# Quantized Frank-Wolfe: Faster Optimization, Lower Communication, and Projection Free

**Mingrui Zhang**[1] **Lin Chen**[1] **Aryan Mokhtari**[2] **Hamed Hassani**[3] **Amin Karbasi**[1]
[1]Yale University [2]University of Texas at Austin [3]University of Pennsylvania

## Abstract

How can we efficiently mitigate the overhead of gradient communications in distributed optimization? This problem is at the heart of training scalable machine learning models and has been mainly studied in the unconstrained setting. In this paper, we propose `Quantized Frank-Wolfe` (QFW), the first projection-free and communication-efficient algorithm for solving *constrained* optimization problems at scale. We consider both convex and non-convex objective functions, expressed as a finite-sum or more generally a stochastic optimization problem, and provide strong theoretical guarantees on the convergence rate of QFW. This is accomplished by proposing novel quantization schemes that efficiently compress gradients while controlling the noise variance intduced during this process. Finally, we empirically validate the efficiency of QFW in terms of communication and the quality of returned solution against natural baselines.

## 1 Introduction

The Frank-Wolfe (FW) method [Frank and Wolfe, 1956], also known as conditional gradient, has recently received considerable attention in the machine learning community, as a projection free algorithm for various *constrained* convex [Jaggi, 2013, Garber and Hazan, 2014, Lacoste-Julien and Jaggi, 2015, Garber and Hazan, 2015, Hazan and Luo, 2016, Mokhtari et al., 2018a] and non-convex [Lacoste-Julien, 2016, Reddi et al., 2016, Mokhtari et al., 2018b, Zhang et al., 2019b, Hassani et al., 2019] optimization problems. In order to apply the FW method to large-scale problems (*e.g.*, training deep neural networks [Ravi et al., 2018, Schramowski et al., 2018, Berrada et al., 2018], RBMs [Ping et al., 2016]) parallelization is unavoidable. To this end, distributed FW variants have been proposed for specific problems, *e.g.*, online learning [Zhang et al., 2017], learning low-rank matrices [Zheng et al., 2018], and optimization under block-separable constraint sets [Wang et al., 2016]. A significant performance bottleneck of distributed optimization methods is the cost of communicating gradients, typically handled by using a parameter-server framework. Intuitively, if each worker in the distributed system transmits the entire gradient, then at least $d$ floating-point numbers are communicated for each worker, where $d$ is the dimension of the problem. This communication cost can be a huge burden on the performance of parallel optimization algorithms [Chilimbi et al., 2014, Seide et al., 2014, Strom, 2015]. To circumvent this drawback, communication-efficient parallel algorithms have received significant attention. One major approach is to quantize the gradients while maintaining sufficient information [De Sa et al., 2015, Abadi et al., 2016, Wen et al., 2017]. For *unconstrained* optimization, when projection is not required for implementing Stochastic Gradient Descent (SGD), several communication-efficient distributed methods have been proposed, including QSGD [Alistarh et al., 2017], SIGN-SGD [Bernstein et al., 2018], and Sparsified-SGD [Stich et al., 2018].

In the constrained setting, and in particular for distributed FW methods, the communication-efficient versions were only studied for specific problems such as sparse learning [Bellet et al., 2015, Lafond et al., 2016]. In this paper, however, we develop `Quantized Frank-Wolfe` (QFW), a general communication-efficient distributed FW for both convex and non-convex objective functions. We study the performance of QFW in two widely recognized settings: 1) stochastic optimization and 2) finite-sum optimization.

To be more specific, let $\mathcal{K} \subseteq \mathbb{R}^d$ be the constraint set. In *constrained stochastic optimization* the goal is to

Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. 

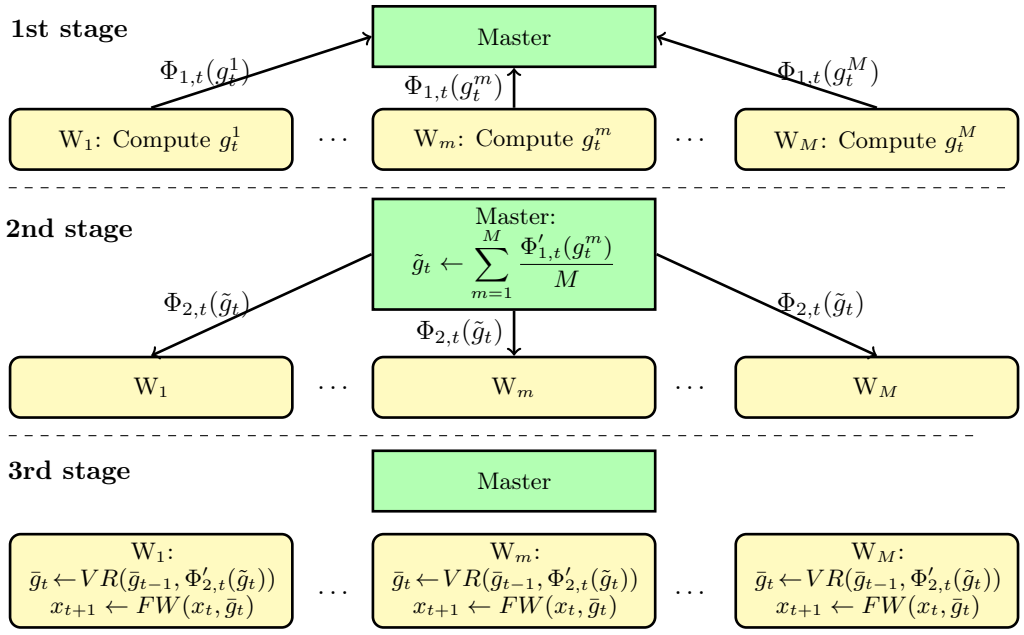



Figure 1: Stages of our general Quantized Frank-Wolfe scheme at time $t$. In the first stage, each worker $m$ computes its local gradient information $g_t^m$ and sends the quantized version $\Phi_{1,t}(g_t^m)$ to the master node. In the second stage, master computes the average of decoded received signals $\Phi'_{1,t}(g_t^m)$, *i.e.*, $\tilde{g}_t \leftarrow (1/M)\sum_{m=1}^{M} \Phi'_{1,t}(g_t^m)$ and then sends its quantized version $\Phi_{2,t}(\tilde{g}_t)$ to the workers. Note that the two quantization schemes $\Phi_{1,t}, \Phi_{2,t}$ depend on $t$ and can be different from each other. In the third stage, workers use the decoded gradient information computed by all workers $\Phi'_{2,t}(\tilde{g}_t)$ and their previous gradient estimation $\bar{g}_{t-1}$ to update their new gradient estimation $\bar{g}_t$ via a variance reduction (VR) scheme. Once the variance reduced gradient approximation $\bar{g}_t$ is evaluated, workers compute the new variable $x_{t+1}$ by following the update of Frank-Wolfe (FW).

