A Decentralized Parallel Algorithm for Training Generative Adversarial Nets

Mingrui Liu[†], **Wei Zhang**[‡], **Youssef Mroueh**[‡], **Xiaodong Cui**[‡], **Jerret Ross**[‡], **Tianbao Yang**[†], **Payel Das**[‡]
[†] Department of Computer Science, The University of Iowa, Iowa City, IA, 52242
[‡] IBM T. J. Watson Research Center, Yorktown Heights, NY, 10598, USA
mingruiliu.ml@gmail.com

Abstract

Generative Adversarial Networks (GANs) are a powerful class of generative models in the deep learning community. Current practice on large-scale GAN training utilizes large models and distributed large-batch training strategies, and is implemented on deep learning frameworks (e.g., TensorFlow, PyTorch, etc.) designed in a centralized manner. In the centralized network topology, every worker needs to either directly communicate with the central node or indirectly communicate with all other workers in every iteration. However, when the network bandwidth is low or network latency is high, the performance would be significantly degraded. Despite recent progress on decentralized algorithms for training deep neural networks, it remains unclear whether it is possible to train GANs in a decentralized manner. The main difficulty lies at handling the nonconvex-nonconcave min-max optimization and the decentralized communication simultaneously. In this paper, we address this difficulty by designing the **first gradient-based decentralized parallel algorithm** which allows workers to have multiple rounds of communications in one iteration and to update the discriminator and generator simultaneously, and this design makes it amenable for the convergence analysis of the proposed decentralized algorithm. Theoretically, our proposed decentralized algorithm is able to solve a class of non-convex non-concave min-max problems with provable non-asymptotic convergence to first-order stationary point. Experimental results on GANs demonstrate the effectiveness of the proposed algorithm.

1 Introduction

Generative Adversarial Networks (GANs) [1] are very effective at modeling high dimensional data, such as images, but are known to be notoriously difficult to train. Recent research on large-scale GAN training by [2] suggests that distributed large-batch training techniques can be beneficial on large models. Their algorithm is based on a centralized network topology [3, 4], in which each worker computes a local stochastic gradient based on its local data and then sends its gradient to a central node. The central node aggregates the local stochastic gradients together, updates its model parameters by first-order methods and then sends the parameters back to each worker. The central node is the busiest node since it needs to communicate with each worker concurrently. This communication is the main bottleneck of centralized algorithms since it could lead to a communication traffic jam when the network bandwidth is low or network latency is high. To address this issue, decentralized algorithms are usually considered as a surrogate when the cost of centralized communication is prohibitively expensive. In decentralized algorithms, instead, each worker only communicates with its neighbors and a central node is not needed. To this end, recently a decentralized algorithm was also designed for training a deep neural network [5]. In addition, decentralized algorithms only require workers to communicate with their trusted neighbors and are usually a good way for maintaining privacy [6, 7, 8].

While decentralized algorithms are beneficial, they are limited from an optimization perspective. All previous decentralized works are designed either for solving convex and non-convex minimization problems [6, 7, 8, 5] or convex-concave min-max problems [9, 10, 11]. However, none of them are directly applicable for non-convex non-concave min-max problems such as GANs. In this paper, we

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

design the first gradient-based decentralized algorithm for solving a class of non-convex non-concave min-max problems with non-asymptotic theoretical convergence guarantees, which we verify with numerical experimentation.

Our problem of interest is to solve the following stochastic optimization problem:

$$\min_{\mathbf{u}} \max_{\mathbf{v}} F(\mathbf{u}, \mathbf{v}) := \mathbb{E}_{\xi \sim \mathcal{D}} \left[f(\mathbf{u}, \mathbf{v}; \xi) \right], \tag{1}$$

where $F(\mathbf{u}, \mathbf{v})$ is possibly non-convex in $\mathbf{u}$ and non-concave in $\mathbf{v}$ while ξ is a random variable following an unknown distribution $\mathcal{D}$. In the context of GANs, $\mathbf{u}$ and $\mathbf{v}$ represent the parameters for the generator and the discriminator respectively. Several works [12, 13, 14, 15] have established a non-asymptotic convergence to an ϵ-first-order stationary point (i.e., a point $(\mathbf{u}, \mathbf{v})$ such that $\|\mathbf{g}(\mathbf{u}, \mathbf{v})\| \leq \epsilon$, where $\mathbf{g}(\mathbf{u}, \mathbf{v}) = [\nabla_{\mathbf{u}} F(\mathbf{u}, \mathbf{v}), -\nabla_{\mathbf{v}} F(\mathbf{u}, \mathbf{v})]^\top$) for a class of nonconvex-nonconcave min-max problems under various assumptions. Other works [16, 17, 18, 19] focus on GAN training and good empirical performance. However, all of them are built upon the single-machine setting. Although the naive centralized parallel algorithm in these works can also apply, it suffers from a high communication cost on the busiest node (e.g., parameter server) and has privacy vulnerabilities. Furthermore, it is nontrivial to design a decentralized parallel algorithm for nonconvex-nonconcave min-max problems. This difficulty is due to decentralized communication only being able to achieve partial consensus among workers, which makes analysis of nonconvex-nonconcave min-max optimization difficult. Our contributions are the following:

- We design a decentralized parallel algorithm called Decentralized Parallel Optimistic Stochastic Gradient (DPOSG) for a class of nonconvex-nonconcave min-max problems, in which both primal and dual variables are updated simultaneously using only first-order information. Our main novelty lies in the design of simultaneous update combined with multiple rounds of decentralized communication, which is the key for our theoretical analysis. This particular design also allows us to utilize the random mixing strategy as proposed in [20, 21] to further improve the performance, which is verified by our experiments.
- Under the similar assumptions in [12, 15], we analyze DPOSG and establish its non-asymptotic convergence to ϵ-first-order stationary point. In addition, our algorithm is communication-efficient since the communication complexity on the busiest node is $O\left(\log(1/\epsilon)\right)$. Although our algorithm is designed for a much more complex nonconvex-nonconcave min-max problem, it has only negligible logarithmic communication complexity when compared to the decentralized algorithm for solving nonconvex minimization problems in [5].
- We empirically demonstrate the effectiveness of the proposed algorithm using a variant of DPOSG implementing Adam updates and show a speedup compared with the single machine baseline for different neural network architectures on several benchmark datasets, including WGAN-GP on CIFAR10 [22] and Self-Attention GAN on ImageNet [23].

2 Related Work

Min-max Optimization and GAN Training Min-max optimization in convex-concave setting was studied thoroughly by a series of seminal works, including the stochastic mirror descent [24], extragradient method [25, 26], dual extrapolation method [27] and stochastic extragradient method [28].

Recently, a wave of studies for min-max optimization without the convexity-concavity assumption has emerged including nonconvex-concave optimization [29, 30, 31, 32] and nonconvex-nonconcave optimization [33, 12, 13, 14, 15]. In addition, there is a line of work attempting to analyze the behavior of min-max optimization algorithms and their applications in training GANs [34, 35, 36, 37, 38, 16, 39, 17, 18, 40, 41, 19, 42, 43]. However, all of these works focus on the single machine setting. Although it is easy to extend some of the works to a centralized parallel version, none of them can be applied in a decentralized setting.

