BYZANTINE-ROBUST FEDERATED DEEP DETERMINISTIC POLICY GRADIENT

Qifeng Lin *Qing Ling*

Sun Yat-Sen University

ABSTRACT

Federated reinforcement learning (FRL) combines multi-agent reinforcement learning (MARL) and federated learning (FL) so that multiple agents can exchange messages with a central server for cooperatively learning their local policies. However, a number of malicious agents may deliberately modify the messages transmitted to the central server so as to hinder the learning process, which is often described by the Byzantine attacks model. To address this issue, we propose to employ robust aggregation to replace the simple average aggregation rule in FRL and enhance Byzantine robustness. To be specific, we focus on the episodic task where the environment and agents are reset in the beginning each episode. First, we extend deep deterministic policy gradient (DDPG) to FRL (termed as F-DDPG), which maintains a global critic and multiple local actors, and is thus computation- and communication-efficient. Then, we introduce geometric median and median to aggregate the gradients received from the agents and propose RF-DDPG, a class of Byzantine-robust FRL methods. Finally, we conduct numerical experiments to validate the robustness of RF-DDPG to Byzantine attacks.

***Index Terms*—** Federated learning (FL), Multi-agent reinforcement learning (MARL), Byzantine attacks.

1. INTRODUCTION

Reinforcement learning (RL) aims at training one or multiple agents to learn how to make decisions by interacting with an environment, and has achieved great improvement by combining with deep neural networks recently [1, 2]. Compared to its single-agent counterpart, multi-agent reinforcement learning (MARL) is more challenging due to the exponentially increasing state-action space. To deal with it, centralized training and decentralized execution (CTDE) gradually becomes a standard framework, where decentralized agents learn their local policies in the reduced state-action spaces with the help of a central server [3–5]. CTDE involves both value function-based and policy gradient-based methods; the former are designed for discrete-action tasks while the latter for continuous-action tasks.

Federated learning (FL) is a popular approach to solving large-scale machine learning problems with decentralized data, using a central server to aggregate messages from multiple local workers [6, 7]. It is advantageous in privacy preservation since the workers exchange with the central server learnable models other than local data. Therefore, federated reinforcement learning (FRL), which combines FL and CTDE of MARL, is naturally developed and gains increasing attention in various fields including but not limited to Internet-of-Things (IoT) [8, 9], edge computing [10], etc. However, most of the existing FRL methods build upon the value function-based deep Q-network (DQN) [1] and cannot deal with continuous-action tasks. Policy gradient-based methods such as actor-critic algorithms are able to handle continuous-action tasks, but need to maintain both actor and critic networks in the central server [10], leading to heavy computation and communication burden. In this paper, we consider continuous-action tasks in FRL, and focus on computation- and communication-efficient algorithms where the central server maintain a global critic and the agents train their decentralized actors.

Due to its decentralized nature, FL (and so does FRL) suffers from potential malicious attacks. Among the local workers, a number of them could be malicious, and deliberately modify the messages sent to the central server. Such malicious attacks hinder the training process, and are often characterized by the Byzantine attacks model [11]. Various methods have been proposed to handle Byzantine attacks, and most of them replace the vulnerable average aggregation of stochastic gradients with robust aggregation rules, such as geometric median [12], median [13], Krum [14], to name a few. To deal with non-i.i.d. (independent and identically distributed) samples, [15] introduces robust stochastic model aggregation, other than robust stochastic model aggregation. However, these works aim at optimizing a global static cost function, which does not necessarily exist on FRL. Hence, how to deal with Byzantine attacks in FRL still remains an open problem.

This paper investigates Byzantine-robust FRL for continuous-action tasks, employing the FL framework to train a global critic and decentralized actors. To be specific, we first extend deep deterministic policy gradient (DDPG) [16] to FRL by maintaining a central server for message aggregation, named F-DDPG. Further, we develop its robust extension termed as RF-DDPG by introducing robust aggregation rules to resist Byzantine attacks for episodic tasks, where the environment and agents are reset in the beginning of each episode. We conduct numerical experiments on the cooperative navigation task [3], showing the effectiveness of our proposed methods.

