Fully Distributed Generative Adversarial Networks

Jorge del Val, Sergio Valcarcel Macua, Javier Zazo, Santiago Zazo

Universidad Politécnica de Madrid

Abstract

We propose a new fully distributed algorithm for generative adversarial networks (GANs) based on distributed stochastic approximation. The framework considers multiple agents, each with a different subset of the data, which cooperate in order to reach a global deep generative model of the data. The agents communicate only among neighbors in a connected graph and no data is transmitted, so that privacy is preserved. The communication cost per agent grows linearly with the neighborhood size. We prove convergence of the distributed algorithm to the equivalent centralized iteration for the case of momentum-based stochastic gradient descent and illustrate its performance over a set of experiments.

1 Introduction

Unsupervised learning is one of the most prominent research fields in machine learning and artificial intelligence nowadays. Particularly, deep generative models have shown the ability to learn useful representations from high-dimensional data [1, 2, 3], such as images [4, 5, 6] or audio [7]. Unsupervised representation learning carries the promise of learning meaningful and useful features of complex data, which can then be used for several problems.

Generative adversarial networks (GANs) [8] constitute a set of techniques that have been proven to be very useful for generative modeling. In contrast with other existing techniques, like [9, 6, 10, 11], GANs model the optimization problem as a two-player minimax game. In the original formulation, the Nash equilibrium of the game was proven to be unique and related to the minimization of the Jensen-Shannon (JS) divergence between the model and data distribution in the infinite model capacity limit [8]. Several variations have been developed since the original formulation [12, 13, 14].

However, most of the developed algorithms to date for training GANs are designed for a centralized or parallel setting. This may not efficient in contexts where the data is inherently distributed among a set of nodes, such as in distributed databases, multi-agent learning [15], or sensor networks [16], where a central node fusing information or load balancing server may not be available. In this paper, we consider the case where data is inherently distributed on a network of N agents. The data of each agent may not be composed of independent samples of the same underlying process, but instead we allow each agent to have a different probability distribution of the data. For instance, each agent could have only data of a particular class or process.

The objective of the network as a whole is to learn a global generative model of the data. This is not possible in general for individual agents, since each individual agent has only access to its own data. Hence, cooperation among the agents is necessary in order to reach their goal. Furthermore, each agent is only able to communicate locally with its neighbors in the network graph, without exchanging any data. Since there is no central authority and all the nodes play the same role in solving the problem, we denote this approach as *fully distributed*.

In order to solve the problem, we use diffusion-based algorithms [17, 18, 19]. These have shown very good performance on distributed optimization problems over networks, providing enhanced stability and performance over consensus algorithms [18, 19, 20, 21]. Our algorithm combines diffusion strategies and generative adversarial networks in order to derive a fully distributed deep generative model. We also highlight that our approach naturally balances the computational load among agents. Up to the best of our knowledge, this is the first work which presents a fully distributed implementation of generative adversarial networks.

Most distributed optimization algorithms rely on convexity of the objective function in order to prove convergence to a global minimum [22, 23, 24, 25]. Other works analyze convergence in non-convex problems [26, 27, 28]. Here, added to the intrinsic non-convexity of the GAN problem, we also face the challenge that the solution is not a stationary point of some optimization problem, but a Nash equilibrium of a two-player minimax game. This fact makes the training of GANs intrinsically unstable, and general convergence guarantees in practical scenarios are yet to be found [29]. As we will see, instead of proving convergence to a Nash equilibrium, we prove convergence to the centralized setting, i.e., a single agent with access to the whole dataset. Hence, the network as a whole will behave asymptotically a single-agent GAN.

Our main contributions are the following: First, we derive a separable formulation of the centralized GAN optimization problem. Secondly, we introduce a fully distributed algorithm to solve the problem. Next, we show that momentum-based stochastic gradient descent converges to the centralized setting under mild assumptions, relying on known results in distributed stochastic approximation. Lastly, we present a case study distributing the MNIST dataset on to a network of agents to illustrate the performance of the proposed approach, showing that every agent is able to learn the global distribution and generate samples from all classes, even when it only had access to data from a single class.