Decentralized Optimization Decentralized optimization algorithms were first studied in [44, 45, 46, 20], where the information is exchanged along the edges in a communication graph. Decentralized algorithms are usually employed to handle the possible failure of the centralized algorithms and to maintain privacy [6, 8]. Several deterministic algorithms are analyzed in the decentralized manner including decentralized gradient descent [47, 48, 8, 49], decentralized dual averaging [50, 51], Alternating Direction Methods of Multipliers [52, 53, 54, 55, 56, 57], decentralized accelerated

coordinate descent [58], and the exact first-order algorithm [59]. Recently several seminal works have been released [60, 61] providing optimal deterministic decentralized first-order algorithms for convex problems.

In large-scale distributed machine learning, people are usually interested in using stochastic gradient methods to update the model parameters. There is a plethora of work trying to analyze decentralized parallel stochastic gradient methods for convex [62, 63, 64, 65, 66] and nonconvex objectives [67, 5, 68, 69, 70, 71, 72, 73, 74, 75]. In particular, Lian et al [5] is the first paper showing that decentralized parallel stochastic gradient is able to outperform its centralized version for nonconvex smooth problems. Besides decentralized communication, several works further consider other techniques to make the decentralized communication more efficient, including allowing asynchrony [68], compression techniques [69, 73, 66], skipping communication rounds [75], and event-triggered communication [76]. For strongly convex objective with finite-sum structure, several decentralized algorithms using variance reduction techniques have also been proposed [77, 78, 79].

However, all of these works are analyzed for minimization problems and none of them can be applied for the class of nonconvex-nonconcave min-max problems as considered in our paper.

Decentralized Optimization for Min-max Problems There are several works considering decentralized min-max optimization where inner maximum function is taken over a set of agents [80] or the objective function is convex-concave [9, 10, 11].

When we were preparing our manuscript, we became aware of a simultaneous and independent work [81] in which another decentralized algorithm for solving a class of nonconvex-nonconcave min-max problems was proposed. In their work, the algorithm uses implicit updates based on the proximal point method [82] and was shown to converge to stationary point and consensus. However, their algorithm is not gradient-based and requires that the sub-problem induced by the proximal point step has a closed-form solution, which is computationally expensive and may not hold in practice. It is unclear whether the analysis in [81] can still work if the sub-problem cannot be solved exactly. In contrast, our algorithm's update rule is simple and does not involve any complicated sub-problem solvers, since it only requires to compute a stochastic gradient and then updates the model parameters in each iteration.

3 Preliminaries and Notations

We use $\|\cdot\|$ to denote the vector ℓ_2 norm or the matrix spectral norm depending on the argument. Define $\mathbf{x} = (\mathbf{u}, \mathbf{v})$, $\mathbf{g}(\mathbf{x}) = [\nabla_{\mathbf{u}} F(\mathbf{u}, \mathbf{v}), -\nabla_{\mathbf{v}} F(\mathbf{u}, \mathbf{v})]^\top$. We say $\mathbf{x}$ is ϵ-first-order stationary point if $\|\mathbf{g}(\mathbf{x})\| \leq \epsilon$. At every point $\mathbf{x}$, we only have access to a noisy observation of $\mathbf{g}$, i.e., $\mathbf{g}(\mathbf{x}; \xi) = [\nabla_{\mathbf{u}} f(\mathbf{u}, \mathbf{v}; \xi), -\nabla_{\mathbf{v}} f(\mathbf{u}, \mathbf{v}; \xi)]^\top$, where ξ is a random variable. In the rest of this paper, we use the term *stochastic gradient* and *gradient* to stand for $\mathbf{g}(\mathbf{x}; \xi)$ and $\mathbf{g}(\mathbf{x})$ respectively.

Throughout the paper, we make the following assumption:

Assumption 1. *(i).* $\mathbf{g}$ *is L-Lipschitz continuous, i.e.* $\|\mathbf{g}(\mathbf{x}_1) - \mathbf{g}(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|$ *for* $\forall \mathbf{x}_1, \mathbf{x}_2$.

(ii). *For* $\forall \mathbf{x}$, $\mathbb{E}\left[\mathbf{g}(\mathbf{x}; \xi)\right] = \mathbf{g}(\mathbf{x})$, $\mathbb{E}\left\|\mathbf{g}(\mathbf{x}; \xi) - \mathbf{g}(\mathbf{x})\right\|^2 \leq \sigma^2$.

(iii). $\|\mathbf{g}(\mathbf{x})\| \leq G$ *for* $\forall \mathbf{x}$.

(iv). *There exists* $\mathbf{x}_*$ *such that* $\langle \mathbf{g}(\mathbf{x}), \mathbf{x} - \mathbf{x}_* \rangle \geq 0$.

Remark: The Assumptions (i), (ii), (iii) are usually made in optimization literature and are standard. The Assumption (iv) is usually used in previous works for solving non-monotone variational inequalities [12] and GAN training [18, 15]. In addition, this assumption holds in some nonconvex minimization problems. For example, it has been shown that this assumption holds in both theory and practice when using SGD for learning neural networks [83, 84, 85].

Single Machine Algorithm Our decentralized algorithm is based on a specific single machine algorithm called Optimistic Stochastic Gradient (OSG) [16, 15] which is designed to solve a class of nonconvex-nonconcave min-max problems. A similar version for convex minimization problem is proposed in [86, 87]. This algorithm keeps two update sequences $\mathbf{z}_k$ and $\mathbf{x}_k$ with the following update rules:

$$\begin{aligned} \mathbf{z}_k &= \mathbf{x}_{k-1} - \eta \mathbf{g}(\mathbf{z}_{k-1}; \xi_{k-1}) \\ \mathbf{x}_k &= \mathbf{x}_{k-1} - \eta \mathbf{g}(\mathbf{z}_k; \xi_k) \end{aligned} \tag{2}$$

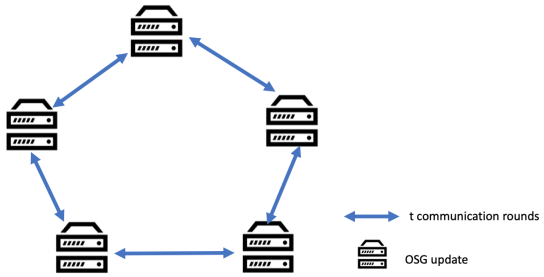

Figure 1: Illustration of DPOSG. Each machine calculates its stochastic gradients and has t communication rounds with its neighbors in parallel. After that each machine conducts OSG update.

It is easy to see that this update is equivalent to the following one line update as in [16]:

$$\mathbf{z}_{k+1} = \mathbf{z}_k - 2\eta \mathbf{g}(\mathbf{z}_k; \xi_k) + \eta \mathbf{g}(\mathbf{z}_{k-1}; \xi_{k-1})$$

4 Decentralized Parallel Optimistic Stochastic Gradient

In this section, inspired by the algorithm in [12, 16, 15] in the single-machine setting, we propose an algorithm named Decentralized Parallel Optimistic Stochastic Gradient (DPOSG), which only allows decentralized communications between workers and there is no central node as in the centralized setting which requires communication with each node concurrently in each iteration. Instead, information is only exchanged between neighborhood nodes in the decentralized setting.