2. RELATED WORK

Our work belongs to MARL with deep neural network approximations for cooperative scenarios, where CTDE has become popular recently. In CTDE, decentralized agents learn their local policies in the reduced state-action spaces with the help of a central server. CTDE methods can be roughly divided into two classes: value function-based and policy gradient-based.

For the value function-based methods, it is critical to learn how to approximate the joint action-value function. Value decomposition networks (VDNs) decompose the joint action-value function to those across the agents [4]. Monotonic value function factorization (QMIX) [17] extends VDNs by introducing a mixing network with non-negative weights to arbitrarily closely approximate any monotonic function. Factorization with transformation [18] removes the intrinsic structural constraints of additivity in VDNs and monotonicity in QMIX, but introduces two extra regularization networks. Based upon QMIX, SMIX(λ) employs a variant of the off-policy SARSA(λ) algorithm to estimate the centralized value func-

Qing Ling (corresponding author) is supported by NSF China Grant 61973324, Guangdong Basic and Applied Basic Research Foundation Grant 2021B1515020094, and Guangdong Province Key Laboratory of Computational Science Grant 2020B1212060032.

tion. These value function-based methods are designed for discrete-action tasks and thus not suitable for continuous-action tasks.

The policy gradient-based methods usually consider continuous-action tasks, and adopt actor-critic as the basic architecture, where the actor is used to learn a policy and the critic aims at estimating the expected long-term average return. Counterfactual multi-agent (COMA) encourages individual agents to deduce their contribution to the team's success, termed as credit assignment. COMA uses the advantage function by comparing the Q-values of the current action and a counterfactual baseline, which marginalizes out a single agent's action but keeps other agents' actions fixed [19]. Different from COMA that employs only one centralized critic, in multi-agent deep deterministic policy gradient (MADDPG), each agent has an individual actor and an individual critic, where the critic is trained using the observations and the actions from all agents [3].

The framework of CTDE naturally adapts to federated learning (FL), yielding federated reinforcement learning (FRL). Federated double deep Q-network (DDQN) equips each agent with a DDQN and aggregates the network parameters at the central server, aiming at minimizing the task completion delay and energy consumption of IoT devices [8]. Federated DDQN is a value function-based method for discrete-action tasks. In actor-critic federated learning (AC-Federate), the agents send the gradients of their local actor and critic networks to the central server, in which global actor and critic networks are updated [10]. Therefore, the computation and communication burden is remarkable. Tightly related to our work is edge federated heterogeneous multi-agent actor-critic (EdgeFed H-MAAC), where the agents' local actor networks are sent to the central server, and the knowledge of different agents is shared [9]. However, in EdgeFed H-MAAC, the global and local actors are eventually the same. In contrast, our works allows the local actors to be different with each other.

Another line of research relevant to our work is robust MARL algorithms. The first class is to defend against perturbations on the agents' observations [20] or adversarial policies of some malicious agents [21]. The second class is to prevent the agents from unsafe states by adding constraints [21–23]. The third class is to resist the attacks on the communication links between the agents [24, 25]. In this paper, we investigate the Byzantine attacks on the communication links between the agents and the central server; that is, some malicious agents may deliberately modify the messages sent to the central to hinder the learning process. Although there have been many algorithms to defend against Byzantine attacks in FL, such as geometric median (GeoMed) [12] and coordinate-wise median (Med) [13], none of them has been studied in the FRL scenarios.

3. PROPOSED METHODS

This section begins with an introduction to MARL. Then, we investigate the extension from DDPG to F-DDPG. Replacing the robust aggregation rule in F-DDPG by robust aggregation yields the proposed RF-DDPG algorithms.