2 Background

2.1 Generative Adversarial Networks

GANs provide a method to train latent variable models based on finding the Nash equilibrium of a two-player minimax game. This is modeled as a coupled unconstrained optimization model between a *generator*, G_θ, which transforms a latent random variable $\boldsymbol{z}$ into an observable vector $\boldsymbol{x'} = G_\theta(\boldsymbol{z})$, and a *discriminator*, D_ϕ, which learns the probability that a sample comes from the dataset or has been generated by the generator. The original model proposed by Goodfellow et al. in [8] solves the following problem

$$\min_\theta \max_\phi \mathbb{E}_{\boldsymbol{x}\sim p_r}[\log(D_\phi(\boldsymbol{x}))] + \mathbb{E}_{\boldsymbol{z}\sim p_z}[\log(1 - D_\phi(G_\theta(\boldsymbol{z})))] \tag{1}$$

where $\boldsymbol{z}$ is a random latent variable of prior distribution p_z and $\boldsymbol{x}$ is a random variable that follows data distribution. If both the generator G_θ and discriminator D_ϕ have infinite model capacity, it can be proven that the cost of the optimal discriminator for a given generator is proportional to the JS divergence between the data and model distribution up to an additive constant [8]. Thus, optimizing the generator is equivalent to minimizing the JS divergence between both distributions, so that it achieves zero divergence at the Nash equilibrium.

However, in practice both generator and discriminator have a finite model capacity and the cost function is highly nonconvex. Thus, typical schemes such as simultaneous gradient-based optimization methods may fail to converge. As a consequence, some variants have been proposed to increase stability [13, 14]. Most of GAN models propose a two player minimax game with the following structure, which generalizes (1) as a general two-player game

$$\begin{aligned}&\min_\theta \mathbb{E}_{\boldsymbol{x}\sim p_r, \boldsymbol{z}\sim p_z} L_G\left(D_\phi(\boldsymbol{x}), D_\phi(G_\theta(\boldsymbol{z})\right)\\&\max_\phi \mathbb{E}_{\boldsymbol{x}\sim p_r, \boldsymbol{z}\sim p_z} L_D\left(D_\phi(\boldsymbol{x}), D_\phi(G_\theta(\boldsymbol{z})\right)\end{aligned} \tag{2}$$

Here, L_G and L_D denote the loss functions of generator and discriminator respectively. In most of GAN models, both optimization problems are unconstrained. However, in some cases it may be necessary to constrain the discriminator [13].

2.2 Training of GANs

Although simultaneous gradient-based stochastic optimization techniques mail fail to converge, the fact is that they are the current preferred approach and could also yield state of the art results if they are properly tunned. Hence, to solve game (2), it is usual to use some sort of simultaneous stochastic gradient descent (SGD) optimization techniques on both the generator and discriminator [8, 3]. This is

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \alpha_t \boldsymbol{g}_t \\ \boldsymbol{\psi}_{t+1} &= \boldsymbol{\psi}_t + \alpha_t \boldsymbol{d}_t \end{aligned} \tag{3}$$

where $\boldsymbol{g}_t$ and $\boldsymbol{d}_t$ are the updates of generator and discriminator respectively, based on the gradients of L_G and L_D, and α_t is the stepsize. The fixed points of iteration (3) are Nash equilibriums of the game. For strongly convex-concave functions, this iteration is known to converge to the unique Nash equilibrium in the case of zero sum games [30]. However, this iteration is usually unstable in the case of GANs, due to the non-convexity of the objective functions, although it is found to work well in most of practical situations [8, 3, 5]. In this work, we will limit ourselves to the case where $\boldsymbol{g}_t$ and $\boldsymbol{d}_t$ are iterations of momentum-based SGD [31, 32].

