Believe What You See: Implicit Constraint Approach for Offline Multi-Agent Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Learning from datasets without interaction with environments (Offline Learning) is an essential step to apply Reinforcement Learning (RL) algorithms in real world scenarios. However, compared with the *single-agent* counterpart, offline *multi-agent* RL introduces more agents with the larger state and action space, which is more challenging but attracts little attention. We demonstrate current offline RL algorithms are ineffective in multi-agent systems due to the accumulated extrapolation error. In this paper, we propose a novel offline RL algorithm, named *Implicit Constraint Q-learning* (ICQ), which effectively alleviates the extrapolation error by only trusting the state-action pairs given in the dataset for value estimation. Moreover, we extend ICQ to multi-agent tasks by decomposing the joint-policy under the implicit constraint. Experimental results demonstrate that the extrapolation error is reduced to almost zero and insensitive to the number of agents. We further show that ICQ achieves the state-of-the-art performance in the challenging multi-agent offline tasks (StarCraft II).

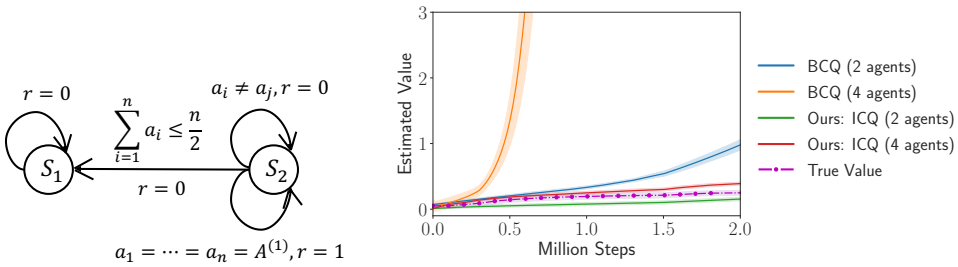
1 Introduction

Recently, reinforcement learning (RL), an active learning process, has achieved massive success in various domains ranging from strategy games [52] to recommendation systems [6]. However, applying RL to real world scenarios poses practical challenges: interaction with the real world, such as autonomous driving, is usually expensive or risky. To solve these issues, offline RL is an excellent choice to deal with practical problems [2, 23, 31, 37, 14, 25, 3, 22, 47, 11], aiming at learning from a fixed dataset without interaction with environments.

The greatest obstacle of offline RL is the distribution shift issue [15], which leads to extrapolation error, a phenomenon in which unseen state-action pairs are erroneously estimated. Unlike the online setting, the inaccurate estimated values of unseen pairs cannot be corrected by interacting with the environment. Therefore, most off-policy RL algorithms fail in the offline tasks due to intractable overestimation. Modern offline methods (e.g., Batch-Constrained deep Q-learning (BCQ) [15]) aim to enforce the learned policy to be close to the behavior policy or suppress the Q-value directly. These methods have achieved massive success in challenging single-agent offline tasks like D4RL [13].

However, many decision processes in real world scenarios belong to multi-agent systems, such as intelligent transportation systems [1], sensor networks [32], and power grids [5]. We demonstrate that *unseen state-action pairs will grow exponentially as the number of agents increases in multi-agent systems, accumulating the extrapolation error quickly*. Moreover, the current offline algorithms are unsuccessful in multi-agent tasks even though they adopt the modern value-decomposition structure [42]. As shown in Figure 1, our results indicate that BCQ, a state-of-the-art offline algorithm, has divergent Q-estimates in a simple multi-agent MDP environment (e.g., BCQ (4 agents)). The

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.



(a) Two-state MMDP. (b) Estimated value of the joint action-value function.

Figure 1: (a) An MMDP where Q-estimates of BCQ will diverge as the number of agents increases. (b) The learning curve of the joint action-value function while running several agents in the given MMDP. The true values are similar in this task with different agent numbers, calculated by averaging the Monte-Carlo estimation under different agents. The Q-estimates of BCQ (4 agents) diverge while our algorithm (ICQ) has accurate Q-estimates. Please refer to Appendix C.5 for the complete results.

extrapolation error for value estimation is accumulated quickly as the number of agents increases, significantly impairing the performance.

Based on these analyses, we propose the Implicit Constraint Q-learning (ICQ) algorithm, which effectively alleviates the extrapolation error as no unseen pairs are involved in estimating Q-value. Besides, ICQ adopts an implicit constraint optimization method, which converts the policy learning into a supervised regression problem and meanwhile guarantees the convergence of Q-function theoretically. By decomposing the joint-policy under the implicit constraint, we extend ICQ to the multi-agent tasks successfully. To the best of our knowledge, our work is the first study analyzing and addressing the extrapolation error in multi-agent reinforcement learning.

We evaluate our algorithm on the challenging multi-agent offline tasks based on StarCraft II [35], where a large number of agents cooperatively complete a task. Experimental results show that ICQ can reduce the extrapolation error to almost zero under any number of agents and learn from complex multi-agent datasets. Further, we evaluate the single-agent version of ICQ in D4RL, a standard single-agent offline benchmark. The results demonstrate the generality of ICQ for a wide range of task scenarios, from single-agent to multi-agent, from discrete to continuous control.

2 Background

Notation. The fully cooperative multi-agent tasks are usually modeled as the Dec-POMDP [28] consisting of the tuple $G = \langle S, A, P, r, \Omega, O, n, \gamma \rangle$. Let $s \in S$ denote the true state of the environment. At each time step $t \in \mathbb{Z}^+$, each agent $i \in N \equiv \{1, \ldots, n\}$ chooses an action $a^i \in A$, forming a joint action $\boldsymbol{a} \in \mathbf{A} \equiv A^n$. Let $P(s' \mid s, \boldsymbol{a}) : S \times \mathbf{A} \times S \to [0, 1]$ denote the state transition function. All agents share the same reward function $r(s, \boldsymbol{a}) : S \times \mathbf{A} \to \mathbb{R}$.

We consider a partially observable scenario in which each agent draws individual observations $o^i \in \Omega$ according to the observation function $O(s, a) : S \times \mathbf{A} \to \Omega$. Each agent has an action-observation history $\tau^i \in \mathbf{T} \equiv (\Omega \times \mathbf{A})^t$, on which it conditions a stochastic policy $\pi^i(a^i \mid \tau^i)$ parameterized by $\theta_i : \mathbf{T} \times \mathbf{A} \to [0, 1]$. The joint action-value function is defined as $Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \boldsymbol{a}) \stackrel{\text{def}}{=} \mathbb{E}_{s_{0:\infty}, \boldsymbol{a}_{0:\infty}} \left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, \boldsymbol{a}_0 = \boldsymbol{a}, \boldsymbol{\pi} \right]$, where $\boldsymbol{\pi}$ is the joint-policy with parameters $\theta = \langle \theta_1, \ldots, \theta_n \rangle$. Let $\mathcal{D}$ denote the replay buffer, which contains trajectories of the behavior policy $\boldsymbol{\mu}$.