solve

$$\min_{x\in\mathcal{K}} f(x) \;:=\; \min_{x\in\mathcal{K}} \mathbb{E}_{z\sim P}[\tilde{f}(x,z)], \tag{1}$$

where $x \in \mathbb{R}^d$ is the optimization variable, $z \in \mathbb{R}^q$ is a random variable drawn from a probability distribution $P$, which determines the choice of a stochastic function $\tilde{f} : \mathbb{R}^d \times \mathbb{R}^q \to \mathbb{R}$. For *constrained finite-sum optimization*, we further assume that $P$ is a uniform distribution over $[N] = \{1, 2, \cdots, N\}$ and the goal is to solve a special case of Problem (1), namely,

$$\min_{x\in\mathcal{K}} f(x) \;:=\; \min_{x\in\mathcal{K}} \frac{1}{N}\sum_{j=1}^{N} f_j(x). \tag{2}$$

In parallel settings, we suppose that we have a computing system consisting of a master node and $M$ workers, and each worker maintains a local copy of $x$. At every iteration of the stochastic case, each worker has access to independent stochastic gradients of $f$; whereas in the finite-sum case, we assume $N = Mn$, thus the objective function can be decomposed as $f(x) = \frac{1}{Mn}\sum_{m\in[M],j\in[n]} f_{m,j}(x)$, and each worker $m$ has access to the exact gradients of $n$ component functions $f_{m,j}(x)$ for all $j \in [n]$.

This way the task of computing gradients is divided among the workers. The master node aggregates local gradients from the workers, and sends the aggregated gradients back to them so that each worker can update the model (*i.e.*, their own iterate) locally. Thus, by transmitting quantized gradients, we can reduce the communication complexity (*i.e.*, number of transmitted bits) significantly. The workflow diagram of the proposed Quantized Frank-Wolfe scheme is summarized in Figure 1. We should highlight that there is a trade-off between gradient quantization and information flow. Intuitively, more intensive quantization reduces the communication cost, but also loses more information, which may decelerate the convergence rate.

**Our contributions:** In this paper, we propose a novel distributed projection-free framework that handles quantization for constrained convex and non-convex optimization problems in finite-sum and stochastic cases. It is well-known that unlike projected gradient-based methods, FW methods may diverge when fed with stochastic gradient [Hazan and Luo, 2016, Mokhtari et al., 2018a]. Indeed, a similar issue arises in a distributed setting where nodes exchange *quantized gradients* which are noisy estimates of the gradients. By in-

Table 1: SFO/IFO Complexity per worker in different settings ($M$ is the number of workers).

| Setting | Function | SFO/IFO Complexity |
|---|---|---|
| Finite-sum | Convex | $\mathcal{O}\left(\frac{N\ln(1/\epsilon)+1/\epsilon^2}{M}\right)$ |
| Finite-sum | Non-convex | $\mathcal{O}\left(\frac{\sqrt{N}}{\epsilon^2\sqrt{M}}\right)$ |
| Stochastic | Convex | $\mathcal{O}\left(\frac{1}{M\epsilon^2}\right)$ |
| Stochastic | Non-convex | $\mathcal{O}\left(\frac{1}{\epsilon^3\sqrt{M}}\right)$ |

corporating appropriate variance reduction techniques, we show that with quantized gradients, we can obtain a provably convergent method which preserves the convergence rates of the state-of-the-art vanilla centralized methods in all the considered cases [Zhang et al., 2019b, Shen et al., 2019b, Hassani et al., 2019, Yurtsever et al., 2019]. We believe our work presents the first quantized, distributed, and projection-free method. Our theoretical results for `Quantized Frank-Wolfe` (QFW) are summarized in Table 1, where the SFO complexity is the required number of stochastic gradients in stochastic case, and the IFO complexity is the number of exact gradients for component functions in finite-sum case. For the convex case, the complexity indicates the number of gradients to achieve an $\epsilon$-suboptimal solution; while in the non-convex case, it refers to the number of gradients to find a first-order $\epsilon$-stationary point. We note that since the $M$ workers compute the gradients simultaneously, the time to obtain gradients is proportional to the SFO/IFO complexity *per worker*. So we report the SFO/IFO complexity per worker, as in many other works on parallel optimization (*e.g.*, SignSGD [Bernstein et al., 2018]). The results in Table 1 show that more workers can decrease the SFO/IFO complexity per worker effectively, and thus accelerate the optimization procedure. **All the proofs in this paper are provided in the appendix**.

## 2 Gradient Quantization Schemes

In most distributed optimization algorithms, the task of computing gradients is divided among the workers, and the master node uses parts of gradients at the workers to update the model (iterate) directly or sends the aggregated gradients to the worker so that each of them can update the model (iterate) locally. Therefore, the information that workers need to send to the master is the elements of the objective function gradient. Thus, by transmitting quantized gradients, we can reduce the communication bits effectively. In this section, we introduce a quantization scheme called s-`Partition Encoding Scheme` and explain how this scheme reduces the overall cost of exchanging gradients. Consider the gradient vector $g \in \mathbb{R}^d$ and let $g_i$ be the $i$-th coordinate of the gradient. The s-`Partition Encoding Scheme` encodes $g_i$ into an element from the set $\{\pm 1, \pm\frac{s-1}{s}, \cdots, \pm\frac{1}{s}, 0\}$ in a random way. To do so, we first compute the ratio $|g_i|/\|g\|_\infty$ and find the indicator $l_i \in \{0, 1, \cdots, s-1\}$ such that $|g_i|/\|g\|_\infty \in [l_i/s, (l_i+1)/s]$. Then we define the random variable $b_i$ as

$$b_i = \begin{cases} l_i/s, & \text{w.p. } 1 - \frac{|g_i|}{\|g\|_\infty}s + l_i, \\ (l_i+1)/s, & \text{w.p. } \frac{|g_i|}{\|g\|_\infty}s - l_i. \end{cases} \tag{3}$$

Finally, instead of transmitting $g_i$, we send $\mathrm{sgn}(g_i)\cdot b_i$, alongside the norm $\|g\|_\infty$. It can be verified that $\mathbb{E}[b_i|g] = |g_i|/\|g\|_\infty$. So we define the corresponding decoding scheme as $\phi'(g_i) = \mathrm{sgn}(g_i)b_i\|g\|_\infty$ to ensure that $\phi'(g_i)$ is an unbiased estimator of $g_i$. We note that the encoding/decoding schemes in Figure 1 are denoted as capital $\Phi/\Phi'$, indicating that they can be any general schemes. The proposed $s$-`Partition Encoding Scheme` is denoted by $\phi/\phi'$. We also note that this quantization scheme is similar to the Stochastic Quantization method in [Alistarh et al., 2017], except that we use $\ell_\infty$-norm while they adopt the $\ell_2$-norm. In the $s$-`Partition Encoding Scheme`, for each coordinate $i$, we need 1 bit to transmit $\mathrm{sgn}(g_i)$. Moreover, since $b_i \in \{0, 1/s, \ldots, (s-1)/s, 1\}$, we need $z = \log_2(s+1)$ bits to send $b_i$. Finally, we need 32 bits to transmit $\|g\|_\infty$. Hence, the total number of communicated bits is $32 + d(z+1)$. Here, by "bits" we mean the number of 0's and 1's transmitted.

