

MULTI-AGENT ADVERSARIAL TRAINING USING DIFFUSION LEARNING

Ying Cao* Elsa Rizk* Stefan Vlaski† Ali H. Sayed*

*School of Engineering, École Polytechnique Fédérale de Lausanne

† Department of Electrical and Electronic Engineering, Imperial College London

ABSTRACT

This work focuses on adversarial learning over graphs. We propose a general adversarial training framework for multi-agent systems using diffusion learning. We analyze the convergence properties of the proposed scheme for convex optimization problems, and illustrate its enhanced robustness to adversarial attacks.

Index Terms— Adversarial training, decentralized optimization, diffusion strategy, multi-agent systems.

1. INTRODUCTION

In many machine learning algorithms, small malicious perturbations that are imperceptible to the human eye can cause classifiers to reach erroneous conclusions [1–5]. To mitigate the negative effect of adversarial examples, one methodology is adversarial training [6], in which clean training samples are augmented by adversarial samples by adding purposefully crafted perturbations. Due to the lack of an explicit definition for the imperceptibility of perturbations, additive attacks are usually restricted within a small bounded region. Most earlier studies, such as [6–10], have focused on studying adversarial training in the context of single agent learning. In this work, we devise a robust training algorithm for multi-agent networked systems by relying on diffusion learning [5, 11, 12], which has been shown to have a wider stability range and improved performance guarantees for adaptation in comparison to other decentralized strategies [5, 11, 12].

There of course exist other works in the literature that applied adversarial learning to a multiplicity of agents, albeit using a different architecture. For example, the works [13–15] employ multiple GPUs and a fusion center, while the works [16–18] consider graph neural networks. In this work, we focus on a fully decentralized architecture where each agent corresponds to a learning unit in its own right, and interactions occur locally over neighborhoods determined by a graph topology.

We formulate a sequential minimax optimization problem involving adversarial samples, and assume in this article that the perturbations are within an ℓ_2 -bounded region. We hasten to add though that the analysis can be extended to other norms, such as ℓ_1 – and ℓ_∞ –bounded perturbations. For simplicity, and due to space limitations, we consider the ℓ_2 –case here.

In the performance analysis, we examine the convergence of the proposed framework for convex settings due to space limitations, but note that similar bounds can be derived for nonconvex environments by showing convergence towards local minimizers. In particular, we show here that with strongly-convex loss functions, the proposed algorithm approaches the global minimizer within $O(\mu)$ after sufficient iterations, where μ is the step-size parameter.

Emails: ying.cao@epfl.ch, elsa.rizk@epfl.ch, s.vlaski@imperial.ac.uk, ali.sayed@epfl.ch

2. PROBLEM SETTING

Consider a collection of K agents where each agent k observes independent realizations of some random data $(\mathbf{x}_k, \mathbf{y}_k)$, where \mathbf{x}_k plays the role of the feature vector and \mathbf{y}_k plays the role of the label variable. Adversarial training in the decentralized setting deals with the following stochastic minimax optimization problem

$$w^* = \operatorname{argmin}_{w \in \mathbb{R}^M} \left\{ J(w) \triangleq \sum_{k=1}^K \pi_k J_k(w) \right\} \quad (1)$$

where $\{\pi_k\}_{k=1}^K$ are positive scaling weights adding up to one, and each individual risk function is defined by

$$J_k(w) = \mathbb{E}_{\{\mathbf{x}_k, \mathbf{y}_k\}} \left\{ \max_{\|\delta_k\| \leq \epsilon} Q_k(w; \mathbf{x}_k + \delta_k, \mathbf{y}_k) \right\} \quad (2)$$

in terms of a loss function $Q_k(\cdot)$. In this formulation, the variable δ_k represents an ℓ_2 norm-bounded perturbation used to generate adversarial examples, and \mathbf{y}_k is the true label of sample \mathbf{x}_k . We refer to w^* as the robust model. In this paper, we assume all agents observe data sampled independently (over time and space) from the same statistical distribution.

