MMPG: MEDIAN-BASED MULTI-AGENT POLICY GRADIENT METHOD ROBUST TO BYZANTINE ATTACKERS

Xinyang Cao and Lifeng Lai

Department of ECE, University of California, Davis, $\{$xyacao,lflai$\}$@ucdavis.edu

ABSTRACT

Most of existing works in multi-agent reinforcement learning assume that agents will follow the pre-established protocols. However, in practice, there is a risk that some agents might be controlled by Byzantine attackers, who can easily prevent the convergence of existing algorithms or lead the existing algorithms to converge to value of attackers' choice. In this paper, we design a robust median-based multi-agent policy gradient method (MMPG) for cooperative reinforcement learning problems in a distributed setting.

1. INTRODUCTION

There are growing numbers of RL problems that require multiple learner operating in a distributed fashion. In general, multi-agent reinforcement learning (MARL) consider the sequential decision-making problem of multiple autonomous agents, each of which aims to optimize its own long-term return by interacting with the environment and other agents [1]. In MARL problem, different agents may have different goals and the overall RL problem may also have a total goal collected from each agent's goal. In the cooperative setting, all agents want to solve similar problems and they collaborate to optimize a common long-term rewards. In this paper, we will consider the distributed reinforcement learning (DRL) problem with cooperative agents.

There have been many recent work on distributed RL under both collaborative RL and parallel RL setting. [2] proposed distributed RL algorithm for the tabular multiagent MDP model with convergence guarantees. [3] developed a distributed Q-learning algorithm, termed QD-learning, in the network setting where agents can only communicate with their neighbors. [4] provided MADDPG which is an adaptation of actor-critic methods for multiagent RL problems.

In multi-agent RL problems, different agents will communicate through a centralized or decentralized network. Most of the existing works on MARL assume that all agents work correctly and can safely send information to the center controller or its neighbors. However, in practice, there is a risk that some of the agents are compromised and send wrong information, which can be called as Byzantine attackers. Byzantine attackers can reduce the long-term reward in cooperative RL setting or lead the algorithms to converge to values chosen by the attackers by modifying or falsifying intermediate results when the center controller requires these intermediate results for updating.

There are many interesting recent works on designing distributed machine learning algorithms for supervised learning problems that can deal with Byzantine attackers [5, 6, 7, 8, 9, 10, 11, 12, 13, 14]. On the other hand, the existing work on the design of RL algorithms that are robust to Byzantine attackers has been limited, except a recent work [15] that develops value iteration based algorithm. In this paper, we focus on distributed problems based on multi-agent cooperative RL where there are Byzantine attackers in the system. In particular, we propose a new multi-agent *policy gradient* method named median-based multi-agent policy gradient method (MMPG), which is robust to Byzantine attacks. In the considered setup, in each iteration, agents will generate trajectory based on the policy decided by the center control and each agent will compute a policy gradient. If an agent is honest, it will send the computed policy gradient information to the center controller. On the other hand, if an agent is a Byzantine attacker, it will send arbitrary information of its choice to the center controller. After receiving information from all agents, the center controller will aggregate information using geometric median. The geometric median enables the server to mitigate the impact of attackers when up to half of the agents are Byzantine attackers. Using this method, we prove that the algorithm can achieve a sublinear convergence to the stationary point. We show this result by proving that the average of norm of overall cumulative loss gradient on iteration is bounded by certain value. Empirically, we evaluate the performance of MMPG using neural network-parameterized policies on the popular multi-agent RL benchmark for the cooperative navigation task.

2. MODEL

Consider a Markov decision process characterized by a quadruple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \rho, l)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state space and the action space for the agent; $\mathcal{P}$ is the space of the state transition probability of the MDP defined as mappings

This work was supported in part by National Science Foundation under Grants CNS-1824553, CCF-1908258 and ECCS-2000415.

$\mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$; $\gamma \in (0,1)$ is the discounting factor; ρ is the initial state distribution; and $l : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the loss for the agent. When considering MARL problems, we assume that there is a central controller and several agents communicating with the central controller. We will also use a similar quadruple to that in RL problem to describe the model $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \rho, \{l_m\}_{m \in \mathcal{M}})$. The difference is that now $\mathcal{S}$ and $\mathcal{A}$ are the state space and the action space for all agents; $l_m : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the loss (or negative reward) for agent m and there are M agents in set $\mathcal{M} = \{1, \ldots, M\}$.