Suppose we have M machines. Denote $W \in \mathbb{R}^{M\times M}$ by a doubly stochastic matrix which satisfies $0 \leq W_{ij} \leq 1$, $W^\top = W$, $\sum_{j=1}^M W_{ij} = 1$ for $i, j = 1, \ldots, M$. In distributed optimization literature, W is used to characterize the decentralized communication topology, in which W_{ij} characterizes the degree of how node j is able to affect node i, and $W_{ij} = 0$ means node i and j are disconnected.

Denote $\lambda_i(\cdot)$ by the i-th largest eigenvalue of W, then we know that $\lambda_1(W) = 1$. In addition, we assume that $\max(|\lambda_2(W)|, |\lambda_M(W)|) < 1$.

Denote $\mathbf{z}_k^i \in \mathbb{R}^{d\times 1}$ (and $\mathbf{x}_k^i \in \mathbb{R}^{d\times 1}$) by the parameters in i-th machine at k-th iteration, and both $\mathbf{z}_k^i$ and $\mathbf{x}_k^i$ have the same shape of trainable parameter of neural networks (the trainable parameters of the discriminator and generator are concatenated together in the GAN setting). Define $Z_k = \left[\mathbf{z}_k^1, \ldots, \mathbf{z}_k^M\right] \in \mathbb{R}^{d\times M}$, $X_k = \left[\mathbf{x}_k^1, \ldots, \mathbf{x}_k^M\right] \in \mathbb{R}^{d\times M}$, $\mathbf{g}(Z_k) = \left[\mathbf{g}(\mathbf{z}_k^1), \ldots, \mathbf{g}(\mathbf{z}_k^M)\right] \in \mathbb{R}^{d\times M}$, $\widehat{\mathbf{g}}(\xi_k, Z_k) = \left[\mathbf{g}(\mathbf{z}_k^1; \xi_k^1), \ldots, \mathbf{g}(\mathbf{z}_k^M; \xi_k^M)\right] \in \mathbb{R}^{d\times M}$, where Z_k, X_k are concatenations of all local variables, $\widehat{\mathbf{g}}(Z_k), \mathbf{g}(Z_k)$ are concatenations of all local unbiased stochastic gradients and their corresponding expectations. The Algorithm is presented Algorithm 1, in which every local worker repeatedly executes the following steps (we use machine i at k-th iteration as an illustrative example):

- **Sampling:** Sample a minibatch according to $\xi_k^i = \left(\xi_k^{i,1}, \xi_k^{i,2}, \ldots, \xi_k^{i,m}\right)$, where m is the minibatch size.
- **Stochastic Gradient Calculation:** Utilize the sampled data to compute the stochastic gradients for both discriminator and generator respectively, which are $\mathbf{g}_\mathbf{u} = \frac{1}{m}\sum_{j=1}^m \nabla_\mathbf{u} f(\mathbf{u}_k, \mathbf{v}_k, \xi_k^{i,j})$, $\mathbf{g}_\mathbf{v} = \frac{1}{m}\sum_{j=1}^m \nabla_\mathbf{v} f(\mathbf{u}_k, \mathbf{v}_k, \xi_k^{i,j})$ respectively. Define $\mathbf{g}(\mathbf{z}_k^i; \xi_k^i) = [\mathbf{g}_\mathbf{u}, -\mathbf{g}_\mathbf{v}]$.
- **Local Averaging and Parameter Update:** Update the model in local memory by $\mathbf{x}_k^i = \tilde{\mathbf{x}}_{k-1}^i - \eta \mathbf{g}(\mathbf{z}_{k-1}^i; \xi_{k-1}^i)$, $\mathbf{z}_k^i = \tilde{\mathbf{x}}_{k-1}^i - \eta \mathbf{g}(\mathbf{z}_k^i; \xi_{k-1}^i)$, where $\tilde{\mathbf{x}}_{k-1}^i$ is calculated via locally averaging the model at $(k-1)$-th iteration over all of its neighbor workers. This local averaging step is done t times according to the matrix W. An illustration of this step is in Figure 1.

We make the following remarks on Algorithm 1:

- In both line 3 and line 4, $X_{k-1}W^t$ is the weight averaging step, which can be implemented in parallel with the stochastic gradient calculation step (evaluating $\widehat{\mathbf{g}}(\xi_{k-1}, Z_{k-1})$ and $\widehat{\mathbf{g}}(\xi_k, Z_k)$). When we encounter a large batch in training deep neural networks, the running

Algorithm 1 Decentralized Parallel Optimistic Stochastic Gradient (DPOSG)

1: **Input:** $Z_0 = X_0 = \mathbf{0}_{d\times M}$
2: **for** $k = 1, \ldots, N$ **do**
3: $\quad Z_k = X_{k-1}W^t - \eta \cdot \widehat{\mathbf{g}}(\xi_{k-1}, Z_{k-1})$
4: $\quad X_k = X_{k-1}W^t - \eta \cdot \widehat{\mathbf{g}}(\xi_k, Z_k)$
5: **end for**

time spent on stochastic gradient calculation usually dominates compared to the weight averaging step during every iteration, so the elapsed time in this case is almost the same as the time spent on the gradient calculation step. This feature makes our algorithm practical and numerically attractive.

- The main differences between the decentralized algorithms for nonconvex-nonconcave min-max problems and minimization problems (e.g., [5]) are two fold. First, we need to introduce two update sequences given in line 3 and line 4 in Algorithm 1, while the algorithm in [5] only requires one update sequence. Second, we need to do the local model averaging t times in each iteration, while one averaging step is sufficient in [5]. Incorporating these ingredients in designing a decentralized algorithm for nonconvex-nonconcave min-max problem is crucial for provable convergence to a stationary point. In addition, we would like to mention that the additional cost and the implementation difficulty incurred by our design are almost negligible compared with [5]. First, the stochastic gradient calculated in line 3 can be reused in the next iteration, which reduces the cost per iteration and shares the similar spirit of one-call stochastic gradient method in [17, 15]. Second, t is only a logarithmic factor of the target accuracy ϵ to ensure the convergence as shown in Theorem 1 presented later, which possesses almost the same communication cost as in the case of communicating once. Third, our algorithm updates the discriminator and generator simultaneously, and this particular design makes it suitable to implement in the decentralized distributed system as in [5].
- When $W \in \mathbb{R}^{M\times M}$ is a matrix whose every entry is $1/M$ and $t = 1$, our Algorithm 1 recovers the Centralized Parallel Optimistic Stochastic Gradient (CPOSG). The same analysis in [15] can be applied in our case and results in $O(\epsilon^{-4})$ computational complexity and $O(\epsilon^{-2})$ communication complexity on the busiest node.
- When $W \in \mathbb{R}^{M\times M}$ is an identity matrix and $M = 1$, Algorithm 1 recovers the single machine version of Optimistic Stochastic Gradient, which is the same as in (2).