3.1. Multi-Agent Reinforcement Learning

MARL aims at training multiple agents to cooperatively learn their local policies. Consider that there are N agents in a common environment. A MARL task can be modelled as a Markov decision process (MDP) and described by a 5-tuple $\langle \mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, \{\mathcal{R}_i\}_{i=1}^N, \mathcal{P}, \gamma \rangle$, where $\mathcal{S}$ is the state space of the common environment and $\mathcal{A}_i$ is the action space of agent i. At time t, each agent i takes its own action $a_i^t \in \mathcal{A}_i$ given the state $s^t \in \mathcal{S}$, and then receives an immediate reward described by a scalar $r_i^t \in \mathcal{R}_i$. Note that different agents may have different immediate rewards. For notational simplicity, we define the joint action as $\boldsymbol{a}^t = \{a_i^t\}_{i=1}^N$ and the joint action space as $\mathcal{A} = \{\mathcal{A}_i\}_{i=1}^N$. After the joint action $\boldsymbol{a}^t \in \mathcal{A}$ taken, the environment will transit to the next state $s^{t+1} \in \mathcal{S}$ with probability $\mathcal{P}(s^{t+1}|s^t, \boldsymbol{a}^t) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$. The discount factor $\gamma \in [0, 1]$ is a constant. The goal of MARL is to train an (approximately) optimal policy $\pi_i : \mathcal{S} \times \mathcal{A} \to [0, 1]$ for each agent i so that the policies jointly maximize the expected long-term average return $\mathbb{E}[\frac{1}{N}\sum_{t=0}^{\infty}\sum_{i=1}^{N}(\gamma)^t r_i^t]$.

Algorithm 1 Robust Federated Deep Deterministic Policy Gradient

Initialization: $\theta_i^0 = \theta^0, \omega_i^0 = \omega^0$ for each agent i; state s^0; time step $t = 0$
while $t < T$, agent i **do**
  Execute action a_i^t in state s^t by policy π_i^t
  Receive an immediate reward r_i^t and the environment transits to next state s^{t+1}
  Agent i is Regular:
    Calculate $\nabla_{\omega_i^t} L(\omega_i^t)$ and send it to central server
  Agent i is Byzantine:
    Send a malicious message to central server
  Receive from central server $\hat{G}^t$ that aggregates all messages in a robust manner
  Update $\omega_i^{t+1} = \omega_i^t + \alpha \cdot \hat{G}^t$
  Update $\theta_i^{t+1} = \theta_i^t + \alpha \cdot \nabla_{\theta_i} J(\theta_i)$
  Update $\tilde{\omega}_i^{t+1} \leftarrow \tau\omega_i^t + (1-\tau)\tilde{\omega}_i^t$
  Update $\tilde{\theta}_i^{t+1} \leftarrow \tau\theta_i^t + (1-\tau)\tilde{\theta}_i^t$
  $t = t + 1$
end while

3.2. Federated Deep Deterministic Policy Gradient

We extend deep deterministic policy gradient (DDPG) [16] to FRL for continuous-action tasks, named as F-DDPG. For single-agent DDPG, the key is to train a critic to accurately estimate the expected long-term return, and then guide an actor to adjust the policy toward a good return. Therefore, in F-DDPG, the basic idea is to maintain a global critic to tell the agents how to adjust their local policies toward a good team return. This approach is computation- and communication-efficient, especially compared with the algorithm that maintains both actor and critic networks in the central server [10]. Core equations of F-DDPG are given as follows.

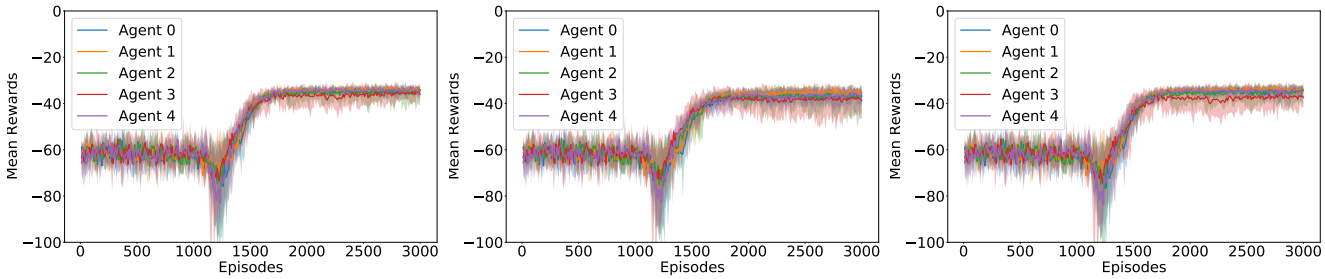
First, the local loss function of the critic at agent i is

