Plan Better Amid Conservatism: Offline Multi-Agent Reinforcement Learning with Actor Rectification

Ling Pan [1] Longbo Huang [1] Tengyu Ma [2] Huazhe Xu [2]

Abstract

Conservatism has led to significant progress in offline reinforcement learning (RL) where an agent learns from pre-collected datasets. However, as many real-world scenarios involve interaction among multiple agents, it is important to resolve offline RL in the multi-agent setting. Given the recent success of transferring online RL algorithms to the multi-agent setting, one may expect that offline RL algorithms will also transfer to multi-agent settings directly. Surprisingly, we empirically observe that conservative offline RL algorithms do not work well in the multi-agent setting—the performance degrades significantly with an increasing number of agents. Towards mitigating the degradation, we identify a key issue that non-concavity of the value function makes the policy gradient improvements prone to local optima. Multiple agents exacerbate the problem severely, since the suboptimal policy by any agent can lead to uncoordinated global failure. Following this intuition, we propose a simple yet effective method, Offline Multi-Agent RL with Actor Rectification (OMAR), which combines the first-order policy gradients and zeroth-order optimization methods to better optimize the conservative value functions over the actor parameters. Despite the simplicity, OMAR achieves state-of-the-art results in a variety of multi-agent control tasks.

1. Introduction

Offline reinforcement learning (RL) has shown great potential in advancing the deployment of RL in real-world tasks where interaction with the environment is prohibitive, costly, or risky (Thomas, 2015). Since an agent has to learn from a given pre-collected dataset in offline RL, it becomes challenging for online off-policy RL algorithms due to extrapolation error (Fujimoto et al., 2019; Lee et al., 2021).

There has been recent progress in tackling the problem based on conservatism. Behavior regularization (Wu et al., 2019; Kumar et al., 2019), *e.g.*, TD3 with Behavior Cloning (TD3+BC) (Fujimoto & Gu, 2021), compels the learning policy to stay close to the data manifold. Yet, its performance highly depends on the data quality. Another line of research incorporates conservatism into the value function by critic regularization (Nachum et al., 2019; Kostrikov et al., 2021a), *e.g.*, Conservative Q-Learning (CQL) (Kumar et al., 2020), which usually learns a conservative estimate of the value function to directly address extrapolation error.

However, many practical scenarios involve multiple agents, *e.g.*, multi-robot control (Amato, 2018), autonomous driving (Pomerleau, 1989; Sadigh et al., 2016). Therefore, offline multi-agent reinforcement learning (MARL) (Yang et al., 2021; Jiang & Lu, 2021; Mathieu et al., 2021) is crucial for solving real-world tasks. Recent results have shown that online RL algorithms can be applied to multi-agent scenarios through either decentralized training or a centralized value function without bells and whistles. For example, PPO (Schulman et al., 2017) leads to the effective methods Independent PPO (Witt et al., 2020) and Multi-Agent PPO (Yu et al., 2021) for the multi-agent setting. Thus, we naturally expect that offline RL algorithms can also transfer easily when applied to multi-agent tasks.

Surprisingly, we find that the performance of the state-of-the-art conservatism-based CQL algorithm in offline RL degrades dramatically with an increasing number of agents, as shown in Figure 2(b) in our experiments. We demonstrate that actor optimization suffers from poor local optima, failing to leverage the global information in the conservative critics well. As a result, it leads to uncoordinated suboptimal learning behavior. The issue is exacerbated severely with more agents and exponentially-sized joint action space (Yang et al., 2021) in the offline setting, because the suboptimal policy of a single agent could lead to a global failure due to lack of coordination. For example, consider a

[1] Institute for Interdisciplinary Information Sciences, Tsinghua University [2] Stanford University. Correspondence to: Ling Pan <pl17@mails.tsinghua.edu.cn>, Longbo Huang <longbohuang@tsinghua.edu.cn>, Tengyu Ma <tengyuma@stanford.edu>, Huazhe Xu <huazhexu@stanford.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

basketball game where there are two competing teams each consisting of five players. When one of the players passes the ball among them, it is important for all the teammates to perform their duties well in their roles to win the game. As a result, if one of the agents in the team fails to learn a good policy, it can fail to cooperate with other agents for coordinated behaviors and lose the ball.

In this paper, we propose a simple yet effective method for offline multi-agent control, Offline MARL with Actor Rectification (OMAR), to better leverage the conservative value functions. Zeroth-order optimization methods, *e.g.*, evolution strategies (Such et al., 2017; Conti et al., 2017; Mania et al., 2018; Salimans et al., 2017), recently emerged as another paradigm for solving decision making tasks that are robust to local optima, while this is usually not the case for first-order policy gradient methods (Nachum et al., 2016; Ge et al., 2017; Safran & Shamir, 2017). Based on this inspiration, we introduce a new combination of the first-order policy gradient and the zeroth-order optimization methods in OMAR so that we can effectively combine the best of both worlds. Towards this goal, in addition to the standard actor gradient update, we encourage the actor to mimic actions from the zeroth-order optimizer that maximize Q-values. Specifically, the zeroth-order optimization part maintains an iteratively updated and refined sampling distribution to find better actions based on Q-values, where we propose an effective sampling mechanism. We then rectify the policy towards these discovered better actions by adding a regularizer to the actor loss.

We conduct extensive experiments in standard continuous control multi-agent particle environments, the complex multi-agent locomotion task, and the challenging discrete control StarCraft II micromanagement benchmark to demonstrate its effectiveness. On all the benchmark tasks, OMAR significantly outperforms strong baselines, including the multi-agent version of current offline RL algorithms including CQL and TD3+BC, as well as a recent offline MARL algorithm MA-ICQ (Yang et al., 2021), and achieves the state-of-the-art performance.

The main contribution of this work can be summarized as follows. We demonstrate the critical challenge of conservatism-based algorithms in the offline multi-agent setting empirically. We propose OMAR, a new algorithm to solve offline MARL tasks. In addition, we theoretically prove that OMAR leads to safe policy improvement. Finally, we conduct extensive experiments to investigate the effectiveness of OMAR. Results show that OMAR significantly outperforms strong baseline methods and achieves state-of-the-art performance in standard continuous and discrete control tasks using offline datasets with different qualities.

2. Background

Partially observable Markov games (POMG) (Littman, 1994; Hu et al., 1998) extend Markov decision processes to the multi-agent setting. A POMG with N agents is defined by a set of global states $\mathcal{S}$, a set of actions $\mathcal{A}_1, \ldots, \mathcal{A}_N$, and a set of observations $\mathcal{O}_1, \ldots, \mathcal{O}_N$ for each agent. At each timestep, each agent i receive an observation o_i and chooses an action based on its policy π_i. The environment transits to the next state according to the state transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A}_1 \times \ldots \times \mathcal{A}_N \times \mathcal{S} \to [0, 1]$. Each agent receives a reward based on the reward function $r_i : \mathcal{S} \times \mathcal{A}_1 \ldots \times \mathcal{A}_N \to \mathbb{R}$ and a private observation $o_i : \mathcal{S} \to \mathcal{O}_i$. The goal is to find a set of optimal policies $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_N\}$, where each agent aims to maximize its own discounted return $\sum_{t=0}^{\infty} \gamma^t r_i^t$ with γ denoting the discount factor. In the offline setting, agents learn from a fixed dataset $\mathcal{D}$ generated by the behavior policy $\boldsymbol{\pi}_\beta$ without interaction with the environments.

2.1. Multi-Agent Actor Critic

Centralized critic. Lowe et al. (2017) propose Multi-Agent Deep Deterministic Policy Gradients (MADDPG) under the centralized training with decentralized execution (CTDE) paradigm by extending the DDPG algorithm (Lillicrap et al., 2016) to the multi-agent setting. In CTDE, agents are trained in a centralized way where they can access to extra global information during training while they need to learn decentralized policies in order to act based only on local observations during execution. In MADDPG, for an agent i, the centralized critic Q_i is parameterized by θ_i. It takes the global state action joint action as inputs, and aims to minimize the temporal difference error defined by $\mathcal{L}(\theta_i) = \mathbb{E}_{\mathcal{D}} \left[(Q_i(s, a_1, \ldots, a_n) - y_i)^2 \right]$, where $y_i = r_i + \gamma \bar{Q}_i(s', a'_1, \cdots, a'_n)|_{a'_j = \bar{\pi}_j(o'_j)}$ and $\bar{Q}_i$ and $\bar{\pi}_i$ denote target networks. To reduce overestimation in MADDPG, MATD3 (Ackermann et al., 2019) estimates the target value based on TD3 (Fujimoto et al., 2018), where $y_i = r_i + \gamma \min_{k=1,2} \bar{Q}_i^k(s', a'_1, \cdots, a'_n)|_{a'_j = \bar{\pi}_j(o'_j)}$. Agents learn decentralized policies π_i parameterized by ϕ_i, which take only local observations as inputs. They are trained by multi-agent policy gradients according to $\nabla_{\phi_i} J(\pi_i) = \mathbb{E}_{\mathcal{D}} \left[\nabla_{\phi_i} \pi_i(a_i|o_i) \nabla_{a_i} Q_i(s, a_1, \ldots, a_n) |_{a_i = \pi_i(o_i)} \right]$, where a_i is predicted from its policy while a_{-i} is sampled from the replay buffer.