One major advantage of the $s$-`Partition Encoding Scheme` is that by tuning the partition parameter $s$ or the corresponding assigned bits $z$, we can smoothly control the trade-off between gradient quantization and information loss, which helps distributed algorithms to attain their best performance. We proceed to characterize the variance of the $s$-`Partition Encoding Scheme`.

**Lemma 1** *The variance of $s$-`Partition Encoding Scheme` $\phi$ for any $g \in \mathbb{R}^d$ is bounded by*

$$\mathrm{Var}[\phi'(g)|g] \le \frac{d}{s^2}\|g\|_\infty^2. \tag{4}$$

Lemma 1 demonstrates the trade-off between the error of quantization and the communication cost for $s$-`Partition Encoding Scheme`. In a nutshell, for larger choices of $s$, the variance is smaller, which in turn results in higher communication cost. If we set $s = 1$, we obtain the `Sign Encoding Scheme`, which requires communicating the encoded scalars $\mathrm{sgn}(g_i)b_i \in \{\pm 1, 0\}$

and the norm $\|g\|_\infty$. Since $z = \log_2(s+1) = 1$, the overall communicated bits for each worker are $32 + 2d$ per round. We characterize its variance in Lemma 2.

**Lemma 2** *The variance of* `Sign Encoding Scheme` *is given by*

$$\mathrm{Var}[\phi'(g)|g] = \|g\|_1\|g\|_\infty - \|g\|_2^2. \tag{5}$$

**Remark 1** *For the probability distribution of the random variable $b_i$, instead of $\|g\|_\infty$, we can use other norms $\|g\|_p$ (where $p \geq 1$). But it can be verified that the $\ell_\infty$-norm leads to the smallest variance for* `Sign Encoding Scheme`*. That is also the reason why we do not use $\ell_2$-norm as in [Alistarh et al., 2017].*

## 3 Convex Minimization

In this section, we analyze the convex minimization problem in both finite-sum and stochastic settings. Note that even in the setting without quantization, if we use stochastic gradients in the update of FW, it might diverge [Hazan and Luo, 2016, Mokhtari et al., 2018a]. So appropriate variance reduction techniques are needed for communicating quantized gradients. Nguyen et al. [2017a,b, 2019] developed the StochAstic Recursive grAdient algoritHm (SARAH), a stochastic recursive gradient update framework. Fang et al. [2018] proposed Stochastic Path-Integrated Differential Estimator (SPIDER) technique, a variant of SARAH, for centralized unconstrained optimization. Recently, Hassani et al. [2019], Shen et al. [2019a], Yurtsever et al. [2019] proposed the SPIDER variants of FW method for both convex and non-convex optimization problems. Similar variance reduction idea was also combined with SGD to solve non-convex finite-sum problems in [Zhou et al., 2018]. In this paper, we generalize SPIDER to the constrained and distributed settings.

We first consider the case where no quantization is performed. Let $\{p_i\} \in \mathbb{N}^+$ be a sequence of period parameters. At the beginning of each period $i$, namely, $t = \sum_{j=1}^{i-1} p_j + 1$, each worker $m$ samples $S_{i,1}$ component functions in finite-sum case, or stochastic functions in stochastic case, which are denoted as $\mathcal{S}_{i,1}^m$. We define the local average gradient on set $\mathcal{S}_{i,1}^m$ as $g_{i,1}^m \triangleq \nabla f_{\mathcal{S}_{i,1}^m}(x_t) = \frac{1}{S_{i,1}} \sum_{j \in \mathcal{S}_{i,1}^m} \nabla f_j(x_t)$. Then each worker $m$ computes the average of all these local gradients $g_{i,1}^m$ and sends it to the master. Then, master node calculates the average of the $M$ received signals and broadcasts it to all workers. Then, the workers update their gradient estimation $\bar{g}_t$ as the averaged signal $\bar{g}_t = \frac{1}{M} \sum_{m=1}^M g_{i,1}^m$.

Note $\bar{g}_t$ is identical for all the workers. In the rest of that period, *i.e.*, $t = \sum_{j=1}^{i-1} p_j + k$, where $2 \leq k \leq p_i$, each worker $m$ samples a set of local functions, denoted as $\mathcal{S}_{i,k}^m$, of size $S_{i,k}$ uniformly at random, and computes the difference of averages of these gradients

$$g_{i,k}^m \triangleq \nabla f_{\mathcal{S}_{i,k}^m}(x_t) - \nabla f_{\mathcal{S}_{i,k}^m}(x_{t-1}), \tag{6}$$

and sends it to master. Then master node calculates the average of the $M$ signals and broadcasts it to all the workers. The workers update their gradient estimation $g_t$ as

$$\bar{g}_t = \bar{g}_{t-1} + \frac{1}{M} \sum_{m=1}^M g_{i,k}^m. \tag{7}$$

So $\bar{g}_t$ is still identical for all the workers. In order to incorporate quantization, each worker simply pushes the quantized version of the average gradients. Then the master decodes the quantizations, encodes the average of decoded signals in a quantized fashion, and broadcasts the quantization. Finally, each worker decodes the quantized signal and updates $x_t$ locally. To be more specific, in the quantized setting, in each iteration $t$ such that $t = \sum_{j=1}^{i-1} p_j + k$ where $1 \leq k \leq p_i$, each worker $m$ sends the quantized version of its local gradient information $\Phi_{1,t}(g_{i,k}^m)$ to the master. Once master collects all the quantized information, it decodes them, *i.e.*, finds $\{\Phi'_{1,t}(g_{i,k}^m)\}_{m=1}^M$, computes their average $\tilde{g}_t$,