Theorem 1. *Suppose Assumption 1 holds and assume* $\|\mathbf{x}_*\| \le \frac{D}{2}$, $\|\bar{\mathbf{z}}_k\| \le \frac{D}{2}$ *hold with some* $D > 0$. *Denote* m *by the size of minibatch used in each machine to estimate the stochastic gradient. Define* $\bar{\mathbf{z}}_k = \frac{1}{M}\sum_{i=1}^{M} \mathbf{z}_k^i$ *and* $\rho = \max(|\lambda_2(W)|, |\lambda_M(W)|) < 1$, *where* $\lambda_i(\cdot)$ *stands for the* i*-th largest eigenvalue. Run Algorithm 1 for* N *iterations, in which* $t \ge \log_{\frac{1}{\rho}}\left(1 + \frac{M\sqrt{mMG^2+\sigma^2}}{4\sigma}\right)$. *Take* $\eta \le \min\left(\frac{1}{6\sqrt{2}L}, \frac{1-\rho^t}{\sqrt{32cML}}, \frac{\sqrt{1-\rho^{2t}}}{4m^{1/4}M^{3/4}L}\right)$ *with* $c = 321$. *Then we have*

$$\frac{1}{N}\sum_{k=0}^{N-1}\mathbb{E}\left\|\mathbf{g}(\bar{\mathbf{z}}_k)\right\|^2 \le 8\left(\frac{\|\mathbf{x}_0 - \mathbf{x}_*\|^2}{\eta^2 N} + \frac{20\sigma^2}{mM} + \frac{48(DL\sigma + \sigma^2)}{\sqrt{mM}}\right).$$

Note that our goal is to make sure that $\frac{1}{N}\sum_{k=0}^{N-1}\mathbb{E}\left\|\mathbf{g}(\bar{\mathbf{z}}_k)\right\|^2 \le \epsilon^2$ and Theorem 1 establishes such a non-asymptotic ergodic convergence. In the single-machine setting, to find the ϵ-stationary point, both the stochastic extra-gradient algorithm in [12] and the OSG algorithm in [15] require the minibatch size to be dependent on ϵ. However, in practice, it is not reasonable to assume m to be dependent on ϵ in single-machine setting since the machine has a memory limit. Handling such a large minibatch could incur some significant system overhead. When m is a constant, independent of ϵ, both the stochastic extragradient algorithm in [12] and the OSG algorithm in [15] cannot be guaranteed to converge to ϵ-stationary point. In the multiple-machines setting, mM is the effective batch size, and we can choose m to be constant and M to be dependent on ϵ (M can be very large, i.e., our algorithm can handle large number of machines). This insight can be summarized in Corollary 1. In addition, the boundedness assumption of $\|\mathbf{x}_*\|$ and $\|\bar{\mathbf{z}}_k\|$ is a realistic assumption in GAN training.

For example, this assumption explicitly holds due to weight clipping in WGAN [88] training, or implicitly holds by using L2 regularization [89].

Corollary 1. *Take $m = O(1)$, $M = O(\epsilon^{-4})$, $N = O(\epsilon^{-8})$ in Theorem 1. To find an ϵ-first-order stationary point, Algorithm 1 has $O(\epsilon^{-12})$ computational complexity and $O(t \times$ degree of the network$) = O(\log(1/\epsilon))$ communication complexity on the busiest node.*

Remark 1 (Non-asymptotic Convergence)**.** *Corollary 1 shows that DPOSG converges to ϵ-stationary point in polynomial time and also enjoys logarithmic communication complexity on the busiest node. The consensus over all nodes can also be achieved due to the logarithmic communication rounds in each iteration.*

Remark 2 (Spectral Gap and Random Mixing Startegy)**.** *The spectral gap ρ depends on the decentralized communication characterized by matrix W. If we use fixed topology where each machine communicates with two neighbors equally, then according to [21], $\rho = \frac{1}{3} + \frac{2}{3}\cos(\frac{2\pi}{M})$ when $M \geq 4$. When using random mixing strategy as in [21], every machine communicates with two random machines each time, and then ρ is much smaller. It implies that t can be chosen to be smaller according to Theorem 1. It further indicates that the Algorithm 1 requires less number of communication rounds in each iteration.*

We also want to mention that the logarithmic communication complexity does not hold for general ring communication topology, but it indeeds holds for the compelete graph case as studied in [21]. For a fixed ring topology, ρ is close to 1 when M is large, and in this case the per-iteration communication complexity is no longer logarithmic. However, we want to emphasize that it is indeed logarithmic in ϵ when using the random mixing strategy with a complete graph as in Rand-DP-OAdam in our experiment, in which any two nodes are connected and each node randomly selects two neighbors to communicate t times in each iteration. In this case, it is shown in [21] that $\mathbb{E}\left\|W_1 \dots W_t - \frac{\mathbf{1}_M \mathbf{1}_M^\top}{M}\right\|_2 \leq \frac{\sqrt{M-1}}{(\sqrt{3})^t}$, where W_i represents the communication topology at the i-th time, and $\mathbf{1}_M$ stands for a M-dimensional vector with all entries being 1. To ensure that RHS$= \frac{\sqrt{M-1}}{(\sqrt{3})^t} \leq \epsilon$, we only need $t = O(\log(1/\epsilon))$ when $M = poly(1/\epsilon)$. Using the random mixing strategy is compatible with the fixed topology as in the proof of Theorem 1, since the two sources of randomness (gradient noise, random mixing) can be decoupled.

4.1 Sketch of the Proof of Theorem 1

We present a high level description here and the detailed proof can be found in Appendix A.4. The key idea in our proof is to approximate the dynamics of a decentralized update to the centralized counterpart, which is of vital importance to establish the convergence of our algorithm. Lemma 1, 2 and 3 are introduced to show how far the decentralized algorithm is away from the centralized counterpart, with proofs included in Appendix A.3. With these lemmas and more refined analysis, we can establish the non-asymptotic convergence of our algorithm. Before introducing those lemmas we introduce the following notations:

Notations Define $\epsilon_k^i = \mathbf{g}(\mathbf{z}_k^i; \xi_k^i) - \mathbf{g}(\mathbf{z}_k^i)$, $\widehat{\mathbf{g}}(\epsilon_k^i, \mathbf{z}_k^i) = \mathbf{g}(\mathbf{z}_k^i; \xi_k^1)$, $\widehat{\mathbf{g}}(\epsilon_k, Z_k) = \left[\mathbf{g}(\mathbf{z}_k^1; \xi_k^1), \dots, \mathbf{g}(\mathbf{z}_k^M; \xi_k^M)\right] \in \mathbb{R}^{d \times M}$, $\bar{\mathbf{z}}_k = \frac{1}{M}\sum_{i=1}^M \mathbf{z}_k^i$. Define $\mathbf{e}_i = [0, \dots, 1, \dots, 0]^\top$, the i-th Canonical basis vector. Denote $\mathbf{1}_M$ by a vector of length M whose every entry is 1.

Lemma 1 bounds the squared error between the average of individual gradients on each machine and the gradient evaluated at the averaged weight.

Lemma 1. *By taking $\eta = \min\left(\frac{1-\rho^t}{\sqrt{32cML}}, \frac{\sqrt{1-\rho^{2t}}}{4m^{1/4}M^{3/4}L}\right)$ with $c \geq 2$, we have*

$$\frac{1}{N}\sum_{k=0}^{N-1}\mathbb{E}\left\|\frac{1}{M}\mathbf{g}(Z_k)\mathbf{1}_M - \mathbf{g}(\bar{\mathbf{z}}_k)\right\|^2 \leq \frac{\sigma^2}{\sqrt{mM}} + \frac{1}{c-1}\cdot\frac{1}{N}\sum_{k=0}^{N-1}\mathbb{E}\left\|\mathbf{g}\left(\bar{\mathbf{z}}_k\right)\right\|^2.$$

The purpose of Lemma 2 is to establish an upper bound for the averaged expected ℓ_2 error between the averaged stochastic gradient and the individual stochastic gradient on each machine.