Decentralized critic. Although centralized critics are widely-adopted in multi-agent methods, they lack scalability because the joint action space is exponentially large in the number of agents (Iqbal & Sha, 2019). On the other hand, independent learning approaches train decentralized critics that take only the local observation and action as inputs. It is shown in Witt et al. (2020); Lyu et al. (2021) that decentralized value functions can result in more robust performance and be beneficial in practice compared with centralized critic approaches. Witt et al. (2020) propose Independent Proximal Policy Optimization (IPPO) based on PPO (Schulman et al., 2017), and show that it can match or even outperform CTDE approaches in the challenging discrete control benchmark tasks (Samvelyan et al., 2019). We can also obtain the Independent TD3 (ITD3) algorithm based on decentralized critics, which is trained to minimize the temporal difference error defined by $\mathcal{L}(\theta_i) = \mathbb{E}_{\mathcal{D}_i}\left[\left(Q_i\left(o_i, a_i\right) - y_i\right)^2\right]$, where $y_i = r_i + \gamma \min_{k=1,2} \bar{Q}_i^k(o_i', \bar{\pi}_i(o_i'))$.

2.2. Conservative Q-Learning

Conservative Q-Learning (CQL) (Kumar et al., 2020) adds a regularizer to the critic loss to address the extrapolation error and learns lower-bounded Q-values. It penalizes Q-values of state-action pairs sampled from a uniform distribution or a policy while encouraging Q-values for state-action pairs in the dataset to be large. Specifically, when built upon decentralized critic methods in MARL, the critic loss is defined as in Eq. (1), where α is the regularization coefficient and $\hat{\pi}_{\beta_i}$ is the empirical behavior policy of agent i.

$$\mathcal{L}(\theta_i) + \alpha \mathbb{E}_{\mathcal{D}_i}[\log \sum_{a_i} \exp(Q_i(o_i, a_i)) - \mathbb{E}_{a_i \sim \hat{\pi}_{\beta_i}}[Q_i(o_i, a_i)]] \tag{1}$$

3. Proposed Method

In this section, we first provide a motivating example where state-of-the-art offline RL methods, including CQL (Kumar et al., 2020) and TD3+BC (Fujimoto & Gu, 2021), can be inefficient in the face of the challenging multi-agent setting. Then, we propose Offline Multi-Agent Reinforcement Learning with Actor Rectification (OMAR), a simple yet effective method for the actors to better optimize the conservative value functions.

3.1. The Motivating Example

We design a Spread environment as shown in Figure 1 which involves n agents and n landmarks ($n \geq 1$) with 1-dimensional action spaces to demonstrate the problem and reveal interesting findings. For the multi-agent setting in the Spread task, n agents need to learn how to cooperate to cover all the landmarks and avoid colliding with each other or arriving at the same landmark. Therefore, it is important for agents to carefully coordinate their actions. The experimental setup is the same as in Section 4.

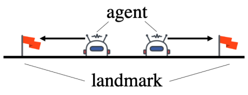
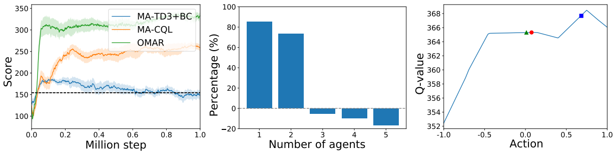


Figure 1: The Spread environment.

Figure 2(a) demonstrates the performance comparison of the multi-agent version of TD3+BC (Fujimoto & Gu, 2021), CQL (Kumar et al., 2020), and OMAR based on ITD3 in the medium-replay dataset from the two-agent Spread environment. As MA-TD3+BC is based on policy regularization that compels the learned policy to stay close to the behavior policy, its performance largely depends on the quality of the dataset. Moreover, it can be detrimental to regularize policies to be close to the dataset in multi-agent settings due to decentralized training and the resulting partial observations. MA-CQL instead learns a lower-bound Q-function to prevent overestimations with additional terms to push down Q-values sampled from a policy while pushing up Q-values for state-action pairs in the dataset. As shown, MA-CQL significantly outperforms MA-TD3+BC in medium-play with more diverse data distribution

However, despite the effectiveness of MA-CQL when exposed to suboptimal trajectories, we surprisingly find that its performance degrades significantly as there are more agents in the cooperative game. This is shown in Figure 2(b), which demonstrates the performance improvement percentage of MA-CQL over the behavior policy with an increasing number of agents from one to five.



(a) (b) (c)

Figure 2: Analysis of MA-TD3+BC, MA-CQL, and OMAR in the medium-replay dataset from Spread. (a) Performance. (b) Performance improvement percentage of MA-CQL over the behavior policy with a varying number of agents. (c) Visualization of the Q-function landscape. The red circle represents the predicted action from the agent using MA-CQL. The green triangle and blue square represent the predicted action from the updated policy of MA-CQL and OMAR.

Towards mitigating the performance degradation, we identify a key issue in MA-CQL that solely regularizing the critic is insufficient for multiple agents to learn good policies for coordination. In Figure 2(c), we visualize the Q-function landscape of MA-CQL during training for an agent in a timestep, with the red circle corresponding to the predicted action from the actor. The green triangle represents the action predicted from the actor after the training step, where the policy gets stuck in a bad local optimum. The first-order policy gradient method is prone to local optima (Dauphin et al., 2014; Ahmed et al., 2019), and we find that the agent can fail to globally leverage the conservative value function well thus leading to suboptimal, uncoordinated learning behavior. Note that the problem is severely exacerbated in the offline multi-agent setting due to the exponentially sized joint action space in the number of agents (Yang et al., 2021). In addition, it usually requires *each* of the agent to learn a good policy for coordination to solve the task, and the suboptimal policy by any agent could result in uncoordinated global failure. Note that we also investigate MA-CQL in a non-cooperative version of the Spread task in Appendix A.2, whose performance does not degrade with an increasing number of agents. This is because the task does not require careful coordination among agents' policies, which highlights the particularly detrimental effect of this problem in offline MARL.

Increasing the learning rate or the number of updates for actors in MA-CQL does not resolve the problem, where results can be found in Appendix A.1. As a result, to solve this critical challenge, it requires a novel solution instead of blindly tuning hyperparameters.

3.2. Offline MARL with Actor Rectification

Our key observations above identify a critical challenge in offline MARL that policy gradient improvements are prone to local optima given a bad value function landscape, since the cooperative task requires careful coordination and is sensitive to suboptimal actions.

Zeroth-order optimization methods, *e.g.*, evolution strategies (Rubinstein & Kroese, 2013; Such et al., 2017; Conti et al., 2017; Salimans et al., 2017; Mania et al., 2018), offer an alternative for policy optimization that is also robust to local optima (Rubinstein & Kroese, 2013). It has shown a welcoming avenue towards using zeroth-order methods for policy optimization in the parameter space that improves exploration in the online RL setting (Pourchot & Sigaud, 2019).

Based on this inspiration, we propose Offline Multi-Agent Reinforcement Learning with Actor Rectification (OMAR), which incorporates sampled actions based on Q-values to rectify the actor so that it can escape from bad local optima. For simplicity of presentation, we demonstrate our method based on the decentralized training paradigm introduced in Section 2.1, which can also be applied to centralized critics as shown in Appendix C.4. Specifically, we propose the following policy objective by introducing a regularizer:

$$\mathbb{E}_{\mathcal{D}_i}\left[(1-\tau)Q_i(o_i,\pi_i(o_i)) - \tau\left(\pi_i(o_i)-\hat{a}_i\right)^2\right] \qquad (2)$$

where $\hat{a}_i$ is the action provided by the zeroth-order optimizer and $\tau \in [0,1]$ denotes the coefficient. Note that TD3+BC (Fujimoto & Gu, 2021) can be interpreted as using the seen action in the dataset for $\hat{a}_i$. The distinction between *optimized* and *seen* actions enables OMAR to perform well even if the dataset quality is from mediocre to low.

We propose our sampling mechanism motivated by the cross-entropy method (CEM) (Rubinstein & Kroese, 2013), which has shown great potential in RL (Lim et al., 2018). However, CEM does not scale to tasks with high-dimensional space well (Nagabandi et al., 2020). We instead propose to sample actions in a softer way motivated by Williams et al. (2015); Lowrey et al. (2018). Specifically, we sample actions according to an iteratively refined Gaussian distribution $\mathcal{N}(\mu_i,\sigma_i)$. At each iteration j, we draw K candidate actions by $a_i^j \sim \mathcal{N}(\mu_i^j,\sigma_i^j)$ and evaluate all their Q-values. The mean and standard deviation of the sampling distribution is updated and refined according to Eq. (3), which produces a softer update and leverages more samples in the update (Nagabandi et al., 2020). Our sampling mechanism indeed outperforms CEM and random, as illustrated in Section 4.1.2. Our method is built upon CQL, which lower-bounds the true Q-function to largely reduce overestimation. The resulting OMAR method is shown in Algorithm 1.