We will use above defined parameters to describe the distributed RL model. In cooperative distributed RL problem, the goal is to find the optimal policy $\pi(\theta)$, which can generate trajectories of state-action pairs $\mathcal{T} := \{s_0, a_0, s_1, a_1, ...\}$, in order to minimize the infinite-horizon discounted loss aggregated over all agents based on parameter θ:

$$\min_{\theta} \sum_{m \in \mathcal{M}} L_m(\theta), \tag{1}$$

where

$$L_m(\theta) := \mathbb{E}_{\mathcal{T} \sim \mathbb{P}(.|\theta)} \left[\sum_{t=0}^{\infty} \gamma^t l_m(s_t, a_t) \right]. \tag{2}$$

Here, $l_m(s_t, a_t)$ is the loss given the state-action pair (s_t, a_t) and $L_m(\pi)$ is the average cumulative loss for agent m over the trajectory. Given a policy π, the probability of generating trajectory $\mathcal{T}$ is given by

$$\mathbb{P}(\mathcal{T}|\theta) = \rho(s_0) \prod_{t=0}^{\infty} \pi(a_t|s_t, \theta) \mathbb{P}(s_{t+1}|s_t, a_t), \tag{3}$$

where $\rho(s_0)$ is the probability of the initial state s_0, and $\mathbb{P}(s_{t+1}|s_t, a_t)$ is the transition probability from the current state s_t to the next state s_{t+1} by taking action a_t.

This problem can be solved by applying the G(po)MDP-based PG stochastic gradient descent method [16]. The approximate policy gradient of each agent's loss $L_m(\theta)$ can be written as:

$$\hat{\nabla}_{N,T} L_m(\theta^k) = \tag{4}$$
$$\frac{1}{N} \sum_{n=1}^{N} \sum_{t=0}^{T} \left(\sum_{\tau=0}^{t} \nabla \log \pi(a_\tau^n | s_\tau^n; \theta^k) \right) \gamma^t l_m(s_t^n, a_t^n).$$

Here N is the number of trajectories used to estimate the policy gradient.

Every agent then sends this value to the center controller. After receiving values from all agents, the center controller computes the sum and updates θ by $\theta^{k+1} = \theta^k - \alpha \hat{\nabla}_{PG}^k$. This process continues until a certain stop criteria is satisfied.

In this paper, we consider a model with Byzantine workers, in which an unknown subset of agents might be compromised. Furthermore, the set of compromised agents might change over time. If an agent is compromised, instead of the approximate policy gradient calculated from local data, it can send arbitrary information to the server. In particular, let $\mathcal{B}_t$ denote the set of compromised agents at iteration k, the server receives data $g_m(\theta^k)$ from m-th agent with

$$g_m(\theta^k) = \begin{cases} \hat{\nabla}_{N,T} L_m(\theta^k) & m \notin \mathcal{B}_t \\ \star & m \in \mathcal{B}_t \end{cases}, \tag{5}$$

in which $\star$ denotes an arbitrary vector chosen by the attacker.

It is easy to see that if the center controller continues to compute the update direction using the summation, the approximate policy gradient algorithm will not converge. The goal of our paper is to design a robust multi-agent policy gradient algorithm that can tolerate Byzantine attackers.

In this paper, we assume the size of $\mathcal{M}$ is M. Among M agents, up to $q < M/2$ of them can be Byzantine attackers. We also assume the center controller know the value q. Furthermore, we will introduce a parameter for the aggregating method $\beta \in (0, \frac{1}{2})$ and use $D(\cdot \| \cdot)$ to denote the Kullback-Leibler (KL) divergence.