One methodology for solving (1) is to first determine the inner maximizer in (2), thus reducing the minimax problem to a standard stochastic minimization formulation. Then, the traditional stochastic gradient method could be used to seek the minimizer. We denote the true maximizer of the perturbed loss function in (2) by

$$\delta_k^*(w) \in \operatorname{argmax}_{\|\delta_k\| \leq \epsilon} Q_k(w; \mathbf{x}_k + \delta_k, \mathbf{y}_k) \quad (3)$$

where the dependence of δ_k^* on w is shown explicitly. To apply the stochastic gradient method, we would need to evaluate the gradient of $Q(w; \mathbf{x}_k + \delta_k^*(w), \mathbf{y}_k)$ relative to w , which can be challenging since $\delta_k^*(w)$ is also dependent on w . This difficulty can be resolved by appealing to Danskin’s theorem [19–21]. Let

$$g(w) \triangleq \max_{\|\delta_k\| \leq \epsilon} Q_k(w; \mathbf{x}_k + \delta_k, \mathbf{y}_k) \quad (4)$$

Then, the theorem asserts that $g(w)$ is convex over w if $Q_k(w; \cdot, \cdot)$ is convex over w . Moreover, $g(w)$ need not be differentiable over w even when $Q_k(w; \cdot, \cdot)$ is differentiable. However, and importantly for our purposes, we can determine a subgradient for $g(w)$ by using the actual gradient of the loss evaluated at the worst perturbation, namely, it holds that

$$\nabla_w Q_k(w; \mathbf{x}_k + \delta_k^*, \mathbf{y}_k) \in \partial_w g(w) \quad (5)$$

where ∂_w refers to the subdifferential set of its argument. In (5), the gradient of $Q_k(\cdot)$ relative to w at the maximizer δ_k^* is computed by

treating δ_k^* as a stand-alone vector and ignoring its dependence on w . When δ_k^* in (3) happens to be unique, then the gradient on the left in (5) will be equal to the right side, so that in that case the function $g(w)$ is differentiable.

Motivated by these properties, and using (5), we can now propose an algorithm to enhance the robustness of multi-agent systems to adversarial perturbations. To do so, we rely on the adapt-then-combine (ATC) version of the diffusion strategy [11, 12] and write down the following adversarial extension to solve (1)–(2)

$$\mathbf{x}_{k,n}^* = \mathbf{x}_{k,n} + \delta_{k,n}^* \quad (6a)$$

$$\phi_{k,n} = \mathbf{w}_{k,n-1} - \mu \nabla_w Q_k(\mathbf{w}_{k,n-1}; \mathbf{x}_{k,n}^*, \mathbf{y}_{k,n}) \quad (6b)$$

$$\mathbf{w}_{k,n} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \phi_{\ell,n} \quad (6c)$$

where

$$\delta_{k,n}^* \in \operatorname{argmax}_{\|\delta_k\| \leq \epsilon} Q_k(\mathbf{w}_{k,n-1}; \mathbf{x}_{k,n} + \delta_k, \mathbf{y}_{k,n}) \quad (7)$$

In this implementation, expression (6a) computes the worst-case adversarial example at iteration n using the perturbation from (7), while (6b) is the intermediate adaptation step in which all agents simultaneously update their parameters with step-size μ . Relation (6c) is the convex combination step where the intermediate states $\phi_{\ell,n}$ from the neighbors of agent k are combined together. The scalars $a_{\ell k}$ are non-negative and they add to one over $\ell \in \mathcal{N}_k$.

3. CONVERGENCE ANALYSIS

This section analyzes the convergence of the adversarial diffusion strategy (6a)–(6c) for the case of strongly convex loss functions. We list the following assumptions, which are commonly used in the literature of decentralized multi-agent learning and single-agent adversarial training [11, 22–25].

Assumption 1. (Strongly-connected graph) *The entries of the combination matrix $A = [a_{\ell k}]$ satisfy $a_{\ell k} \geq 0$ and the entries on each column add up to one, which means that A is left-stochastic. Moreover, the graph is assumed to be strongly-connected, meaning that there exists a path with nonzero weights $\{a_{\ell k}\}$ linking any pair of agents and, in addition, at least one node k in the network has a self-loop with $a_{kk} > 0$.*