We adopt the *centralized training and decentralized execution* (CTDE) paradigm [39]. During training, the algorithm has access to the true state s and every agent's action-observation history τ_i, as well as the freedom to share all information between agents. However, during execution, each agent has access only to its action-observation history.

Batch-constrained deep Q-learning (BCQ) is a state-of-the-art offline RL method, which aims to avoid selecting an unfamiliar action at the next state during a value update. Specifically, BCQ

optimizes π by introducing a perturbation model $\xi(\tau, a)$ as follows

$$\pi(\tau) = \underset{a^{[i]}+\xi(\tau,a^{[i]})}{\arg\max}\ Q^{\pi}(\tau, a^{[i]} + \xi(\tau, a^{[i]}); \phi, \psi), \tag{1}$$

where $a^{[i]} \sim G(\tau;\varphi)_{i=1}^{m}$, $G(\tau;\varphi)$ denotes the generative model (e.g., conditional variational auto-encoder [21, 38]) parameterized by φ, m denotes the size of the action set and $a^{[i]}$ denotes the selected action. The perturbation model $\xi(\tau, a^{[i]})$ is adopted to adjust action. The generative model $G(\tau;\varphi)$ is trained to match the state-action pairs sampled from $\mathcal{D}$ by minimizing the cross-entropy loss function [7]. Let $\{a_t^{[i]}\}_{i=0}^{m}$ denote the action set. As for the discrete version of BCQ, if $\frac{a^{[i]}\sim G(\tau;\psi)}{\max\{a^{[i]}\sim G(\tau;\varphi)\}_{i=1}^{m}} \leq \zeta$, $a^{[i]}$ is considered an unfamiliar action and $\xi(\tau, a^{[i]})$ will mask $a^{[i]}$ in the maximizing the Q-value operation. The action-value function is updated by minimizing the following loss

$$L(\phi,\psi) = \mathbb{E}_{\mu}\left[\left(r + \gamma \max_{a^{[i]}+\xi(\tau,a^{[i]})} Q^{\pi}(\tau', a^{[i]} + \xi(\tau, a^{[i]}); \phi', \psi') - Q^{\pi}(\tau, a; \phi, \psi)\right)^2\right]. \tag{2}$$

3 Analysis of Accumulated Extrapolation Error in Multi-Agent RL

In this section, we theoretically analyze that the extrapolation error is gradually accumulated as the number of agents increases, which lays the basis for Section 4. The extrapolation error in off-policy learning mainly attributes that the evaluation of Q^{π} uses out-of-distribution (OOD) actions [15]. To quantify OOD actions, we define (s, a) contained in the datasets as *seen* state-action pairs. Otherwise, we name them as *unseen* pairs. Our theoretical result indicates that the key of offline RL is to reduce the effect of unseen pairs when evaluating the seen ones.

3.1 Extrapolation Error Propagation in Offline RL

Let $Q_{\mathcal{B}}^{\pi}$ denote the value-function computed with the batch $\mathcal{B}$, and we define the tabular extrapolation error as $\epsilon_{\mathrm{MDP}}(s,a) \overset{\mathrm{def}}{=} Q^{\pi}(s,a) - Q_{\mathcal{B}}^{\pi}(s,a)$, where we do not consider history information τ for analytical clarity. We partition the extrapolation error as $\epsilon_{\mathrm{MDP}} = [\epsilon_{\mathrm{s}}, \epsilon_{\mathrm{u}}]^{\mathrm{T}}$ according to seen and unseen state-action pairs. As interaction with the environment is not allowed, we can only optimize ϵ_{s} while not eliminating ϵ_{u}. Then, let $P^{\pi}(s', a' \mid s, a) = P_M(s' \mid s, a)\pi(a' \mid s')$ denote the transition matrix, where the subscript M denotes the true MDP. We decompose the transition matrix as $P^{\pi}(s', a' \mid s, a) = \begin{bmatrix} P_{\mathrm{s,s}}^{\pi} & P_{\mathrm{s,u}}^{\pi} \\ P_{\mathrm{u,s}}^{\pi} & P_{\mathrm{u,u}}^{\pi} \end{bmatrix}$ according to state-action pairs' property (e.g., $P_{\mathrm{s,u}}^{\pi} = P_M(s'_{\mathrm{u}} \mid s_{\mathrm{s}}, a_{\mathrm{s}})\pi(a'_{\mathrm{u}} \mid s'_{\mathrm{u}})$ denotes the transition probability from seen to unseen pairs). We suppose unseen pairs will have inaccurate estimates in each iteration. For conceptual clarity, we formalize the inaccurate estimates as the fixed additional error $[0, \epsilon_{\mathrm{b}}]^{\mathrm{T}}$ at each iteration, which has the same dimension as ϵ_{MDP}. Based on the above definitions, we have the following conclusion:

Theorem 1 *Given a deterministic MDP, ϵ_{s} is related to ϵ_{u}, ϵ_{b} with a probability proportional to $P_{\mathrm{s,u}}^{\pi}$,*

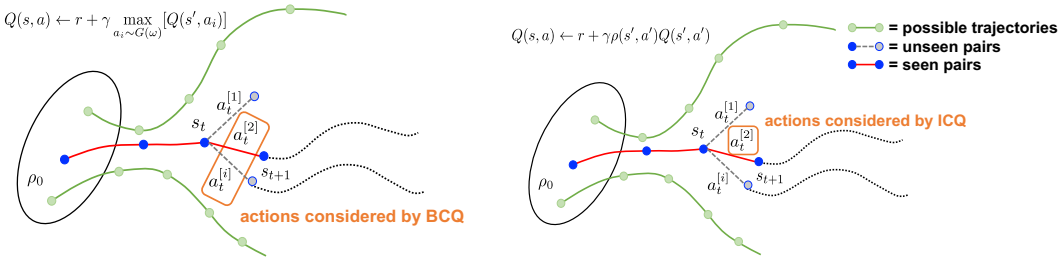
$$\epsilon_{\mathrm{s}}^{k} = \sum_{s',a'}\left[\gamma P_{\mathrm{s,s}}^{\pi}\epsilon_{\mathrm{s}}^{k+1} + \gamma^{K-k}P_{\mathrm{s,u}}^{\pi}\left(P_{\mathrm{u,u}}^{\pi}\right)^{K-k-1}\epsilon_{\mathrm{u}}^{K} + \sum_{i=1}^{K-k-1}\gamma^{i}P_{\mathrm{s,u}}^{\pi}\left(P_{\mathrm{u,u}}^{\pi}\right)^{i-1}\left(\gamma\epsilon_{\mathrm{s}}^{k+i+1} + \epsilon_{\mathrm{b}}\right)\right], \tag{3}$$

where k indexes the iteration and K denotes the total number of iterations.