Algorithm 1 Median-based multi-agent approximate policy gradient (MMPG) algorithm

```
**Center controller:**
Initialize randomly selects $\theta^0 \in \Theta$.
**for** $k = 0, 1, 2, ..., K$ **do**
    1: Broadcasts the current model parameter estimator $\theta^k$ to all agents;
    2: Waits to receive policy gradients from agents; $g_m(\theta^k)$ denotes the value received from agent $m$;
    3: Computes $g(\theta^k) = \text{med}\{Mg_1(\theta^k), ..., Mg_M(\theta^k))\}$;
    7:Updates $\theta^{k+1} = \theta^k - \alpha g(\theta^k)$;
**end for**
**Agent** $m$**:**
1: Receives model parameter estimator $\theta^k$, computes the approximate policy gradient $\hat{\nabla}_{N,T} L_m(\theta^k)$;
2: If worker $m$ is honest, it sends $\hat{\nabla}_{N,T} L_m(\theta^k)$; If not, it sends the value determined by the attacker;
```

3. ALGORITHM

In this section, we describe our algorithm in distributed RL problem that can tolerate Byzantine attacks. Main steps of the algorithm are listed in Algorithm 1.

The main idea of our algorithm is to use geometric median of the received information as the aggregation method. In particular, after receiving the policy gradient from agents, the center controller computes

$$g(\theta^k) = \text{med}\{Mg_1(\theta^k), ..., Mg_M(\theta^k))\},$$

in which med$\{\cdot\}$ is the geometric median of the vectors. The center controller will then use $g(\theta^k)$ to update the parameters:

$$\theta^{k+1} = \theta^k - \alpha g(\theta^k). \tag{6}$$

After that, the center controller broadcasts new parameter θ^{k+1}.

The formal convergence proof will be provided in the convergence analysis section. Here we highlight the high level idea why the algorithm works. Geometric median is a generalization of median in one-dimension to multiple dimensions, and has been widely used in robust statistics. In particular, let $x_i \in \mathbb{R}^d, i = 1, \cdots, n$, then the geometric median of the set $\{x_1, x_2, ..., x_n\}$ is define as

$$\text{med}\{x_1, x_2, ..., x_n\} := \arg\min_x \sum_{i=1}^{n} \|x_i - x\|. \tag{7}$$

The geometric median has a very nice property that will be useful for our analysis. For example, when the dimension is one, then if strictly more than half points are in $[-r, r]$, the geometric median must lie in $[-r, r]$. When the dimension is larger than one, the geometric median has following property.

Lemma 1. *[17] Let $x_1, x_2, ..., x_n$ be n points in a Hilbert space. Let x^* denote the geometric median of these points. For any $\beta \in (0, 1/2)$, and given $r > 0$, if $\sum_{i=1}^{n} \mathbf{1}_{\{\|x_i\| \le r\}} \ge (1-\beta)n$, then $\|x^*\| \le \mathcal{C}_\beta r$, where $\mathcal{C}_\alpha = 2(1-\beta)/(1-2\beta)$.*

From Lemma 1, we can see that, if a majority number $(1-\beta)n$ of points are inside the Euclidean ball of radius r centered at origin, then the geometric median must be inside the Euclidean ball of radius $\mathcal{C}_\beta r$. From this property we can upper bound geometric median by $\mathcal{C}_\beta r$.

4. CONVERGENCE ANALYSIS

In this section, we analyze the convergence property of the proposed algorithm MMPG with Byzantine attackers.

Before presenting detailed analysis, here we introduce some assumptions and high level ideas. For

$$L(\theta) = \sum_{m \in \mathcal{M}} L_m(\theta),$$

if the sum of $\|\nabla L(\theta)\|$ on iterations can be bounded by some constant value, the algorithm can successfully converge. Hence, we will show this bound by using geometric median method. Now, we introduce two assumptions. These assumptions are similar to those used in [18, 16, 19, 20, 21].

Assumption 1. *For each state-action pair (s,a), the loss $l_m(s, a)$ is bounded as $l_m(s, a) \in [0, \bar{l}]$.*

Assumption 1 requires the boundedness of the instantaneous loss, which makes sense in practice since the systems in real life commonly output finite magnitudes of responses. [18, 16, 19].