Algorithm 1 Offline Multi-Agent Reinforcement Learning with Actor Rectification (OMAR).

```
Initialize $Q$-networks $Q_i^1$, $Q_i^2$, policy networks $\pi_i$ with random parameters $\theta_1^i$, $\theta_2^i$, $\phi_i$, and target networks with $\bar{\theta}_i^1 \leftarrow \theta_i^1$, $\bar{\theta}_i^2 \leftarrow \theta_i^2$, $\bar{\phi}_i \leftarrow \phi_i$ for each agent $i \in [1,N]$
for training step $t = 1$ to $T$ do
    for agent $i = 1$ to $N$ do
        Sample a random minibatch of $S$ samples $(o_i, a_i, r_i, o_i')$ from $\mathcal{B}$
        Set $y = r_i + \gamma \min_{j=1,2}\left(\bar{Q}_i^j(o_i', \pi_i(o_i' + \epsilon))\right)$
        Update critics $\theta_i$ to minimize Eq. (1)
        Initialize $\mathcal{N}(\mu_i, \sigma_i)$
        for iteration $j = 1$ to $J$ do
            Draw a population with $K$ individuals $\hat{\mathcal{A}}_i = \{\hat{a}_i^k \sim \mathcal{N}(\mu_i,\sigma_i)\}_{k=1}^K$
            Estimate $Q$-values for $K$ individuals in the population $\{Q_i^1(o_i, \hat{a}_i^k)\}_{k=1}^K$
            Update $\mu_i$ and $\sigma_i$ according to Eq. (3)
        end for
        Obtain the picked candidate action $\hat{a}_i = \arg\max_{\hat{a}_i \in \hat{\mathcal{A}}_i \cup \pi_i(o_i)} Q_i^1(o_i, \hat{a}_i)$
        Update the actor $\phi_i$ to minimize Eq. (2)
        Update target networks: $\bar{\theta}_i^j \leftarrow \rho\theta_i^j + (1-\rho)\bar{\theta}_i^j$ and $\bar{\phi}_i \leftarrow \rho\phi_i + (1-\rho)\bar{\phi}_i$
    end for
end for
```

$$\mu_i^{j+1} = \frac{\sum_{k=1}^K \exp(\beta Q_i^k) a_i^k}{\sum_{m=1}^K \exp(\beta Q_i^m)}, \quad \sigma_i^{j+1} = \sqrt{\sum_{k=1}^K \left(a_i^k - \mu_i^j\right)^2}. \qquad (3)$$

Next, we theoretically justify that OMAR provides a safe policy improvement guarantee. Let $J(\pi_i)$ denote the discounted return of a policy π_i in the empirical MDP $\hat{M}_i$ which is induced by transitions in the dataset $\mathcal{D}_i$, i.e.,

$\hat{M}_i = \{(o_i, a_i, r_i, o_i') \in \mathcal{D}_i\}$. In Theorem 3.1, we give a lower bound on the difference between the policy performance of OMAR over the empirical behavior policy $\hat{\pi}_{\beta_i}$ in the empirical MDP $\hat{M}_i$. The proof is in Appendix B.

Theorem 3.1. *Let π_i^* denote the policy obtained by optimizing Eq. (2), $D(\pi_i, \hat{\pi}_{\beta_i})(o_i) = \frac{1-\hat{\pi}_{\beta_i}(\pi_i(o_i)|o_i)}{\hat{\pi}_{\beta_i}(\pi_i(o_i)|o_i)}$, and $d^{\pi_i}(o_i)$ denote the marginal discounted distribution of observations of policy π_i. Then, we have that* $J(\pi_i^*) - J(\hat{\pi}_{\beta_i}) \geq \frac{\alpha}{1-\gamma}\mathbb{E}_{o_i \sim d^{\pi_i^*}(o_i)}\left[D(\pi_i^*, \hat{\pi}_{\beta_i})(o_i)\right]$
$+\frac{\tau}{1-\tau}\mathbb{E}_{o_i \sim d^{\pi_i^*}(o_i)}\left[(\pi_i^*(o_i) - \hat{a}_i)^2\right]$
$-\frac{\tau}{1-\tau}\mathbb{E}_{o_i \sim d^{\hat{\pi}_{\beta_i}}(o_i), a_i \sim \hat{\pi}_{\beta_i}}\left[(a_i - \hat{a}_i)^2\right]$.

Remark. From Theorem 3.1, the first term on the right-hand side is non-negative, and the difference between the second and third terms is the difference between two expected distances. The former corresponds to the gap between the action from our zeroth-order optimizer $\hat{a}_i$ and the optimal action $\pi_i^*(o_i)$. The latter corresponds to the gap between $\hat{a}_i$ and the action from the behavior policy. Since both terms can be bounded and controlled, we find that OMAR gives a safe policy improvement guarantee over $\hat{\pi}_{\beta_i}$.

It is interesting for future work to theoretically study the identified issue about local optima in MARL. We note that multi-agent optimization has been shown to be theoretically much more challenging than single-agent optimization—it is well-known that finding approximate Nash equilibrium of general two-player games is PPAD-hard (Daskalakis, 2013). Recent works (Daskalakis et al., 2021) also showed that finding local stationary solutions for multi-player optimization is intractable. These results even hold when the environments are given, let alone any unknown environments. Admittedly, these theoretical results are not exactly in the same setting as ours, but we also note that a rigorous analysis of the training dynamics is extremely challenging because all the optimization objectives are non-convex.

3.2.1. The Effect of OMAR in the Spread Task

Now, we investigate whether OMAR can successfully address the identified problem using the Spread environment as an example. We further analyze its effect in offline/online, multi-agent/single-agent settings for a better understanding of the potential of our method.

Can OMAR address the identified problem? In Figure 2(c), the blue square corresponds to the action from the updated actor using OMAR according to Eq. (2). In contrast to the policy update in MA-CQL, OMAR can better leverage the global information in the critic and help the actor to escape from the bad local optimum. Figure 2(a) further validates that OMAR significantly improves MA-CQL in terms of both performance and efficiency. The upper part in Figure 3 shows the performance improvement percentage of OMAR over MA-CQL (y-axis) with a varying number of agents (x-axis), where OMAR always outperforms MA-CQL. We also notice that the performance improvement of OMAR over MA-CQL is much more significant in the multi-agent setting in Spread than in the single-agent setting. This echoes with what is discussed above that the problem becomes more critical with more agents, as it requires each of the agents to learn a good policy based on the conservative value function for a successful joint policy for coordination. Otherwise, it can lead to an uncoordinated global failure.

Is OMAR effective in offline/online, multi-agent/single-agent settings? We next investigate the effectiveness of OMAR in the following settings corresponding to different quadrants in Figure 3: i) offline multi-agent, ii) offline single-agent, iii) online single-agent, iv) online multi-agent. For the online setting, we build our method upon MATD3 based on clipped double estimators with our proposed policy objective in Eq. (2). We evaluate the performance improvement percentage of our method over MATD3. The results for the online setting are shown in the lower part in Figure 3.

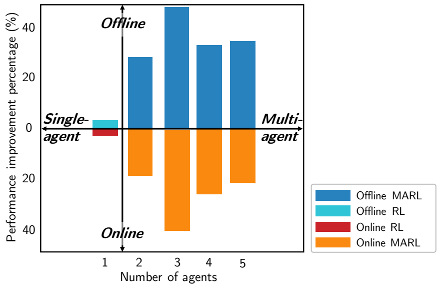


Figure 3: Performance improvement percentage of our method over MA-CQL in the offline (upper part) setting and MATD3 in the online setting (lower part) with a varying number of agents in the Spreak task. The first to fourth quadrants correspond to the offline MARL, offline RL, online RL, and online MARL settings.

As shown in Figure 3, our method is generally applicable in all the settings, with a much more significant performance improvement in the offline setting (upper part) than the online case (lower part). Intuitively, in the online setting, if the actor has not well exploited the global information in the value function, it can still explore and interact with the environment to collect better experiences for improving the value estimation and provides better guidance for the policy. However, in the offline setting, it is much harder for an agent to escape from a bad local optimum and difficult for the actor to best leverage the global information in the conservative critic. As expected, we find that the performance gain is the largest in the offline multi-agent domain.

Table 1: Averaged normalized score of OMAR and baselines in multi-agent particle environments.

		MA-ICQ	MA-TD3+BC	MA-CQL	OMAR
Random	Cooperative navigation	6.3 ± 3.5	9.8 ± 4.9	24.0 ± 9.8	$\mathbf{34.4} \pm 5.3$
	Predator-prey	2.2 ± 2.6	5.7 ± 3.5	5.0 ± 8.2	$\mathbf{11.1} \pm 2.8$
	World	1.0 ± 3.2	2.8 ± 5.5	0.6 ± 2.0	$\mathbf{5.9} \pm 5.2$
Medium -replay	Cooperative navigation	13.6 ± 5.7	15.4 ± 5.6	20.0 ± 8.4	$\mathbf{37.9} \pm 12.3$
	Predator-prey	34.5 ± 27.8	28.7 ± 20.9	24.8 ± 17.3	$\mathbf{47.1} \pm 15.3$
	World	12.0 ± 9.1	17.4 ± 8.1	29.6 ± 13.8	$\mathbf{42.9} \pm 19.5$
Medium	Cooperative navigation	29.3 ± 5.5	29.3 ± 4.8	34.1 ± 7.2	$\mathbf{47.9} \pm 18.9$
	Predator-prey	63.3 ± 20.0	65.1 ± 29.5	61.7 ± 23.1	$\mathbf{66.7} \pm 23.2$
	World	71.9 ± 20.0	73.4 ± 9.3	58.6 ± 11.2	$\mathbf{74.6} \pm 11.5$
Expert	Cooperative navigation	104.0 ± 3.4	108.3 ± 3.3	98.2 ± 5.2	$\mathbf{114.9} \pm 2.6$
	Predator-prey	113.0 ± 14.4	115.2 ± 12.5	93.9 ± 14.0	$\mathbf{116.2} \pm 19.8$
	World	109.5 ± 22.8	110.3 ± 21.3	71.9 ± 28.1	$\mathbf{110.4} \pm 25.7$

4. Experiments

We conduct a series of experiments to study the following key questions:

- How does OMAR compare against state-of-the-art offline RL and MARL methods?
- What is the effect of critical hyperparameters, our sampling mechanism, and the size of the dataset?
- Is OMAR generally applicable to other conservatism-based algorithms?
- Can OMAR scale to the more complex multi-agent locomotion tasks?
- Can OMAR be applied to the challenging discrete control StarCraft II micromanagement benchmarks?
- Is OMAR compatible in single-agent tasks?