Please refer to Appendix B.1 for the detailed proof.

The above theorem indicates ϵ_{u} and ϵ_{b} will influence ϵ_{s} with a probability proportional to $P_{\mathrm{s,u}}^{\pi}$ and $P_{\mathrm{u,u}}^{\pi}$. For this reason, we have to alleviate ϵ_{u} and ϵ_{b} indirectly. Researchers are interested in $P_{\mathrm{s,u}}^{\pi}$ as we can change it by optimizing $\pi(a'_{\mathrm{u}} \mid s'_{\mathrm{u}})$ while it is hard to quantify $P_{\mathrm{u,u}}^{\pi}$ as we do not know $P_M(s'_{\mathrm{u}} \mid s_{\mathrm{u}}, a_{\mathrm{u}})$. For example, BCQ aims to optimize $P_{\mathrm{s,u}}^{\pi} \to 0$ by enforcing $\pi(a'_{\mathrm{u}} \mid s'_{\mathrm{u}}) \to 0$.

Accumulated Extrapolation Error with Value Decomposition. In multi-agent tasks, MARL algorithms prefer to individually optimize π^i by decomposing $\boldsymbol{\pi}$ to solve high-dimensional joint-action space. For example, VDN [42] supposes the joint action-value function can be additively decomposed into value functions across agents: $Q^{\boldsymbol{\pi}}(s, \boldsymbol{a}) \approx \sum_i^n Q_i(\tau_i, a_i)$. We assume



(a) BCQ. (b) ICQ.

Figure 2: The comparison between ICQ and BCQ for the target Q-value estimation. The spots denote states, and the connections between spots indicate actions. The red solid-lines denote seen pairs, and the gray dotted-lines are unseen pairs. (a) BCQ estimates Q-value in a defined similar action set (orange) while unseen pairs still exist in the set with low probability. (b) ICQ only adopts seen pairs (orange) in the training set for Q-value estimation.

that the value-decomposition error is $\epsilon_d = Q^{\boldsymbol{\pi}}(s, \boldsymbol{a}) - \sum_i^n Q_i(\tau_i, a_i)$ by applying the value-decomposition assumption like VDN. Let define the individual extrapolation error of agent i as $\epsilon_{\mathrm{s}}^{k,i}(\tau_i, a_i) = Q_i^{\pi_k}(\tau_i, a_i) - Q_{\mathcal{B},i}^{\pi_k}(\tau_i, a_i)$, where $\epsilon_{\mathrm{s}}^{k,i}(\tau_i, a_i)$ is calculated based on the Theorem 1. Then, we have the following conclusion:

Theorem 2 *If the joint action-value function $Q^{\boldsymbol{\pi}}$ is decomposed as individual action-value functions, the total extrapolation error ϵ_{s} will be accumulated in the number of agents $\epsilon_{\mathrm{s}}^k \propto \sum_i^n \epsilon_{\mathrm{s}}^{k,i}$,*

$$\epsilon_{\mathrm{s}}^k = \sum_i^n \epsilon_{\mathrm{s}}^{k,i} + \sum_{s',\boldsymbol{a}'} \left(\gamma P_{\mathrm{s,s}}^{\boldsymbol{\pi}} + \gamma^{K-k} P_{\mathrm{s,u}}^{\boldsymbol{\pi}} \left(P_{\mathrm{u,u}}^{\boldsymbol{\pi}}\right)^{K-k-1} + \sum_{i=1}^{K-k-1} \gamma^i P_{\mathrm{s,u}}^{\boldsymbol{\pi}} \left(P_{\mathrm{u,u}}^{\boldsymbol{\pi}}\right)^{i-1} \right) \cdot \Delta_d, \tag{4}$$

where $\epsilon_{\mathrm{s}}^{k,i}$ denotes the individual extrapolation error of agent i at k-th iteration, $\Delta_d = \epsilon_d(s', \boldsymbol{a}') - \tilde{\epsilon}_d(s', \boldsymbol{a}')$ and $\tilde{\epsilon}_d(s, \boldsymbol{a}) = Q_{\mathcal{B}}^{\boldsymbol{\pi}}(s, \boldsymbol{a}) - \sum_i^n Q_{\mathcal{B},i}(\tau_i, a_i)$.

Please refer to Appendix B.2 for the detailed proof.

3.2 Toy Example

We design a toy two states Multi-Agent Markov Decision Process (MMDP) to illustrate the accumulated extrapolation error in multi-agent tasks (see Figure 1a). All agents start at state s_2 and explore rewards for 100 environment steps by taking actions $A^{(0)} = 0$ or $A^{(1)} = 1$. The optimal policy is all agents select $A^{(1)}$, and the reward is +1 in this case. Otherwise, the reward is 0. The MMDP task has sparse rewards. State s_2 will transfer to s_1 if the joint policy satisfies $\sum_{i=1}^n a_i \leq \frac{n}{2}$ at state s_2. State s_1 never returns to s_2. We run BCQ and our method ICQ on limited datasets while running several agents. The datasets only contain 32 trajectories generated by QMIX, and the number of unseen state-action pairs exponentially grows as the number of agents increases. We control the amount of valuable information ($r = 1$) in different datasets equal for fair comparisons. The multi-agent version of BCQ shares the same value-decomposition structure as ICQ (see Appendix D.2).

As shown in Figure 1b, the joint action-value function learned by BCQ gradually diverges as the number of agents increases while ICQ maintains a reasonable Q-value. The experimental result is consistent with Theorem 2, and we present the additional analysis for the toy example in Appendix B.3. In summary, we show theoretically and empirically that the extrapolation error is accumulated quickly as the number of agents increases and makes the Q-estimates of algorithms easier to diverge.

4 Implicit Constraint Approach for Offline Multi-Agent RL

In this section, we give an effective method to solve the accumulated extrapolation error in Multi-Agent RL based on the analysis of Section 3. From the algorithmic perspective, we point out that the critical challenge in offline RL is estimating target Q-value without sampled actions from π. We propose Implicit Constraint Q-learning (ICQ), which only trusts the seen state-action pairs in datasets for value function estimation. Further, we extend ICQ to multi-agent tasks.