$$\begin{aligned} L(\omega_i) &= \mathbb{E}_{(s,\boldsymbol{a},r_i,s',\boldsymbol{a}')\sim\mathcal{B}_i}[(Q(s,\boldsymbol{a}|\omega_i) - y_i)^2], \\ y_i &= r_i + \gamma Q(s', \boldsymbol{a}'|\tilde{\omega}_i), \end{aligned} \tag{1}$$

where $\mathcal{B}_i$ is the local replay buffer to store the transitions and $Q(\cdot)$ represents the action-value function. Agent i maintains a quickly-changed critic network parameterized by ω_i and a slowly-changed target critic network parameterized by $\tilde{\omega}_i$, with $\tilde{\omega}_i \leftarrow \tau\omega_i + (1-\tau)\tilde{\omega}_i$ where $\tau \in [0, 1)$. The transition from the state-action pair $(s, \boldsymbol{a})$ to $(s', \boldsymbol{a}')$ yields reward r_i of agent i. Since r_i is private for agent i, transmitting it for training the global critic is not acceptable due to the risk of privacy leakage. Thus, agent i sends the gradient $\nabla_{\omega_i} L(\omega_i)$ to the central server. After the central server receives all the gradients of the agents' critics, it averages them to obtain a global gradient, given by

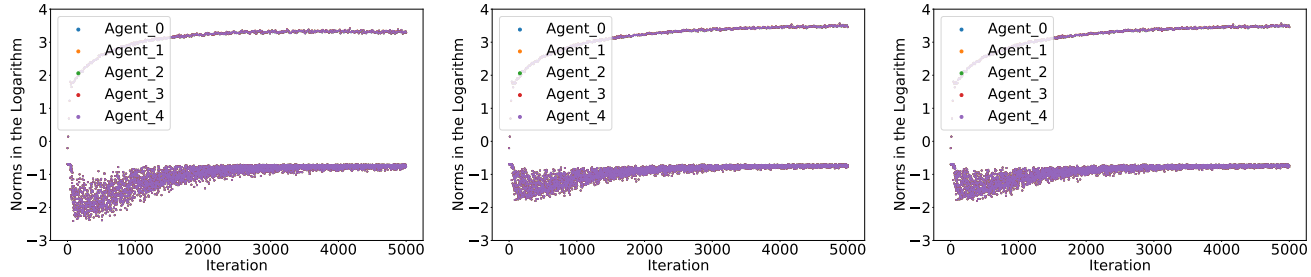
$$G = \frac{1}{N}\sum_{i=1}^{N} \nabla_{\omega_i} L(\omega_i). \tag{2}$$

Fig. 1. Mean rewards without attack.



(a) average (b) median (c) geometric median

Fig. 2. Norms of gradients in logarithm scale under Gaussian attack performed by agent 4.



(a) average (b) median (c) geometric median

Thereafter, the central server sends the aggregated gradient G to all agents for updating their learnable weight of the local critics. In the algorithm, the parameters of the agents are initialized to the same. This way, each agent is able to learn the same estimation of the global expected long-term average return. Since only the gradients of the agents' critic networks are required, this step is computation- and communication- efficient.

