Joint Adaptive Neighborhood Constraint for Offline Multi-Agent Reinforcement Learning

Xiancheng Gao[1], **Mingxiao Feng**[*,2], **Lin Liu**[1], **Yuanrui Duan**[1]
Yufeng Shi[1], **Wengang Zhou**[*,1,2], **Houqiang Li**[1,2]

[1]University of Science and Technology of China, Hefei, China
[2]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China

Abstract

Offline reinforcement learning suffers from distributional shift and extrapolation errors. These issues are particularly severe in multi-agent settings. As the number of agents increases, the joint action space grows exponentially while agent behaviors become highly coupled. Consequently, even if individual actions remain within the data distribution, their combination may still result in out-of-distribution (OOD) joint actions. Directly extending single-agent constraints to multi-agent settings is difficult, as they fail to effectively constrain joint actions. This often results in poor coordination or excessive conservatism. To bridge this gap, we propose the joint adaptive neighborhood constraint (JANC), which explicitly constructs controlled neighborhoods in the joint action space to suppress extrapolation while preserving reliable generalization. Moreover, JANC adaptively scales neighborhood radii based on joint advantages to align generalization with value structures. In practice, our method first performs adaptive neighborhood Q learning to explore high-value behaviors and then conducts value and policy learning based on the refined actions. Experiments on offline benchmarks, including multi-agent Mujoco and StarCraft II, show that JANC outperforms state-of-the-art methods on the majority of tasks in complex cooperative scenarios.

1 Introduction

In recent years, online reinforcement learning (RL) has gained significant attention in multiple domains, including video games [1], nuclear fusion [2] and LLM alignment [3]. However, it still faces challenges such as high exploration risk and expensive data collection [4, 5]. Offline reinforcement learning focuses on learning policies solely from pre-collected datasets. This method avoids interactions with the environment, effectively reducing cost and safety risks [6, 7].

Nevertheless, offline RL still has a fundamental challenge: the distributional shift between the learned policy and the behavior policy. This problem may cause the agent to choose out-of-distribution (OOD) actions that are rare or do not appear in the dataset, resulting in value overestimation and extrapolation errors [8, 9]. This problem becomes more serious in multi-agent reinforcement learning (MARL) [10]. As the number of agents increases, the joint action space grows exponentially [11], and agent behaviors are highly coupled [12]. Even if individual agent actions are well-covered by the dataset, their combination may fall into low-density regions. This can lead to severe extrapolation errors in the joint action space and limit the overall policy performance. Therefore, in offline MARL, it is important to have constraint at the joint-action level to ensure the reliability and stability.

Directly extending single-agent constraint-based methods to multi-agent scenarios often leads to structural challenges. Some approaches extend single-agent policy constraint methods to multi-

*Corresponding authors: fengmx@iai.ustc.edu.cn, zhwg@ustc.edu.cn.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

agent scenarios [13]. However, in the Centralized Training with Decentralized Execution (CTDE) setting, constraints are commonly restricted to each agent's local policy as agents are assumed to act independently. Consequently, this approach fails to enforce coordination at the joint-action level [14]. Other methods extend value regularization methods to penalize OOD actions [15]. However, in high-dimensional joint action spaces, such penalties require considering a large number of joint action combinations, resulting in substantial computational complexity. Although recent work attempts to mitigate this issue, it does not impose explicit constraints on joint policy [16]. Furthermore, in-sample methods learn policies strictly using actions contained within the dataset. This approach limits the exploration of the joint action space, making it difficult to discover high-value behaviors that the behavior policy never explored [17, 18]. When joint action spaces expand and the dataset is sparse, these methods often fail to achieve an effective balance between conservatism and generalization.

To tackle these issues, this paper proposes an offline multi-agent joint adaptive neighborhood constraint (JANC) method. JANC constructs controlled neighborhoods around observed joint actions in the dataset, enabling explicit joint-action constraints without modeling the joint policy. We demonstrate that under certain conditions, controlled generalization within these neighborhoods can reliably enhance performance by allowing the agents to explore high-value joint actions. Furthermore, JANC adaptively adjusts the neighborhood radius using a joint advantage function: low-advantage joint actions are assigned larger neighborhoods to encourage exploration, while high-advantage joint actions use smaller neighborhoods to limit extrapolation errors. This enables fine-grained control of generalization at the joint-action level and effectively prevents coordination failure caused by excessive expansion of individual agent actions.

In terms of algorithmic implementation, we adopt the CTDE paradigm, which is widely used in multi-agent reinforcement learning. Based on this framework, our proposed JANC method operates in two main steps. First, we perform adaptive neighborhood Q learning to explore high-value collaborative behaviors within the neighborhood. Subsequently, we conduct value learning and policy learning based on the refined actions. This process improves the overall performance of decision-making while maintaining training stability. Experimental results show that JANC significantly outperforms state-of-the-art methods, on multi-agent offline benchmarks, including multi-agent Mujoco and SMACv2. These results demonstrate its effectiveness in enforcing joint action constraints and achieving agent coordination.

2 Related Works

2.1 Offline Reinforcement Learning

Offline reinforcement learning focuses on learning policies from a fixed dataset without online interaction. However, this approach suffers from severe distributional shift and extrapolation error when the learned policy deviates from the behavior policy [8, 9]. To address this issue, previous methods introduce different forms of conservatism into offline policy learning. A common approach is to constrain the learned policy to remain close to the behavior policy by aligning their probability densities. This is typically enforced through Kullback–Leibler (KL) divergence [19, 20] or behavior cloning [13, 21]. Other approaches impose conservatism by penalizing value functions with OOD actions, thereby discouraging the exploitation of unreliable value estimates [15, 22, 23]. Recently, in-sample methods have been proposed to mitigate the distributional shift by restricting learning to actions available in the dataset. This approach prevents value overestimation by avoiding queries for OOD actions [24, 25, 26]. Recently, neighborhood constraints have emerged to balance conservatism and generalization [27]. However, extending this paradigm to multi-agent systems remains a question due to the exponential explosion of joint action spaces and complex agent coupling.

2.2 Offline Multi-agent Reinforcement Learning

Multi-agent reinforcement learning is challenging due to the exponential growth of the joint action space and the strong coupling between agent behaviors [28]. Extending MARL methods to the offline setting exacerbates the distributional shift and extrapolation error. Several approaches have been proposed to solve these problems. For example, OMAR [14] incorporates zeroth-order optimization to local policy learning. CFCQL [16] extends CQL by applying conservative regularization in a counterfactual manner. However, these methods do not impose explicit constraints on the joint-action level. This may fail to provide an effective constraint on multi-agent joint behaviors. Some recent

approaches follow the in-sample learning paradigm and restrict updates strictly within the dataset. ICQ [29] uses importance sampling to implicitly constrain local policy updates on OOD samples. OMIGA [17] converts the global-level value regularization into equivalent implicit local regularizations for each agent. ComaDICE [18] introduces a regularizer in the space of stationary distributions to enforce consistency across the joint action distribution. Although effective in improving training stability, these methods can be overly conservative and may limit the generalization capability of neural networks in unseen but relevant regions of the joint action space.