Lemma 2. *The following inequality holds for the stochastic gradient:*

$$\frac{1}{M}\sum_{i=1}^M\mathbb{E}\left[\left\|\frac{1}{M}\widehat{\mathbf{g}}(\epsilon_{k-1}, Z_{k-1})\mathbf{1}_M - \widehat{\mathbf{g}}(\epsilon_{k-1}, Z_{k-1})\mathbf{e}_i\right\|\right] \leq \frac{2\sigma}{\sqrt{mM}} + 2\mathbb{E}\left[\frac{1}{M}\sum_{i=1}^M\left\|\mathbf{g}(\mathbf{z}_{k-1}^i) - \mathbf{g}(\bar{\mathbf{z}}_{k-1})\right\|\right].$$

Finally Lemma 3 bounds the averaged ℓ_2 error between the individual gradient on each machine and the gradient evaluated at the averaged weight.

Lemma 3. *Define* $\mu_k = \frac{1}{M}\sum_{i=1}^{M}\left\|\mathbf{g}(\mathbf{z}_k^i) - \mathbf{g}(\bar{\mathbf{z}}_k)\right\|$. *Suppose* $\eta < \frac{1}{4L}$ *and* $t \geq \log_{\frac{1}{\rho}}\left(1 + \frac{M\sqrt{mMG^2+\sigma^2}}{4\sigma}\right)$. *We have* $\frac{1}{N}\sum_{k=0}^{N-1}\mathbb{E}\left[\mu_k\right] < \frac{8\eta L\sigma}{\sqrt{mM}(1-4\eta L)}$.

Note that the ℓ_2 errors in Lemma 2 and Lemma 3 do not appear in the analysis of decentralized algorithms for minimization problems [5]. However, for the nonconvex-nonconcave min-max problem we consider, we need to carefully bound this ℓ_2 error, which requires t rounds of local decentralized communication.

5 Experiments

5.1 Experimental Settings

Although our convergence is proved for DPOSG which is not an adaptive algorithm, we implement an adaptive gradient variant of DPOSG implementing Adam updates and its decentralized versions in our experiments, since they provide better empirical performance [16, 15]. We implemented three algorithms: Centralized Parallel Optimistic Adam (CP-OAdam), Decentralized Parallel Optimistic Adam (DP-OAdam) and Randomization Decentralized Parallel Adam (Rand-DP-OAdam) inspired by [20, 21]. We use the term 'learner' to represent 'GPU' in our experiment.

In CP-OAdam, after each minibatch update, every learner sums their weights and calculates the average, which is used as the new set of weights. The summation is implemented using an all-reduce call [1].

We implemented the DP-OAdam algorithm similar to [5]: We arrange all the learners in a communication ring. After each mini-batch update, each learner sends its weights to its left neighbor and right neighbor and sets the average of its weight and the weights of its neighbors as its new weight. In addition, we overlap the gradients computation with the weights exchanging and averaging to further improve run-time performance. An implicit barrier is enforced at the end of each iteration so that every learner advances in a lock-step. Finally, one noteworthy implementation detail is that we arrange the weights update steps for both generator and discriminator such that they occur immediately together. In this way, we could treat the union of two networks as one entire network and plug it in the system that deals with the single objective function as originally proposed in [5].

In Rand-DP-OAdam, we follow the same implementation as in DP-OAdam except that in each iteration each learner randomly selects two neighbors to communicate its weights instead of using a fixed communication topology as in DP-OAdam.

We consider two experiments. The first one is WGAN-GP [22] on CIFAR10 dataset, and the second one is Self-Attention GAN [23] on ImageNet dataset. In each worker, we store both the discriminator and the generator, in which these two neural networks are simultaneously updated in our algorithm. The size of the combined generator and discriminator for WGAN-GP with CIFAR10 is 8.36MB, while the model for Self-Attention GAN with ImageNet is 315.07MB. For both experiments, we tune the learning rate in $\{1\times10^{-3}, 4\times10^{-4}, 2\times10^{-4}, 1\times10^{-4}, 4\times10^{-5}, 2\times10^{-5}, 1\times10^{-5}\}$ and choose the one which delivers the best performance for the centralized baseline (CP-OAdam), and decentralized algorithms (DP-OAdam, Rand-DP-OAdam) are using the same learning rate as CP-OAdam. For Self-Attention GAN on ImageNet, we tune different learning rates for discriminator and generator respectively and choose to use 10^{-3} for generator and 4×10^{-5} for the discriminator. We fix the total batch size as 256 (i.e. the product of batch size per learner and number of learners is 256).

5.2 Convergence and Speedup results in HPC environment

In this section, we conduct experiments in a HPC environment where the network has low latency (1 μs). We compare convergence and speedup results of DP-OAdam and Rand-DP-OAdam with the centralized baseline CP-OAdam, which is presented in Figure 2. In the two figures on top, the x-axis is number of epochs and the y-axis is the Inception Score (IS-score). In the two figures on bottom, the x-axis is the number of learners and the y-axis is speedup. Since the models on each learner are different at any time, the orange band shows the lowest IS across learners and the highest

[1] all-reduce is a reduction operation followed by a broadcast operation. An reduction operation is both associative and commutative (e.g., summation).

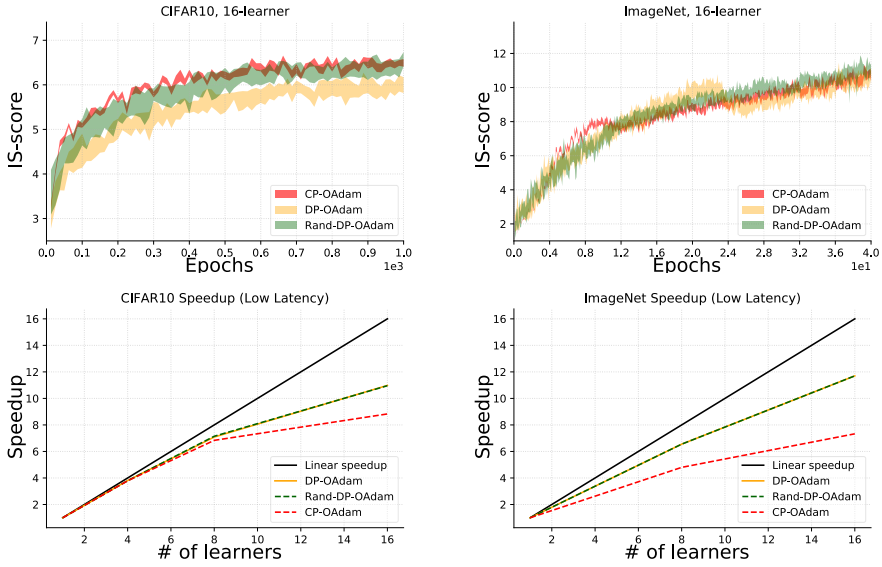

Figure 2: Convergence and speedup comparison between CP-OAdam, DP-OAdam and Rand-DP-OAdam for 16 learners on CIFAR10 and ImageNet. In terms of epochs, DP-OAdam matches the CP-OAdam convergence, and Rand-DP-OAdam further improves over DP-OAdam. Both DP-OAdam and Rand-DP-OAdam have better speedup than CP-OAdam.