3 Fully distributed algorithm

Consider a set $\mathcal{I}$ of N agents. Each agent $i \in \mathcal{I}$ has a subset of n_i data samples, where each data sample x_k^i may be distributed according to different probability distributions p_r^i depending on the node i. Understanding $\boldsymbol{x}$ in (2) as a random sample over the whole distributed dataset, i.e., a *mixture* over the agents, we can easily see that $p(x_n) = p(i)p(x_n|i) = w_i p_r^i(x)$, where $w_i = p(i) = \frac{n_i}{n}$ is the probability that the sample x_n is in node i. Consequently, we can rewrite the unconstrained GAN optimization problem (2) as a sum over the local objectives of each agent

$$\begin{aligned} &\min_{\theta} \sum_{i\in\mathcal{I}} w_i \mathbb{E}_{\boldsymbol{x}_i \sim p_r^i, \boldsymbol{z}\sim p_z} L_G\left(D_\phi(\boldsymbol{x}_i), D_\phi(G_\theta(\boldsymbol{z}))\right) \\ &\max_{\phi} \sum_{i\in\mathcal{I}} w_i \mathbb{E}_{\boldsymbol{x}_i \sim p_r^i, \boldsymbol{z}\sim p_z} L_D\left(D_\phi(\boldsymbol{x}_i), D_\phi(G_\theta(\boldsymbol{z}))\right) \end{aligned} \tag{4}$$

For the sake of clarity in subsequent derivations, we introduce shorthands for the individual objective functions of each node's generator and discriminator as

$$F_G^i(\theta, \phi) \triangleq N w_i \mathbb{E}_{\boldsymbol{x}_i \sim p_r^i, \boldsymbol{z}\sim p_z} L_G\left(D_\phi(\boldsymbol{x}_i), D_\phi(G_\theta(\boldsymbol{z}))\right) \tag{5}$$

$$F_D^i(\theta, \phi) \triangleq N w_i \mathbb{E}_{\boldsymbol{x}_i \sim p_r^i, \boldsymbol{z}\sim p_z} L_D\left(D_\phi(\boldsymbol{x}_i), D_\phi(G_\theta(\boldsymbol{z}))\right) \tag{6}$$

so that we can write problem (4) as follows

$$\begin{aligned} &\min_{\theta} \frac{1}{N} \sum_{i\in\mathcal{I}} F_G^i(\theta, \phi) \\ &\max_{\phi} \frac{1}{N} \sum_{i\in\mathcal{I}} F_D^i(\theta, \phi) \end{aligned} \tag{7}$$

In this way, we achieve a separable formulation of (2). To solve (7), each agent $i \in \mathcal{I}$ can only communicate with the agents in its *neighborhood*, which consists of the agents which are directly connected to i and we denote as $\mathcal{N}_i \subseteq \mathcal{I}$. Each agent i at timestep t will have its own estimation of the solution to (7), which we denote as θ_t^i and ϕ_t^i. Among the available fully distributed algorithmic templates, we choose the adapt-and-combine (ATC) scheme, which consists in two steps (see, e.g., [17, 18, 19]). First, at the adaptation step, every agent moves in the direction of the gradient of its local function (adaptation), obtaining intermediate estimates:

$$\begin{aligned} \widehat{\boldsymbol{\theta}}^i_{t+1} &= \boldsymbol{\theta}^i_t - \alpha_t \boldsymbol{g}^i_t \\ \widehat{\boldsymbol{\psi}}^i_{t+1} &= \boldsymbol{\psi}^i_t + \alpha_t \boldsymbol{d}^i_t \end{aligned} \tag{8}$$

where $\boldsymbol{g}^i_t$ and $\boldsymbol{d}^i_t$ are momentum updates based on noisy estimations of $\nabla_{\theta^i} F^i_G(\theta^i, \phi^i)$ and $\nabla_{\phi^i} F^i_D(\theta^i, \phi^i)$ respectively, which we remark that must be calculated locally by agent i with mini-batches of its own data.

The second step consists in performing a convex combination of the parameter estimated by its neighbors and itself:

$$\begin{aligned} \boldsymbol{\theta}^i_{t+1} &= \sum_{j \in \mathcal{N}_i} c_{ji} \widehat{\boldsymbol{\theta}}^j_{t+1} \\ \boldsymbol{\psi}^i_{t+1} &= \sum_{j \in \mathcal{N}_i} c_{ji} \widehat{\boldsymbol{\psi}}^j_{t+1} \end{aligned} \tag{9}$$

where $c_{ji} \in [0, 1]$ denotes the weight given by node i to the information coming from node j. We also introduce the combination matrix $C = [c_{ij}]$.