4.1 The Implicit Constraint Q-learning (ICQ) Framework

Comparison between Offline and Off-policy RL. Based on the analysis of Section 3, we can effectively alleviate extrapolation error by setting $\pi(a'_{\mathrm{u}} \mid s'_{\mathrm{u}}) = 0$, which is the most significant difference between offline and off-policy RL. For expressive clarity, we first introduce the standard Bellman operator $\mathcal{T}^{\pi}$ as follows:

$$(\mathcal{T}^{\pi}Q)(\tau, a) \stackrel{\text{def}}{=} Q(\tau, a) + \mathbb{E}_{\tau' \sim \mu}[r + \gamma \mathbb{E}_{a' \sim \pi}[Q(\tau', a')] - Q(\tau, a)]. \tag{5}$$

Many off-policy evaluation methods, such as the Tree Backup [9] and Expected SARSA [36], are designed based on this operator. However, when coming into the offline setting, the standard Bellman operator suffers from the OOD issue as the current policy π is adopted for target Q-value estimation. A natural way to avoid the OOD issue is by adopting the importance sampling method [27]:

$$(\mathcal{T}^{\pi}Q)(\tau, a) = Q(\tau, a) + \mathbb{E}_{\tau' \sim \mu}[r + \gamma \mathbb{E}_{a' \sim \mu}[\rho(\tau', a')Q(\tau', a')] - Q(\tau, a)], \tag{6}$$

where $\rho(\tau', a') \stackrel{\text{def}}{=} \frac{\pi(a'|\tau')}{\mu(a'|\tau')}$ denotes the importance sampling weight. If we can calculate $\rho(\tau', a')$ *without sampled actions from* π, the unseen pairs will be avoided for target Q-value estimation and the target action a' will be sampled from μ. In this case, $\pi(a'_{\mathrm{u}} \mid s'_{\mathrm{u}}) = 0$ and $P^{\pi}_{\mathrm{s,u}} = 0$. Unfortunately, in most real world scenarios, it is extremely tough to obtain the exact behavior policy when collecting the training data, e.g., using expert demonstrations. To solve these issues, ICQ adopts the implicit constraint optimization method to construct the relationship between π and μ.

Implicit Constraint Optimization. In offline tasks, the policies similar to the behavior policy, i.e., $D_{\mathrm{KL}}(\pi \| \mu)[\tau] \leq \epsilon$ are preferred while maximizing the accumulated reward $Q^{\pi}(\tau, a)$, which can be described in the following optimization problem:

$$\pi_{k+1} = \arg\max_{\pi} \mathbb{E}_{a \sim \pi(\cdot|\tau)}[Q^{\pi_k}(\tau, a)] \qquad \text{s.t.} \quad D_{\mathrm{KL}}(\pi \| \mu)[\tau] \leq \epsilon. \tag{7}$$

First, note that the objective is a linear function of the decision variables π. All constraints are convex functions. Thus Equation 7 is a convex optimization problem, and we can solve the optimal policy π^* related to μ through the KKT condition [8], whose proof is in Appendix B.4:

$$\pi^*_{k+1}(a \mid \tau) = \frac{1}{Z(\tau)} \mu(a \mid \tau) \exp\left(\frac{Q^{\pi_k}(\tau, a)}{\alpha}\right), \tag{8}$$

where $\alpha > 0$ is the Lagrangian coefficient and $Z(\tau)$ is the normalizing partition function, which is calculated approximately by softmax operation over mini-batch samples. Therefore, we can calculate the ratio of π and μ by relocating μ to the left-hand side:

$$\rho(\tau, a) = \frac{\pi^*_{k+1}(a \mid \tau)}{\mu(a \mid \tau)} = \frac{1}{Z(\tau)} \exp\left(\frac{Q^{\pi_k}(\tau, a)}{\alpha}\right). \tag{9}$$

Note that the derived importance sampling weight ρ does not involve π. Therefore, compared with BCQ, which learns a similar action generator to filter unseen pairs in Q-value estimation, ICQ does not use any unseen pairs (see Figure 2).

Final Loss. Based on the derived importance sampling weight $\rho(\tau, a)$ in Equation 9, we can learn $Q^{\pi}(\tau, a; \phi)$ by minimizing

$$J_Q(\phi) = \mathbb{E}_{\tau' \sim \mu}\left[r + \gamma \mathbb{E}_{a' \sim \mu}\left[\frac{1}{Z(\tau')} \exp\left(\frac{Q(\tau', a'; \phi')}{\alpha}\right) Q(\tau', a'; \phi')\right] - Q(\tau, a; \phi)\right]^2. \tag{10}$$

Since we have derived the optimal policy π^*_{k+1} in Equation 8, we project π^*_{k+1} into the parameterized policy space by minimizing the loss function:

$$\begin{aligned} J_{\pi}(\theta) &= \mathbb{E}_{\varrho_{\mu}(\tau)}\left[D_{\mathrm{KL}}\left(\pi^*_{k+1} \| \pi_{\theta}\right)[\tau]\right] \stackrel{(a)}{=} \mathbb{E}_{\varrho_{\mu}(\tau)}\left[\sum_a \frac{\pi^*_{k+1}(a \mid \tau)}{\mu(a \mid \tau)} \mu(a \mid \tau)\left(-\log \pi_{\theta}(a \mid \tau)\right)\right] \\ &\stackrel{(b)}{=} \mathbb{E}_{\varrho_{\mu}(\tau), a \sim \mu}\left[-\frac{1}{Z(\tau)} \log(\pi(a \mid \tau; \theta)) \exp\left(\frac{Q^{\pi}(\tau, a)}{\alpha}\right)\right], \end{aligned} \tag{11}$$

where $\varrho_{\mu}(\tau)$ denotes data distribution under μ, (a) ignores $\mathbb{E}_{\varrho_{\mu}(\tau)}\left[\sum_a \pi^*_{k+1}(a \mid \tau) \log \pi^*_{k+1}(a \mid \tau)\right]$ that is not related to θ, and (b) applies the importance sampling weight derived in Equation 9 under forward KL constraint. See Appendix A for the complete workflow of the ICQ algorithm.

4.2 Extending ICQ to Multi-Agent Tasks

This section extends ICQ to multi-agent tasks and proposes a novel offline multi-agent reinforcement learning algorithm. For notational clarity, we name the Multi-Agent versions of ICQ as ICQ-MA. The complete algorithm is shown in Appendix A.