Assumption 2. *For each state-action pair (s,a), and any policy parameter $\theta \in \mathbb{R}^d$, there exist constants G and F such that*

$$\|\nabla \log \pi(a|s;\theta)\| \le G \tag{8}$$

and

$$\left| \frac{\partial^2}{\partial\theta_i \partial\theta_j} \log \pi(a|s;\theta) \right| \le F. \tag{9}$$

Assumption 2 requires the score function and its partial derivatives to be bounded, so that the variance of the estimator is bounded.

From Assumptions 1 and 2, the overall accumulated loss function $L(\theta)$ is also M_L-smooth, where

$$M_L = \left(F + G^2 + \frac{2\gamma G^2}{1-\gamma}\right) \frac{\gamma M \bar{l}}{(1-\gamma)^2}. \tag{10}$$

Since the overall accumulated loss function $L(\theta)$ is M_L-smooth, we have the following lemma.

Lemma 2. *Under Assumptions 1 and 2, if the step-size is selected such that $\alpha \le 1/M_L$, then the objective values satisfy*

$$\begin{aligned} L(\theta^{k+1}) - L(\theta^k) &\le -\frac{\alpha}{2}\|\nabla L(\theta^k)\|^2 \\ &+\alpha\|g(\theta^k) - \nabla_T L(\theta^k)\|^2 + \alpha\|\nabla_T L(\theta^k) - \nabla L(\theta^k)\|^2. \end{aligned} \tag{11}$$

This lemma shows the distance of accumulated loss function in each iteration. For the third term in Lemma 2, the distance between infinite policy gradient and finite policy gradient is bounded by the following lemma.

Lemma 3. *[22] For the infinite policy gradient and its finite approximation, for any θ, the distance of them is bounded by*

$$\|\nabla_T L(\theta) - \nabla L(\theta)\| \le \sigma_T, \tag{12}$$

where

$$\sigma_T = MG\bar{l}\left(T + \frac{\gamma}{1-\gamma}\right)\frac{\gamma^T}{1-\gamma}. \tag{13}$$

For the second term in Lemma 2, for each term in $g(\theta^k)$, we have the following lemma.

Lemma 4. *Under Assumptions 1 and 2, for any scalar $\frac{\delta}{K} \in (0, 1)$, the distance between each term in $g(\theta)$ and $\nabla_T L(\theta)$ is*

$$Pr\left(\|M\hat{\nabla}_{N,T} L_m(\theta) - \nabla_T L(\theta)\|^2 \le \sigma^2_{N,\delta/K}\right) \ge 1 - \frac{\delta}{K}.$$

For $\beta \in (0, 1/2)$, define good event as

$$\begin{aligned} \mathcal{E}_\beta = &\left\{ \sum_{m=1}^{M} 1_{\{\|M\hat{\nabla}_{N,T} L_m(\theta) - \nabla_T L(\theta)\|^2 \le \sigma^2_{N,\delta/K}\}} \right\} \\ &\ge M(1-\beta) + q, \end{aligned} \tag{14}$$

where q is the number of Byzantine attackers. When this event happens, the received information from at least $M(1-\beta) + q$ agents is close to the $\nabla_T L(\theta)$. The following lemma gives a lower bound to the probability of good event $\mathcal{E}_\beta$.

Lemma 5. *Suppose for all m, we have*

$$Pr\left(\|M\hat{\nabla}_{N,T}L_m(\theta) - \nabla_T L(\theta)\|^2 \leq \sigma^2_{N,\delta/K}\right) \geq 1 - \frac{\delta}{K},$$

for any $\beta \in (q/M, 1/2)$ and $0 < \delta/K \leq \beta - q/M$. Then

$$Pr\left(\mathcal{E}_\beta \geq M(1-\beta) + q\right) \geq 1 - \exp^{-MD(\beta - q/M||(\delta/K))},$$

where

$$D(\delta'\|\delta) = \delta' \log\frac{\delta'}{\delta} + (1-\delta')\log\frac{1-\delta'}{1-\delta}. \quad (15)$$

Now we can define a new variable $\mathbb{V}^k = L(\theta^k) - L(\theta^*)$ to show the distance between accumulation loss function and the optimal accumulation loss function in each iteration. From the above lemmas, we have the following convergence result.