**Algorithm 1** `Quantized Frank-Wolfe` (QFW)

**Input:** constraint set $\mathcal{K}$, total iteration number $T$, No. of workers $M$, period parameters $\{p_i\}$, sample sizes $\{S_{i,k}\}$, learning rate $\eta_t$, initial point $x_1 \in \mathcal{K}$
**Output:** $x_{T+1}$ or $x_o$, where $x_o$ is chosen from $\{x_1, x_2, \cdots, x_T\}$ uniformly at random
**for** $t = 1$ **to** $T$ **do**
  Set $x_{i,k} \leftarrow x_t$, where $t = \sum_{j=1}^{i-1} p_j + k, 1 \leq k \leq p_i$
  Each worker $m$ computes local gradient $g_{i,k}^m$ by $g_{i,1}^m = \nabla f_{\mathcal{S}_{i,1}^m}(x_{i,k}) = \nabla f_{\mathcal{S}_{i,1}^m}(x_t)$ for $k = 1$, or $g_{i,k}^m \triangleq \nabla f_{\mathcal{S}_{i,k}^m}(x_{i,k}) - \nabla f_{\mathcal{S}_{i,k}^m}(x_{i,k-1}) = \nabla f_{\mathcal{S}_{i,k}^m}(x_t) - \nabla f_{\mathcal{S}_{i,k}^m}(x_{t-1})$ for $k \geq 2$
  Each worker $m$ encodes $g_{i,k}^m$ as $\Phi_{1,i,k}(g_{i,k}^m)$ and pushes it to the master
  Master decodes $\Phi_{1,i,k}(g_{i,k}^m)$ as $\Phi'_{1,i,k}(g_{i,k}^m)$, and computes $\tilde{g}_{i,k} \leftarrow \frac{1}{M} \sum_{m=1}^M \Phi'_{1,i,k}(g_{i,k}^m)$
  Master encodes $\tilde{g}_{i,k}$ as $\Phi_{2,i,k}(\tilde{g}_{i,k})$, and broadcasts it to all workers
  Workers decode $\Phi_{2,i,k}(\tilde{g}_{i,k})$ as $\Phi'_{2,i,k}(\tilde{g}_{i,k})$
  **if** $k = 1$ **then**
    Workers update $\bar{g}_{i,k} \leftarrow \Phi'_{2,i,k}(\tilde{g}_{i,k})$
  **else**
    Workers update $\bar{g}_{i,k} \leftarrow \Phi'_{2,i,k}(\tilde{g}_{i,k}) + \bar{g}_{i,k-1}$
  **end if**
  Each worker updates $x_{t+1} \leftarrow x_t + \eta_t(v_t - x_t) = x_{i,k} + \eta_{i,k}(v_{i,k} - x_{i,k})$ where $v_{i,k} \leftarrow \mathrm{argmin}_{v \in \mathcal{K}} \langle v, \bar{g}_{i,k} \rangle$
**end for**

and sends its quantized version $\Phi_{2,t}(\tilde{g}_t)$ to all workers. Then, all the workers decode the received quantized signal and use it as their new gradient approximation $\bar{g}_t$ and update their variable according to the update of Frank-Wolfe, *i.e.*,

$$x_{t+1} = x_t + \eta_t(v_t - x_t), \tag{8}$$

where $v_t \leftarrow \text{argmin}_{v\in\mathcal{K}}\langle v, \bar{g}_t\rangle$. The description of our proposed `Quantized Frank-Wolfe` (QFW) method is shown in Figure 1 and outlined in Algorithm 1.

**Remark 2** *The model update (Line 15 in Algorithm 1) should be performed at each worker. Since all the linear programming problems (to obtain $v_t$) are solved simultaneously, the total running time is the same with that where the model updating is performed in the master node. However, additional variance would be introduced if the master node updates the model and broadcasts it in quantized manners. Thus the master-updating method lacks theoretical justification regarding convergence, and we adopt the worker-updating approach.*

### 3.1 Finite-Sum Setting

Now we proceed to establish the convergence properties of our proposed QFW in the finite-sum setting. Recall that we assume that there are $N$ functions and $M$ workers in total, and each worker $m$ has access to $n = N/M$ functions $f_{m,j}$ for $j \in [n]$. We first make two assumptions on the constraint set and component functions. Let $\|\cdot\|$ denote the $l_2$ norm in the Euclidean space through out the paper.

**Assumption 1** *The constraint set $\mathcal{K}$ is convex and compact, with diameter $D = \sup_{x,y\in\mathcal{K}}\|x - y\|$.*

**Assumption 2** *The functions $f_{m,i}$ are convex, $L$-smooth on $\mathcal{K}$, and satisfy that $\|\nabla f_{m,i}(x)\|_\infty \leq G_\infty$, for all $m \in [M], i \in [n], x, y \in \mathcal{K}$.*

**Theorem 1 (Finite-Sum Convex)** *Consider QFW outlined in Algorithm 1. Under Assumptions 1 and 2, if we set $p_i = 2^{i-1}, \mathcal{S}^m_{i,1} = \{f_{m,j} : j \in [n]\}$ (*i.e.*, each worker $m$ samples all its $n$ component functions), $S_{i,k} = p_i/M = 2^{i-1}/M$, for all $i \geq 1, k \geq 2$, and $\eta_{i,k} = 2/(p_i + k) = 2/(2^{i-1} + k)$, and use the $s_{1,i,1} = (\sqrt{\frac{dp_i^2}{M}})$-`Partition Encoding Scheme`, $s_{2,i,1} = (\sqrt{dp_i^2})$-`Partition Encoding Scheme` for $k = 1$, and $s_{1,i,k} = (\sqrt{\frac{dp_i}{M}})$-`Partition Encoding Scheme`, $s_{2,i,k} = (\sqrt{dp_i})$-`Partition Encoding Scheme` for $k \geq 2$ as $\Phi_{1,i,k}$ and $\Phi_{2,i,k}$ in Algorithm 1, then the output $x_{T+1} \in \mathcal{K}$ satisfies*

$$\mathbb{E}[f(x_{T+1})] - f(x^*) \leq \frac{4D\sqrt{2(G_\infty^2 + 6L^2D^2)} + 2LD^2}{T},$$

*where $x^*$ is a minimizer of $f$ on $\mathcal{K}$.*

**Corollary 1** *To obtain an $\epsilon$-suboptimal solution, we need to run the QFW method for at most $\mathcal{O}(1/\epsilon)$ iterations. The IFO complexity per worker in this case is $\mathcal{O}(\frac{N\ln(1/\epsilon)+1/\epsilon^2}{M})$.*

Corollary 1 shows IFO complexity per worker is linear in $1/M$, which implicates that increasing the number of workers $M$ will decrease the IFO complexity per worker effectively, thus accelerate the optimization procedure. Also, our numerical experiments in Section 5 showed that our proposed method requires significantly fewer bits than the unquantized version to achieve a specific accuracy.