With the learned global critic, at agent i, the actor parameterized by θ_i is updated along a policy gradient direction

$$\begin{aligned}&\nabla_{\theta_i} J(\theta_i)\\ =&\mathbb{E}_{(s,\boldsymbol{a})\sim\mathcal{B}_i}[\nabla_{\theta_i} \log \pi_{\theta_i}(a_i|s)\nabla_{a_i} Q_{\pi_{\theta_i}}(s,\boldsymbol{a};\omega_i)|_{a_i=\pi_{\theta_i}(s)}],\end{aligned} \tag{3}$$

where $J(\cdot)$ denotes the global expected long-term average return and π_{θ_i} denotes the local policy of agent i parameterized by θ_i. With policy gradient, each agent can adjust its local policy to achieve a better estimation of the expected long-term average return. Like the introduction of $\tilde{\omega}_i$ to ω_i, we also stabilize the learning process by introducing the target actor network parameterized by $\tilde{\theta}_i$, and update it with $\tilde{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\tilde{\theta}_i$.

3.3. Robust Aggregation

The simple average aggregation rule in (2) is critical to estimate the global expected long-term average return. Although performing well without Byzantine attacks, it is vulnerable to Byzantine attacks. The Byzantine agents can deliberately send malicious gradients to the central server, for example, values generated following the Gaussian distribution, so as to hinder or even destroy the learning process.

Motivated by the Byzantine-robust algorithms in FL, we consider applying the robust aggregation rules rather than simple average aggregation. We denote a robust aggregation rule as $\mathrm{RA}(\cdot)$ that operates on N messages $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$. The first is geometric median (GeoMed) given by

$$\mathrm{GeoMed}(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N) = \arg\max_{\boldsymbol{x}} \sum_{i=1}^{N} ||\boldsymbol{x} - \boldsymbol{x}_j||, \tag{4}$$

which finds a point that has the smallest sum of Euclidean distances between itself and the N points. We also consider coordinate-wise median (Med). For every coordinate j, the output is

$$\mathrm{Med}(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)_j = median(\boldsymbol{x}_{1j}, \boldsymbol{x}_{2j}, \ldots, \boldsymbol{x}_{Nj}), \tag{5}$$

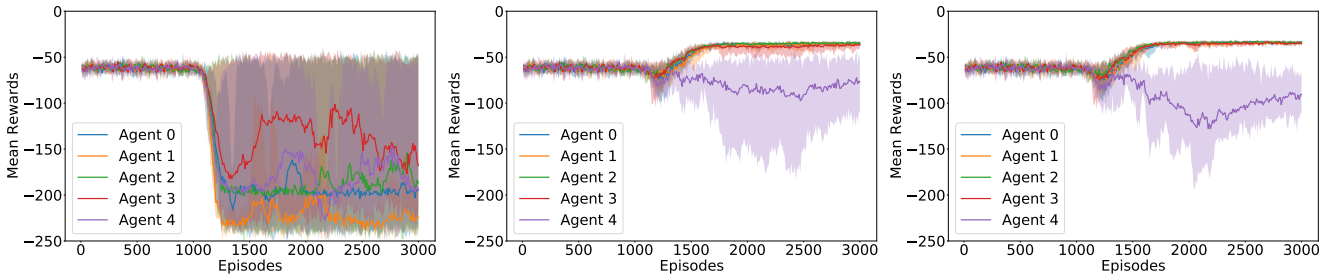
where $\boldsymbol{x}_{ij}$ represents the j-th element of $\boldsymbol{x}_i$ and $median(\cdot)$ selects the median among multiple scalars.

The overview of RF-DDPG is given in Algorithm 1.