Assumption 2. (Strong convexity) *For each agent k , the loss function $Q_k(w; \cdot)$ is ν -strongly convex over w , namely, for any $w_1, w_2, x \in \mathbb{R}^M$ and $y \in \mathbb{R}$, it holds that*

$$Q_k(w_2; x, y) \geq Q_k(w_1; x, y) + \nabla_{w^\top} Q_k(w_1; x, y)(w_2 - w_1) + \frac{\nu}{2} \|w_2 - w_1\|^2 \quad (8)$$

□

We remark that it also follows from Danskin’s theorem [19–21] that, when $Q_k(w; \cdot, \cdot)$ is ν -strongly convex over w , then the adversarial risk $J_k(w)$ defined by (2) will be strongly convex. As a result, the aggregate risk $J(w)$ in (1) will be strongly-convex as well.

Assumption 3. (Smooth loss functions): *For each agent k , the gradients of the loss function relative to w and x are Lipschitz in relation to the three variables $\{w, x, y\}$ in the following manner:*

$$\|\nabla_w Q_k(w_2; x + \delta, y) - \nabla_w Q_k(w_1; x + \delta, y)\| \leq L_1 \|w_2 - w_1\| \quad (9a)$$

$$\|\nabla_w Q_k(w; x_2 + \delta, y) - \nabla_w Q_k(w; x_1 + \delta, y)\| \leq L_2 \|x_2 - x_1\| \quad (9b)$$

$$\|\nabla_w Q_k(w; x + \delta, y_2) - \nabla_w Q_k(w; x + \delta, y_1)\| \leq L_3 \|y_2 - y_1\| \quad (9c)$$

and

$$\|\nabla_x Q_k(w_2; x + \delta, y) - \nabla_x Q_k(w_1; x + \delta, y)\| \leq L_4 \|w_2 - w_1\| \quad (10a)$$

$$\|\nabla_x Q_k(w; x_2 + \delta, y) - \nabla_x Q_k(w; x_1 + \delta, y)\| \leq L_5 \|x_2 - x_1\| \quad (10b)$$

where $\|\delta\| \leq \epsilon$. For later use, we use $L = \max\{L_1, L_2, L_3, L_4, L_5\}$.

Assumption 4. (Bounded gradient disagreement) *For any pair of agents k and ℓ , the squared gradient disagreements are uniformly bounded on average, namely, for any $w \in \mathbb{R}^M$ and $\|\delta\| \leq \epsilon$, it holds that*

$$\mathbb{E}_{\{x, y\}} \|\nabla_w Q_k(w; x + \delta, y) - \nabla_w Q_\ell(w; x + \delta, y)\|^2 \leq C^2 \quad (11)$$

□

Note that (11) is automatically satisfied when all agents use the same loss function and collect data independently from the same distribution.

To evaluate the performance of the proposed framework (6a)–(6c), it is critical to compute the inner maximizer $\delta_{k,n}^*$ defined by (7). Fortunately, for some convex problems, such as logistic regression, the maximization in (7) has a unique closed-form solution, which will be shown in the simulation section. Thus, we analyze the convergence properties of (6a)–(6c) when $\delta_{k,n}^*$ is unique. We first establish the following affine Lipschitz result for the risk function in (2); proofs are omitted due to space limitations.

Lemma 1. (Affine Lipschitz) *For each agent k , the gradient of $J_k(w)$ is affine Lipschitz, namely, for any $w_1, w_2 \in \mathbb{R}^M$, it holds that*

$$\|\nabla_w J_k(w_2) - \nabla_w J_k(w_1)\|^2 \leq 2L^2 \|w_2 - w_1\|^2 + 8L^2 \epsilon^2 \quad (12)$$

□

Contrary to the traditional analysis of decentralized learning algorithms where the risk functions $J_k(w)$ are generally Lipschitz, it turns out from (12) that under adversarial perturbations, the risks in (2) are now affine Lipschitz. This requires adjustments to the convergence arguments. A similar situation arises, for example, when one studies the convergence of decentralized learning under non-smooth losses — see [5, 26, 27].