It is convenient to write the iterations (8) and (9) into a single network iteration. To do so, we first introduce vectors of length $(M_G + M_D)$ which gather variables of the generator and discriminator for each agent, where M_G and M_D are the dimensions of θ^i_t and ϕ^i_t respectively:

$$\boldsymbol{\vartheta}^i_t \triangleq \begin{pmatrix} \boldsymbol{\theta}^i_t \\ \boldsymbol{\psi}^i_t \end{pmatrix} \tag{10}$$

$$\boldsymbol{v}^i_t \triangleq \begin{pmatrix} -\boldsymbol{g}^i_t \\ \boldsymbol{d}^i_t \end{pmatrix} \tag{11}$$

The momentum update for every agent i is given by

$$\boldsymbol{v}^i_t = \mu_t \boldsymbol{v}^i_{t-1} - \boldsymbol{h}^i_t \tag{12}$$

where $\mu_t \in [0, 1)$ is the momentum parameter and $\boldsymbol{h}^i_t$ is a noisy estimation of the gradient of the generator and discriminator objective functions. The point in which the gradient is evaluated may vary depending on the method. For standard momentum [32] we have

$$\mathbb{E}[\boldsymbol{h}^i_t | \vartheta_t] = \begin{pmatrix} \nabla_\theta F^i_G(\vartheta_t) \\ \nabla_\phi F^i_D(\vartheta_t) \end{pmatrix} \tag{13}$$

and for Nesterov accelerated gradient (NAG) [31] we have

$$\mathbb{E}[\boldsymbol{h}^i_t | \vartheta_t] = \begin{pmatrix} \nabla_\theta F^i_G(\vartheta_t + \mu_t v_t) \\ \nabla_\phi F^i_D(\vartheta_t + \mu_t v_t) \end{pmatrix} \tag{14}$$

As we will see in section 4, our results are independent of either method, and thus we will not make explicit mention in subsequent derivations. Now we introduce the vectors of length $N(M_G + M_D)$ that gather every variable of every node in the network:

$$\boldsymbol{\vartheta}_t \triangleq \left(\boldsymbol{\vartheta}^i_t\right)_{i \in \mathcal{I}}, \quad \boldsymbol{v_t} \triangleq \left(\boldsymbol{v}^i_t\right)_{i \in \mathcal{I}}, \quad \boldsymbol{h}^i_t = \left(\boldsymbol{h}^i_t\right)_{i \in \mathcal{I}} \tag{15}$$

Hence, we can express the ATC algorithms run by all the agents as a single recursion:

$$\begin{aligned} \boldsymbol{\vartheta}_{t+1} &= (C^T \otimes I_M)\left(\boldsymbol{\vartheta}_t + \alpha_t \boldsymbol{v}_t\right) \\ \boldsymbol{v}_t &= \mu_t \boldsymbol{v}_{t-1} - \boldsymbol{h}_t \end{aligned} \tag{16}$$

4 Convergence with momentum-based optimization

In this section we show that previous results on distributed stochastic approximation due to [26] can be applied to momentum-based simultaneous gradient methods. In particular, we show that iteration (16) is asymptotically equivalent to the iteration (3). To do so, we need the following assumptions.

Assumption 1. *(Combination weights). The $N \times N$ matrix $C = [c_{ij}]$ satisfies:*

$$c_{ij} \geq 0, \quad c_{ij} = 0 \quad \text{if} \quad j \notin \mathcal{N}_i, \quad C^T \mathbb{1} = \mathbb{1}, \quad C\mathbb{1} = \mathbb{1} \tag{17}$$

$$\rho\left(C(I - \mathbb{1}\mathbb{1}^T/N)C^T\right) < 1 \tag{18}$$

Assumption 2. *(Stepsize). The stepsize parameter α_t from (16) satisfies*

$$\alpha_t > 0 \quad \forall t, \quad \sum_{t=0}^{\infty} \alpha_t \to \infty, \quad \sum_{t=0}^{\infty} \alpha_t^2 < \infty \tag{19}$$