4.2.1 Multi-Agent Policy Evaluation with λ-return

Based on the analysis in Section 4.1, we can effectively alleviate ϵ_{u} and ϵ_{b} by optimizing $\pi(a'_{\mathrm{u}} \mid s'_{\mathrm{u}}) = 0$. The extrapolation error in this case is changed to $\epsilon_{\mathrm{s}}^{k} \propto \sum_{i}^{n} \epsilon_{\mathrm{s}}^{k,i} = \sum_{i}^{n} \sum_{o'_i, a'_i} \gamma P_{\mathrm{s,s}}^{\pi^i} \epsilon_{\mathrm{s}}^{k+1,i}$ according to the Theorem 1 and Theorem 2. However, it is not easy to optimize $\epsilon_{\mathrm{s}}^{k}$ directly because the bias in $\epsilon_{\mathrm{s}}^{k+1,i}$ is accumulated as the number of agents increases, which makes learning unstable [48]. Therefore, we adopt the n-step evaluation to eliminate the accumulated bias. The λ-return Bellman operator is

$$(\mathcal{T}_{\lambda}^{\boldsymbol{\pi}} Q)(\boldsymbol{\tau}, \boldsymbol{a}) \stackrel{\text{def}}{=} Q(\boldsymbol{\tau}, \boldsymbol{a}) + \mathbb{E}_{\boldsymbol{\mu}} \left[\sum_{t \geq 0} (\gamma\lambda)^t \left(r_t + \gamma \rho(\boldsymbol{\tau}_{t+1}, \boldsymbol{a}_{t+1}) Q(\boldsymbol{\tau}_{t+1}, \boldsymbol{a}_{t+1}) - Q(\boldsymbol{\tau}_t, \boldsymbol{a}_t) \right) \right], \tag{12}$$

where $\rho(\boldsymbol{\tau}_t, \boldsymbol{a}_t) = \frac{1}{Z(\boldsymbol{\tau}_t)} \exp(\frac{1}{\alpha} Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}_t, \boldsymbol{a}_t))$. The hyper-parameter $0 \leq \lambda \leq 1$ provides the balance between bias and variance. Let $Q^{\boldsymbol{\pi}}$ denote the true Q-function of the current policy. The following theorem indicates that if $\boldsymbol{\pi}$ is not far from $\boldsymbol{\mu}$, the sequence $Q_k = (\mathcal{T}_{\lambda}^{\boldsymbol{\pi}})^k Q$ converges to $Q^{\boldsymbol{\pi}}$.

Theorem 3 *Let $\epsilon = \max_{\boldsymbol{\tau}} D_{\mathrm{KL}}(\boldsymbol{\pi} \| \boldsymbol{\mu})[\boldsymbol{\tau}]$. Assume that the algorithm satisfies the minimum visit frequency, finite trajectories and bounded step-size assumptions. If $\lambda < \frac{1-\gamma}{\gamma\sqrt{2\epsilon}}$, repeating $\mathcal{T}_{\lambda}^{\boldsymbol{\pi}}$ infinitely from arbitrary Q_0 leads to $Q^{\boldsymbol{\pi}}$: $\lim_{k \to \infty} Q_k(\boldsymbol{\tau}, \boldsymbol{a}) = Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \boldsymbol{a})$.*

Please refer to Appendix B.5 for the detailed proof.

Since we train joint-policy $\boldsymbol{\pi}$ according to the multi-agent version of Equation 7, the implicit constraint $D_{\mathrm{KL}}(\boldsymbol{\pi} \| \boldsymbol{\mu})[\boldsymbol{\tau}] \leq \epsilon$ promotes the convergence of Q-function in Theorem 3.

4.2.2 Decomposed Multi-Agent Joint-Policy under Implicit Constraint

Under the CTDE framework, we have to train individual policies for decentralized execution. Besides, it is also challenging to compute $\mathbb{E}_{\boldsymbol{\mu}}[\rho(\boldsymbol{\tau}', \boldsymbol{a}') Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}', \boldsymbol{a}')]$ in multi-agent policy evaluation as its computational complexity is $O(|A|^n)$. To address the above issues, we first define the joint-policy as $\boldsymbol{\pi}(\boldsymbol{a} \mid \boldsymbol{\tau}) \stackrel{\text{def}}{=} \Pi_{i \in N} \pi^i(a^i \mid \tau^i)$, and then introduce a mild value-decomposition assumption:

$$Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \boldsymbol{a}) = \sum_{i} w^i(\boldsymbol{\tau}) Q^i(\tau^i, a^i) + b(\boldsymbol{\tau}), \tag{13}$$

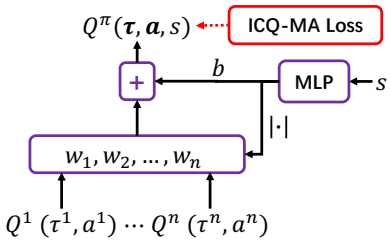


Figure 3: Mixer Network.

where $w^i(\boldsymbol{\tau}) \geq 0$ and $b(\boldsymbol{\tau})$ are generated by the Mixer Network whose inputs are global observation-action history (see Figure 3). Based on the above assumptions, we propose the decomposed multi-agent joint-policy under implicit constraint in the following theorem:

Theorem 4 *Assuming the joint action-value function is linearly decomposed, we can decompose the multi-agent joint-policy under implicit constraint as follows*

$$\boldsymbol{\pi} = \arg\max_{\pi^1, \ldots, \pi^n} \sum_{i} \mathbb{E}_{\tau^i, a^i \sim \mathcal{D}} \left[\frac{1}{Z^i(\tau^i)} \log(\pi^i(a^i \mid \tau^i)) \exp\left(\frac{w^i(\boldsymbol{\tau}) Q^i(\tau^i, a^i)}{\alpha} \right) \right], \tag{14}$$

where $Z^i(\tau^i)$ is the normalizing partition function, which is calculated approximately by softmax operation over mini-batch samples.

Please refer to Appendix B.6 for the detailed proof.

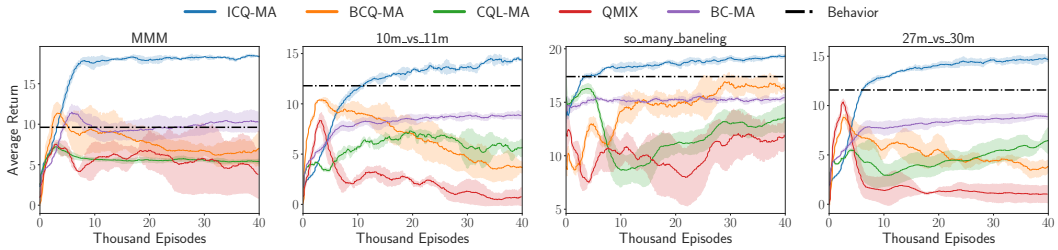


Figure 4: Performance comparison in StarCraft II environments.