In contrast, our work proposes JANC to construct a controlled neighborhood in the joint action space. It effectively suppresses extrapolation error while enhancing the generalization of multi-agent behaviors.

3 Preliminaries

3.1 Offline MARL

We focus on cooperative multi-agent tasks, which can be formalized as a multi-agent Partially Observable Markov Decision Process (POMDP) [30]. This problem is defined by the tuple $\mathcal{G} = \langle \mathcal{S}, \mathcal{A}, P, r, Z, \mathcal{O}, n, \gamma \rangle$. $s \in \mathcal{S}$ denotes the global state. Each agent $i \in \{1, \dots, n\}$ receives a local observation $o_i \in \mathcal{O}$ by function $Z(s, i) : \mathcal{S} \times \{1, \dots, n\} \to \mathcal{O}$. We denote $\mathbf{o} = (o_1, \dots, o_n) \in \mathcal{O}^n$. At each step, an agent executes an action $a \in \mathcal{A}$ to form a joint action $\mathbf{a} \in \mathcal{A}^n$. The environment then transitions to a next state s' according to the transition dynamics $P(s' \mid s, \mathbf{a}) : \mathcal{S} \times \mathcal{A}^n \times \mathcal{S} \to [0, 1]$. All agents share the same global reward $r(\mathbf{o}, \mathbf{a}) : \mathcal{O}^n \times \mathcal{A}^n \to \mathbb{R}$. The goal in cooperative MARL is to learn a set of policies $\pi_{\text{tot}} = \{\pi_1, \dots, \pi_n\}$ that maximizes the expected discounted return $\mathbb{E}_{\mathbf{a} \sim \pi_{\text{tot}}, \mathbf{o} \sim O^n} \big[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{o}_t, \mathbf{a}_t) \big]$. In the offline setting, we leverage a pre-collected dataset $\mathcal{D}$ to learn policies. Since no additional interaction with the environment is allowed, policy optimization must rely on the transitions contained in $\mathcal{D}$.

3.2 CTDE Framework and Value Decomposition

Centralized training with decentralized execution (CTDE) is a standard framework in MARL. Agents are allowed to access global information during training, while each agent i executes its policy based solely on its local observation o_i [31, 32]. However, learning a global value function remains challenging due to the growth of the joint action space and the difficulty of credit assignment. To address these issues, value decomposition methods are proposed to factorize the global value function (Q or V) into agent-wise components, effectively decomposing the joint action space [33, 11, 34].

Following recent offline MARL methodologies, the relationship between global functions $Q_{\text{tot}}(\mathbf{o}, \mathbf{a})$, $V_{\text{tot}}(\mathbf{o})$ and local utilities $Q_i(o_i, a_i)$, $V_i(o_i)$ can be expressed as:

$$\begin{cases} Q_{\text{tot}}(\mathbf{o}, \mathbf{a}) = \sum_{i=1}^{n} w_i(\mathbf{o}) Q_i(o_i, a_i) + b(\mathbf{o}) \\ V_{\text{tot}}(\mathbf{o}) = \sum_{i=1}^{n} w_i(\mathbf{o}) V_i(o_i) + b(\mathbf{o}) \end{cases} \text{s.t. } w_i(\mathbf{o}) \geq 0, \forall i. \tag{1}$$

where Q_i and V_i represent the local value functions of agent i, and the weights $w_i \geq 0$ and bias b are obtained from a network taking joint observation $\mathbf{o}$ as input.

4 Method

4.1 Neighborhood Constraint for Offline MARL

In offline MARL, policies are learned from a fixed dataset $\mathcal{D}$. A central challenge is how to learn policies that avoid OOD actions. Previous methods often impose constraints independently on each agent, either through policy regularization or value-based penalties. However, such independent constraints fail to correctly capture the feasible set of joint actions in coordinated decision-making.

We begin by formalizing this mismatch from a measure-theoretic perspective. We define joint support as $\mathcal{S}_{\text{joint}}(\mathbf{o}) := \text{supp}(\pi_\beta(\mathbf{a} \mid \mathbf{o}))$ and independent product support as $\mathcal{S}_{\text{indep}}(\mathbf{o}) := \prod_{i=1}^{n} \text{supp}(\pi_\beta(a_i \mid o_i))$. Specifically, the marginal support $\text{supp}(\pi_\beta(a_i \mid o_i))$ consists of individual agent actions a_i and $\mathcal{S}_{\text{indep}}$ represents the Cartesian product of these independent supports. Next, we introduce an assumption to discuss the relationship between the two.

Assumption 1 (Joint Support Non-Factorization)**.** *In coordinated multi-agent datasets, the joint action support $\mathcal{S}_{joint}(\mathbf{o})$ forms a strict subset of the Cartesian product formed by the individual marginal supports $\mathcal{S}_{indep}(\mathbf{o})$:*

$$\mathcal{S}_{joint}(\mathbf{o}) \subsetneq \mathcal{S}_{indep}(\mathbf{o}) := \prod_{i=1}^{n} \{a_i \mid \exists \mathbf{a}_{-i}, (\mathbf{o}, (a_i, \mathbf{a}_{-i})) \in \mathcal{D}\}. \tag{2}$$

Intuitively, for any action $a_i \in \text{supp}(\pi_\beta(a_i \mid o_i))$, it must be backed by at least one valid joint action $(a_i, \mathbf{a}_{-i}) \in \mathcal{S}_{\text{joint}}(\mathbf{o})$. Here $\mathbf{a}_{-i}$ denotes the joint action of all other agents. This relationship indicates that any joint action in $\mathcal{S}_{\text{joint}}(\mathbf{o})$ must be an element of the independent product support $\mathcal{S}_{\text{indep}}(\mathbf{o})$. Thus we observe the containment $\mathcal{S}_{\text{joint}}(\mathbf{o}) \subseteq \mathcal{S}_{\text{indep}}(\mathbf{o})$. Furthermore, the strong coupling of actions in cooperative tasks and the sparsity of offline samples often result in a discrepancy between individual and joint supports. There exist joint actions composed of marginally supported individual actions but have zero joint probability in the dataset: $\exists \tilde{\mathbf{a}} \in \mathcal{S}_{\text{indep}}(\mathbf{o})$ such that $\pi_\beta(\tilde{\mathbf{a}} \mid \mathbf{o}) = 0$. Since such $\tilde{\mathbf{a}}$ are excluded from $\mathcal{S}_{\text{joint}}(\mathbf{o})$ but contained in $\mathcal{S}_{\text{indep}}(\mathbf{o})$, the inclusion is strict.