To proceed with the convergence analysis, we introduce the gradient noise process, which is defined by

$$\mathbf{s}_{k,n}(\mathbf{w}_{k,n-1}) = \nabla_w Q_k(\mathbf{w}_{k,n-1}; \mathbf{x}_{k,n}^*, \mathbf{y}_{k,n}) - \nabla_w J_k(\mathbf{w}_{k,n-1}) \quad (13)$$

This quantity measures the difference between the approximate gradient (represented by the gradient of the loss) and the true gradient (represented by the gradient of the risk). The following result establishes some useful properties for the gradient noise process, namely, it has zero mean and bounded second-order moment (conditioned on past history).

Lemma 2. (Moments of gradient noise) *For each agent k , the gradient noise defined in (13) is zero mean and its variance satisfies*

$$\mathbb{E} \{ \mathbf{s}_{k,n}(\mathbf{w}_{k,n-1}) | \mathcal{F}_{n-1} \} = 0 \quad (14)$$

$$\mathbb{E} \{ \|\mathbf{s}_{k,n}(\mathbf{w}_{k,n-1})\|^2 | \mathcal{F}_{n-1} \} \leq \beta_{k,\epsilon}^2 \|\tilde{\mathbf{w}}_{k,n-1}\|^2 + \sigma_{k,\epsilon}^2 \quad (15)$$

for some non-negative scalars $\beta_{k,\epsilon}^2$ and $\sigma_{k,\epsilon}^2$ that depend on ϵ and can vary across agents. In the above notation, the quantity \mathcal{F}_{n-1} is the filtration generated by the past history of the random process $\text{col}\{\mathbf{w}_{k,n}\}$, and

$$\tilde{\mathbf{w}}_{k,n-1} = \mathbf{w}^* - \mathbf{w}_{k,n-1} \quad (16)$$

□

The main convergence result is stated next; the proof is again omitted due to space limitations.

Theorem 1. (Network mean-square-error stability) Consider a network of K agents running the adversarial diffusion learning algorithm (6a)–(6c). Under Assumptions 1– 4 and for sufficiently small step size μ , the network converges asymptotically to an $O(\mu)$ -neighborhood around the global minimizer \mathbf{w}^* at an exponential rate, namely,

$$\limsup_{n \rightarrow \infty} \mathbb{E} \|\tilde{\mathbf{w}}_{k,n-1}\|^2 \leq O(\mu) \quad (17)$$

□

The above theorem indicates that the proposed algorithm enables the network to approach an $O(\mu)$ -neighborhood of the robust minimizer \mathbf{w}^* after enough iterations, so that the worst-case performance over all possible perturbations in the small region bounded by ϵ can be effectively minimized.

4. COMPUTER SIMULATIONS

In this section, we illustrate the performance of the proposed algorithm using a logistic regression application. Let γ be a binary variable that takes values from $\{-1, 1\}$, and $\mathbf{h} \in \mathbb{R}^M$ be a feature variable. The robust logistic regression problem by a network of agents employs the risk functions:

$$J_k(w) = \mathbb{E} \max_{\|\delta\| \leq \epsilon} \left\{ \ln(1 + e^{-\gamma(\mathbf{h} + \delta)^\top w}) \right\} \quad (18)$$

The analytical solution for the inner maximizer (i.e., the worst-case perturbation) is given by

$$\delta^* = -\epsilon \gamma \frac{\mathbf{w}}{\|\mathbf{w}\|} \quad (19)$$

which is consistent with the perturbation computed from the fast gradient method (FGM) [7].