Assumption 3. *(Boundedness). There exists a nonnegative scalar B such that*

$$\mathbb{E}[\|\boldsymbol{h}_t^i\|^2|] \leq B \quad \forall t \in \{0, ..., \infty\}, \forall i \in \mathcal{I} \tag{20}$$

Assumption 4. *(Independence). Conditionally on $\mathcal{F}_t$, $\boldsymbol{h}_{t+1}^i$ are independent for every $i \in \mathcal{I}$.*

Assumption 5. *The initial value of the process v_t is bounded. This is, $\|v_0\| \leq \infty$.*

Assumption 1 is satisfied when matrix C is primitive and doubly stochastic, which can be easily satisfied in practice, even when the matrix is designed by the agents in a fully distributed manner (e.g., by using Metropolis-Hasting rule [33]). Assumptions 2 and 3 are standard for studying convergence of stochastic approximations. In particular, Assumption 2 states that the parameter must be updated slow enough, while ensuring that the noise is asymptotically averaged. Assumption 3 ensures that the second moment of the noisy gradient is bounded. We state the following theorem:

Theorem 1. *Under assumptions 3 and 5 there exists a nonnegative scalar L such that*

$$\mathbb{E}[\|\boldsymbol{v}_t\|^2] \leq L \quad \forall t \in \{0, ..., \infty\} \tag{21}$$

Proof. Using Minkowski's inequality on (12) we have

$$\sqrt{\mathbb{E}[\|\boldsymbol{v}_t\|^2]} \leq \mu_{t-1}\sqrt{\mathbb{E}[\|\boldsymbol{v}_{t-1}\|^2]} + \sqrt{\mathbb{E}[\|\boldsymbol{h}_t\|^2]} \tag{22}$$

Since $\mu_t \in [0, 1)$ by definition, then there exists a constant $U \in [0, 1)$ such that $\mu_t \leq U < 1$ for every timestep t. Then, from Assumption 3, we have:

$$\sqrt{\mathbb{E}[\|\boldsymbol{v}_t\|^2]} \leq U\sqrt{\mathbb{E}[\|\boldsymbol{v}_{t-1}\|^2]} + \sqrt{B} \tag{23}$$

Now define the constant $K = \frac{\sqrt{B}}{1-U}$. If $\sqrt{\mathbb{E}[\|\boldsymbol{v}_{t-1}\|^2]} \leq K$, then

$$\sqrt{\mathbb{E}[\|\boldsymbol{v}_t\|^2]} \leq U\frac{\sqrt{B}}{1-U} + \sqrt{B} = \frac{\sqrt{B}}{1-U} = K \tag{24}$$

On the other hand, if $\sqrt{\mathbb{E}[\|\boldsymbol{v}_{t-1}\|^2]} \geq K$, then $\sqrt{B} \leq (1-U)\sqrt{\mathbb{E}[\|\boldsymbol{v}_{t-1}\|^2]}$ and we have

$$\sqrt{\mathbb{E}[\|\boldsymbol{v}_t\|^2]} \leq U\sqrt{\mathbb{E}[\|\boldsymbol{v}_{t-1}\|^2]} + (1-U)\sqrt{\mathbb{E}[\|\boldsymbol{v}_{t-1}\|^2]} = \sqrt{\mathbb{E}[\|\boldsymbol{v}_{t-1}\|^2]} \tag{25}$$

Thus, by induction we have that

$$\mathbb{E}[\|\boldsymbol{v}_t\|^2] \leq \max(\|v_0\|^2, K^2) \quad \forall t \in \{0, ..., \infty\} \tag{26}$$

□

Proposition 1. *Under assumptions 1-5, the sequence of parameters given by (16) converges almost surely to an equivalent centralized iteration in the form (3), with a variance in the gradients equal to* $\frac{1}{N}\sum_{i\in\mathcal{I}} \mathit{Var}[\boldsymbol{h}_t^i]$.