Consider a Two-Key Door example to illustrate this reliability gap: two agents must simultaneously turn keys in the same direction (both left L or both right R) to open a door. The dataset contains only (L, L) and (R, R), such that $\mathcal{S}_{\text{joint}}(\mathbf{o}) = \{(L, L), (R, R)\}$. Under independent constraints, Both agents see that L and R are supported. Their independent product support becomes $\mathcal{S}_{\text{indep}}(\mathbf{o}) = \{(L, L), (R, R), (L, R), (R, L)\}$. Consequently, the two agents may mistakenly execute a joint action (L, R), which is OOD and fails the task.

In general, constraints must be imposed at the joint-action level to correctly construct empirical support and mitigate extrapolation errors. To this end, we propose the joint-action neighborhood constraint.

Definition 1 (Joint-action Neighborhood Constraint)**.** *Let $\mathcal{A}^n$ denote the joint action space. We define the constrained action set for a joint observation $\mathbf{o}$ as:*

$$\mathcal{C}_N(\mathbf{o}) := \{\tilde{\mathbf{a}} \in \mathcal{A}^n \mid \|\tilde{\mathbf{a}} - \mathbf{a}\| \leq \epsilon, (\mathbf{o}, \mathbf{a}) \in \mathcal{D}\}. \tag{3}$$

which consists of all joint actions located within the ϵ-neighborhoods of dataset actions corresponding to a given observation $\mathbf{o} \in \mathcal{D}$.

This approach leverages the joint actions in the dataset to explicitly construct a controlled neighborhood, allowing global constraints without the need for joint policy modeling.

4.2 Controlled Neighborhood Generalization

Building upon the joint-action neighborhood constraint, we now evaluate its potential to enhance performance. In offline multi-agent settings, in-sample methods achieve robust performance through their inherent conservatism [17, 18]. However, by restricting learning strictly to observed actions, they often fail to fully exploit the generalization capabilities of neural networks. We point out that controlled generalization can be performed based on joint-action neighborhood constraints. This can be viewed as endowing in-sample learning with reliable generalization. We now analyze the dynamic programming properties of two operators, $\mathcal{T}_{JNC}$ and $\mathcal{T}_{In}$, which are defined as follows.

Definition 2 (JNC Operator)**.** *The Joint-action Neighborhood Constraint (JNC) operator $\mathcal{T}_{JNC}$ is defined as:*

$$\mathcal{T}_{JNC} Q_{tot}(\mathbf{o}, \mathbf{a}) := r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}' \sim P(\cdot|\mathbf{o},\mathbf{a})} \left[\max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} Q_{tot}(\mathbf{o}', \tilde{\mathbf{a}}') \right], \tag{4}$$

where the constrained action set $\mathcal{C}_N(\mathbf{o}')$ is defined in Eq. (3)*.*

Definition 3 (In-sample Operator)**.** *The Joint-action In-sample operator $\mathcal{T}_{In}$ is defined as:*

$$\mathcal{T}_{In} Q_{tot}(\mathbf{o}, \mathbf{a}) := r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}' \sim P(\cdot|\mathbf{o},\mathbf{a})} \left[\max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} Q_{tot}(\mathbf{o}', \mathbf{a}') \right], \tag{5}$$

where $\mathcal{C}_{\mathcal{D}}(\mathbf{o}) := \{\mathbf{a} \in \mathcal{A}^n \mid (\mathbf{o}, \mathbf{a}) \in \mathcal{D}\}$ represents the set of joint actions available in the dataset for observation $\mathbf{o}$. By construction, $\mathcal{C}_{\mathcal{D}}(\mathbf{o}) \subseteq \mathcal{C}_N(\mathbf{o})$, which implies that the JNC operator explores a larger action space compared to the In-sample operator.

To demonstrate the effectiveness of the $\mathcal{T}_{JNC}$ operator, we first examine the Q value updates within the action neighborhood. Specifically, we analyze a single-step gradient update of the generic backup operator $\mathcal{T}_{u_{tot}}$ on a sample $(\mathbf{o}, \mathbf{a}) \in \mathcal{D}$ with a small learning rate η. We characterize its generalization effect on a nearby joint action $(\mathbf{o}, \tilde{\mathbf{a}}) \notin \mathcal{D}$ as follows:

Theorem 1 (Local Consistency)**.** *Under specific continuity conditions, for a sufficiently small learning rate η and a joint action $\tilde{\mathbf{a}}$ in the proximity of $\mathbf{a}$, the following relationship holds:*

$$Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta') = Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) + C_1 \Big(\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) - Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) + C_2 \|\tilde{\mathbf{a}} - \mathbf{a}\| \Big) + O(\|\theta' - \theta\|^2), \tag{6}$$

where $C_1 \in [0, 1]$ and C_2 denotes a finite constant.

The formal theoretical proof is provided in Appendix A.1. Theorem 1 suggests that as the perturbed joint action $\tilde{\mathbf{a}}$ remains within a restricted distance from the dataset sample $\mathbf{a}$, the impact of the perturbation term $C_2\|\tilde{\mathbf{a}} - \mathbf{a}\|$ remains bounded. Given that $C_1 \in [0, 1]$, the update rule effectively approximates the true Bellman operator. Consequently, the global Q function achieves reliable generalization specifically within this controlled neighborhood.

Building upon this local consistency, we next investigate the convergence properties and performance guarantees of the $\mathcal{T}_{JNC}$ operator.

Theorem 2 (Contraction)**.** *$\mathcal{T}_{JNC}$ is a γ-contraction operator in the generalized neighborhood region $\mathcal{C}_N$ according to the L_∞ norm. Therefore, by repeatedly applying $\mathcal{T}_{JNC}$, any initial function Q will converge to a unique fixed point Q^*_{JNC}.*

Finally, we demonstrate that by expanding the search space from the in-sample action set $\mathcal{C}_\mathcal{D}$ to the controlled neighborhood $\mathcal{C}_N$, JNC achieves an optimal value no lower than that of strict in-sample optimization.