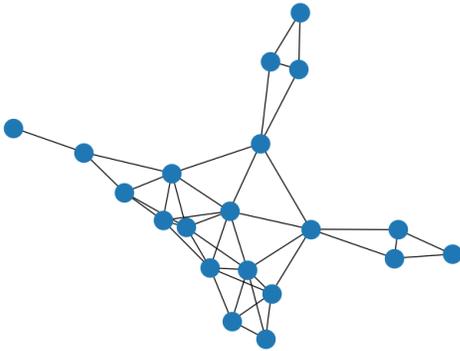


Fig. 1. A randomly generated graph structure used in the simulations

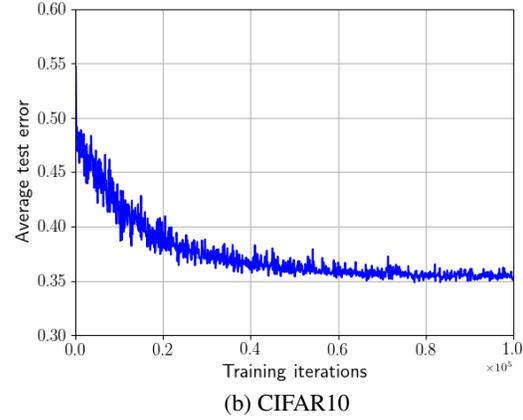
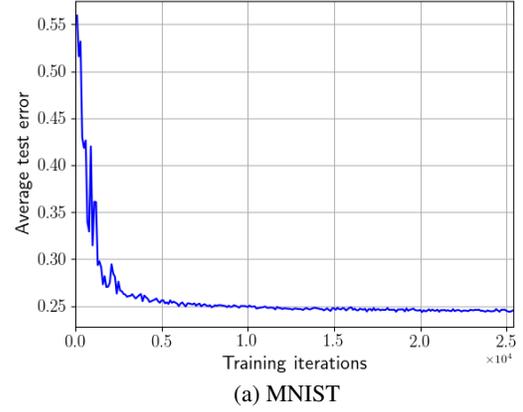
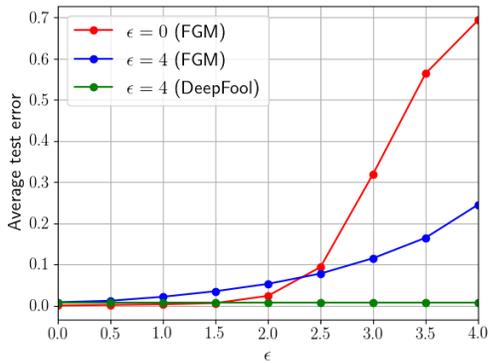


Fig. 2. The convergence plots for the two datasets: (a) The evolution of the average classification error over adversarial examples bounded by $\epsilon = 4$ during training for MNIST; (b) The evolution of the average classification error of adversarial examples bounded by $\epsilon = 1.5$ during training for CIFAR10.

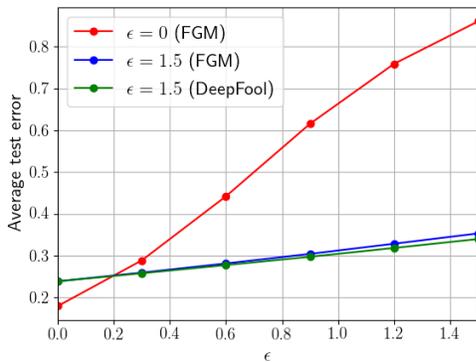
In our experiments, we use both the MNIST [28] and CIFAR10 [29] datasets, and randomly generate a graph with 20 nodes, shown in Fig. 1. We limit our simulations to binary classification in this example. For this reason, we consider samples with digits 0 and 1 from MNIST, and images for airplanes and automobiles from CIFAR10. We set the perturbation bound in (18) to $\epsilon = 4$ for MNIST and $\epsilon = 1.5$ for CIFAR10. In the test phase, we compute the average classification error across the network to measure the performance of the multi-agent system against perturbations of different strengths.

We first illustrate the convergence of our algorithm, as anticipated by Theorem 1. From Fig. 2, we observe a steady decrease in the classification error towards a limiting value.

The robust behavior of the proposed algorithm is illustrated in Fig. 3 for both MNIST and CIFAR10. We explain the curves for MNIST and similar remarks hold for CIFAR10. In the simulation, we use perturbations generated in one of two ways: using the FGM worst-case construction and also using the DeepFool construction [5, 30]. The figure shows three curves. The red curve is obtained by training the network using the traditional diffusion learning strategy



(a) MNIST



(b) CIFAR10

Fig. 3. The robustness plots for the two datasets: (a) Classification error versus perturbation size for MNIST; (b) Classification error versus perturbation size for CIFAR10. The graphs show three plots illustrating the behavior of the traditional (nonrobust) algorithm to worst-case perturbations generated by means of FGM, as well as the performance of the proposed adversarial diffusion strategy (6a)–(6c) to attacks generated by FGM and DeepFool.

without accounting for robustness. The network is subsequently fed with worst-case perturbed samples (generated using FGM) during testing corresponding to different levels of ϵ . The red curve shows that the classification error deteriorates rapidly. The blue curve repeats the same experiment except that the network is now trained with the adversarial diffusion strategy (6a)–(6c). It is seen in the blue curve that the testing error is more resilient and the degradation is better controlled. The same experiment is repeated using the same adversarially trained network, where the perturbed samples are now generated using DeepFool as opposed to FGM. Here again it is seen that the network is resilient and the degradation in performance is better controlled.