Proof. By theorem 1 we have that $\mathbb{E}[\|\boldsymbol{v}_t\|^2]$ is bounded. Thus, from [26, Lemma 1], we have that the iteration (16) converges to the average iteration

$$\bar{\boldsymbol{\vartheta}}_{t+1} = \bar{\boldsymbol{\vartheta}}_t - \alpha_t \bar{\boldsymbol{v}}_t \tag{27}$$

where $\bar{\boldsymbol{\vartheta}}_{\boldsymbol{t}} \triangleq \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\vartheta}_t^i$. This is,

$$\lim_{t\to\infty} \mathbb{E}[\|\boldsymbol{\vartheta}_t^i - \bar{\boldsymbol{\vartheta}}_t\|^2] = 0 \quad \forall i \in \mathcal{I} \tag{28}$$

Following (12), we have that

$$\bar{\boldsymbol{v}}_t = \mu_t \bar{\boldsymbol{v}}_{t-1} - \bar{\boldsymbol{h}}_t \tag{29}$$

Since each $\boldsymbol{h}_{\boldsymbol{t}}^{\boldsymbol{i}}$ is an unbiased estimator of the gradient of $F_G^i(\theta,\psi)$ and $F_D^i(\theta,\psi)$, it follows from (7) that $\bar{\boldsymbol{h}}_t$ is an unbiased estimator of the gradient of the global cost functions. Also, from assumption 4 we have that $\mathrm{Var}[\bar{\boldsymbol{h}}_t] = \frac{1}{N}\sum_{i\in\mathcal{I}} \mathrm{Var}[\boldsymbol{h}_t^i]$. Hence, (27) is equivalent to (3) with a fraction of the noise in the gradient estimations. □

5 Experiments

In this section we realize experiments which illustrate the proposed approach. We focus on showing the following points:

- The norm of the difference between the parameters of every agent and the average parameter vector decreases toward zero.
- Every agent is able to learn the global distribution, even if it is just given samples from a single class.

We consider two scenarios, namely, distributed and centralized. In the distributed scenario, we consider a network of $N = 10$ agents. Each agent is given samples of the MNIST dataset [34] corresponding only to a single class, i.e., digits of a single number (see Fig. 1). The dataset is distributed equally among all agents, so that every agent has the same number of samples. As can be seen in Fig. 1, the number of neighbors per agent is less than the total number of nodes, so that for example agent 9 is three jumps away from agent 6. Every agent runs a DCGAN [5] with mini-batch SGD optimization accelerated by Nesterov momentum. The centralized scenario consists on a DCGAN with the same architecture that every agent in the distributed scenario, but with access to the whole dataset. For equivalence of both scenarios, the batch sizes of each agent in the distributed scenario are $\frac{1}{N}$ times the batch size of the centralized scenario (see proposition 1). We will equivalently denote the distributed model by FD-DCGAN and the centralized model by DCGAN.

The cost functions of discriminator and generator respectively in all cases are

$$L_D(D_\phi(\boldsymbol{x}), D_\phi(G_\theta(\boldsymbol{z}))) = \log(D_\phi(\boldsymbol{x})) + \log(1 - D_\phi(G_\theta(\boldsymbol{z}))) \tag{30}$$
$$L_G(D_\phi(\boldsymbol{x}), D_\phi(G_\theta(\boldsymbol{z}))) = \log(D_\phi(G_\theta(\boldsymbol{z}))) \tag{31}$$

which are a heuristic, non-saturating modification of (1) to avoid the vanishing of the generator's gradient [3] when the discriminator is strong. The weights of the connection matrix c_{ij} are obtained by the Metropolis-Hastings rule [33]. Also, all the models sample $\boldsymbol{z}$ from a standard normal distribution.