Theorem 1. *Under Assumptions 1 and 2, if the stepsize $\alpha \leq \frac{1}{M_L}$, for any $\beta \in (q/M, 1/2)$ and $0 < \delta/K \leq \beta - q/M$, with probability at least $1 - \exp^{-MD(\beta - q/M||(\delta/K))}$,*

$$\frac{1}{K}\sum_{k=1}^{K}\|\nabla L(\theta)\|^2 \leq \frac{2M_L}{K}\mathbb{V}^1 + 2C_\beta\sigma^2_{N,\delta/K} + 2\sigma^2_T. \quad (16)$$

This theorem shows that the algorithm achieves a sub-linear convergence to the stationary point when there are at most half number of total agents are Byzantine attackers.

5. NUMERICAL RESULTS

In this section, we provide numerical examples to illustrate the analytical results. We consider the simulation environment of Cooperative Navigation task, which is from the popular OpenAI Gym paradigm [23]. In this task, there are M agents and M landmarks, each agent has a different landmark goal and has reward based on the goal. Agents are connected to a central coordinator. The goal of the whole system is to maximize the reward which is obtained by minimizing the negative of loss function. In the simulation, we assume the state is globally observable. For the parameter setting, we set the environment with $M = 5$ agents. The policy is parameterized by a three-layer neural network where the first and the second hidden layers contain 50 and 20 neurons with ReLU as the activation function and using softmax operator as the output layer. The discounting factor is $\gamma = 0.99$ in all the tests and the stepsize $\alpha = 0.001$. For each episode, both algorithms terminate after $T = 100$ iterations. We run in total $N = 10$ batch episodes in each Monte Carlo run, and report the globally averaged reward from 5 agents. In our numerical simulation, we consider applying two different kinds of attacks during communication : 1) Inverse attack, in which each attacker first calculates the policy gradient but sends the inverse value to the server; and 2) Random attack, in which the attacker randomly generates policy gradient value. In our simulation, we assume 2 agents are Byzantine attackers,and we compare the performance between two algorithms: 1) The proposed median-based multi-agent policy gradient (MMPG) and 2) G(PO)MDP.

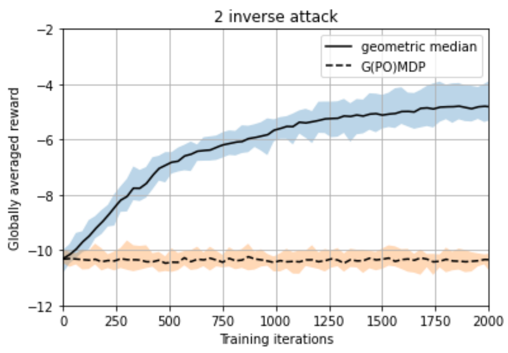

Fig. 1. Globally averaged reward: 2 Inverse attack.

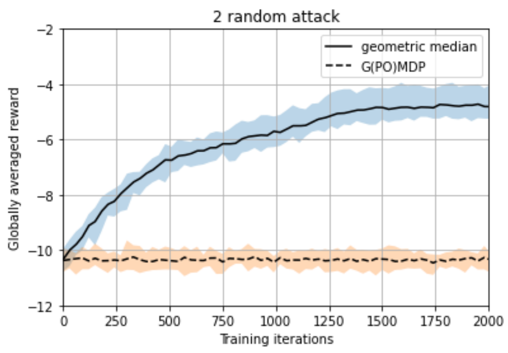

Fig. 2. Globally averaged reward: 2 Random attack.

Figures 1 and 2 plot the value of the globally averaged reward vs iteration with 2 inverse attacks and 2 random attacks respectively in the multi-agents RL task, where all agents solve the shared Cooperative Navigation task with 5 agents. From Figures 1 and 2, we can see that G(PO)MDP method does not work. On the other hand, the proposed MMPG can still benefit from honest agents and shows a better globally averaged reward.

6. CONCLUSION

In this paper, we have proposed a robust median-based multi-agent policy gradient algorithm that can tolerant at most half number of total agents being Byzantine attackers in a cooperative RL setting. We have proved that the proposed algorithm can converge to stationary point. We have also provided numerical examples to illustrate the performance of the proposed algorithm.