In Fig. 4, we plot some randomly selected CIFAR10 images, their perturbed versions, and the classification decisions generated by the nonrobust algorithm and its adversarial version (6a)–(6c). We observe from the figure that no matter which attack method is applied, the perturbations are always imperceptible to the human eye.

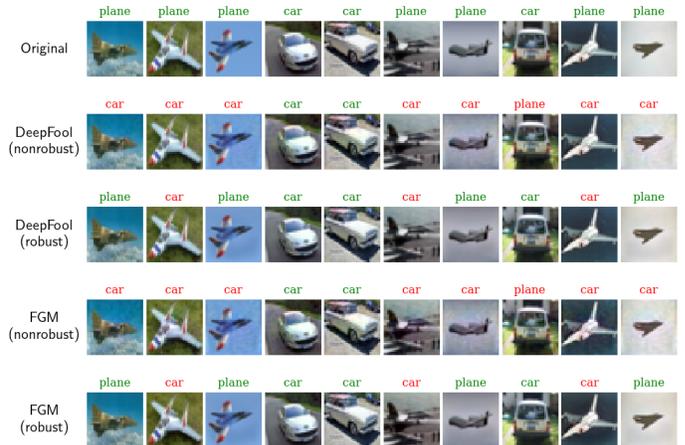


Fig. 4. Visualization of the original and adversarial samples. The first row consists of 10 random original samples with the titles representing their true classes. The second row shows the adversarial examples generated by DeepFool and applied to the standard (nonrobust) algorithm. The third row shows the results obtained by the adversarial (robust) algorithm. The titles are the predictions by the corresponding models. The same construction is repeated in the last two rows using FGM. If the prediction of an image is wrong, the title is shown in red color. It is seen that the adversarial algorithm fails less frequently.

Moreover, while the nonrobust algorithm fails to classify correctly in most cases, the adversarial algorithm is more robust and leads to fewer classification errors.

5. CONCLUSION

In this paper, we proposed a diffusion defense mechanism for adversarial attacks. We analyzed the convergence of the proposed method under convex losses and showed that it approaches a small $O(\mu)$ neighborhood around the robust solution. We further illustrated the behavior of the trained network to perturbations generated by FGM and DeepFool constructions and observed the enhanced robust behavior. Similar results are applicable to nonconvex losses and will be described in future work.

6. REFERENCES

- [1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in *Proc. International Conference on Learning Representations*, Banff, 2014, pp. 1–10.
- [2] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, “Pixeldefend: Leveraging generative models to understand and defend against adversarial examples,” in *Proc. International Conference on Learning Representations*, Vancouver, 2018, pp. 1–20.
- [3] R. Jia and P. Liang, “Adversarial examples for evaluating reading comprehension systems,” in *Proc. Empirical Methods in Natural Language Processing*, Copenhagen, 2017, pp. 1–11.