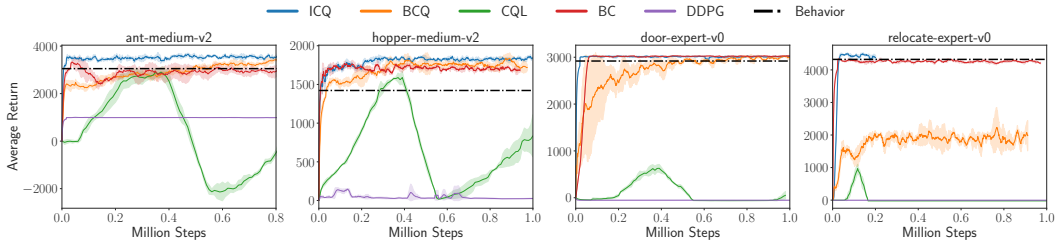


Figure 5: Performance comparison in D4RL.

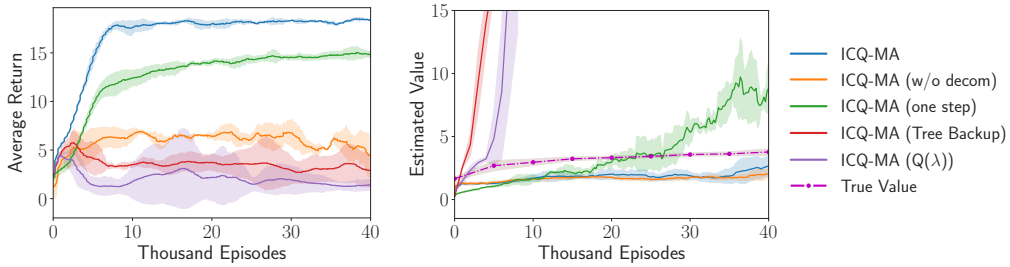
The decomposed multi-agent joint-policy has a concise form. We can train individual policies π^i by minimizing $J_{\boldsymbol{\pi}}(\theta) = \sum_i \mathbb{E}_{\boldsymbol{\mu}}\left[-\frac{1}{Z^i(\tau^i)}\log(\pi^i(a^i \mid \tau^i;\theta_i))\exp\left(\frac{w^i Q^i(\tau^i,a^i)}{\alpha}\right)\right]$. Besides, $w^i(\boldsymbol{\tau})$ achieves the trade-off between the roles of agents. If some agents have important roles, the value of corresponding $w^i(\boldsymbol{\tau})$ is relatively large. Also, if $w^i(\boldsymbol{\tau}) \to 0$, π^i is approximately considered as the behavior cloning policy. As for the policy evaluation, we train $Q(\boldsymbol{\tau}, \boldsymbol{a}; \phi, \psi)$ by minimizing $J_Q(\phi,\psi) = \mathbb{E}_{\boldsymbol{\mu}}\left[\sum_{t\geq 0}(\gamma\lambda)^t\left(r_t + \gamma\frac{\exp\left(\frac{1}{\alpha}Q(\boldsymbol{\tau}_{t+1},\boldsymbol{a}_{t+1};\phi',\psi')\right)}{Z(\boldsymbol{\tau}_{t+1};\phi',\psi')}Q(\boldsymbol{\tau}_{t+1},\boldsymbol{a}_{t+1};\phi',\psi') - Q(\boldsymbol{\tau}_t,\boldsymbol{a}_t;\phi,\psi)\right)\right]^2$, where $Q(\boldsymbol{\tau}_{t+1},\boldsymbol{a}_{t+1};\phi',\psi') = \sum_i w^i(\boldsymbol{\tau}_{t+1};\psi')Q^i(\tau^i_{t+1},a^i_{t+1};\phi'_i) - b(\boldsymbol{\tau}_{t+1};\psi')$.

5 Related Work

As ICQ-MA seems to be the first work addressing the accumulated extrapolation error issue in offline MARL, we briefly review the prior single-agent offline RL works here, which can be divided into three categories: dynamic programming, model-based, and safe policy improvement methods.

Dynamic Programming. Policy constraint methods in dynamic programming [20, 2, 51, 44, 17] are most closely related to our work. They attempt to enforce π to be close to μ under KL-divergence, Wasserstein distance [46], or MMD [41], and then only use actions sampled from π in dynamic programming. For example, BCQ [15] constrains the mismatch between the state-action visitation of the policy and the state-action pairs contained in the batch by using a state-conditioned generative model to produce only previously seen actions. AWR [31] and ABM [37] attempt to estimate the value function of the behavior policy via Monte-Carlo or TD(λ). Unlike these methods, our algorithm, ICQ, estimates the Q-function of the current policy using actions sampled from μ, enabling much more efficient learning. Another series of methods [45, 29, 30] aim to estimate uncertainty to determine the trustworthiness of a Q-value prediction. However, the high-fidelity requirements for uncertainty estimates limit the performance of algorithms.

Model-based and Safe Policy Improvement. Model-based methods [18, 43, 12, 49, 19] attempt to learn the model from offline data, with minimal modification to the algorithm. Nevertheless, modeling MDPs with very high-dimensional image observations and long horizons is a major open problem, which leads to limited algorithm performance [23]. Besides, safe policy improvement methods [22, 40, 4, 10] require a separately estimated model to μ to deal with unseen actions. However, accurately estimating μ is especially hard if the data come from multiple sources [26]. In contrast, ICQ does not require μ, and it can effectively deal with multi-source data.



(a) Performance comparison.

(b) Estimates of Q^{π} and true values.

Figure 6: Module ablation study on MMM map.

6 Experiments

In this section, we evaluate ICQ-MA and ICQ on multi-agent (StarCraft II) and single-agent (D4RL) offline benchmarks and compare them with state-of-the-art methods. Then, we conduct ablation studies on ICQ-MA. We aim to better understand each component's effect and further analyze the main driver for the performance improvement.

