. *Neural Information Processing Systems* .

Seide, F.; Fu, H.; Droppo, J.; Li, G.; and Yu, D. 2014. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Tang, H.; Lian, X.; Qiu, S.; Yuan, L.; Zhang, C.; Zhang, T.; and Liu, J. 2019. DeepSqueeze: Decentralization Meets Error-Compensated Compression. *arXiv:1907.07346* .

Tang, H.; Lian, X.; Yan, M.; Zhang, C.; and Liu, J. 2018. D^2: Decentralized Training over Decentralized Data. *International Conference on Machine Learning* .

Wai, H.-T.; Lafond, J.; Scaglione, A.; and Moulines, E. 2017. Decentralized Frank–Wolfe algorithm for convex and nonconvex problems. *IEEE Transactions on Automatic Control* 62(11): 5522–5537.

Xu, J.; Zhu, S.; Soh, Y. C.; and Xie, L. 2015. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes. In *2015 54th IEEE Conference on Decision and Control (CDC)*, 2055–2060. IEEE.

Xu, M.; Jin, R.; and Zhou, Z.-H. 2013. Speedup matrix completion with side information: Application to multi-label learning. In *Advances in neural information processing systems*, 2301–2309.

Yan, F.; Sundaram, S.; Vishwanathan, S.; and Qi, Y. 2012. Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. *IEEE Transactions on Knowledge and Data Engineering* 25(11): 2483–2493.

Zhang, M.; Chen, L.; Mokhtari, A.; Hassani, H.; and Karbasi, A. 2019. Quantized Frank-Wolfe: Faster Optimization, Lower Communication, and Projection Free. *arXiv:1902.06332* .