### 3.2 Stochastic Setting

QFW can also be applied to the stochastic case. Recall that in the stochastic setting we assume that the objective function is $f(x) = \mathbb{E}_{z\sim P}[\tilde{f}(x, z)]$ and each worker has access to independent samples $\tilde{f}(x, z)$. Before proving the convergence properties of QFW for the stochastic setting, we first make a standard assumption on $\tilde{f}(x, z)$.

**Assumption 3** *The stochastic function $\tilde{f}(x, z)$ is convex, $L$-smooth on $\mathcal{K}$. The gradient $\nabla\tilde{f}(x, z)$ is an unbiased estimate of $\nabla f(x)$ with bounded variance $\sigma^2$, and satisfies that $\|\nabla\tilde{f}(x, z)\|_\infty \leq G_\infty$, for all $x \in \mathcal{K}, z \in \mathbb{R}^q$.*

**Theorem 2 (Stochastic Convex)** *Consider QFW outlined in Algorithm 1. Under Assumptions 1 and 3, if we set $p_i = 2^{i-1}, S_{i,1} = \frac{\sigma^2 p_i^2}{ML^2D^2}$, $S_{i,k} = p_i/M = 2^{i-1}/M$, for all $i \geq 1, k \geq 2$, and $\eta_{i,k} = 2/(p_i + k) = 2/(2^{i-1} + k)$, and use the $s_{1,i,1} = (\sqrt{\frac{dp_i^2}{M}})$-`Partition Encoding Scheme`, $s_{2,i,1} = (\sqrt{dp_i^2})$-`Partition Encoding Scheme` for $k = 1$, and $s_{1,i,k} = (\sqrt{\frac{dp_i}{M}})$-`Partition Encoding Scheme`, $s_{2,i,k} = (\sqrt{dp_i})$-`Partition Encoding Scheme` for $k \geq 2$ as $\Phi_{1,i,k}$ and $\Phi_{2,i,k}$ in Algorithm 1, then the output $x_{T+1} \in \mathcal{K}$ satisfies*

$$\mathbb{E}[f(x_{T+1})] - f(x^*) \leq \frac{4D\sqrt{13L^2D^2 + 2G_\infty^2} + 2LD^2}{T},$$

*where $x^*$ is a minimizer of $f$ on $\mathcal{K}$.*

**Corollary 2** *To obtain an $\epsilon$-suboptimal solution, we need to run the QFW method outlined in Algorithm 1 for at most $\mathcal{O}(1/\epsilon)$ iterations. The SFO complexity per worker in this case is $\mathcal{O}(1/(M\epsilon^2))$*

Corollary 2 shows SFO complexity per worker is linear in $1/M$, which implies a speed-up for distributed

settings. It also shows that the dependency of QFW's complexity on $\epsilon$ for convex settings is optimal.

**Remark 3** *In theory, the partitioning levels of quantization do depend on the number of iterations. Thus more transmission bits are required over the optimization procedure. But it will not render our QFW method communication-expensive. We set the partitioning levels conservatively to achieve the theoretical guarantees. However, as shown in the experiments (Section 5), much smaller quantization levels (which are actually constants) are usually preferred in practice.*

## 4 Non-Convex Optimization

With slightly different parameters, QFW can be applied to non-convex settings as well. In *unconstrained* non-convex optimization problems, the gradient norm $\|\nabla f\|$ is usually a good measure of convergence as $\|\nabla f\| \to 0$ implies convergence to a stationary point. However, in the constrained setting we study the Frank-Wolfe Gap [Jaggi, 2013, Lacoste-Julien, 2016] defined as

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

For constrained optimization problem, if a point $x$ satisfies $\mathcal{G}(x) = 0$, then it is a first-order stationary point. Also, by definition, we have $\mathcal{G}(x) \geq 0$, for all $x \in \mathcal{K}$. We first analyze the finite-sum setting and then the more general stochastic setting.

### 4.1 Finite-Sum Setting

To extend our results to the nonconvex setting we first assume that the following condition is satisfied.

**Assumption 4** *The component functions $f_{m,i}$ are $L$-smooth on $\mathcal{K}$ and uniformly bounded,* i.e.*, $\sup_{x \in \mathcal{K}} |f_{m,i}(x)| \leq M_0$. Further, $\|\nabla f_{m,i}(x)\|_\infty \leq G_\infty$, for all $m \in [M], i \in [n], x, y \in \mathcal{K}$.*

**Theorem 3 (Finite-Sum Non-Convex)** *Under Assumptions 1 and 4, if we set $p_i = \sqrt{n}$, $\mathcal{S}^m_{i,1} = \{f_{m,j} : j \in [n]\}$ (*i.e.*, each worker $m$ samples all its $n$ component functions), $S_{i,k} = \sqrt{n}/M$ for all $i \geq 1, k \geq 2, \eta_t = T^{-1/2}$ for all $t$, and use the $s_{1,i,1} = (\sqrt{\frac{Td}{M}})$-`Partition Encoding Scheme`, $s_{2,i,1} = (\sqrt{Td})$-`Partition Encoding Scheme` for $k = 1$, and $s_{1,i,k} = (\frac{d^{1/2} n^{1/4}}{\sqrt{M}})$-`Partition Encoding Scheme`, $s_{2,i,k} = (d^{1/2} n^{1/4})$-`Partition Encoding Scheme` for $k \geq 2$ as $\Phi_{1,i,k}$ and $\Phi_{2,i,k}$ in Algorithm 1, then the output $x_o \in \mathcal{K}$ satisfies*

$$\mathbb{E}[\mathcal{G}(x_o)] \leq \frac{2M_0 + D\sqrt{3L^2D^2 + 2G_\infty^2} + \frac{LD^2}{2}}{\sqrt{T}}.$$

**Algorithm 2** `Stochastic Non-Convex Quantized Frank-Wolfe` (SNC-QFW)

1: **Input:** constraint set $\mathcal{K}$, iteration number $T$, No. of workers $M$, initial point $x_1 \in \mathcal{K}$
2: Obtain $T$ independent samples of $z_i$, and define finite-sum $\hat{f}(x) = \frac{1}{T}\sum_{i=1}^T \tilde{f}(x, z_i)$
3: Apply Algorithm 1 on $\hat{f}$ with $N = T$ and all other parameters being the same as in Theorem 3
4: **Output:** $x_o$, where $x_o$ is chosen from $\{x_1, x_2, \cdots, x_T\}$ uniformly at random

**Corollary 3** *Algorithm 1 finds an $\epsilon$-first order stationary point after at most $\mathcal{O}(1/\epsilon^2)$ iterations. The IFO complexity per worker is $\mathcal{O}(\sqrt{N}/(\epsilon^2\sqrt{M}))$.*

Corollary 3 shows IFO complexity per worker is linear in $1/\sqrt{M}$, implicating that increasing the number of workers $M$ will decrease the IFO complexity per worker effectively, thus accelerate the optimization procedure.