6.1 Multi-Agent Offline Tasks on StarCraft II

We first construct the multi-agent offline datasets based on ten maps in StarCraft II (see Table 3 in Appendix E). The datasets are made by collecting DOP [48] training data. All maps share the same reward function, and each map includes 12000 trajectories. We are interested in non-expert data or multi-source data. Therefore, we artificially divide behavior policies into three levels based on the average episode return (see Table 4 in Appendix E). Then, we evenly mix data of three levels.

We compare our method against QMIX [34], multi-agent version of BCQ (BCQ-MA), CQL (CQL-MA), and behavior cloning (BC-MA). To maintain consistency, BCQ-MA, CQL-MA, and BC-MA share the same linear value decomposition structure with ICQ-MA. Details for baseline implementations are in Appendix D.2. Each algorithm is run with five seeds, where the performance is evaluated ten times every 50 episodes. To better demonstrate the effectiveness of ICQ-MA, we employ the find-tuned hyper-parameters provided by authors of BCQ and CQL. Details for hyper-parameters are in Appendix D.1.

We investigate ICQ-MA's performance compared to state-of-the-art baselines in different maps, including two super hard maps, MMM2, and 27m_vs_30m. Results in Figure 4 show that ICQ-MA significantly outperforms all baselines and achieves state-of-the-art performance in all maps. QMIX, BCQ-MA, and CQL-MA have poor performances due to the accumulated extrapolation error. Interestingly, since BC does not depend on the policy evaluation, it is not subject to extrapolation error. Thus BC-MA has a sound performance as StarCraft II is deterministic.

To understand the reasons for ICQ-MA's better performance, we analyze its value estimation in Section 6.3 and Appendix C.2, which demonstrates that the extrapolation error of ICQ-MA is reduced to almost zero. Please refer to Appendix C.1 and Appendix C.2 for additional results on StarCraft II.

6.2 Single-Agent Offline Tasks on D4RL

To compare with standard offline methods, we also evaluate ICQ in the single offline tasks (e.g., D4RL), including gym domains, adroit tasks [33], kitchen tasks [16], and antmaze (see Figure 7). Specifically, adroit tasks require controlling a 24-DoF robotic hand to imitate human behavior. Kitchen tasks require controlling a 9-DoF robot to manipulate multiple objects sequentially with only a sparse 0-1 completion reward. Antmaze requires composing parts of sub-optimal trajectories to form more optimal policies for reaching goals on a MuJoco Ant robot. These tasks are extremely challenging. Figure 5 shows that ICQ significantly and constantly outperforms baselines in perfor-

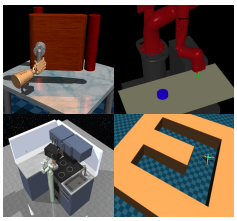


Figure 7: D4RL Task.

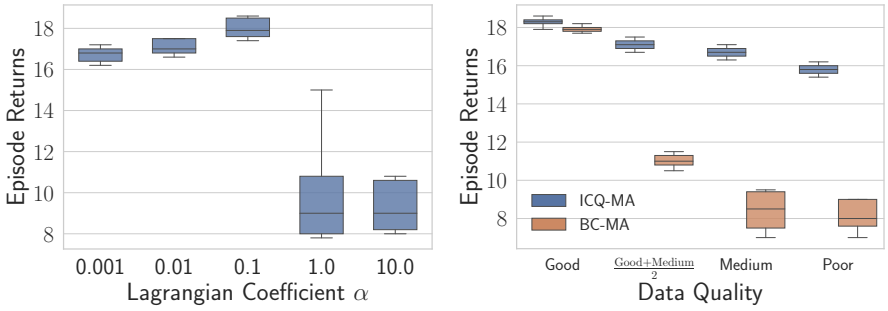


Figure 8: Ablation study on MMM map. Left: Parameter α. Right: Data quality.

mance and sample efficiency and achieves stable learning behavior. We reproduce the BCQ and CQL according to the official code[1]. Please refer to Appendix C.3 for additional results on D4RL.

6.3 Ablation Study

We conduct ablation studies of ICQ-MA in the MMM map of StarCraft II to study the effect of different modules, value estimation, important hyper-parameters, and data quality.

Module and Value Estimation Analysis. From Figure 6, we find that if we adopt other Q-value estimation methods in implicit constraint policies (e.g., $Q(\lambda)$ [24] or Tree Backup), the corresponding algorithms (ICQ-MA ($Q(\lambda)$) or ICQ-MA (Tree Backup)) have poor performances and incorrect estimated value. Suppose we train ICQ-MA without decomposed implicit constraint module (e.g., ICQ-MA ($\mathrm{w/o}$ decom)). In that case, the algorithm's performance is poor, although the estimated value is smaller than the true value, confirming the necessity of decomposed policy. Besides, the performance of one-step estimation (ICQ-MA (one step)) indicates n-step estimation is not the critical factor for improving ICQ-MA, while one-step estimation will introduce more bias.

The Parameter α. The Lagrangian coefficient α of implicit constraint operator directly affects the intensity of constraint, which is a critical parameter for the performance. A smaller α leads to a relaxing constraint and tends to maximize reward. If $\alpha \to 0$, ICQ-MA is simplified to Q-learning [50] while $\alpha \to \infty$ results in that ICQ-MA is equivalent to behavior cloning. Indeed, there is an intermediate value that performs best that can best provide the trade-off as in Figure 8.

Data Quality. It is also worth studying the performance of ICQ-MA and BC-MA with varying data quality. Specifically, we make the datasets from behavior policies of different levels (e.g., Good, Medium, and Poor). As shown in Figure 8, ICQ-MA is not sensitive to the data quality, while the performance of BC-MA drops drastically with the data quality deteriorates. Results confirm that ICQ-MA is robust to the data quality while BC-MA strongly relies on the data quality.

Computational Complexity. With the same training steps in SMAC, BCQ-MA consumes 70% time of ICQ-MA. Although ICQ-MA takes a little long time compared with BCQ-MA, it achieves excellent performance in benchmarks. The computing infrastructure for running experiments is a server with an AMD EPYC 7702 64-Core Processor CPU.

7 Conclusion

In this work, we demonstrate a critical problem in multi-agent off-policy reinforcement learning with finite data, where it introduces accumulated extrapolation error in the number of agents. We empirically show the current offline algorithms are ineffective in the multi-agent offline setting. Therefore, we propose the Implicit Constraint Q-learning (ICQ) method, which effectively alleviates extrapolation error by only trusting the state-action pairs in datasets. To the best of our knowledge, the multi-agent version of ICQ is the first multi-agent offline algorithm capable of learning from complex multi-agent datasets. We put the discussion of our work's limitation in Appendix F as the limited space. Due to the importance of offline tasks and multi-agent systems, we sincerely hope our algorithms can be a solid foothold for applying RL to practical applications.