4. NUMERICAL EXPERIMENTS

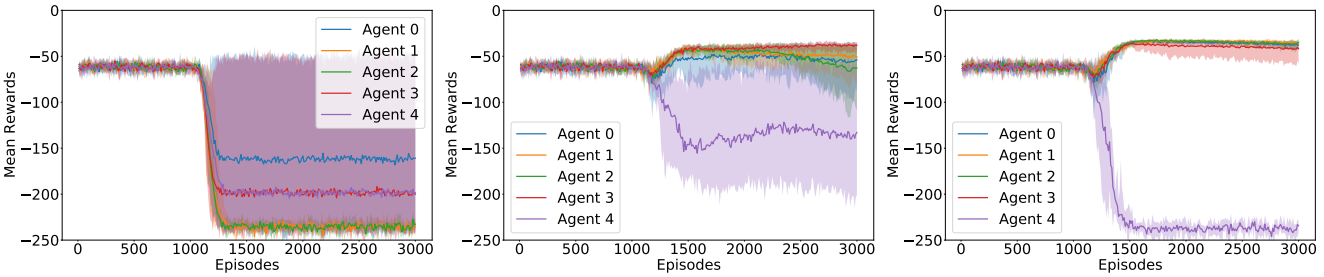
We adopt the cooperative navigation task [3] as our environment. In each episode, N agents and N landmarks are initialized uniformly randomly inside a rectangular region of size 2×2. The agents aim at occupying their target landmarks. In the numerical experiments, we generate $N = 5$ agents, number them from 0 to 4 and also generate 5 corresponding landmarks. The state of the environment consists of positions and velocities of all agents. The agents can move continuously within the region. The immediate reward of each agent is composed of two parts: 1) the negative of the distance between itself and its target landmark; 2) whether a collision happens or not. Each collision contributes -1 to the immediate reward.

Fig. 3. Mean rewards under Gaussian attack performed by agent 4.



(a) average (b) median (c) geometric median

Fig. 4. Mean rewards under sign-flip attack performed by agent 4.



(a) average (b) median (c) geometric median

Each agent is equipped with four neural networks for an actor, a target actor, a critic, and a target critic, respectively. Further, each neural network consists of two hidden layers containing 64 neural units, where the rectified linear unit (ReLU) is chosen as the activation function. The output layers for the actor networks are softmax with random noise, and those for the critic networks are linear. The state and actions of all agents are concatenated to form the input of the critic networks.

We use the Adam optimizer to train the actor and critic networks, where the learning rate is set as $1e^{-4}$. Further, we set $\gamma = 0.9$ and the batch size as 1000. For each experiment, we run 5 random realizations and report the results in average.

We first study the effectiveness of F-DDPG as shown in Fig. 1(a). All agents can achieve relatively high mean rewards. We further consider RF-DDPG with the Med and GeoMed aggregation rules when there are no Byzantine agents. As shown in Fig. 1(b) and (c), GeoMed can achieve nearly the same performance as the simple average aggregation rule while Med is slightly worse than GeoMed. To study why GeoMed and Med are effective in RF-DDPG, for all agents, we plot their norms of gradients in the logarithm scale, as shown in Fig. 2. There is a higher curve due to episodically resetting the environment and agents. We can see that all norms of gradients under three aggregation rules gradually converge to their steady values, which is critical for the robust aggregation rules to filter out malicious messages.

We randomly select the agent 4 as the malicious agent and perform two types of Byzantine attacks. Meanwhile, the malicious agent still updates its local model using the global gradient to track the newest model.

Gaussian attack. Each element of the message sent by a Byzantine agent is generated following the Gaussian distribution $\sim \mathcal{N}(0, \sigma^2)$. We set $\sigma = 100$ here.

Sign-flip attack. The sign of each element of the message sent by a Byzantine agent is flipped, and the value is then multiplied by 10.

Experimental results are given in Figs. 3 and 4. Observe that the simple average aggregation rule is vulnerable to both attacks. The reason is that the messages sent by the malicious agent are totally different from the normal ones, leading the evolutions of the regular agents' local models to be far away from the correct directions. Med and GeoMed are both helpful to defend against the Byzantine attacks. Compared with the simple average aggregation rule, they can yield nearly correct gradients and thus lead the local models to reasonable ones. Meanwhile, GeoMed performs better than Med, suggesting that the Euclidean distance is a suitable metric to measure whether a received message is malicious or not.

Due to the page limit, we have to leave more numerical experiments, including the effects of other Byzantine attacks, other robust aggregation rules and the number of Byzantine agents, to an extended version of this paper. Source code is available online[1].