### 4.2 Stochastic Setting

For non-convex objective function, the stochastic optimization problem in (1) can be solved approximately by the QFW method described in Algorithm 1. Specifically, the objective function $f(x) = \mathbb{E}_{z \sim P}[\tilde{f}(x, z)]$ can be approximated by a finite-sum problems with $B$ samples where the samples $\{z_1, \ldots, z_B\}$ are independently drawn according to the probability distribution $P$. Thus we define the surrogate function $\hat{f}$

$$\hat{f}(x) = \frac{1}{B}\sum_{i=1}^B \tilde{f}(x, z_i), \tag{10}$$

as the finite-sum approximation of $f(x)$. As a result, we can apply QFW on $\hat{f}$, thus optimize $f$ approximately. The algorithm is outlined in Algorithm 2

In the non-convex setting, if we further assume that $\tilde{f}(x, z)$ is $G$-Lipschitz for all $z \in \mathbb{R}^q$, then we have the following lemma:

**Lemma 3 (Theorem 5 of [Reddi et al., 2016])** *If we define $\hat{\mathcal{G}}(x) = \max_{v \in \mathcal{K}} \langle v - x, -\nabla \hat{f}(x) \rangle$, then $\mathbb{E}[\mathcal{G}(x) - \hat{\mathcal{G}}(x)] \leq \frac{GD}{\sqrt{B}}$. Recall that $D$ is the diameter of $\mathcal{K}$ as defined in Assumption 1, $\mathcal{G}(x) = \max_{v \in \mathcal{K}} \langle v - x, -\nabla f(x) \rangle$. Thus for the output $x_o$, we have*

$$\mathbb{E}[\mathcal{G}(x_o)] \leq \mathbb{E}[\hat{\mathcal{G}}(x_o)] + \frac{GD}{\sqrt{B}}. \tag{11}$$

Baesd on Theorem 3 and Lemma 3, we have the following theoretical guarantee for stochastic non-convex minimization.

**Theorem 4 (Stochastic Non-Convex)** *Assuming that for all $z \in \mathbb{R}^q$, $\tilde{f}(x,z)$ is G-Lipschitz, L-smooth, and satisfies $|\tilde{f}(x,z)| \leq M_0$ for all $x \in \mathcal{K}$. If we obtain $T$ independent samples of $z_i$, and apply Algorithm 1 on $\hat{f}(x) = \frac{1}{T}\sum_{i=1}^{T} \tilde{f}(x,z_i)$ with $N = T, n = T/M$, and all the other parameters set the same as in Theorem 3, then after $T$ iterations, the output $x_o \in \mathcal{K}$ satisfies*

$$\mathbb{E}[\mathcal{G}(x_o)] \leq \frac{2M_0 + D\sqrt{3L^2D^2 + 2G^2} + \frac{LD^2}{2}}{\sqrt{T}} + \frac{GD}{\sqrt{T}}.$$

We note that the algorithm finds an $\epsilon$-first order stationary point with at most $\mathcal{O}(1/\epsilon^2)$ rounds. The SFO complexity per worker is $\mathcal{O}(\sqrt{N}/(\sqrt{M}\epsilon^2)) = \mathcal{O}(\frac{1}{\epsilon^3\sqrt{M}})$. Thus the SFO complexity per worker is linear in $1/\sqrt{M}$, which implicates that increasing the number of workers $M$ will decrease the SFO complexity per worker.

## 5 Experiments

We evaluate the performance of algorithms by visualizing their loss $f(x_t)$ vs. the number of transmitted bits. The experiments were performed on 20 Intel Xeon E5-2660 cores and thus the number of workers is 20. For each curve in the figures below, we ran at least 50 repeated experiments, and the height of shaded regions represents two standard deviations.

In our first setup, we consider a multinomial logistic regression problem. Consider the dataset $\{(x_i, y_i)\}_{i=1}^N \subseteq \mathbb{R}^d \times \{1, \ldots, C\}$ with $N$ samples that have $C$ different labels. We aim to find a model $w$ to classify these sample points under the condition that the solution has a small $\ell_1$-norm. Therefore, we aim to solve the following convex problem

$$\min_w f(w) := -\sum_{i=1}^{N}\sum_{c=1}^{C} 1\{y_i = c\} \log \frac{\exp(w_c^\top x_i)}{\sum_{j=1}^{C}\exp(w_j^\top x_i)}, \quad \text{subject to} \quad \|w\|_1 \leq 1. \tag{12}$$

In our experiments, we use the MNIST dataset and assume that each worker stores 3000 images. Therefore, the overall number of samples in the training set is $N = 60000$.

In our second setup, our goal is to minimize the loss of a three-layer neural network under some conditions on the norm of the solution. Before stating the problem precisely, let us define the log-loss function as $h(y,p) \triangleq -\sum_{c=1}^{C} 1\{y = c\} \log p_c$ for $y \in \{1, \ldots, C\}$ and a $C$-dimensional probability vector $p := (p_1, \cdots, p_C)$. We aim to solve the following non-convex problem

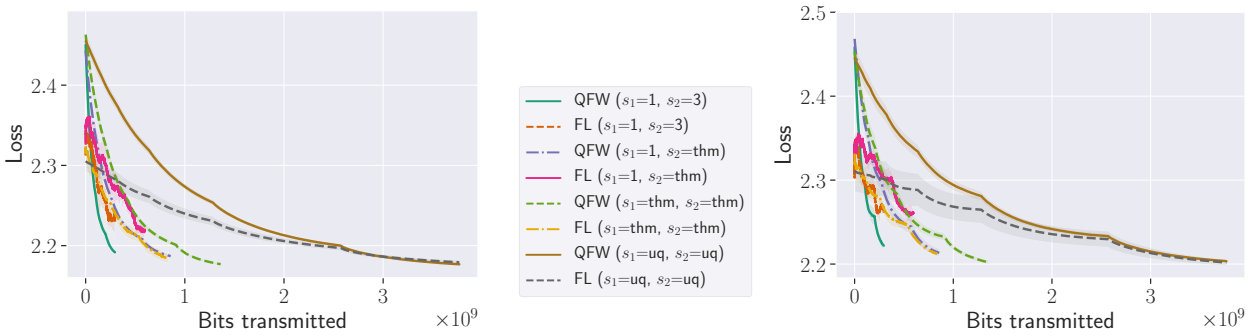
$$\min_{W_1, W_2} \sum_{i=1}^{N} h(y_i, \phi(W_2\, \mathrm{relu}(W_1 x_i + b) + b_2)), \quad \text{subject to} \quad \|W_i\|_1 \leq a_1, \|b_i\|_1 \leq a_2, \tag{13}$$

where $\mathrm{relu}(x) \triangleq \max\{0, x\}$ is the ReLU function and $\phi$ is the softmax function. The imposed $\ell_1$ constraint on the weights leads to a sparse network. We further remark that Frank-Wolfe methods are suitable for training a neural network subject to an $\ell_1$ constraint as they are equivalent to a dropout regularization [Ravi et al., 2018]. In our setup, the size of matrices $W_1$ and $W_2$ are $784 \times 50$ and $50 \times 10$, respectively, and the constraints parameters are $a_1 = a_2 = 5$.