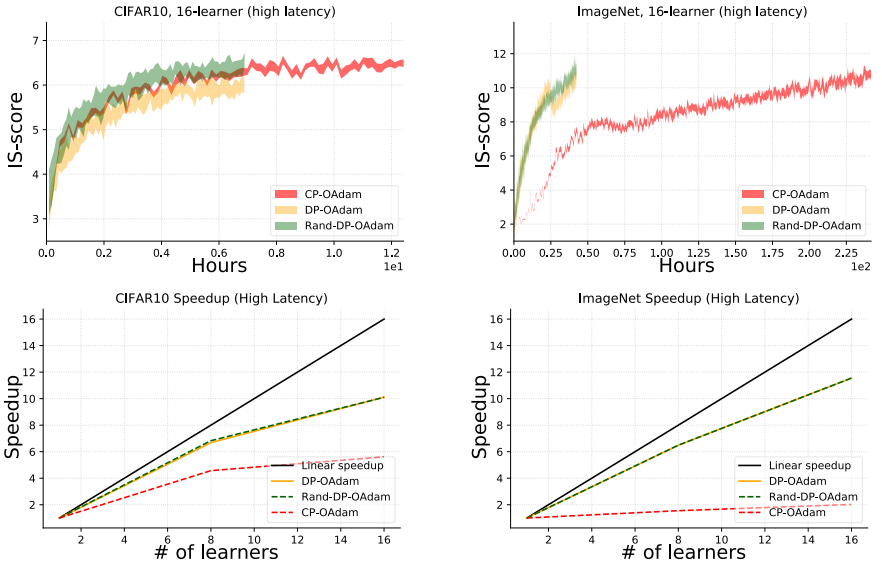

Figure 3: Run-time and speedup comparison for CIFAR10 and ImageNet in a high latency environment. Both DP-OAdam and Rand-DP-OAdam significantly outperform CP-OAdam in terms of both run-time and speedup.

IS across learners when measured for DP-OAdam. Similarly, the green (red) band shows the IS range for Rand-DP-OAdam (CP-OAdam) respectively.

As we can see from Figure 2, in terms of epochs, DP-OAdam for decentralized Parallel GAN training have similar convergence speed as its centralized counterpart (CP-OAdam). Rand-DP-OAdam further improves DP-OAdam significantly due to the usage of random mixing strategy. In terms of speedup, decentralized algorithms (DP-OAdam and Rand-DP-OAdam) are much faster than its centralized counterpart (CP-OAdam). Detailed experiment settings and more experimental results can be found in Appendix B. We also include the generated images in Appendix C.

5.3 Run-time and Speedup Results in Cloud Environment

Low-latency network is common in HPC environment whereas high-latency network is common in commodity cloud systems. In this section, we report run-time and speedup results of CIFAR10 and ImageNet experiments on a cloud computing environment where the network has high-latency ($1ms$). The results are presented in Figure 3. Decentralized algorithms (DP-OAdam, Rand-DP-OAdam) deliver significantly better run-time and speedup than CP-OAdam in the high-latency network setting. Allreduce relies on chunking a large message to smaller pieces to enable software pipe-lining to achieve optimal throughput [90], which inevitably leads to many more hand-shake messages on the network than decentralized algorithms and worse performance when latency is high [5, 68, 70]. We recommend practitioners to consider DP-OAdam when deploying Training as a Service system [91] on cloud.

6 Conclusion

In this paper, we have introduced a decentralized parallel algorithm for training GANs. Theoretically, our decentralized algorithm is proven to have non-asymptotic convergence guarantees for a class of nonconvex-nonconcave min-max problems. Empirically, our proposed decentralized algorithms are shown to significantly outperform its centralized counterpart and deliver good empirical performance on GAN training tasks on CIFAR10 and ImageNet.

Acknowledgment

We thank anonymous reviewers for their constructive comments. This work was partially supported by NSF #1933212, NSF CAREER Award #1844403.

Broader Impact

In this paper, researchers introduce a decentralized parallel algorithm for training Generative Adversarial Nets (GANs). The proposed scheme can be proved to have a non-asymptotic convergence to first-order stationary points in theory, and outperforms centralized counterpart in practice.

Our proposed decentralized algorithm is a class of foundational research, since the algorithm design and analysis are proposed for a general class of nonconvex-nonconcave min-max problems and not necessarily restricted for training GANs. Both the algorithm design and the proof techniques are novel, and it may inspire future research along this direction.

Our decentralized algorithm has broader impacts in a variety of machine learning tasks beyond GAN training. For example, our algorithm is promising in other machine learning problems whose objective function has a min-max structure, such as adversarial training [92], robust machine learning [93], etc.

Our decentralized algorithm can be applied in several real-world applications such as image-to-image generation [94], text-to-image generation [95], face aging [96], photo inpainting [97], dialogue systems [98], etc. In all these applications, GAN training is an indispensable backbone. Training GANs in these applications usually requires to leverage centralized large batch distributed training which could suffer from inefficiency in terms of run-time, and our algorithm is able to address this issue by drastically reducing the running time in the whole training process.

These real-world applications have a broad societal implications. First, it can greatly help people's daily life. For example, many companies provide online service, where an AI chatbot is usually utilized to answer customer's questions. However, the existing chatbot may not be able to fully understand customer's question and its response is usually not good enough. One can adopt our decentralized algorithms to efficiently train a generative adversarial network based on the human-to-human chatting history, and the learned model is expected to answer customer's questions in a better manner. This system can help customers and significantly enhance users' satisfaction. Second, it can help protect users' privacy. One benefit of decentralized algorithms is that it does not need the central node to collect all users' information and every node only communicates with its trusted neighbors. In this case, our proposed decentralized algorithms naturally preserve users' privacy.

We encourage researchers to further investigate the merits and shortcomings of our proposed approach. In particular, we recommend researchers to design new algorithms for training GANs with faster convergence guarantees.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[3] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pages 1223–1231, 2012.

[4] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pages 583–598, 2014.

[5] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 5330–5340, 2017.

[6] S Sundhar Ram, A Nedić, and Venugopal V Veeravalli. Asynchronous gossip algorithms for stochastic optimization. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 3581–3586. IEEE, 2009.

[7] Feng Yan, Shreyas Sundaram, SVN Vishwanathan, and Yuan Qi. Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2483–2493, 2012.

[8] Kun Yuan, Qing Ling, and Wotao Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

[9] Alec Koppel, Felicia Y Jakubiec, and Alejandro Ribeiro. A saddle point algorithm for networked online convex optimization. *IEEE Transactions on Signal Processing*, 63(19):5149–5164, 2015.

[10] David Mateos-Núnez and Jorge Cortés. Distributed subgradient methods for saddle-point problems. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 5462–5467. IEEE, 2015.