5. CONCLUSION

In this paper, we first propose a federated deep deterministic policy gradient (F-DDPG) algorithm for FRL. Then, we develop robust federated deep deterministic policy gradient (RF-DDPG) algorithms to defend against Byzantine attacks, where the central server employs geometric median (GeoMed) and coordinate-wise median (Med) to aggregate gradients received from the agents, rather than using the simple average aggregation rule. Finally, we conduct numerical experiments to validate our proposed algorithms and the results show that RF-DDPG is robust to Byzantine attacks.

[1]https://github.com/smielqf/RF-DDPG

6. REFERENCES

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[2] Yaser Keneshloo, Tian Shi, Naren Ramakrishnan, and Chandan K Reddy, "Deep reinforcement learning for sequence-to-sequence models," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 7, pp. 2469–2489, 2019.

[3] Ryan Lowe, YI WU, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, 2017, vol. 30, pp. 6379–6390.

[4] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al., "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 2085–2087.

[5] Xinghu Yao, Chao Wen, Yuhui Wang, and Xiaoyang Tan, "Smix(λ): Enhancing centralized value functions for cooperative multiagent reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2021.

[6] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[7] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2020.

[8] Sheyda Zarandi and Hina Tabassum, "Federated double deep Q-learning for joint delay and energy minimization in iot networks," *arXiv preprint arXiv:2104.11320*, 2021.

[9] Zheqi Zhu, Shuo Wan, Pingyi Fan, and Khaled B Letaief, "Federated multi-agent actor-critic learning for age sensitive mobile edge computing," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[10] Kai-Hsiang Liu, Yi-Huai Hsu, Wan-Ni Lin, and Wanjiun Liao, "Fine-grained offloading for multi-access edge computing with actor-critic federated learning," in *IEEE Wireless Communications and Networking Conference*, 2021, pp. 1–6.

[11] Leslie Lamport, Robert Shostak, and Marshall Pease, "The Byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982.

[12] Yudong Chen, Lili Su, and Jiaming Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, pp. 1–25, 2017.

[13] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta, "Generalized Byzantine-tolerant SGD," *arXiv preprint arXiv:1802.10116*, 2018.

[14] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 118–128.

[15] Liping Li, Wei Xu, Tianyi Chen, Georgios B Giannakis, and Qing Ling, "RSA: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," in *AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 1544–1551.

[16] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[17] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson, "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *International Conference on Machine Learning*, 2018, pp. 4295–4304.

[18] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi, "QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *International Conference on Machine Learning*, 2019, pp. 5887–5896.

[19] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson, "Counterfactual multi-agent policy gradients," in *AAAI Conference on Artificial Intelligence*, vol. 32, pp. 2974–2982.

[20] Jieyu Lin, Kristina Dzeparoska, Sai Qian Zhang, Alberto Leon-Garcia, and Nicolas Papernot, "On the robustness of cooperative multi-agent reinforcement learning," in *IEEE Security and Privacy Workshops*, 2020, pp. 62–68.

[21] Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell, "Adversarial policies: Attacking deep reinforcement learning," in *International Conference on Learning Representations*, 2019.

[22] Songtao Lu, Kaiqing Zhang, Tianyi Chen, Tamer Basar, and Lior Horesh, "Decentralized policy gradient descent ascent for safe multi-agent reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, vol. 35, pp. 8767–8775.

[23] Raghuram Bharadwaj Diddigi, D Sai Koti Reddy, Prabuchandran KJ, and Shalabh Bhatnagar, "Actor-critic algorithms for constrained multi-agent reinforcement learning," in *International Conference on Autonomous Agents and MultiAgent Systems*, 2019, pp. 1931–1933.

[24] Zhaoxian Wu, Han Shen, Tianyi Chen, and Qing Ling, "Byzantine-resilient decentralized td learning with linear function approximation," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3839–3853, 2021.

[25] Martin Figura, Krishna Chaitanya Kosaraju, and Vijay Gupta, "Adversarial attacks in consensus-based multi-agent reinforcement learning," *arXiv preprint arXiv:2103.06967*, 2021.