For all of the considered settings, we vary the quantization level, use the $s_1$-partition encoding scheme when workers send encoded tensors to the master and use the $s_2$-partition encoding scheme when the master broadcasts encoded tensors to the workers ($s_i = uq$ indicates FW without quantization and $s_i = thm$ indicates `QFW` with the quantization level recommended by our theorems, where $i = 1, 2$). We also propose the federated learning approach `FL`, an effective heuristic based on `QFW`, where each worker performs its local Frank-Wolfe update autonomously without communicating with each other and synchronizes the model only at the end of each round. This method may not enjoy the strong theoretical guarantees of `QFW` and we observe in our experiments that it is even prone to divergence. In stochastic minimization, each worker samples 1000 images uniformly at random and without replacement.

In Figure 2, we observe the performance of `FL`, FW without quantization, and different variants of `QFW` for solving the multinomial logistic regression problem in (12). The stochastic minimization is presented in Figure 2a and the finite-sum minimization is shown in Figure 2b. We observe that `QFW` with `Partition Encoding Scheme` ($s_1 = 1, s_2 = 3$) has the best performance in terms of the amount of transmitted bits. Specifically, `QFW` with `Partition Encoding Scheme` ($s_1 = 1, s_2 = 3$) requires $3 \times 10^8$ bits to hit the lowest loss in Figures 2a and 2b, while `FL` with the same level of quantization only achieves a suboptimal loss (approximately 2.24) with the same amount of communication. Furthermore, FW without quantization requires more than $3.8 \times 10^9$ bits to reach the same error, *i.e.*, quantization reduces communication load by at least an order of magnitude.

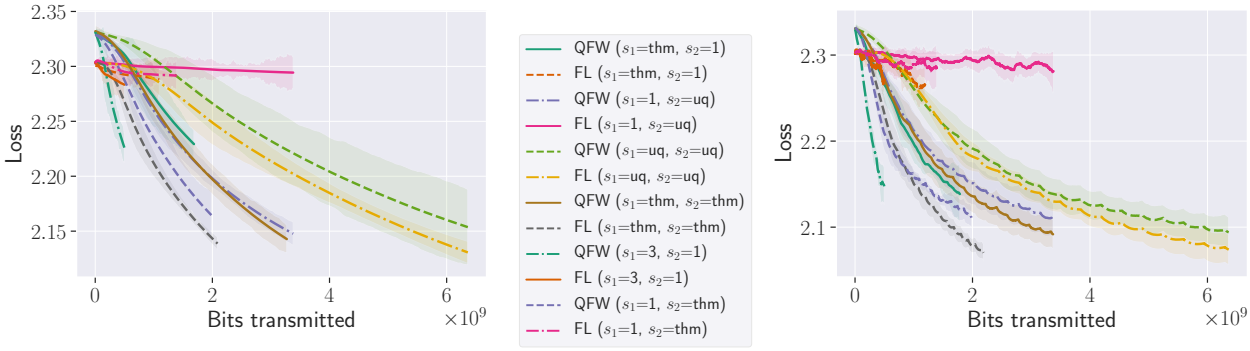
Figure 3 demonstrates the performance of `FL`, FW without quantization, and different variants of `QFW` for solving the three-layer neural network in (13). Again we show the stochastic minimization on the left (Figure 3a) and the finite-sum minimization on the right (Figure 3b). We observe four divergent curves of the federated learning method `FL` ($s_1 = 3, s_2 = 1$; $s_1 = thm, s_2 = 1$; $s_1 = 1, s_2 = thm$; and $s_1 = 1, s_2 = uq$), while all `QFW` curves converge. This observation is in accordance with the fact that `FL` has no theoretical guarantee, in contrast to the proposed `QFW` method. FW




(a) Stochastic convex minimization

(b) Finite-sum convex minimization.

Figure 2: Comparison in terms of the loss versus the number of transmitted bits for a multinomial logistic regression problem. The best performance belongs to `QFW` with `Partition Encoding Scheme` ($s_1 = 1, s_2 = 3$), and FW without quantization has the worst performance.




(a) Stochastic non-convex minimization

(b) Finite-sum non-convex minimization.

Figure 3: Comparison of algorithms in terms of the loss versus the number of transmitted bits for a three-layer neural network. FW without quantization ($s = uq$) significantly underperforms the quantized FW methods. We observe four divergent curves of the federated learning method `FL` ($s_1 = 3, s_2 = 1$; $s_1 = thm, s_2 = 1$; $s_1 = 1, s_2 = thm$; and $s_1 = 1, s_2 = uq$).

without quantization consumes approximately $6.5 \times 10^9$ bits to achieve the lowest loss. Its amount of communication is twice that of `QFW` with the quantization levels recommended by Theorems 3 and 4.

In both convex and non-convex setups, the theoretically guaranteed quantization levels recommended by our theorems may be conservative. In fact, a `Partition Encoding Scheme` with partitions much fewer than our theorems recommend achieves a similar loss level and saves even more communication bits. For example, `QFW` with $s_1 = 1, s_2 = 3$ and $s_1 = 1, s_2 = thm$ exhibits a higher communication efficiency than `QFW` with $s_1 = thm, s_2 = thm$ in Figures 2a and 2b.

We provide more experimental results in the longer version of this paper [Zhang et al., 2019a].

## 6 Conclusion

In this paper, we developed `Quantized Frank-Wolfe` (QFW), the first general-purpose projection-free and communication-efficient framework for constrained optimization. Along with proposing various quantization schemes, QFW can address both convex and non-convex optimization settings in stochastic and finite-sum cases. We provided theoretical guarantees on the convergence rate of QFW and validated its efficiency empirically on training multinomial logistic regression and neural networks. Our theoretical results highlighted the importance of variance reduction techniques to stabalize FW and achieve a sweet trade-off between the communication complexity and convergence rate in distributed settings. We also note that it might be possible to design simpler Quantized FW methods based on the new developments [Zhang et al., 2019b].