The code is publicly available at https://github.com/ling-pan/OMAR, and videos are available at https://sites.google.com/view/omar-videos.

4.1. Multi-Agent Particle Environments

We first conduct a series of experiments in the widely-adopted multi-agent particle environments (Lowe et al., 2017) where the agents need to cooperate to solve the task. The cooperative navigation task includes 3 agents and 3 landmarks, where agents are rewarded based on the distance to the landmarks and penalized for colliding with each other. Thus, it is important for agents to cooperate to cover all landmarks without collision. In predator-prey, 3 predators aim to catch the prey. The predators need to cooperate to surround and catch the prey as the predators are slower than the prey. The world task involves 4 slower cooperating agents that aim to catch 2 faster adversaries, where adversaries desire to eat foods while avoiding being captured.

We construct a variety of datasets according to behavior policies with different qualities based on adding noises to MATD3 to increase diversity following (Fu et al., 2020). The random dataset is generated by rolling out a randomly initialized policy for 1 million (M) steps. We obtain the medium-replay dataset by recording all samples in the replay buffer during training until the policy reached the medium level of the performance. The medium or expert datasets consist of 1M samples by unrolling a partially-pretrained policy with a medium performance level or a fully-trained policy.

We compare OMAR against state-of-the-art offline RL methods including CQL (Kumar et al., 2020) and TD3+BC (Fujimoto & Gu, 2021). We also compare it with a recent offline MARL algorithm MA-ICQ (Yang et al., 2021). We build all the methods on independent TD3 based on decentralized critics, while we also consider centralized critics based on MATD3 with a detailed evaluation in Appendix C.4 that achieves similar performance improvement. Each algorithm is run for five random seeds, and we report the mean performance with standard deviation. A detailed description of hyperparameters and setup can be found in Appendix C.1.2.

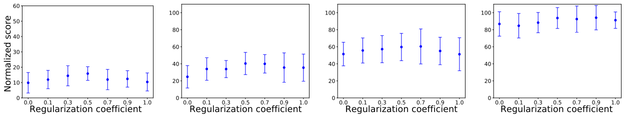
4.1.1. Performance Comparison

Table 1 summarizes the average normalized scores in different datasets in multi-agent particle environments, where the learning curves are shown in Appendix C.2. The normalized score is computed as $100 \times (S - S_{\text{random}})/(S_{\text{expert}} - S_{\text{random}})$ following Fu et al. (2020). As shown, the performance of MA-TD3+BC highly depends on the quality of the dataset. MA-ICQ is based on only trusting seen state-action pairs in the dataset. As shown, it does not perform well in datasets with more diverse data distribution (random and medium-replay), while generally matching the performance of MA-TD3+BC in datasets with narrower distribution (medium and expert). MA-CQL matches or outperforms MA-TD3+BC in datasets with lower quality except for the expert dataset, as it does not rely on constraining the learning policy to stay close to the behavior policy. OMAR significantly outperforms all the baselines and achieves state-of-the-art performance. We attribute the performance gain to the actor rectification scheme that is independent of data quality and improves global optimization. In addition, OMAR does not incur much computation cost and only takes 4.7% more runtime on average compared with that of MA-CQL.

4.1.2. Ablation Study

We now investigate how sensitive OMAR is to key hyperparameters including the regularization coefficient τ and the effect of the sampling mechanism. We also analyze the effect of the size of the dataset in Appendix C.3.

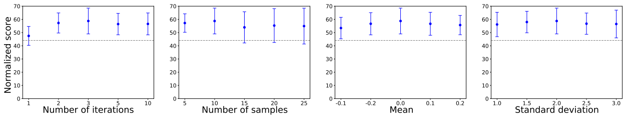
The effect of the regularization coefficient. Figure 4 shows the averaged normalized score of OMAR over different tasks with different values of the regularization coefficient τ in each kind of dataset. As shown, OMAR is sensitive to this hyperparameter, which controls the exploitation level of the critic. We find the best value of τ is neither close to 1 nor 0, showing that it is the combination of both policy gradients and the actor rectification that performs well. We also notice that the optimal value of τ is smaller for datasets with lower quality and more diverse data distribution including random and medium-replay, but larger for medium and expert datasets. In addition, the performance of OMAR with all values of τ matches or outperforms that of MA-CQL. Note that this is the only hyperparameter that needs to be tuned in OMAR beyond MA-CQL.



(a) Random. (b) Med-replay. (c) Medium. (d) Expert.

Figure 4: Ablation study on the regularization coefficient in different types of datasets.

The effect of key hyperparameters in the sampling mechanism. Core hyperparameters for our sampling scheme involve the number of iterations, the number of sampled actions, and the initial mean and standard deviation of the Gaussian distribution. Figure 5 shows the performance comparison of OMAR with different values of these hyperparameters in the cooperative navigation task, where the grey dotted line corresponds to the normalized score of MA-CQL. As shown, our sampling mechanism is not sensitive to these hyperparameters, and we therefore fix them for all types of the tasks to be the same set with the best performance.



(a) Iterations. (b) Samples. (c) Mean. (d) Std.

Figure 5: Ablation study on key hyperparameters in the sampling mechanism averaged over different types of datasets.

The effect of the sampling mechanism. We now analyze the effect of the zeroth-order optimizer in OMAR, and compare our sampling scheme against random sampling and the cross-entropy method (CEM) (De Boer et al., 2005) in the cooperative navigation task. As shown in Table 2, our sampling mechanism significantly outperforms CEM and random sampling in all types of datasets with different qualities. It enjoys a larger margin in datasets with lower quality including random and medium-replay. This is because the proposed sampling mechanism incorporates more samples into the distribution in a softer way, which updates in a more effective way.

Table 2: Ablation study of OMAR with different sampling mechanisms in different types of datasets.

	OMAR (random)	OMAR (CEM)	OMAR
Random	24.3 ± 7.0	25.8 ± 7.3	$\mathbf{34.4} \pm 5.3$
Med-rep	23.5 ± 5.3	32.6 ± 5.1	$\mathbf{37.9} \pm 5.3$
Medium	41.2 ± 11.1	45.0 ± 13.3	$\mathbf{47.9} \pm 18.9$
Expert	101.0 ± 5.2	106.4 ± 13.8	$\mathbf{114.9} \pm 2.6$

4.2. Applicability to Other Algorithms

To demonstrate that the proposed approach is generally applicable to different algorithms, we build OMAR upon another recent conservatism-based offline RL method, EDAC (An et al., 2021), to study its effect. Table 3 illustrates the comparison results of the multi-agent version of EDAC (MA-EDAC) and a variant of OMAR built upon MA-EDAC in multi-agent particle environments. As shown, OMAR provides consistent performance improvement over MA-EDAC, which demonstrates that our method is versatile. As a result, OMAR is generally applicable to different methods with conservative value estimates.

Table 3: Averaged normalized score of MA-EDAC and OMAR based on MA-EDAC in multi-agent particle environments.

		MA-EDAC	OMAR (based on MA-EDAC)
Random	Co-navi	29.9 ± 13.3	**35.7** ± 12.2
	Pred-prey	10.1 ± 6.5	**11.5** ± 1.7
	World	13.5 ± 5.3	**16.5** ± 4.0
Med -rep	Co-navi	34.1 ± 8.2	**38.5** ± 6.7
	Pred-prey	31.4 ± 8.9	**58.0** ± 14.9
	World	34.5 ± 15.0	**43.5** ± 11.3
Medium	Co-navi	39.9 ± 14.2	**51.6** ± 13.8
	Pred-prey	64.4 ± 14.6	**70.8** ± 11.1
	World	72.1 ± 7.0	**76.3** ± 13.0
Expert	Co-navi	99.6 ± 7.7	**114.7** ± 3.7
	Pred-prey	93.7 ± 11.0	**114.5** ± 8.1
	World	93.5 ± 20.3	**103.7** ± 21.2

4.3. Multi-Agent MuJoCo

In this section, we investigate whether OMAR can scale to the more complex continuous control multi-agent task. We consider the multi-agent HalfCheetah task from the multi-agent MuJoCo environment (Peng et al., 2020), which extends the high-dimensional MuJoCo locomotion tasks in the single-agent setting to the multi-agent case. In this environment, agents control different parts of joints of the robot as shown in Appendix C.1.1. These agents need to cooperate to make the robot run forward by coordinating their actions. Different types of datasets are constructed following the same way as in Section 4.1.