[11] Hoi-To Wai, Zhuoran Yang, Zhaoran Wang, and Mingyi Hong. Multi-agent reinforcement learning via double averaging primal-dual optimization. In *Advances in Neural Information Processing Systems*, pages 9649–9660, 2018.

[12] AN Iusem, Alejandro Jofré, Roberto I Oliveira, and Philip Thompson. Extragradient method with variance reduction for stochastic variational inequalities. *SIAM Journal on Optimization*, 27(2):686–724, 2017.

[13] Qihang Lin, Mingrui Liu, Hassan Rafique, and Tianbao Yang. Solving weakly-convex-weakly-concave saddle-point problems as weakly-monotone variational inequality. *arXiv preprint arXiv:1810.10207*, 2018.

[14] Maziar Sanjabi, Meisam Razaviyayn, and Jason D Lee. Solving non-convex non-concave min-max games under polyak-{\L} ojasiewicz condition. *arXiv preprint arXiv:1812.02878*, 2018.

[15] Mingrui Liu, Youssef Mroueh, Jerret Ross, Wei Zhang, Xiaodong Cui, Payel Das, and Tianbao Yang. Towards better understanding of adaptive gradient algorithms in generative adversarial nets. In *International Conference on Learning Representations*, 2020.

[16] Constantinos Daskalakis, Andrew Ilyas, Vasilis Syrgkanis, and Haoyang Zeng. Training gans with optimism. *arXiv preprint arXiv:1711.00141*, 2017.

[17] Gauthier Gidel, Hugo Berard, Gaëtan Vignoud, Pascal Vincent, and Simon Lacoste-Julien. A variational inequality perspective on generative adversarial networks. *arXiv preprint arXiv:1802.10551*, 2018.

[18] Panayotis Mertikopoulos, Houssam Zenati, Bruno Lecouat, Chuan-Sheng Foo, Vijay Chandrasekhar, and Georgios Piliouras. Mirror descent in saddle-point problems: Going the extra (gradient) mile. *arXiv preprint arXiv:1807.02629*, 2018.

[19] Tatjana Chavdarova, Gauthier Gidel, Francois Fleuret, and Simon Lacoste-Julien. Reducing noise in gan training with variance reduced extragradient. *arXiv preprint arXiv:1904.08598*, 2019.

[20] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE/ACM Transactions on Networking (TON)*, 14(SI):2508–2530, 2006.

[21] Wei Zhang, Xiaodong Cui, Abdullah Kayi, Mingrui Liu, Ulrich Finkler, Brian Kingsbury, George Saon, Youssef Mroueh, Alper Buyuktosunoglu, Payel Das, David Kung, and Michael Picheny. Improving efficiency in large-scale decentralized distributed training. In *ICASSP'2020*, May 2020.

[22] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[23] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

[24] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.

[25] GM Korpelevich. The extragradient method for finding saddle points and other problems. *Matecon*, 12:747–756, 1976.

[26] Arkadi Nemirovski. Prox-method with rate of convergence o (1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004.

[27] Yurii Nesterov. Dual extrapolation and its applications to solving variational inequalities and related problems. *Mathematical Programming*, 109(2-3):319–344, 2007.

[28] Anatoli Juditsky, Arkadi Nemirovski, Claire Tauvel, et al. Solving variational inequalities with stochastic mirror-prox algorithm. *Stochastic Systems*, 1(1):17–58, 2011.

[29] Hassan Rafique, Mingrui Liu, Qihang Lin, and Tianbao Yang. Non-convex min-max optimization: Provable algorithms and applications in machine learning. *arXiv preprint arXiv:1810.02060*, 2018.

[30] Tianyi Lin, Chi Jin, and Michael I Jordan. On gradient descent ascent for nonconvex-concave minimax problems. *arXiv preprint arXiv:1906.00331*, 2019.

[31] Songtao Lu, Ioannis Tsaknakis, and Mingyi Hong. Block alternating optimization for non-convex min-max problems: algorithms and applications in signal processing and communications. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4754–4758. IEEE, 2019.

[32] Kiran K Thekumparampil, Prateek Jain, Praneeth Netrapalli, and Sewoong Oh. Efficient algorithms for smooth minimax optimization. In *Advances in Neural Information Processing Systems*, pages 12659–12670, 2019.

[33] Cong D Dang and Guanghui Lan. On the convergence properties of non-euclidean extragradient methods for variational inequalities with generalized monotone operators. *Computational Optimization and applications*, 60(2):277–310, 2015.

[34] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[35] Constantinos Daskalakis and Ioannis Panageas. The limit points of (optimistic) gradient descent in min-max optimization. In *Advances in Neural Information Processing Systems*, pages 9236–9246, 2018.

[36] Eric V Mazumdar, Michael I Jordan, and S Shankar Sastry. On finding local nash equilibria (and only local nash equilibria) in zero-sum games. *arXiv preprint arXiv:1901.00838*, 2019.

[37] Vaishnavh Nagarajan and J Zico Kolter. Gradient descent gan optimization is locally stable. In *Advances in Neural Information Processing Systems*, pages 5585–5595, 2017.

[38] Paulina Grnarova, Kfir Y Levy, Aurelien Lucchi, Thomas Hofmann, and Andreas Krause. An online learning approach to generative adversarial networks. *arXiv preprint arXiv:1706.03269*, 2017.

[39] Abhay Yadav, Sohil Shah, Zheng Xu, David Jacobs, and Tom Goldstein. Stabilizing adversarial nets with prediction methods. *arXiv preprint arXiv:1705.07364*, 2017.

[40] Tengyuan Liang and James Stokes. Interaction matters: A note on non-asymptotic local convergence of generative adversarial networks. In *AISTATS*, 2019.

[41] Gauthier Gidel, Reyhane Askari Hemmat, Mohammad Pezeshki, Rémi Le Priol, Gabriel Huang, Simon Lacoste-Julien, and Ioannis Mitliagkas. Negative momentum for improved game dynamics. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1802–1811. PMLR, 16–18 Apr 2019.

[42] Waïss Azizian, Ioannis Mitliagkas, Simon Lacoste-Julien, and Gauthier Gidel. A tight and unified analysis of extragradient for a whole spectrum of differentiable games. *arXiv preprint arXiv:1906.05945*, 2019.

[43] Florian Schäfer and Anima Anandkumar. Competitive gradient descent. In *Advances in Neural Information Processing Systems*, pages 7623–7633, 2019.

[44] John Nikolas Tsitsiklis. Problems in decentralized decision making and computation. Technical report, Massachusetts Inst of Tech Cambridge Lab for Information and Decision Systems, 1984.

[45] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 482–491. IEEE, 2003.

[46] Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53(1):65–78, 2004.

[47] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48, 2009.

[48] Duvsan Jakovetic, Joao Xavier, and JoseMF Moura. Fast distributed gradient methods. *IEEE Transactions on Automatic Control*, 59(5):1131–1146, 2014.

[49] Tianyu Wu, Kun Yuan, Qing Ling, Wotao Yin, and Ali H Sayed. Decentralized consensus optimization with asynchrony and delays. *IEEE Transactions on Signal and Information Processing over Networks*, 4(2):293–307, 2017.

[50] John C Duchi, Alekh Agarwal, and Martin J Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic control*, 57(3):592–606, 2011.