Theorem 3 (Performance)**.** *For a controlled neighborhood $\mathcal{C}_N$ satisfying the local conditions of Theorem 1, the value functions induced by the optimal JNC policy π^*_{JNC} and the optimal In-sample policy π^*_{In} satisfy:*

$$V^*_{JNC}(\mathbf{o}) \geq V^*_{In}(\mathbf{o}), \quad \forall \mathbf{o} \in \mathcal{D}. \tag{7}$$

The neighborhood construction and local consistency analysis motivate controlled generalization around observed joint actions, while Theorem 3 establishes the optimal-value benefit relative to strict in-sample optimization. The proofs of Theorem 2 and Theorem 3 are provided in Appendix A.2.

4.3 Adaptive Neighborhood Constraint

To improve the flexibility of $\mathcal{T}_{JNC}$ under heterogeneous data quality, we extend the fixed neighborhood constraint in $\mathcal{C}_N$ to an adaptive formulation. We propose a joint adaptive neighborhood constraint that adjusts the neighborhood radius based on the joint advantage of each sample:

Definition 4 (Joint Adaptive Neighborhood Constraint)**.** *Within the Bellman objective, the selection of joint actions is confined to an adaptive neighborhood:*

$$\mathcal{C}_{AN}(\mathbf{o}) := \{\tilde{\mathbf{a}} \in \mathcal{A}^n \mid \|\tilde{\mathbf{a}} - \mathbf{a}\| \leq \epsilon \exp(-\alpha A_{tot}(\mathbf{o}, \mathbf{a})), (\mathbf{o}, \mathbf{a}) \in \mathcal{D}\} \tag{8}$$

where α represents the inverse temperature and $A_{tot}(\mathbf{o}, \mathbf{a})$ denotes the joint advantage function.

In practice, we impose bounds on the neighborhood radius to ensure that the resulting neighborhood remains within a controlled range, consistent with the proposed theorems. Within this bounded range, this adaptive mechanism balances exploration and conservatism by dynamically modulating the neighborhood radius. For joint actions with low advantage, the constraint relaxes the radius to facilitate a broader search, enhancing the ability to discover higher-quality targets from suboptimal data. In contrast, for high-advantage joint actions, the neighborhood is tightened to ensure the precision of policy updates and mitigate extrapolation errors. By adjusting the search space according to data quality, this approach effectively handles data heterogeneity and improves policy generalization.

4.4 Overall Algorithm

Building upon the in-sample learning framework, we propose a practical algorithmic realization of the JNC Operator $\mathcal{T}_{JNC}$.

4.4.1 Adaptive Neighborhood Q Learning

For each joint observation-action pair $(\mathbf{o}, \mathbf{a}) \in \mathcal{D}$, we seek a high-value joint action within $\mathcal{C}_{AN}(\mathbf{o})$ to approximate the neighborhood maximization in $\mathcal{T}_{JNC}$. Rather than explicitly searching the joint-action neighborhood for each Bellman backup, we learn a parameterized auxiliary policy $\boldsymbol{\delta}(\mathbf{o}, \mathbf{a})$ that directly predicts an action refinement, efficiently approximating the constrained optimization and reducing the computational cost of repeated neighborhood searches. The neighboring action is parameterized as $\tilde{\mathbf{a}} = \mathbf{a} + \boldsymbol{\delta}(\mathbf{o}, \mathbf{a})$, and the auxiliary policy is trained to increase the centralized joint value while limiting the perturbation magnitude:

$$\max_{\boldsymbol{\delta}} \quad Q_{\text{tot}}(\mathbf{o}, \mathbf{a} + \boldsymbol{\delta}(\mathbf{o}, \mathbf{a})) \quad \text{s.t.} \quad \|\boldsymbol{\delta}(\mathbf{o}, \mathbf{a})\| \exp(\alpha A_{\text{tot}}(\mathbf{o}, \mathbf{a})) \leq \epsilon, \quad \forall (\mathbf{o}, \mathbf{a}) \in \mathcal{D}, \tag{9}$$

where $A_{\text{tot}}(\mathbf{o}, \mathbf{a}) = Q_{\text{tot}}(\mathbf{o}, \mathbf{a}) - V_{\text{tot}}(\mathbf{o})$, and $\bar{Q}_{\text{tot}}$ denotes the target global Q value network.

The constraint limits the perturbation magnitude to $\epsilon \exp(-\alpha A_{\text{tot}}(\mathbf{o}, \mathbf{a}))$, permitting a broader local search around lower-advantage joint actions while refining high-advantage actions more conservatively. In practice, we optimize the auxiliary policy using a Lagrangian penalty with coefficient $\lambda > 0$, with the penalty weight computed from the target Q value:

$$\max_{\boldsymbol{\delta}} \mathbb{E}_{(\mathbf{o},\mathbf{a}) \sim \mathcal{D}} \big[Q_{\text{tot}}(\mathbf{o}, \mathbf{a} + \boldsymbol{\delta}(\mathbf{o}, \mathbf{a})) - \lambda \exp\big(\alpha(\bar{Q}_{\text{tot}}(\mathbf{o}, \mathbf{a}) - V_{\text{tot}}(\mathbf{o}))\big) \|\boldsymbol{\delta}(\mathbf{o}, \mathbf{a})\| \big]. \tag{10}$$

The auxiliary policy $\boldsymbol{\delta}$ is used only during training to refine dataset joint actions into locally improved neighboring actions for subsequent value and policy learning. Task execution relies on the final decentralized policies and does not require $\boldsymbol{\delta}$.

4.4.2 Multi-Agent Value Function Learning

In the multi-agent setting, the global Q function and V function follow the value decomposition architecture defined in Eq. (1), where Q_i and V_i represent the local value functions of agent i, and the mixing parameters $\boldsymbol{w}, b$ are generated by a hypernetwork taking joint observation $\mathbf{o}$ as input. The mixing network aggregates the local utilities into centralized joint values, allowing action refinement and value learning to be guided by the estimated value of the joint action rather than individual actions alone.