Table 4 summarizes the average normalized scores in each kind of dataset in multi-agent HalfCheetah. As shown, OMAR significantly outperforms baseline methods in random, medium-replay, and medium datasets, and matches the performance of MA-TD3+BC in expert, demonstrating its effectiveness to scale to more complex control tasks. It is also worth noting that the performance of MA-TD3+BC depends on the quality of the data, which underperforms OMAR in other types of dataset except for expert.

Table 4: Average normalized score of different methods in multi-agent HalfCheetah.

	ICQ	TD3+BC	CQL	OMAR
Random	7.4 ± 0.0	7.4 ± 0.0	7.4 ± 0.0	**13.5** ± 7.0
Med-rep	35.6 ± 2.7	27.1 ± 5.5	41.2 ± 10.1	**57.7** ± 5.1
Medium	73.6 ± 5.0	75.5 ± 3.7	50.4 ± 10.8	**80.4** ± 10.2
Expert	110.6 ± 3.3	**114.4** ± 3.8	64.2 ± 24.9	**113.5** ± 4.3

4.4. StarCraft II Micromanagement Benchmark

In this section, we further study the effectiveness of OMAR in larger-scale tasks based on the StarCraft II micromanagement benchmark (Samvelyan et al., 2019) on maps with an increasing number of agents and difficulties including 2s3z, 3s5z, 1c3s5z, and 2c_vs_64zg. We compare OMAR and the most competitive method, MA-CQL, based on the evaluation protocol in Kumar et al. (2020); Agarwal et al. (2020); Gulcehre et al. (2020), where datasets are constructed following Agarwal et al. (2020); Gulcehre et al. (2020) by recording samples observed during training. Each dataset consists of 1M samples. We use the Gumbel-Softmax reparameterization method (Jang et al., 2016) to generate discrete actions for MATD3 since it requires differentiable policies (Lowe et al., 2017; Iqbal & Sha, 2019; Peng et al., 2020). A detailed description of the tasks and implementation details can be found in Appendix C.1.

Figure 6 demonstrates the comparison results in test win rates. As shown, OMAR significantly outperforms MA-CQL in performance and learning efficiency. The average performance gain of OMAR compared to MA-CQL is 76.7% in all tested maps, showing that OMAR is also effective in the challenging discrete control StarCraft II micromanagement tasks.

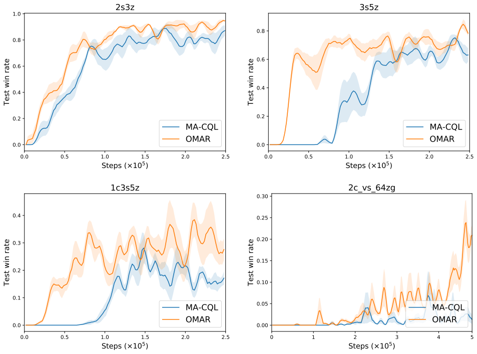


Figure 6: Comparison of test win rates in StarCraft II micromanagement tasks.

4.5. Compatibility in Single-Agent D4RL Environments

Besides the single-agent setting of the Spread task we have studied in Figure 3, we also evaluate the effectiveness of our method in single-agent tasks in the Maze2D domain from the D4RL benchmark (Fu et al., 2020). Table 5 shows the comparison results of each method in an increasing order of complexity of the maze including umaze, medium, and large. Based on the results in Table 5 and Figure 3, we find that OMAR also outperforms CQL, indicating that OMAR is also compatible for single-agent control as discussed in the previous section.

Table 5: Averaged normalized score of OMAR and baselines in the single-agent Maze2D domain.

	umaze	medium	large
TD3+BC	41.1 ± 4.9	75.5 ± 27.1	103.9 ± 31.4
ICQ	4.8 ± 3.8	13.0 ± 7.9	9.2 ± 20.0
CQL	109.8 ± 23.9	106.4 ± 11.0	94.6 ± 44.6
OMAR	**124.7** ± 7.6	**125.7** ± 12.3	**157.7** ± 12.3

5. Related Work

Offline reinforcement learning. Many recent papers achieve improvements in offline RL (Wu et al., 2019; Kumar et al., 2019; Yu et al., 2020; Kidambi et al., 2020; Wang et al., 2020; Argenson & Dulac-Arnold, 2020; Kostrikov et al., 2021a;b; Wu et al., 2021; An et al., 2021) that address the extrapolation error. Behavior regularization typically compels the learning policy to stay close to the behavior policy. Yet, its performance relies heavily on the quality of the dataset. Critic regularization approaches typically add a regularizer to the critic loss which pushes down Q-values for actions sampled from a given policy (Kumar et al., 2020) for learning conservative values. An et al. (2021) propose an ensemble method based on clipped Q-learning for uncertainty penalization. As discussed above, it can be difficult for the actor to best leverage the global information in the conservative critic as policy gradient methods are prone to local optima, which is important in the offline multi-agent setting.

Multi-agent reinforcement learning. A number of multi-agent policy gradient algorithms train agents based on centralized value functions (Lowe et al., 2017; Foerster et al., 2018; Rashid et al., 2018; Yu et al., 2021; Pan et al., 2021) while another line of research focuses on decentralized training (Witt et al., 2020). Yang et al. (2021) show that the extrapolation error in offline RL can be more severe in the multi-agent setting than the single-agent case due to the exponentially sized joint action space w.r.t. the number of agents. In addition, it presents a critical challenge in the decentralized setting when the datasets for each agent only consist of its own action instead of the joint action (Jiang & Lu, 2021). Jiang & Lu (2021) address the challenges based on the behavior regularization BCQ (Fujimoto et al., 2019) algorithm while Yang et al. (2021) propose to estimate the target value based on the next action from the dataset, where both methods can largely depend on the quality of the dataset. OMAR do not restrict its value estimate based only on seen state-action pairs in the dataset, and therefore performs well in datasets with diffrent quality.

Zeroth-order optimization method. It has been recently shown in (Such et al., 2017; Conti et al., 2017; Mania et al., 2018) that evolutionary strategies (ES) emerge as another paradigm for continuous control. Recent research shows that it has the potential to combine RL with ES to reap the best of both worlds (Khadka & Tumer, 2018; Pourchot & Sigaud, 2019) in the high-dimensional parameter space for the actor. Sun et al. (2020) replace the policy gradient update via supervised learning based on sampled noises from random shooting. Kalashnikov et al. (2018); Lim et al. (2018); Simmons-Edler et al. (2019); Peng et al. (2020) extend Q-learning based approaches to handle continuous action space based on the popular cross-entropy method (CEM) in ES.

6. Conclusion

In this paper, we study the important and challenging offline multi-agent RL setting, where we identify that directly extending current conservatism-based RL algorithms to offline multi-agent scenarios results in severe performance degradation along with an increasing number of agents through empirical analysis. We propose a simple yet effective method, OMAR, to tackle the problem by combining the first-order policy gradient with the zeroth-order optimization methods. We find that OMAR successfully helps the actor escape from bad local optima and consequently find better actions. Extensive experiments show that OMAR significantly outperforms state-of-the-art baselines on a variety of multi-agent control tasks. Interesting future directions include theoretical study for our identified problem in the offline multi-agent case with deep neural networks , utilizing a more general class of distributions for the sampling mechanism, and the application of OMAR to other multi-agent RL methods.

Acknowledgements

We thank Bei Peng for help with results of MATD3 in the StarCraft II micromanagement benchmark. We also thank Kefan Dong, Colin Wei, Yuping Luo, Garrett Thomas, and Jeff Z. Haochen for insightful discussion. The work of Ling Pan and Longbo Huang is supported by the Technology and Innovation Major Project of the Ministry of Science and Technology of China under Grant 2020AAA0108400 and 2020AAA0108403, and the Tsinghua University Initiative Scientific Research Program. Tengyu Ma acknowledges the support of Google Faculty Award, NSF IIS 2045685, the Sloan Fellowship, and JD.com.

References

Ackermann, J., Gabler, V., Osa, T., and Sugiyama, M. Reducing overestimation bias in multi-agent domains using double centralized critics. *arXiv preprint arXiv:1910.01465*, 2019.

Agarwal, R., Schuurmans, D., and Norouzi, M. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pp. 104–114. PMLR, 2020.

Ahmed, Z., Le Roux, N., Norouzi, M., and Schuurmans, D. Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning*, pp. 151–160. PMLR, 2019.

Amato, C. Decision-making under uncertainty in multi-agent and multi-robot systems: Planning and learning. In *IJCAI*, pp. 5662–5666, 2018.

An, G., Moon, S., Kim, J.-H., and Song, H. O. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in Neural Information Processing Systems*, 34, 2021.

Argenson, A. and Dulac-Arnold, G. Model-based offline planning. *arXiv preprint arXiv:2008.05556*, 2020.

Conti, E., Madhavan, V., Such, F. P., Lehman, J., Stanley, K. O., and Clune, J. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. *arXiv preprint arXiv:1712.06560*, 2017.

Daskalakis, C. On the complexity of approximating a nash equilibrium. *ACM Transactions on Algorithms (TALG)*, 9(3):1–35, 2013.

Daskalakis, C., Skoulakis, S., and Zampetakis, M. The complexity of constrained min-max optimization. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1466–1478, 2021.

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2933–2941, 2014.

De Boer, P.-T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *arXiv preprint arXiv:2106.06860*, 2021.

Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596. PMLR, 2018.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062. PMLR, 2019.