[51] Konstantinos I Tsianos, Sean Lawlor, and Michael G Rabbat. Push-sum distributed dual averaging for convex optimization. In *2012 ieee 51st ieee conference on decision and control (cdc)*, pages 5453–5458. IEEE, 2012.

[52] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[53] Ermin Wei and Asuman Ozdaglar. Distributed alternating direction method of multipliers. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5445–5450. IEEE, 2012.

[54] Ruiliang Zhang and James T Kwok. Asynchronous distributed admm for consensus optimization. In *ICML*, pages 1701–1709, 2014.

[55] Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, 2014.

[56] Franck Iutzeler, Pascal Bianchi, Philippe Ciblat, and Walid Hachem. Explicit convergence rate of a distributed alternating direction method of multipliers. *IEEE Transactions on Automatic Control*, 61(4):892–904, 2015.

[57] Necdet Serhat Aybat, Zi Wang, Tianyi Lin, and Shiqian Ma. Distributed linearized alternating direction method of multipliers for composite convex consensus optimization. *IEEE Transactions on Automatic Control*, 63(1):5–20, 2017.

[58] Hadrien Hendrikx, Laurent Massoulié, and Francis Bach. Accelerated decentralized optimization with local updates for smooth and strongly convex objectives. *arXiv preprint arXiv:1810.02660*, 2018.

[59] Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

[60] Kevin Seaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3027–3036. JMLR. org, 2017.

[61] Kevin Scaman, Francis Bach, Sébastien Bubeck, Laurent Massoulié, and Yin Tat Lee. Optimal algorithms for non-smooth distributed optimization in networks. In *Advances in Neural Information Processing Systems*, pages 2740–2749, 2018.

[62] Ohad Shamir and Nathan Srebro. Distributed stochastic optimization and learning. In *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 850–857. IEEE, 2014.

[63] Michael Rabbat. Multi-agent mirror descent for decentralized stochastic optimization. In *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 517–520. IEEE, 2015.

[64] Benjamin Sirb and Xiaojing Ye. Consensus optimization with delayed and stochastic gradients on decentralized networks. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 76–85. IEEE, 2016.

[65] Guanghui Lan, Soomin Lee, and Yi Zhou. Communication-efficient algorithms for decentralized and stochastic optimization. *Mathematical Programming*, pages 1–48, 2018.

[66] Anastasia Koloskova, Sebastian U Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. *arXiv preprint arXiv:1902.00340*, 2019.

[67] Zhanhong Jiang, Aditya Balu, Chinmay Hegde, and Soumik Sarkar. Collaborative deep learning in fixed topology networks. In *Advances in Neural Information Processing Systems*, pages 5904–5914, 2017.

[68] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. Asynchronous decentralized parallel stochastic gradient descent. In *ICML*, 2018.

[69] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. D2: Decentralized training over decentralized data. *arXiv preprint arXiv:1803.07068*, 2018.

[70] Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Michael Rabbat. Stochastic gradient push for distributed deep learning. *arXiv preprint arXiv:1811.10792*, 2018.

[71] Jianyu Wang and Gauri Joshi. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

[72] Jianyu Wang, Anit Kumar Sahu, Zhouyi Yang, Gauri Joshi, and Soummya Kar. Matcha: Speeding up decentralized sgd via matching decomposition sampling. *arXiv preprint arXiv:1905.09435*, 2019.

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

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

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

[76] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Sparq-sgd: Event-triggered and compressed communication in decentralized stochastic optimization. *arXiv preprint arXiv:1910.14280*, 2019.

[77] Aryan Mokhtari and Alejandro Ribeiro. Dsa: Decentralized double stochastic averaging gradient algorithm. *The Journal of Machine Learning Research*, 17(1):2165–2199, 2016.

[78] Kun Yuan, Bicheng Ying, Jiageng Liu, and Ali H Sayed. Variance-reduced stochastic learning by networked agents under random reshuffling. *IEEE Transactions on Signal Processing*, 67(2):351–366, 2018.

[79] Ran Xin, Usman A Khan, and Soummya Kar. Variance-reduced decentralized stochastic optimization with gradient tracking–part ii: Gt-svrg. *arXiv preprint arXiv:1910.04057*, 2019.

[80] Kunal Srivastava, Angelia Nedic, and Duvsan Stipanovic. Distributed min-max optimization in networks. In *2011 17th International Conference on Digital Signal Processing (DSP)*, pages 1–8. IEEE, 2011.

[81] Weijie Liu, Aryan Mokhtari, Asuman Ozdaglar, Sarath Pattathil, Zebang Shen, and Nenggan Zheng. A decentralized proximal point-type method for saddle point problems. *arXiv preprint arXiv:1910.14380*, 2019.

[82] R Tyrrell Rockafellar. Monotone operators and the proximal point algorithm. *SIAM journal on control and optimization*, 14(5):877–898, 1976.

[83] Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with relu activation. In *Advances in Neural Information Processing Systems*, pages 597–607, 2017.

[84] Robert Kleinberg, Yuanzhi Li, and Yang Yuan. An alternative view: When does sgd escape local minima? *arXiv preprint arXiv:1802.06175*, 2018.

[85] Yi Zhou, Junjie Yang, Huishuai Zhang, Yingbin Liang, and Vahid Tarokh. Sgd converges to global minimum in deep learning via star-convex path. *arXiv preprint arXiv:1901.00451*, 2019.

[86] Chao-Kai Chiang, Tianbao Yang, Chia-Jung Lee, Mehrdad Mahdavi, Chi-Jen Lu, Rong Jin, and Shenghuo Zhu. Online optimization with gradual variations. In *Conference on Learning Theory*, pages 6–1, 2012.

[87] Sasha Rakhlin and Karthik Sridharan. Optimization, learning, and games with predictable sequences. In *Advances in Neural Information Processing Systems*, pages 3066–3074, 2013.

[88] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[89] Karol Kurach, Mario Lucic, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. A large-scale study on regularization and normalization in gans. *arXiv preprint arXiv:1807.04720*, 2018.

[90] Pitch Patarasuk and Xin Yuan. Bandwidth optimal all-reduce algorithms for clusters of workstations. *J. Parallel Distrib. Comput.*, 69:117–124, 2009.

[91] Wei Zhang, Minwei Feng, Yunhui Zheng, Yufei Ren, Yandong Wang, Ji Liu, Peng Liu, Bing Xiang, Li Zhang, Bowen Zhou, and Fei Wang. Gadei: On scale-up training as a service for deep learning. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. The IEEE International Conference on Data Mining series(ICDM'2017), 2017.

[92] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

[93] Hongseok Namkoong and John C Duchi. Variance-based regularization with convex objectives. In *Advances in neural information processing systems*, pages 2971–2980, 2017.

[94] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[95] Ayushman Dash, John Cristian Borges Gamboa, Sheraz Ahmed, Marcus Liwicki, and Muhammad Zeshan Afzal. Tac-gan-text conditioned auxiliary classifier generative adversarial network. *arXiv preprint arXiv:1703.06412*, 2017.

[96] Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Face aging with conditional generative adversarial networks. In *2017 IEEE international conference on image processing (ICIP)*, pages 2089–2093. IEEE, 2017.

[97] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[98] Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. *arXiv preprint arXiv:1701.06547*, 2017.