We use the V function to learn a log-sum-exp aggregation over Q values corresponding to the previously obtained high-value neighbor actions. This provides a smooth approximation of the max operator in $\mathcal{T}_{JNC}$. Meanwhile, following [17], we introduce a local regularization term into the learning of $V_i(o_i)$ to ensure training stability. The optimization objective is given as follows:

$$\min_{V_i} \mathbb{E}_{(o_i,a_i) \sim \mathcal{D}} \left[\exp\left(\frac{w_i(\mathbf{o})}{\beta} \left(\bar{Q}_i(o_i, a_i + \bar{\delta}_i) - V_i(o_i) \right) \right) + \frac{w_i(\mathbf{o})}{\beta} V_i(o_i) \right], \tag{11}$$

where $\bar{Q}_i$ is the target local Q function and $\bar{\delta}_i$ is the target local auxiliary policy, and β controls how closely V_i approximates the maximum Q value. Based on the learned V network, the Q learning objective for the global network is defined as:

$$\min_{\substack{Q_i, w_i, b \\ i=1,\ldots,n}} \mathbb{E}_{(\mathbf{o},\mathbf{a},\mathbf{o}') \sim \mathcal{D}} \left[\left(Q_{\text{tot}}(\mathbf{o}, \mathbf{a}) - (r(\mathbf{o}, \mathbf{a}) + \gamma V_{\text{tot}}(\mathbf{o}')) \right)^2 \right], \tag{12}$$

where $V_{\text{tot}}(\mathbf{o}')$ provides a smooth approximation of the neighborhood value at the next joint observation, allowing the value of refined actions to be propagated through Bellman updates.

4.4.3 Multi-Agent Policy Learning

Based on the learned value functions, we optimize the multi-agent policy by targeting the discovered high-value behaviors. Instead of directly mimicking the dataset actions, the individual policy networks $\pi_i(a_i|o_i)$ are trained to recover the high-value neighbor actions $\tilde{a}_i = a_i + \delta_i$ via a weighted behavior cloning objective [21, 35]:

$$\max_{\pi_i} \mathbb{E}_{(o_i,a_i) \sim \mathcal{D}} \left[\exp\left(\frac{w_i(\mathbf{o})}{\beta} \left(\bar{Q}_i(o_i, a_i + \delta_i) - V_i(o_i) \right) \right) \cdot \log \pi_i(a_i + \delta_i | o_i) \right]. \tag{13}$$

Algorithm 1 JANC

Input: Offline dataset $\mathcal{D}$, hyperparameters λ and α.
Initialize: mixing parameters $\boldsymbol{w}, b$; $\delta_i, \bar{\delta}_i, \pi_i, \bar{\pi}_i, Q_i, \bar{Q}_i, V_i$ for each agent $i = 1, 2, ..., n$.
for each gradient step **do**
    Update local state-value function V_i by minimizing Eq. (11);
    Update local action-value function Q_i and mixing parameters $\boldsymbol{w}, b$ by minimizing Eq. (12);
    Update local auxiliary policy δ_i by maximizing Eq. (10);
    Update local policy π_i by maximizing Eq. (13);
    Soft update $\bar{Q}_i$, $\bar{\pi}_i$, and $\bar{\delta}_i$ using Q_i, π_i, and δ_i for each agent i;
end for
Output: Final policies $\{\pi_i\}_{i=1}^n$.

Table 1: Average returns of JANC and baselines with the mean and standard deviation on MaMujoco tasks.

Task	Dataset	MABCQ	MACQL	ICQ	OMIGA	ComaDICE	JANC(ours)
HalfCheetah	expert	2992.7±629.7	1189.5±1034.5	2955.9±459.2	3383.6±552.7	4082.9±45.7	**4781.4±132.4**
	m-expert	3543.7±780.9	1194.2±1081.0	2834.0±420.3	2948.5±518.9	3889.7±81.6	**4667.1±278.4**
	medium	2590.5±1110.4	1011.3±1016.9	2549.3±96.3	3608.1±237.4	2664.7±54.2	**4833.3±181.7**
	m-replay	-333.6±152.1	1998.7±693.9	1922.4±612.9	2504.7±83.5	2855.0±242.2	**3941.7±93.6**
Hopper	expert	77.9±58.0	159.1±313.8	754.7±806.3	859.6±709.5	**2827.7±62.9**	2664.4±578.1
	m-expert	54.3±23.7	64.8±123.3	355.4±373.9	709.0±595.7	1362.4±522.9	**2392.8±450.4**
	medium	44.6±20.6	401.3±199.9	501.8±14.0	1189.3±544.3	822.6±66.2	**2243.8±267.6**
	m-replay	26.5±24.0	31.4±15.2	195.4±103.6	774.2±494.3	906.3±242.1	**1084.9±255.7**
Ant	expert	1317.7±286.3	1042.4±2021.6	2050.0±11.9	2055.5±1.6	2056.9±5.9	**2108.3±27.0**
	m-expert	1020.9±242.7	800.2±1621.5	1590.2±85.6	1720.3±110.6	1813.9±68.4	**2094.6±48.2**
	medium	1059.6±91.2	533.9±1766.4	1412.4±10.9	1418.4±5.4	1425.0±2.9	**1454.0±37.9**
	m-replay	950.8±48.8	234.6±1618.3	1016.7±53.5	1105.1±88.9	**1122.9±61.0**	1114.8±81.2

The key idea is that $\tilde{a}_i = a_i + \delta_i$ allows the policy to capture the enhanced cooperative strategies discovered during adaptive neighborhood Q learning. By focusing on these high-value actions within the trusted neighborhood, agents can generalize beyond the original data while staying consistent with the Bellman updates. The complete procedure is summarized in Algorithm 1. Training alternates between action refinement, value updates, and policy updates within the same loop, rather than separating them into distinct training phases.

5 Experiments

In this section, we evaluate the performance of our proposed JANC method. We aim to demonstrate its effectiveness in balancing conservatism and exploration across both continuous and discrete cooperative multi-agent tasks.

5.1 Experimental Setup

We conduct experiments on two widely recognized benchmarks in multi-agent reinforcement learning. All experiments are conducted on NVIDIA 2080 Ti and NVIDIA 3090 GPUs.

Multi-Agent Mujoco (MaMujoco) As a benchmark for continuous robotic control, MaMujoco [36] decomposes Mujoco environments into multi-agent joint control tasks where each agent controls specific joints. We utilize datasets collected by [17] using the HAPPO algorithm, covering the Hopper, Ant, and HalfCheetah tasks. Each task includes four data quality levels: expert, medium-expert, medium, and medium-replay.

StarCraft Multi-Agent Challenge v2 (SMACv2) For discrete multi-agent coordination, we evaluate our method on SMACv2 [37]. SMACv2 extends the original SMAC by introducing procedural generation to enhance diversity in unit types, initial positions, and team compositions. The datasets are collected by [38] using the MAPPO algorithm. It consists of three representative scenarios:

Table 2: Average win rate (%) of JANC and baselines with the mean and standard deviation on SMACv2 tasks.