Ge, R., Lee, J. D., and Ma, T. Learning one-hidden-layer neural networks with landscape design. *arXiv preprint arXiv:1711.00501*, 2017.

Gulcehre, C., Wang, Z., Novikov, A., Le Paine, T., Gomez Colmenarejo, S., Zolna, K., Agarwal, R., Merel, J., Mankowitz, D., Paduraru, C., et al. Rl unplugged: Benchmarks for offline reinforcement learning. *arXiv e-prints*, pp. arXiv–2006, 2020.

Hu, J., Wellman, M. P., et al. Multiagent reinforcement learning: theoretical framework and an algorithm. In *ICML*, volume 98, pp. 242–250. Citeseer, 1998.

Iqbal, S. and Sha, F. Actor-attention-critic for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 2961–2970. PMLR, 2019.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Jiang, J. and Lu, Z. Offline decentralized multi-agent reinforcement learning. *arXiv preprint arXiv:2108.01832*, 2021.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Khadka, S. and Tumer, K. Evolution-guided policy gradient in reinforcement learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 1196–1208, 2018.

Kidambi, R., Rajeswaran, A., Netrapalli, P., and Joachims, T. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kostrikov, I., Fergus, R., Tompson, J., and Nachum, O. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pp. 5774–5783. PMLR, 2021a.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021b.

Kumar, A., Fu, J., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949*, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191, 2020.

Lee, S., Seo, Y., Lee, K., Abbeel, P., and Shin, J. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. *arXiv preprint arXiv:2107.00591*, 2021.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In *ICLR (Poster)*, 2016.

Lim, S., Joseph, A., Le, L., Pan, Y., and White, M. Actor-expert: A framework for using q-learning in continuous action spaces. *arXiv preprint arXiv:1810.09103*, 2018.

Littman, M. L. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*, 30:6379–6390, 2017.

Lowrey, K., Rajeswaran, A., Kakade, S., Todorov, E., and Mordatch, I. Plan online, learn offline: Efficient learning and exploration via model-based control. *arXiv preprint arXiv:1811.01848*, 2018.

Lyu, X., Xiao, Y., Daley, B., and Amato, C. Contrasting centralized and decentralized critics in multi-agent reinforcement learning. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 844–852, 2021.

Mania, H., Guy, A., and Recht, B. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

Mathieu, M., Ozair, S., Srinivasan, S., Gulcehre, C., Zhang, S., Jiang, R., Le Paine, T., Zolna, K., Powell, R., Schrittwieser, J., et al. Starcraft ii unplugged: Large scale offline reinforcement learning. In *Deep RL Workshop NeurIPS 2021*, 2021.

Nachum, O., Norouzi, M., and Schuurmans, D. Improving policy gradient by exploring under-appreciated rewards. *arXiv preprint arXiv:1611.09321*, 2016.

Nachum, O., Dai, B., Kostrikov, I., Chow, Y., Li, L., and Schuurmans, D. Algaedice: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019.

Nagabandi, A., Konolige, K., Levine, S., and Kumar, V. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pp. 1101–1112. PMLR, 2020.

Pan, L., Rashid, T., Peng, B., Huang, L., and Whiteson, S. Regularized softmax deep multi-agent q-learning. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

Peng, B., Rashid, T., de Witt, C. A. S., Kamienny, P.-A., Torr, P. H., Böhmer, W., and Whiteson, S. Facmac: Factored multi-agent centralised policy gradients. *arXiv preprint arXiv:2003.06709*, 2020.

Pomerleau, D. A. Alvinn: An autonomous land vehicle in a neural network. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA ARTIFICIAL INTELLIGENCE AND PSYCHOLOGY ..., 1989.

Pourchot and Sigaud. CEM-RL: Combining evolutionary and gradient-based methods for policy search. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BkeU5j0ctQ.

Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., and Whiteson, S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 4295–4304. PMLR, 2018.

Rubinstein, R. Y. and Kroese, D. P. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media, 2013.

Sadigh, D., Sastry, S., Seshia, S. A., and Dragan, A. D. Planning for autonomous cars that leverage effects on human actions. In *Robotics: Science and Systems*, volume 2, pp. 1–9. Ann Arbor, MI, USA, 2016.

Safran, I. and Shamir, O. Spurious local minima are common in two-layer relu neural networks. *arXiv preprint arXiv:1712.08968*, 2017.

Salimans, T., Ho, J., Chen, X., Sidor, S., and Sutskever, I. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Samvelyan, M., Rashid, T., Schroeder de Witt, C., Farquhar, G., Nardelli, N., Rudner, T. G., Hung, C.-M., Torr, P. H., Foerster, J., and Whiteson, S. The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2186–2188, 2019.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Simmons-Edler, R., Eisner, B., Mitchell, E., Seung, S., and Lee, D. Q-learning for continuous actions with cross-entropy guided policies. *arXiv preprint arXiv:1903.10605*, 2019.

Such, F. P., Madhavan, V., Conti, E., Lehman, J., Stanley, K. O., and Clune, J. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint arXiv:1712.06567*, 2017.

Sun, H., Xu, Z., Song, Y., Fang, M., Xiong, J., Dai, B., Zhang, Z., and Zhou, B. Zeroth-order supervised policy improvement. *arXiv preprint arXiv:2006.06600*, 2020.

Thomas, P. S. *Safe reinforcement learning*. PhD thesis, University of Massachusetts Libraries, 2015.

Wang, Z., Novikov, A., Zolna, K., Merel, J. S., Springenberg, J. T., Reed, S. E., Shahriari, B., Siegel, N., Gulcehre, C., Heess, N., et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33: 7768–7778, 2020.

Williams, G., Aldrich, A., and Theodorou, E. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015.

Witt, C. S., Gupta, T., Makoviichuk, D., Makoviychuk, V., Torr, P. H., Sun, M., and Whiteson, S. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

Wu, Y., Tucker, G., and Nachum, O. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Wu, Y., Zhai, S., Srivastava, N., Susskind, J., Zhang, J., Salakhutdinov, R., and Goh, H. Uncertainty weighted actor-critic for offline reinforcement learning. *arXiv preprint arXiv:2105.08140*, 2021.

Yang, Y., Ma, X., Li, C., Zheng, Z., Zhang, Q., Huang, G., Yang, J., and Zhao, Q. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. *arXiv preprint arXiv:2106.03400*, 2021.

Yu, C., Velu, A., Vinitsky, E., Wang, Y., Bayen, A., and Wu, Y. The surprising effectiveness of mappo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J. Y., Levine, S., Finn, C., and Ma, T. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

A. Additional Results for Spread

A.1. Results with Larger Learning Rates and Number of Updates of Actors in MA-CQL

Table 6 shows the result of MA-CQL with larger learning rates, where we also include results for using smaller learning rates for reference. Table 7 demonstrates the result of MA-CQL with larger numbers of updates for actors.

Table 6: Performance of MA-CQL with larger learning rate for the actor.

Learning rate	$5e-4$	$1e-3$	$5e-3$	$1e-2$	$5e-2$	$1e-1$
Performance	152.3 ± 17.1	164.0 ± 14.5	256.2 ± 34.2	267.9 ± 19.0	202.0 ± 38.9	100.1 ± 36.4

Table 7: Performance of MA-CQL with larger number of updates for the actor.

# Updates	1	5	20
Performance	267.9 ± 19.0	278.6 ± 14.8	263.7 ± 23.1

A.2. The Performance of MA-CQL in a Non-Cooperative Version of the Multi-Agent Spread Task

We consider a non-cooperative version of the Spread task in Figure 1 which involves n agents and n landmarks, where each of the agents aims to navigate to its own unique target landmark. In contrast to the Spread task that requires cooperation, the reward function for each agent only depends on its distance to its target landmark. This is a variant of Spread that consists of multiple independent learning agents, and the performance is measured by the average return over all agents.

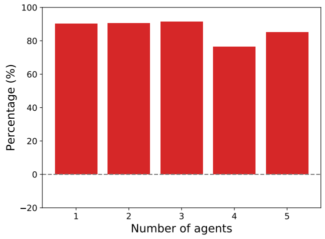


Figure 7: Performance improvement percentage of MA-CQL over the behavior policy with a varying number of agents in a non-cooperative version of the Spread task.

Figure 7 shows the result of the performance improvement percentage of MA-CQL over the behavior policy in the independent Spread task. As shown, the performance of CQL does not degrade with an increasing number of agents in this setting that does not require cooperation, unlike a dramatic performance decrease in the cooperative Spread task in Figure 2(b). The result further confirms that the issue we discovered is due to the failure of coordination.