[1]BCQ: https://github.com/sfujim/BCQ, CQL: https://github.com/aviralkumar2907/CQL.

References

[1] Jeffrey L Adler and Victor J Blue. A cooperative multi-agent transportation management and route guidance system. *Transportation Research Part C: Emerging Technologies*, 10(5-6):433–454, 2002.

[2] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pages 104–114. PMLR, 2020.

[3] Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. *arXiv preprint arXiv:2010.13611*, 2020.

[4] Felix Berkenkamp, Matteo Turchetta, Angela P Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. *arXiv preprint arXiv:1705.08551*, 2017.

[5] Duncan S Callaway and Ian A Hiskens. Achieving controllability of electric loads. *Proceedings of the IEEE*, 99(1):184–199, 2010.

[6] Yuanjiang Cao, Xiaocong Chen, Lina Yao, Xianzhi Wang, and Wei Emma Zhang. Adversarial attacks and detection on reinforcement learning-based interactive recommender systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1669–1672, 2020.

[7] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

[8] Axel Dreves, Francisco Facchinei, Christian Kanzow, and Simone Sagratella. On the solution of the kkt conditions of generalized nash equilibrium problems. *SIAM Journal on Optimization*, 21(3):1082–1108, 2011.

[9] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

[10] Benjamin Eysenbach, Shixiang Gu, Julian Ibarz, and Sergey Levine. Leave no trace: Learning to reset for safe and autonomous reinforcement learning. *arXiv preprint arXiv:1711.06782*, 2017.

[11] Rasool Fakoor, Pratik Chaudhari, and Alexander J Smola. P3o: Policy-on policy-off policy optimization. In *Uncertainty in Artificial Intelligence*, pages 1017–1027. PMLR, 2020.

[12] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, pages 1447–1456. PMLR, 2018.

[13] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[14] Scott Fujimoto, Edoardo Conti, Mohammad Ghavamzadeh, and Joelle Pineau. Benchmarking batch deep reinforcement learning algorithms. *arXiv preprint arXiv:1910.01708*, 2019.

[15] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019.

[16] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.

[17] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

[18] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 652–661. PMLR, 2016.

[19] Gregory Kahn, Adam Villaflor, Pieter Abbeel, and Sergey Levine. Composable action-conditioned predictors: Flexible off-policy learning for robot navigation. In *Conference on Robot Learning*, pages 806–816. PMLR, 2018.

[20] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[21] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[22] Romain Laroche, Paul Trichelair, and Remi Tachet Des Combes. Safe policy improvement with baseline bootstrapping. In *International Conference on Machine Learning*, pages 3652–3661. PMLR, 2019.

[23] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[24] Rémi Munos. Q (λ) with off-policy corrections. In *Algorithmic Learning Theory: International Conference, ALT 2016, Bari, Italy, October 19-21, 2016, Proceedings*, volume 9925, page 305. Springer, 2016.

[25] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *arXiv preprint arXiv:1906.04733*, 2019.

[26] Ashvin Nair, Murtaza Dalal, Abhishek Gupta, and Sergey Levine. Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[27] Radford M Neal. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

[28] Frans A Oliehoek, Christopher Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

[29] Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 8626–8638, 2018.

[30] Brendan O'Donoghue, Ian Osband, Remi Munos, and Volodymyr Mnih. The uncertainty bellman equation and exploration. In *International Conference on Machine Learning*, pages 3836–3845, 2018.

[31] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

[32] Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 20–27, 2004.

[33] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

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

[35] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 2186–2188, 2019.

[36] Juan C Santamaria, Richard S Sutton, and Ashwin Ram. Experiments with reinforcement learning in problems with continuous state and action spaces. *Adaptive behavior*, 6(2):163–217, 1997.

[37] Noah Y Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, and Martin Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396*, 2020.

[38] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.

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

[40] Aaron Sonabend-W, Junwei Lu, Leo A Celi, Tianxi Cai, and Peter Szolovits. Expert-supervised reinforcement learning for offline policy learning and evaluation. *arXiv preprint arXiv:2006.13189*, 2020.

[41] Bharath K Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, Gert RG Lanckriet, et al. On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics*, 6:1550–1599, 2012.

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

[43] Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR, 2016.

[44] Emanuel Todorov. Linearly-solvable markov decision problems. In *Advances in neural information processing systems*, pages 1369–1376, 2007.

[45] Ahmed Touati, Harsh Satija, Joshua Romoff, Joelle Pineau, and Pascal Vincent. Randomized value functions via multiplicative normalizing flows. In *Uncertainty in Artificial Intelligence*, pages 422–432. PMLR, 2020.

[46] SS Vallender. Calculation of the wasserstein distance between probability distributions on the line. *Theory of Probability & Its Applications*, 18(4):784–786, 1974.

[47] Quan Vuong, Yiming Zhang, and Keith W Ross. Supervised policy update for deep reinforcement learning. *arXiv preprint arXiv:1805.11706*, 2018.

[48] Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. Off-policy multi-agent decomposed policy gradients. *arXiv preprint arXiv:2007.12322*, 2020.

[49] Yu-Xiang Wang, Alekh Agarwal, and Miroslav Dudık. Optimal and adaptive off-policy evaluation in contextual bandits. In *International Conference on Machine Learning*, pages 3589–3597. PMLR, 2017.

[50] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[51] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[52] Deheng Ye, Zhao Liu, Mingfei Sun, Bei Shi, Peilin Zhao, Hao Wu, Hongsheng Yu, Shaojie Yang, Xipeng Wu, Qingwei Guo, et al. Mastering complex control in moba games with deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6672–6679, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We show contributions and scope clearly in the abstract and introduction.
   (b) Did you describe the limitations of your work? [Yes] Due to the space limits, we put it in the Appendix F.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work aims to drive Reinforcement Learning algorithms into practical applications, and it is not related to the environmental impact, security considerations, and other negative societal factors.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] Please refer to the description in front of each Theorem.
   (b) Did you include complete proofs of all theoretical results? [Yes] Please refer to Appendix B.1, Appendix B.2, Appendix B.3, Appendix B.5, Appendix B.6.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Please refer to the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyper-parameters, how they were chosen)? [Yes] Please refer to Section 6 and Appendix D.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We run each experiment with five random seeds.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please refer to the Computational Complexity in Section 6.3
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] Please refer to Section 6.2.
   (b) Did you mention the license of the assets? [Yes] Please refer to Section 6.2.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] We did not use crowdsourcing or conduct research with human subjects.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]