Task	Dataset	BC	ICQ	OMAR	OMIGA	ComaDICE	JANC(ours)
Protoss_5_vs_5	expert	35.0±4.6	7.5±3.7	11.9±1.2	37.5±10.8	22.5±3.6	**44.4±5.1**
	medium	26.9±5.4	5.6±6.1	7.5±2.5	26.2±4.2	19.4±1.2	**36.9±5.1**
	m-expert	35.6±6.7	6.2±4.0	8.8±2.3	24.4±9.1	20.0±9.2	**38.8±3.2**
	random	3.8±3.1	4.4±4.2	3.8±1.3	5.0±2.5	4.4±4.7	**10.0±2.6**
Terran_5_vs_5	expert	26.9±4.2	20.6±6.4	15.0±1.1	32.5±5.3	33.1±3.2	**41.9±9.5**
	medium	21.2±3.1	10.0±7.2	9.4±5.7	23.8±1.5	25.0±2.8	**30.6±2.5**
	m-expert	28.7±6.1	16.9±5.4	10.0±6.5	31.9±7.0	29.4±4.7	**40.0±3.8**
	random	6.2±2.0	4.5±3.2	3.8±1.8	5.0±1.5	**13.8±3.2**	11.3±4.2
Zerg_5_vs_5	expert	15.6±6.2	7.5±2.5	12.5±6.8	22.5±9.4	18.1±1.2	**28.8±2.6**
	medium	13.1±5.4	5.9±2.5	10.6±2.7	18.8±6.6	17.5±1.5	**23.8±5.0**
	m-expert	15.6±5.9	6.2±6.6	11.3±5.3	19.4±5.4	18.8±5.2	**28.1±3.8**
	random	4.4±3.8	0.6±1.3	3.1±0.4	3.8±3.6	**6.9±1.3**	6.3±5.0

Protoss_5_vs_5, Terran_5_vs_5, and Zerg_5_vs_5. Similar to MaMujoco, the data is categorized into expert, medium-expert, medium, and random levels.

Dataset construction details and summary statistics, including average returns, are provided in Appendix B.1 and Table 3.

We report raw episodic returns rather than normalized scores on MaMujoco, and win rates expressed as percentages on SMACv2. For both benchmarks, each method is evaluated using five random seeds, with 32 evaluation episodes per seed.

5.2 Baselines

We compare JANC against several standard and state-of-the-art (SOTA) offline MARL algorithms. These include Behavioral Cloning (BC) and multi-agent extensions of popular single-agent methods, specifically MABCQ [23] and MACQL [15]. Building on MACQL, OMAR [14] introduces a zeroth-order optimization method via the Cross-Entropy Method (CEM) for actor rectification. We also consider methods rooted in the in-sample learning paradigm: ICQ [29] decomposes the joint policy under implicit constraints to mitigate extrapolation errors; OMIGA [17] converts global-level value regularization into equivalent implicit local value regularizations; and COMADICE [18] achieves coordinated behavior by combining global stationary distribution regularization with offline policy learning. In particular, ComaDICE is the SOTA algorithm in this field. These baselines represent a diverse set of approaches for addressing the fundamental challenges of offline multi-agent reinforcement learning.

We additionally implement Independent ANQ, a per-agent extension of the single-agent method ANQ [27], to compare independent and joint-action neighborhood constraints. It uses the same datasets, training budget, and backbone network architectures as JANC.

5.3 Comparative Results

We first evaluate the performance of different methods in the MaMujoco environment, with results summarized in Table 1. The results indicate that simply extending single-agent policy constraints or value penalization algorithms to multi-agent systems leads to suboptimal performance. This failure stems from the intricate coupling of joint actions and the significantly increased complexity of the multi-agent environment. In-sample learning methods offer enhanced stability through conservatism. However, this inherent property limits their ability to explore high-potential actions beyond the dataset. Our method, JANC, achieves consistent performance gains across a wide range of tasks. This method enables exploration of high-potential joint actions while maintaining proximity to the empirical data distribution.

We further examine joint-action coordination by comparing JANC with Independent ANQ. Joint-action support need not factorize into the Cartesian product of individual action supports: even when each agent's action is supported by the dataset, their combination may form an unseen joint behavior. Independent ANQ constrains marginal neighborhoods separately and may therefore admit

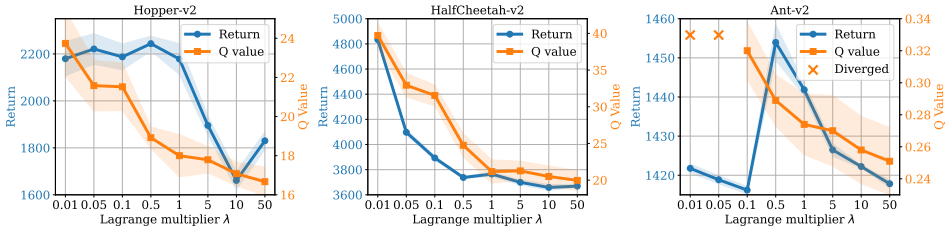


Figure 1: Returns and Q values of JANC with varying Lagrange multiplier λ. The results are averaged over five random seeds. The crosses × mean that the value functions diverge.

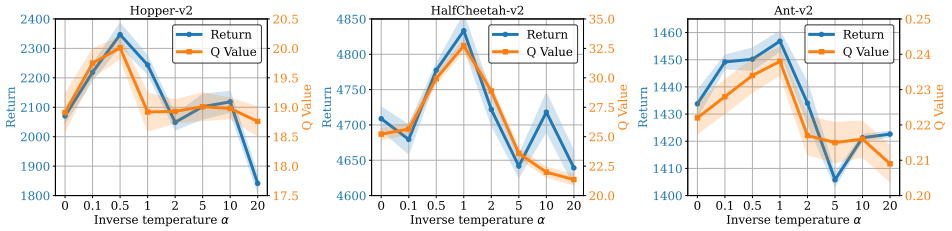


Figure 2: Returns and Q values of JANC with varying inverse temperature α. The results are averaged over five random seeds.

such coordination-level OOD actions. JANC instead constructs neighborhoods around observed joint-action tuples, with refinement guided by centralized joint values and advantages. JANC achieves higher mean returns across all 12 MaMujoco task–dataset settings (Table 4 in the appendix), supporting the benefit of joint-action rather than independent neighborhood constraints.