B. Proof of Theorem 3.1

Theorem 3.1. *Let π_i^* denote the policy obtained by optimizing Eq. (2), $D(\pi_i, \hat{\pi}_{\beta_i})(o_i) = \frac{1-\hat{\pi}_{\beta_i}(\pi_i(o_i)|o_i)}{\hat{\pi}_{\beta_i}(\pi_i(o_i)|o_i)}$, and $d^{\pi_i}(o_i)$ denote the marginal discounted distribution of observations of policy π_i. Then, we have that $J(\pi_i^*) - J(\hat{\pi}_{\beta_i}) \geq \frac{\alpha}{1-\gamma}\mathbb{E}_{o_i \sim d^{\pi_i^*}(o_i)}\left[D(\pi_i^*, \hat{\pi}_{\beta_i})(o_i)\right] + \frac{\tau}{1-\tau}\mathbb{E}_{o_i \sim d^{\pi_i^*}(o_i)}\left[(\pi_i^*(o_i) - \hat{a}_i)^2\right] - \frac{\tau}{1-\tau}\mathbb{E}_{o_i \sim d^{\hat{\pi}_{\beta_i}}(o_i), a_i \sim \hat{\pi}_{\beta_i}}\left[(a_i - \hat{a}_i)^2\right].$*

Proof. For OMAR, we have the following iterative update for agent i:

$$\hat{Q}_i^{k+1} \leftarrow \arg\min_{Q_i} \alpha \mathbb{E}_{o_i \sim \mathcal{D}_i}\left[\mathbb{E}_{a_i \sim \tilde{\pi}_i(a_i|o_i)}\left[Q_i(o_i, a_i)\right] - \mathbb{E}_{a_i \sim \hat{\pi}_{\beta_i}(a_i|o_i)}\left[Q_i(o_i, a_i)\right]\right]$$
$$+ \frac{1}{2}\mathbb{E}_{o_i, a_i, o_i' \sim \mathcal{D}}\left[\left(Q_i(o_i, a_i) - \hat{\mathcal{B}}^{\pi_i}\hat{Q}_i^k(o_i, a_i)\right)^2\right], \tag{4}$$

where $\tilde{\pi}_i(a_i|o_i) = 1$ if and only if $a_i = \pi_i(o_i)$.

Let $\hat{Q}_i^{k+1}$ be the fixed point of solving Equation (4) by setting the derivative of Eq. (4) with respect to Q_i to be 0, then we have that

$$\hat{Q}_i^{k+1}(o_i, a_i) = \hat{\mathcal{B}}^{\pi_i}\hat{Q}_i^k(o_i, a_i) - \alpha\left(\frac{I_{a_i=\pi_i(o_i)}}{\hat{\pi}_{\beta_i}(a_i|o_i)} - 1\right), \tag{5}$$

where I is the indicator function.

Denote $D(\pi_i, \hat{\pi}_{\beta_i})(o_i) = \frac{1}{\hat{\pi}_{\beta_i}(\pi_i(o_i)|o_i)} - 1$, and we obtain the difference between the value function $\hat{V}_i(o_i)$ and the original value function as:

$$\hat{V}_i(o_i) = V_i(o_i) - \alpha D(\pi_i, \hat{\pi}_{\beta_i})(o_i), \tag{6}$$

Then, the policy that minimizes the loss function defined in Eq. (2) is equivalently obtained by maximizing

$$(1-\tau)\left(J(\pi_i) - \alpha\frac{1}{1-\gamma}\mathbb{E}_{o_i\sim d^{\pi_i}_{\hat{M}_i}(o_i)}\left[D(\pi_i, \hat{\pi}_{\beta_i})(o_i)\right]\right) - \tau\mathbb{E}_{o_i\sim d^{\pi_i}_{\hat{M}_i}(o_i)}\left[(\pi_i(o_i) - \hat{a}_i)^2\right]. \tag{7}$$

Therefore, we obtain that

$$\begin{aligned}&(1-\tau)\left(J(\pi_i^*) - \alpha\frac{1}{1-\gamma}\mathbb{E}_{o_i\sim d^{\pi_i^*}_{\hat{M}_i}(o_i)}\left[D(\pi_i^*, \hat{\pi}_{\beta_i})(o_i)\right]\right) - \tau\mathbb{E}_{o_i\sim d^{\pi_i^*}_{\hat{M}_i}(o_i)}\left[(\pi_i^*(o_i) - \hat{a}_i)^2\right] \\ \geq&(1-\tau)J(\hat{\pi}_{\beta_i}) - \tau\mathbb{E}_{o_i\sim d^{\hat{\pi}_{\beta_i}}_{\hat{M}_i}(o_i), a_i\sim\hat{\pi}_{\beta_i}(a_i|o_i)}\left[(a_i - \hat{a}_i)^2\right].\end{aligned} \tag{8}$$

Then, from Eq. (8) we obtain the result. □

C. Experimental Details

C.1. Experimental Setup

C.1.1. Tasks.

We adopt the open-source implementations for multi-agent particle environments[1] from (Lowe et al., 2017), Multi-Agent MuJoCo[2] from (Peng et al., 2020), and StarCraft II Micromanagement Benchmark[3] from (Samvelyan et al., 2019). Figures 8(a)-(c) illustrate tasks from multi-agent particle environments. The two-agent HalfCheetah task is shown in Figure 8(d) while Figures 8(e)-(g) illustrate the Maze2D environments from the D4RL[4] benchmark (Fu et al., 2020). The expert and random scores for cooperative navigation, predator-prey, world, and two-agent HalfCheetah are $\{516.8, 159.8\}$, $\{185.6, -4.1\}$, $\{79.5, -6.8\}$, and $\{3568.8, -284.0\}$, respectively. Tested maps in StarCraft II micromanagement benchmark are summarized in Table 8.

Table 8: Specs of tested maps in the StarCraft II micromanagement benchmark.

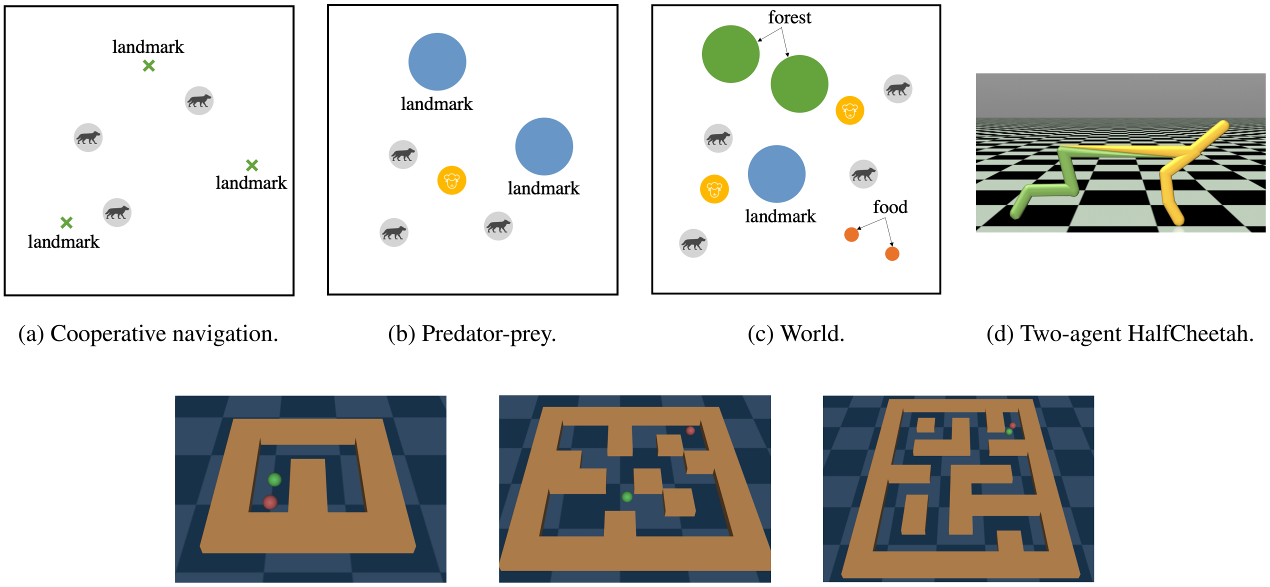
Name	Agents	Enemies
2s3z	2 Stalkers and 3 Zealots	2 Stalkers and 3 Zealots
3s5z	3 Stalkers and 5 Zealots	3 Stalkers and 5 Zealots
1c3s5z	1 Colossi, 3 Stalkers and 5 Zealots	1 Colossi, 3 Stalkers and 5 Zealots
2c_vs_64zg	2 Colossi	64 Zerglings

[1] https://github.com/openai/multiagent-particle-envs

[2] https://github.com/schroederdewitt/multiagent_mujoco

[3] https://github.com/oxwhirl/smac

[4] https://github.com/rail-berkeley/d4rl



(a) Cooperative navigation. (b) Predator-prey. (c) World. (d) Two-agent HalfCheetah.

(e) Maze2D-Umaze-Dense. (f) Maze2D-Medium-Dense. (g) Maze2D-Large-Dense.

Figure 8: Multi-agent particle environments and Multi-Agent HalfCheetah.