## Acknowledgements

Lin Chen is supported by Google PhD Fellowship. Hamed Hassani is supported by AFOSR Award 19RT0726, NSF HDR TRIPODS award 1934876, NSF award CPS-1837253, NSF award CIF-1910056, and NSF CAREER award CIF-1943064. Amin Karbasi is partially supported by NSF (IIS-1845032), ONR (N00014-19-1-2406), and AFOSR (FA9550-18-1-0160).

## References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pages 1709–1720, 2017.

Aurélien Bellet, Yingyu Liang, Alireza Bagheri Garakani, Maria-Florina Balcan, and Fei Sha. A distributed frank-wolfe algorithm for communication-efficient sparse learning. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 478–486. SIAM, 2015.

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Anima Anandkumar. signsgd: compressed optimisation for non-convex problems. *arXiv preprint arXiv:1802.04434*, 2018.

Leonard Berrada, Andrew Zisserman, and M Pawan Kumar. Deep frank-wolfe for neural network optimization. *arXiv preprint arXiv:1811.07591*, 2018.

Trishul M Chilimbi, Yutaka Suzue, Johnson Apacible, and Karthik Kalyanaraman. Project adam: Building an efficient and scalable deep learning training system. In *OSDI*, volume 14, pages 571–582, 2014.

Christopher M De Sa, Ce Zhang, Kunle Olukotun, and Christopher Ré. Taming the wild: A unified analysis of hogwild-style algorithms. In *Advances in neural information processing systems*, pages 2674–2682, 2015.

Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Advances in Neural Information Processing Systems*, pages 687–697, 2018.

Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics (NRL)*, 3(1-2):95–110, 1956.

Dan Garber and Elad Hazan. Faster rates for the frank-wolfe method over strongly-convex sets. *arXiv preprint arXiv:1406.1305*, 2014.

Dan Garber and Elad Hazan. Faster rates for the frank-wolfe method over strongly-convex sets. In *ICML*, volume 15, pages 541–549, 2015.

Hamed Hassani, Amin Karbasi, Aryan Mokhtari, and Zebang Shen. Stochastic continuous greedy++: When upper and lower bounds match. In *Advances in Neural Information Processing Systems*, pages 13066–13076, 2019.

Elad Hazan and Haipeng Luo. Variance-reduced and projection-free stochastic optimization. In *ICML*, pages 1263–1271, 2016.

Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML*, pages 427–435, 2013.

Simon Lacoste-Julien. Convergence rate of frank-wolfe for non-convex objectives. *arXiv preprint arXiv:1607.00345*, 2016.

Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of frank-wolfe optimization variants. In *Advances in Neural Information Processing Systems*, pages 496–504, 2015.

Jean Lafond, Hoi-To Wai, and Eric Moulines. D-fw: Communication efficient distributed algorithms for high-dimensional sparse optimization. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4144–4148. IEEE, 2016.

Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Stochastic conditional gradient methods: From convex minimization to submodular maximization. *arXiv preprint arXiv:1804.09554*, 2018a.

Aryan Mokhtari, Asuman Ozdaglar, and Ali Jadbabaie. Escaping saddle points in constrained optimization. In *Advances in Neural Information Processing Systems*, pages 3629–3639, 2018b.

Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Sarah: A novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2613–2621. JMLR. org, 2017a.

Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Stochastic recursive gradient algorithm for nonconvex optimization. *arXiv preprint arXiv:1705.07261*, 2017b.

Lam M Nguyen, Marten van Dijk, Dzung T Phan, Phuong Ha Nguyen, Tsui-Wei Weng, and Jayant R Kalagnanam. Optimal finite-sum smooth non-convex optimization with sarah. *arXiv preprint arXiv:1901.07648*, 2019.

Wei Ping, Qiang Liu, and Alexander T Ihler. Learning infinite rbms with frank-wolfe. In *Advances in Neural Information Processing Systems*, pages 3063–3071, 2016.

Sathya N Ravi, Tuan Dinh, Vishnu Sai Rao Lokhande, and Vikas Singh. Constrained deep learning using conditional gradient and applications in computer vision. *arXiv preprint arXiv:1803.06453*, 2018.

Sashank J Reddi, Suvrit Sra, Barnabás Póczos, and Alex Smola. Stochastic frank-wolfe methods for nonconvex optimization. *arXiv preprint arXiv:1607.08254*, 2016.

Patrick Schramowski, Christian Bauckhage, and Kristian Kersting. Neural conditional gradients. *arXiv preprint arXiv:1803.04300*, 2018.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 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*, 2014.

Zebang Shen, Cong Fang, Peilin Zhao, Junzhou Huang, and Hui Qian. Complexities in projection-free stochastic non-convex minimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2868–2876, 2019a.

Zebang Shen, Cong Fang, Peilin Zhao, Junzhou Huang, and Hui Qian. Complexities in projection-free stochastic non-convex minimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2868–2876, 2019b.

Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. In *Advances in Neural Information Processing Systems*, pages 4452–4463, 2018.

Nikko Strom. Scalable distributed dnn training using commodity gpu cloud computing. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

Yu-Xiang Wang, Veeranjaneyulu Sadhanala, Wei Dai, Willie Neiswanger, Suvrit Sra, and Eric Xing. Parallel and distributed block-coordinate frank-wolfe algorithms. In *International Conference on Machine Learning*, pages 1548–1557, 2016.

Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. In *Advances in neural information processing systems*, pages 1509–1519, 2017.

Alp Yurtsever, Suvrit Sra, and Volkan Cevher. Conditional gradient methods via stochastic path-integrated differential estimator. In *International Conference on Machine Learning*, pages 7282–7291, 2019.

Mingrui Zhang, Lin Chen, Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Quantized frank-wolfe: Faster optimization, lower communication, and projection free. *arXiv preprint arXiv:1902.06332*, 2019a.

Mingrui Zhang, Zebang Shen, Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. One sample stochastic frank-wolfe. *arXiv preprint arXiv:1910.04322*, 2019b.

Wenpeng Zhang, Peilin Zhao, Wenwu Zhu, Steven CH Hoi, and Tong Zhang. Projection-free distributed online learning in networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 4054–4062. JMLR. org, 2017.

Wenjie Zheng, Aurélien Bellet, and Patrick Gallinari. A distributed frank–wolfe framework for learning low-rank matrices with the trace norm. *Machine Learning*, 107(8-10):1457–1475, 2018.

Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic nested variance reduction for nonconvex optimization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 3925–3936. Curran Associates Inc., 2018.