- [4] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, “Robust adversarial reinforcement learning,” in *Proc. International Conference on Machine Learning*, Sydney, 2017, pp. 2817–2826.
- [5] A. H. Sayed, *Inference and Learning from Data*. Cambridge University Press, 2022.
- [6] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in *Proc. International Conference on Learning Representations*, Vancouver, 2018, pp. 1–23.
- [7] T. Miyato, A. M. Dai, and I. Goodfellow, “Adversarial training methods for semi-supervised text classification,” in *Proc. International Conference on Learning Representations*, Toulon, 2017, pp. 1–11.
- [8] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in *Proc. International Conference on Learning Representations*, San Diego, 2015, pp. 1–11.
- [9] P. Maini, E. Wong, and Z. Kolter, “Adversarial robustness against the union of multiple perturbation models,” in *Proc. International Conference on Machine Learning*, 2020, pp. 6640–6650.
- [10] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, “Theoretically principled trade-off between robustness and accuracy,” in *Proc. International Conference on Machine Learning*, California, 2019, pp. 7472–7482.
- [11] A. H. Sayed, “Adaptation, learning, and optimization over networks,” *Foundations and Trends in Machine Learning*, vol. 7, pp. 311–801, 2014.
- [12] A. H. Sayed, “Adaptive networks,” *Proc. IEEE*, vol. 102, no. 4, pp. 460–497, 2014.
- [13] C. Qin, J. Martens, S. Gowal, D. Krishnan, K. Dvijotham, A. Fawzi, S. De, R. Stanforth, and P. Kohli, “Adversarial robustness through local linearization,” in *Proc. Advances in Neural Information Processing Systems*, Vancouver, 2019, pp. 13 824–13 833.
- [14] G. Zhang, S. Lu, S. Liu, X. Chen, P.-Y. Chen, L. Martie, L. Horesh, and M. Hong, “Distributed adversarial training to robustify deep neural networks at scale,” in *Proc. Conference on Uncertainty in Artificial Intelligence*, Eindhoven, 2022, pp. 2353–2363.
- [15] Y. Liu, X. Chen, M. Cheng, C.-J. Hsieh, and Y. You, “Concurrent adversarial learning for large-batch training,” in *Proc. International Conference on Learning Representations*, 2022, pp. 1–17.
- [16] F. Feng, X. He, J. Tang, and T.-S. Chua, “Graph adversarial training: Dynamically regularizing based on graph structure,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 6, pp. 2493–2504, 2021.
- [17] K. Xu, H. Chen, S. Liu, P. Y. Chen, T. W. Weng, M. Hong, and X. Lin, “Topology attack and defense for graph neural networks: An optimization perspective,” in *Proc. International Joint Conference on Artificial Intelligence*, Macao, 2019, pp. 3961–3967.
- [18] X. Wang, X. Liu, and C.-J. Hsieh, “Graphdefense: Towards robust graph convolutional networks,” *arXiv preprint arXiv:1911.04429*, pp. 1–9, 2019.
- [19] T. Lin, C. Jin, and M. Jordan, “On gradient descent ascent for nonconvex-concave minimax problems,” in *Proc. International Conference on Machine Learning*, 2020, pp. 6083–6093.
- [20] R. T. Rockafellar, *Convex Analysis*. Princeton University Press, 2015.
- [21] K. K. Thekumparampil, P. Jain, P. Netrapalli, and S. Oh, “Efficient algorithms for smooth minimax optimization,” in *Proc. Advances in Neural Information Processing Systems*, Vancouver, 2019, pp. 12 659–12 670.
- [22] A. Sinha, H. Namkoong, R. Volpi, and J. Duchi, “Certifying some distributional robustness with principled adversarial training,” in *Proc. International Conference on Learning Representations*, Vancouver, 2018, pp. 1–34.
- [23] S. Vlaski and A. H. Sayed, “Diffusion learning in non-convex environments,” in *Proc. IEEE ICASSP*, Brighton, 2019, pp. 5262–5266.
- [24] S. Vlaski and A. H. Sayed, “Distributed learning in non-convex environments—part i: Agreement at a linear rate,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 1242–1256, 2021.
- [25] S. Vlaski and A. H. Sayed, “Distributed learning in non-convex environments—part ii: Polynomial escape from saddle-points,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 1257–1270, 2021.
- [26] B. Ying and A. H. Sayed, “Performance limits of stochastic sub-gradient learning, Part I: Single agent case,” *Signal Processing*, vol. 144, pp. 271–282, 2018.
- [27] B. Ying and A. H. Sayed, “Performance limits of stochastic sub-gradient learning, Part II: Multi-agent case,” *Signal Processing*, vol. 144, pp. 253–264, 2018.
- [28] L. Deng, “The mnist database of handwritten digit images for machine learning research,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.
- [29] A. Krizhevsky, V. Nair, and G. Hinton, “Cifar-10 (canadian institute for advanced research).” [Online]. Available: <http://www.cs.toronto.edu/~kriz/cifar.html>
- [30] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “DeepFool: a simple and accurate method to fool deep neural networks,” in *Proc. IEEE CVPR*, Las Vegas, 2016, pp. 2574–2582.