In the SMACv2 environment, we implement a discrete variant of our framework to handle discrete action spaces. Under local observations and decentralized execution, we use independent modeling as a feasible approximation. Specifically, we first learn a factorized joint behavior policy from the dataset: $\pi_{\text{BC}}(\mathbf{a}|\mathbf{o}) = \prod_{i=1}^{n} \pi_{\text{BC},i}(a_i|o_i)$, where $\pi_{\text{BC},i}(a_i|o_i)$ denotes the behavior policy of agent i. Based on this factorized policy, the joint neighborhood constraint is formulated as:

$$\mathcal{C}_N(\mathbf{o}) := \left\{ \prod_{i=1}^{n} \pi_{\delta,i} \;\middle|\; \sum_{i=1}^{n} D_{\text{KL}}\left(\pi_{\delta,i} \,\|\, \pi_{\text{BC},i}\right) \leq \epsilon \right\}, \tag{14}$$

where $\pi_{\delta,i}$ represents the perturbed action distribution for agent i. Although the behavior policy is factorized for tractable parameterization, the neighborhood constraint is imposed jointly across all agents through the aggregated KL divergence budget ϵ. Therefore, the resulting neighborhood does not degenerate into independent marginal constraints, and still preserves coordination at the joint-policy level. The complete algorithmic implementation is provided in Appendix B.2. As shown in Table 2, JANC consistently achieves apparent improvements in win rate, confirming its efficacy in discrete environments.

The reduced gains on medium-replay or random datasets and their potential connection to data diversity are discussed in Section 6.

5.4 Ablation Study

We examine the sensitivity of JANC to its core hyperparameters: the Lagrange multiplier λ and the inverse temperature α. For the results in Table 1, we set $\lambda = 0.5$ for Hopper and Ant, $\lambda = 0.01$ for HalfCheetah. For the experiments in Table 2, we fix $\lambda = 0.5$. α is configured as 1.0 across all tasks.

Lagrange multiplier λ We first investigate the influence of λ on medium-level datasets across several MaMujoco tasks (Figure 1). The results demonstrate that λ is a critical parameter for balancing exploration with conservatism. Small λ weakens the neighborhood constraint, enabling aggressive value propagation and overly optimistic Q estimates. In extreme cases, this leads to catastrophic divergence, as observed in the Ant environment. In contrast, as λ increases, the constraint becomes

progressively tighter. This effectively suppresses extrapolation errors and induces more conservative Q value estimates. The boundary cases further clarify this mechanism: $\lambda = 0$ reduces to unconstrained Q learning, while $\lambda \to \infty$ corresponds to strict in-sample learning. Notably, intermediate values of λ yield the best performance, indicating effective control of extrapolation and generalization.

Inverse temperature α We further analyze the effect of α, which controls how joint-action advantage modulates the neighborhood radius (Figure 2). When $\alpha = 0$, the exponential weighting term degenerates to a constant, resulting in a fixed neighborhood radius. As α increases, the neighborhood radius becomes advantage-dependent. Large positive advantages tighten the neighborhood, making the updates more conservative. Conversely, actions with low or negative advantages relax the constraint and allow for larger exploratory updates. Overly large α makes the radius highly sensitive to advantage differences, leading to inconsistent updates and instability. In contrast, smaller values of α reduce the influence of advantage information. This results in smoother neighborhood adaptation and achieves a more robust trade-off between exploration and stability.

6 Limitations

The effectiveness of JANC depends on whether the offline dataset provides meaningful local structure around observed joint actions. When such structure is insufficient, neighborhood refinement may provide limited benefits. Our framework achieves larger performance improvements on datasets with relatively unified behavior policies, such as the expert, medium-expert, and medium levels, while the gains are reduced on medium-replay or random datasets. A possible explanation is that highly diverse or noisy data collected from multiple behavior policies make it more challenging to define a stable, Bellman-consistent neighborhood.

The adaptive neighborhood mechanism also relies on manually selected hyperparameters α and λ to control its adaptation and constraint strength. Automatically selecting these hyperparameters remains a direction for future work.

7 Conclusion

In this paper, we identify the limitations of existing offline MARL methods in imposing joint-action constraints. To tackle this issue, we propose JANC, which forms controlled neighborhoods around joint actions observed in the dataset. Moreover, we show that under certain conditions, generalization within the controlled neighborhood can be trusted to improve performance. In particular, we propose an adaptive constraint method based on joint advantages to adjust the constraint neighborhood radii. Empirical results on multi-agent Mujoco and SMACv2 benchmarks show that JANC achieves better performance than most state-of-the-art baselines on the majority of tasks. Our method provides a promising direction for safe and effective offline MARL.

8 Acknowledgements

This work was supported by the National Key RD Program of China under Contract 2022ZD0119802 and the Youth Innovation Promotion Association CAS. It was also supported by the GPU cluster built by MCC Lab of Information Science and Technology Institution and the Supercomputing Center of USTC.The authors declare no competing interests.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[2] Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

[3] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

[4] Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

[5] Wen-Chi Yang, Giuseppe Marra, Gavin Rens, and Luc De Raedt. Safe reinforcement learning via probabilistic logic shields. In *International Joint Conference on Artificial Intelligence*, pages 5739–5749, 2023.

[6] Wonchul Shin and Yusung Kim. Guide to control: Offline hierarchical reinforcement learning using subgoal generation for long-horizon and sparse-reward tasks. In *International Joint Conference on Artificial Intelligence*, pages 4217–4225, 2023.

[7] Wenhui Liu, Zhijian Wu, Jingchao Wang, Dingjiang Huang, and Shuigeng Zhou. Imagination-limited q-learning for offline reinforcement learning. In *International Joint Conference on Artificial Intelligence*, pages 5770–5778, 2025.

[8] 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.

[9] Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8):10237–10257, 2023.

[10] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pages 321–384, 2021.

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

[12] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, 2017.

[13] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 20132–20145, 2021.

[14] Ling Pan, Longbo Huang, Tengyu Ma, and Huazhe Xu. Plan better amid conservatism: Offline multi-agent reinforcement learning with actor rectification. In *International conference on machine learning*, pages 17221–17237. PMLR, 2022.

[15] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020.

[16] Jianzhun Shao, Yun Qu, Chen Chen, Hongchang Zhang, and Xiangyang Ji. Counterfactual conservative q-learning for offline multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[17] Xiangsen Wang, Haoran Xu, Yinan Zheng, and Xianyuan Zhan. Offline multi-agent reinforcement learning with implicit global-to-local value regularization. In *Advances in Neural Information Processing Systems*, 2023.

[18] The Viet Bui, Thanh Hong Nguyen, and Tien Anh Mai. ComaDICE: Offline cooperative multi-agent reinforcement learning with stationary distribution shift regularization. In *International Conference on Learning Representations*, 2025.

[19] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11761–11771, 2019.

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

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

[22] Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pages 5774–5783, 2021.

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

[24] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

[25] Divyansh Garg, Joey Hejna, Matthieu Geist, and Stefano Ermon. Extreme Q-learning: MaxEnt RL without entropy. In *International Conference on Learning Representations*, 2023.