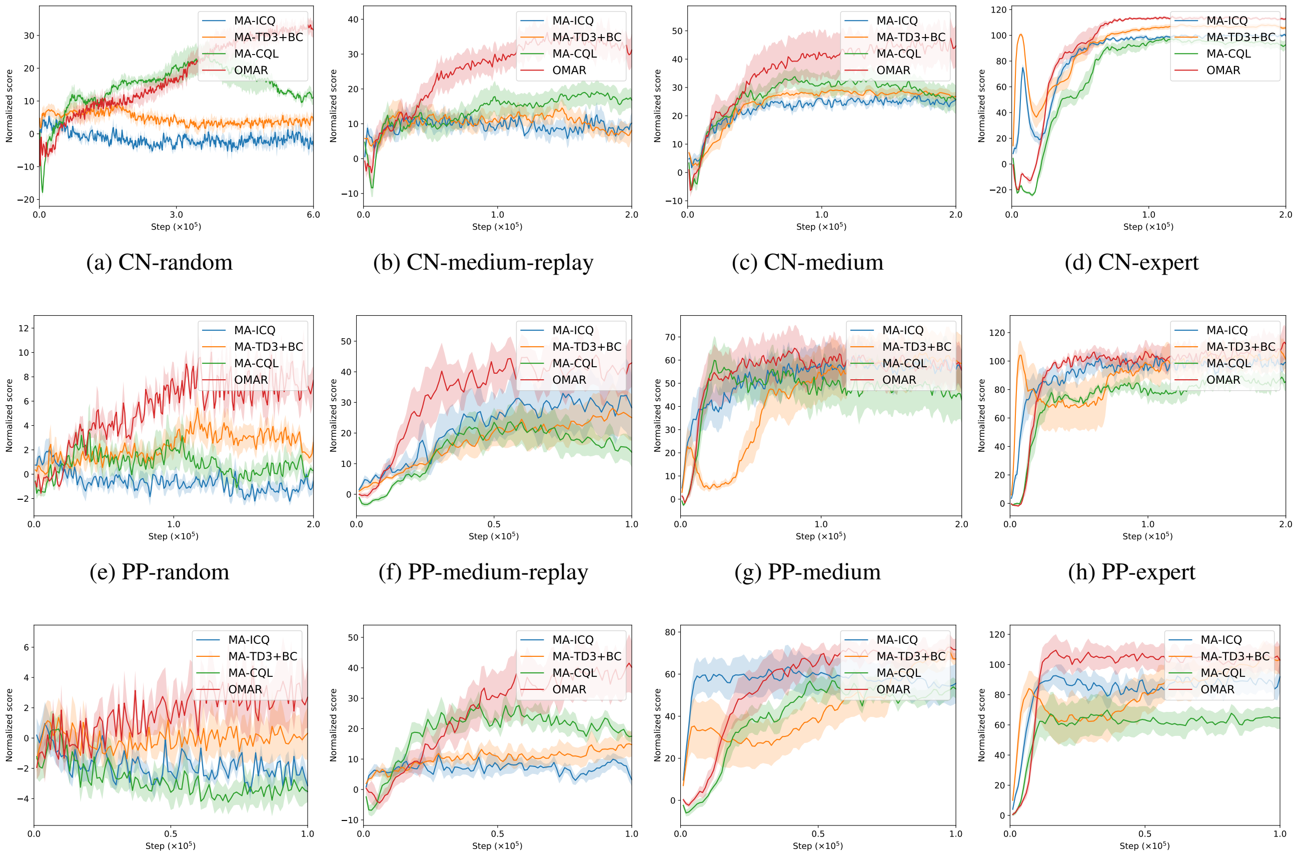
C.1.2. Baselines.

All baseline methods are implemented based on an open-source implementation[5] from (Iqbal & Sha, 2019), where we implement MA-TD3+BC[6], MA-CQL[7], and MA-ICQ[8] based on open-source implementations with fine-tuned hyperparameters. For MA-CQL, we tune a best critic regularization coefficient from $\{0.1, 0.5, 1.0, 5.0\}$ following (Kumar et al., 2020) for each task. Specifically, we use the discount factor γ of 0.99. We sample a minibatch of 1024 samples from the dataset for updating each agent's actor and critic using the Adam (Kingma & Ba, 2014) optimizer with the learning rate to be 0.01. The target networks for the actor and critic are soft updated with the update rate to be 0.01. Both the actor and critic networks are feedforward networks consisting of two hidden layers with 64 neurons per layer using ReLU activation. For OMAR, the only hyperparameter that requires tuning is the regularization coefficient λ, where we use a smaller value for datasets with more diverse data distribution in random and medium-replay with a value of 0.5, while we use a larger value for datasets with more narrow data distribution in medium and expert with values of 0.7 and 0.9 respectively. As OMAR is insensitive to the hyperparameters of the sampling mechanism, we set them to a fixed set of values for all types of datasets in all tasks, where the number of iterations is 3, the number of samples is 10, the mean is 0.0, and the standard deviation is 2.0. For OMAR in the StarCraft II micromanagement benchmark, we follow the fine-tuned set of hyperparameters for MATD3 in (Peng et al., 2020). The code will be released upon publication of the paper.

C.2. Learning Curves

Figure 9 demonstrates the learning curves of MA-ICQ, MA-TD3+BC, MA-CQL and OMAR in different types of datasets in multi-agent particle environments, where the solid line and shaded region represent mean and standard deviation, respectively.

[5] https://github.com/shariqiqbal2810/maddpg-pytorch
[6] https://github.com/sfujim/TD3_BC
[7] https://github.com/aviralkumar2907/CQL
[8] https://github.com/YiqinYang/ICQ



(a) CN-random (b) CN-medium-replay (c) CN-medium (d) CN-expert

(e) PP-random (f) PP-medium-replay (g) PP-medium (h) PP-expert

(i) W-random (j) W-medium-replay (k) W-medium (l) W-expert

Figure 9: Learning curves of MA-ICQ, MA-TD3+BC, MA-CQL, and OMAR in multi-agent particle environments (CN, PP, and W is abbreviated for cooperative navigation, predator-prey, and world respectively).

C.3. Additional Ablation Study on the Effect of the Size of the Dataset

In this section, we conduct an ablation study to investigate the effect of the size of the dataset following the experimental protocol in Agarwal et al. (2020). We first generate a full replay dataset by recording all samples in the replay buffer encountered during the training course for 1 million steps. Then, we randomly sample $N\%$ experiences from the full replay dataset and obtain several smaller datasets with the same data distribution, where $N \in \{0.1, 1, 10, 20, 50, 100\}$.

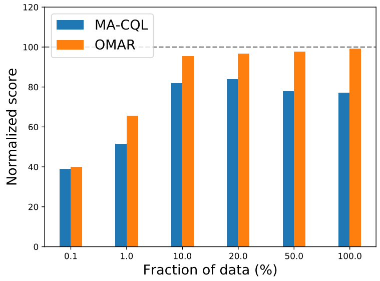


Figure 10: Normalized score of OMAR and MA-CQL trained using a fraction of the entire replay dataset.

Figure 10 shows that the performance of MA-CQL increases given more data points for $N \in \{1, 10, 20\}$. However, it does not further increase given an even larger amount of data, which performs much worse than the fully-trained online agents and fails to recover their performance. On the contrary, OMAR always outperforms MA-CQL by a large margin when $N > 1\%$, whose performance is much closer to the fully-trained online agents given more data points. Therefore, the optimality issue still persists when dataset size becomes larger (e.g., it can take a very long time to escape from them if the objective contains very flat regions (Ahmed et al., 2019)). In addition, the zeroth-order optimizer part in OMAR can better guide the actor given a larger amount of data points with a more accurate value function.

C.4. Applicability on Centralized Training with Decentralized Execution

C.4.1. Results based on MATD3

In this section, we demonstrate the versatility of the method and show that it can also be applied and beneficial to methods based on centralized critics under the CTDE paradigm. Specifically, all baseline methods are built upon the MATD3 algorithm (Ackermann et al., 2019) using centralized critics as detailed in Section 2.1. Note that performance comparison and discussion of a centralized value function and a decentralized one is in Appendix C.4.2. Table 9 summarizes the averaged normalized score of different algorithms in each kind of dataset. As shown, OMAR (centralized) also significantly outperforms MA-ICQ (centralized) and MA-CQL (centralized), and matches the performance of MA-TD3+BC (centralized) in the expert dataset while outperforming it in other datasets.

Table 9: The average normalized score of different methods based on MATD3 with centralized critics under the CTDE paradigm.

	Random	Medium-reply	Medium	Expert
MA-ICQ	5.2 ± 5.5	10.1 ± 4.6	27.4 ± 5.3	96.7 ± 4.1
MA-TD3+BC	7.9 ± 2.2	9.3 ± 9.1	29.4 ± 3.7	$\mathbf{108.1} \pm 3.3$
MA-CQL	12.8 ± 4.9	11.2 ± 6.6	26.3 ± 13.3	69.5 ± 15.7
OMAR	$\mathbf{21.6} \pm 4.6$	$\mathbf{19.1} \pm 9.2$	$\mathbf{33.7} \pm 14.5$	$\mathbf{105.9} \pm 3.6$

C.4.2. Discussion about the centralized and decentralized critics in offline multi-agent RL

We attribute the lower performance in Table 9 (based on a centralized value function) compared to Table 1 (based on a decentralized value function) due to the base algorithm, where Table 10 shows the performance comparison of offline independent TD3 and offline multi-agent TD3 in different types of dataset in cooperative navigation. As shown, utilizing centralized critics underperforms decentralized critics in the offline setting. There has also been recent research (Witt et al., 2020; Lyu et al., 2021) showing the benefits of decentralized value functions compared to a centralized one, which leads to more robust performance. We attribute the performance loss of CTDE in the offline setting due to a more complex and higher-dimensional value function conditioning on all agent's actions and the global state that is harder to learn well without exploration.

Table 10: Averaged normalized score of ITD3 and MATD3 in cooperative navigation.

	Random	Medium-replay	Medium	Expert
ITD3	$\mathbf{18.7} \pm 8.0$	$\mathbf{19.9} \pm 4.7$	$\mathbf{18.6} \pm 4.4$	$\mathbf{75.5} \pm 7.9$
MATD3	16.1 ± 5.6	12.7 ± 6.1	12.1 ± 14.2	1.6 ± 2.7