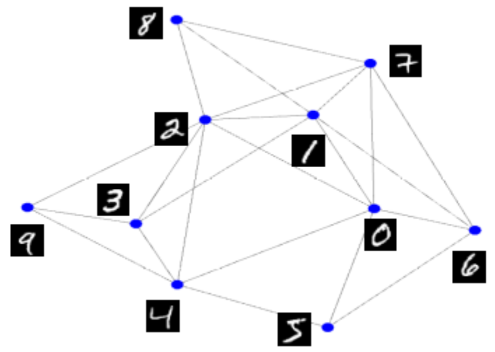

Fig. 1: Experimental setting

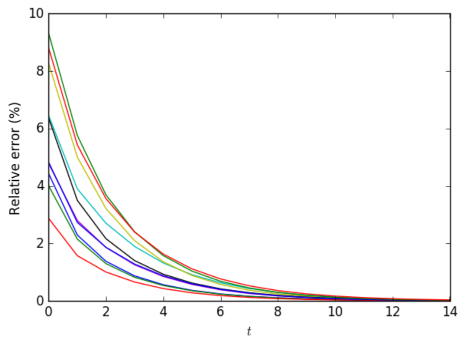

Fig. 2: Relative error between ϑ_t^i and $\bar{\vartheta}_t$

The DCGAN models in all cases consist of a generator with a fully connected layer and two convolutional layers equipped with $\tanh$ nonlinearity and followed by batch normalization [35]. The discriminator is made of two convolutional layers followed by two fully connected layers equipped with $\tanh$ activations and sigmoid output.

In Fig. 2 we show the relative error between ϑ_t^i and $\bar{\vartheta}_t$ for all the agents $i \in \mathcal{I}$. We observe a very fast convergence, reaching an error of 0.026% at iteration 15. In addition, we have observed that the error remains in those levels during all the training process, in concordance with proposition 1.

In Fig. 3 we show samples of different agents after ten thousand iterations, in addition to the centralized GAN. We observe that the agents are able to generate sharp samples from all the classes, even though each agent only saw samples of a given digit. In addition, we see that the agents are able to generate digits of agents which are three hops away in the network, such as digit 9 from agent 6 or digit 7 from agent 3.

Lastly, on Fig. 4 we show the mean cost of the generators and discriminators of all the agents versus the cost of the generator and discriminator of the centralized GAN, which due to (7) are equivalent. All the costs are averaged for 30 iterations. As can be noted, the distributed algorithm arrives to a solution with the same cost values than the centralized GAN. Also, we note that both sequences are very similar up to a displacement constant, which reinforces our claim that the network behaves as a single agent GAN. We also observed that our approach seems to be less noisy than a central approach.

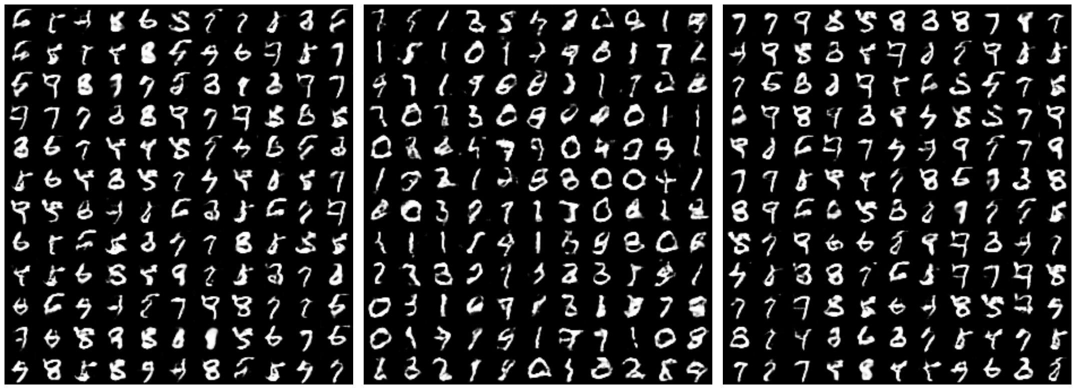

Fig. 3: Comparison of scenarios. (left) Samples from agent 6. (center) Samples from agent 3. (right) Samples of the centralized GAN.

6 Conclusions

We developed a model for fully distributed generative adversarial networks, which works for arbitrary unconstrained GAN models with momentum-based stochastic gradient descent and any connected topology without any central coordination or exchange of data among agents. We also proved almost