7. REFERENCES

[1] L. Busoniu and B. Babuska, R.and De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 156–172, Feb. 2008.

[2] M. Lauer and M. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems," in *In Proceedings of the Seventeenth International Conference on Machine Learning*. Citeseer, 2000.

[3] S. Kar, J. Moura, and H. Poor, "Qd-learning: A collaborative distributed strategy for multi-agent reinforcement learning through consensus," *arXiv preprint arXiv:1205.0047*, 2012.

[4] R. Lowe, Y. Wu, A. Tamar, J. Harb, OpenAI Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[5] Y. Chen, L. Su, and J. 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. 44, Dec. 2017.

[6] L. Chen, Z. Charles, D. Papailiopoulos, et al., "Draco: Robust distributed training via redundant gradients," *arXiv preprint arXiv:1803.09877*, Jun. 2018.

[7] L. Su and J. Xu, "Securing distributed gradient descent in high dimensional statistical learning," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 1, pp. 12, Mar. 2019.

[8] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Defending against saddle point attack in Byzantine-robust distributed learning," *arXiv preprint arXiv:1806.05358*, Sep. 2018.

[9] X. Cao and L. Lai, "Distributed gradient descent algorithm robust to an arbitrary number of Byzantine attackers," *IEEE Trans. Signal Processing*, vol. 67, no. 22, pp. 5850–5864, Nov. 2019.

[10] L. Li, W. Xu, T. Chen, G. Giannakis, and Q. Ling, "Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Jul. 2019, vol. 33, pp. 1544–1551.

[11] R. Jin, Y. Huang, X. He, H. Dai, and T. Wu, "Stochastic-sign SGD for federated learning with theoretical guarantees," *arXiv preprint arXiv:2002.10940*, Feb. 2020.

[12] Z. Yang, A. Gang, and W. Bajwa, "Adversary-resilient inference and machine learning: From distributed to decentralized," *arXiv preprint arXiv:1908.08649*, Feb. 2020.

[13] R. Jin, X. He, and H. Dai, "Distributed Byzantine tolerant stochastic gradient descent in the era of big data," in *Proc. IEEE Intl. Conf. on Communication*, Shanghai, China, May 2019, pp. 1–6.

[14] X. Cao and L. Lai, "Distributed approximate newton's method robust to byzantine attackers," *IEEE Trans. Signal Processing*, vol. 68, pp. 6011–6025, Oct. 2020.

[15] Y. Chen, X. Zhang, K. Zhang, M. Wang, and X. Zhu, "Byzantine-robust online and offline distributed reinforcement learning," 2022, https://arxiv.org/abs/2206.00165.

[16] J. Baxter and P. Bartlett, "Infinite-horizon policy-gradient estimation," *Journal of Artificial Intelligence Research*, vol. 15, pp. 319–350, Nov. 2001.

[17] S. Minsker et al., "Geometric median and robust estimation in banach spaces," *Bernoulli*, vol. 21, no. 4, pp. 2308–2335, Mar. 2015.

[18] Z. Zhang, K.and Yang, H. Liu, T. Zhang, and T. Basar, "Fully decentralized multi-agent reinforcement learning with networked agents," in *Proc. Intl. Conf. on Machine Learning*. PMLR, 2018, pp. 5872–5881.

[19] S. Lu, K. Zhang, T. Chen, T. Başar, and L. 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.

[20] M. Papini, D. Binaghi, G. Canonaco, M. Pirotta, and M. Restelli, "Stochastic variance-reduced policy gradient," in *Proc. Intl. Conf. on Machine Learning*. PMLR, 2018, pp. 4026–4035.

[21] R. Babanezhad Harikandeh, M. Ahmed, A. Virani, M. Schmidt, J. Konečnỳ, and S. Sallinen, "Stopwasting my gradients: Practical svrg," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[22] T. Chen, K. Zhang, G. Giannakis, and T. Basar, "Communication-efficient policy gradient methods for distributed reinforcement learning," *IEEE Transactions on Control of Network Systems*, pp. 917–929, May 2021.

[23] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.