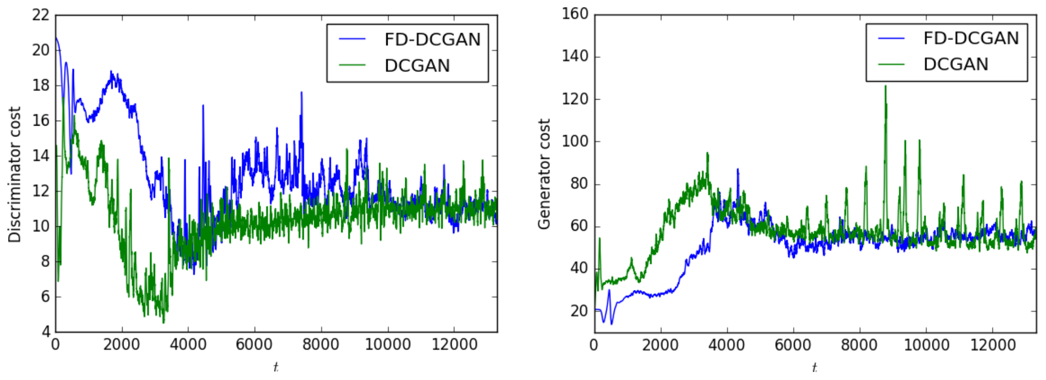

Fig. 4: Cost functions of distributed and centralized scenarios. (left) Discriminator cost. (right) Generator cost.

sure asymptotic convergence of the network iteration towards an equivalent centralized iteration with access to the global dataset. Our experiments indicate that our framework allows the network of agents to learn the global probability distribution.

This work is, up to the best of our knowledge, the first to propose and analyze such framework. Future work includes generalizing to optimization methods other than momentum SGD, application to other distributed learning domains such as cooperative multi-agent reinforcement learning or distributed semi-supervised learning.

References

[1] R. Salakhutdinov, *Learning deep generative models*. PhD thesis, University of Toronto, 2009.

[2] Y. Bengio *et al.*, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[3] I. Goodfellow, "NIPS 2016 Tutorial: Generative Adversarial Networks," *ArXiv e-prints*, Dec. 2017. arXiv:1701.00160.

[4] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," *arXiv preprint arXiv:1502.04623*, 2015.

[5] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," *ICLR*, 2016.

[6] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," *arXiv preprint arXiv:1601.06759*, 2016.

[7] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," *ArXiv e-prints*, Sept. 2016. arXiv:1609.03499.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27* (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds.), pp. 2672–2680, Curran Associates, Inc., 2014.

[9] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[10] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[11] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th annual international conference on machine learning*, pp. 609–616, ACM, 2009.

[12] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2016.

[13] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[14] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," *arXiv preprint arXiv:1609.03126*, 2016.

[15] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous agents and multi-agent systems*, vol. 11, no. 3, pp. 387–434, 2005.

[16] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[17] A. H. Sayed, "Diffusion adaptation over networks," *E-Reference Signal Processing*. R. Chellapa and S. Theodoridis, editors, Elsevier, 2013. Also available as arXiv:1205.4220v1, May 2012.

[18] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks," *IEEE Signal Processing Magazine*, vol. 30, no. 3.

[19] A. H. Sayed, "Adaptive networks," *Proceedings of the IEEE*, vol. 112, no. 4.

[20] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 12.

[21] S. V. Macua, S. Zazo, and J. Zazo, "Distributed black-box optimization of nonconvex functions," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3591–3595, IEEE, 2015.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[23] M. Zhu and S. Martínez, "On distributed convex optimization under inequality and equality constraints," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 151–164, 2012.

[24] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[25] T.-H. Chang, A. Nedic, and A. Scaglione, "Distributed constrained optimization by consensus-based primal-dual perturbation method," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1524–1538, 2014.

[26] P. Bianchi and J. Jakubowicz, "Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 2.

[27] P. Di Lorenzo and G. Scutari, "Next: In-network nonconvex optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.

[28] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Asynchronous gossip algorithms for stochastic optimization," in *Proceedings of the 48th IEEE Conference on Decision and Control, held jointly with the 28th Chinese Control Conference.*, pp. 3581–3586, IEEE, 2009.

[29] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *ICLR*, 2017.

[30] D. Feijer and F. Paganini, "Stability of primal–dual gradient dynamics and applications to network optimization," *Automatica*, vol. 46, no. 12, pp. 1974–1981, 2010.

[31] Y. Nesterov, "A method of solving a convex programming problem with convergence rate o (1/k2)," in *Soviet Mathematics Doklady*, vol. 27, pp. 372–376, 1983.

[32] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[33] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[34] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[35] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.