[26] Yixiu Mao, Hongchang Zhang, Chen Chen, Yi Xu, and Xiangyang Ji. Supported value regularization for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[27] Yixiu Mao, Yun Qu, Cheems Wang, and Xiangyang Ji. Adaptive neighborhood-constrained Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025.

[28] Yiqun Chen, Hangyu Mao, Jiaxin Mao, Shiguang Wu, Tianle Zhang, Bin Zhang, Wei Yang, and Hongxing Chang. PTDE: Personalized training with distilled execution for multi-agent reinforcement learning. In *International Joint Conference on Artificial Intelligence*, pages 31–39, 2024.

[29] Yiqin Yang, Xiaoteng Ma, Chenghao Li, Zewu Zheng, Qiyuan Zhang, Gao Huang, Jun Yang, and Qianchuan Zhao. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, 2021.

[30] Craig Boutilier. Planning, learning and coordination in multiagent decision processes. In *TARK*, volume 96, pages 195–210, 1996.

[31] Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. Optimal and approximate q-value functions for decentralized pomdps. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

[32] Landon Kraemer and Bikramjit Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 2016.

[33] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinícius Flores Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, and Thore Graepel. 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.

[34] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. QPLEX: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations*, 2021.

[35] 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.

[36] Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. FACMAC: Factored multi-agent centralised policy gradients. In *Advances in Neural Information Processing Systems*, pages 12208–12221, 2021.

[37] Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob N Foerster, and Shimon Whiteson. SMACv2: An improved benchmark for cooperative multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[38] Yuanrui Duan, Wengang Zhou, Yufeng Shi, Xiancheng Gao, Lin Liu, and Houqiang Li. Offline multi-agent reinforcement learning with global moderate generalization. In *International Conference on Autonomous Agents and Multiagent Systems*, 2025.

[39] Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J Cree. Regularisation of neural networks by enforcing lipschitz continuity. *Machine Learning*, 110(2):393–416, 2021.

[40] Francois Dufour and Tomas Prieto-Rumeau. Finite linear programming approximations of constrained discounted markov decision processes. *SIAM Journal on Control and Optimization*, 51(2):1298–1324, 2013.

[41] Huaqing Xiong, Tengyu Xu, Lin Zhao, Yingbin Liang, and Wei Zhang. Deterministic policy gradient: Convergence analysis. In *Uncertainty in Artificial Intelligence*, pages 2159–2169. PMLR, 2022.

[42] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

A Proofs

A.1 Proof of Theorem 1

To establish the theoretical foundation for Theorem 1, we first introduce several standard continuity assumptions commonly used in reinforcement learning.

Assumption 2. *The learned value function $Q_{tot}(\mathbf{o}, \mathbf{a}; \theta)$ is K_Q-Lipschitz and upper bounded by $Q_{\max}$, such that $\forall \mathbf{o} \in \mathcal{O}^n, \forall \mathbf{a}_1, \mathbf{a}_2 \in \mathcal{A}^n$: $|Q_{tot}(\mathbf{o}, \mathbf{a}_1; \theta) - Q_{tot}(\mathbf{o}, \mathbf{a}_2; \theta)| \leq K_Q \|\mathbf{a}_1 - \mathbf{a}_2\|$.*

Assumption 3. *$Q_{tot}(\mathbf{o}, \mathbf{a}; \theta)$ has a K_g-Lipschitz continuous gradient. Specifically, for $\forall \mathbf{o} \in \mathcal{O}^n, \forall \mathbf{a}_1, \mathbf{a}_2 \in \mathcal{A}^n$, the following holds: $\|\nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a}_1; \theta) - \nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a}_2; \theta)\| \leq K_g \|\mathbf{a}_1 - \mathbf{a}_2\|$.*

Assumption 4. *The value function and its gradient are bounded, i.e., $|Q_{tot}(\mathbf{o}, \mathbf{a})| \leq Q_{\max}$ and $\|\nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a})\| \leq g_{\max}$ for $\forall \mathbf{o}, \mathbf{a}$.*

Assumption 5. *The transition dynamics $P(\mathbf{o}'|\mathbf{o}, \mathbf{a})$ is K_P-Lipschitz, i.e., $\forall \mathbf{o}, \mathbf{o}' \in \mathcal{O}^n, \forall \mathbf{a}_1, \mathbf{a}_2 \in \mathcal{A}^n$: $\|P(\mathbf{o}'|\mathbf{o}, \mathbf{a}_1) - P(\mathbf{o}'|\mathbf{o}, \mathbf{a}_2)\| \leq K_P \|\mathbf{a}_1 - \mathbf{a}_2\|$.*

Assumption 6. *The reward function $r(\mathbf{o}, \mathbf{a})$ is K_r-Lipschitz, i.e., $\forall \mathbf{o} \in \mathcal{O}^n, \forall \mathbf{a}_1, \mathbf{a}_2 \in \mathcal{A}^n$: $|r(\mathbf{o}, \mathbf{a}_1) - r(\mathbf{o}, \mathbf{a}_2)| \leq K_r \|\mathbf{a}_1 - \mathbf{a}_2\|$.*

The value functions are commonly modeled using neural networks or linear parameters. Assumptions 2 and 3 can be relatively easily satisfied [39]. Assumptions 4, 5, and 6 are also common in the theoretical studies of RL [40, 41] and optimization [42]. These continuity assumptions serve as the foundation for the subsequent analysis of value network generalization. As a preliminary step to proving Theorem 4, we first establish the following lemma.

Lemma 1. *$\forall \mathbf{o} \in \mathcal{D}, \forall \mathbf{a}_1, \mathbf{a}_2 \in \mathcal{A}^n$, the generic backup operator $\mathcal{T}_{u_{tot}}$ satisfies the Lipschitz condition: $|\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}_1) - \mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}_2)| \leq K_{\mathcal{T}} \|\mathbf{a}_1 - \mathbf{a}_2\|$, where $K_{\mathcal{T}} := K_r + \gamma K_P |\mathcal{S}| Q_{\max}$ is a positive bounded constant.*

Proof. $\forall \mathbf{o} \in \mathcal{D}, \forall \mathbf{a}_1, \mathbf{a}_2 \in \mathcal{A}^n$, the difference of the backup operator is bounded as follows:

$$
\begin{aligned}
&|\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}_1) - \mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}_2)| \\
&= \left| r(\mathbf{o}, \mathbf{a}_1) - r(\mathbf{o}, \mathbf{a}_2) + \gamma \mathbb{E}_{\mathbf{o}' \sim P(\cdot|\mathbf{o}, \mathbf{a}_1)} \left[\max_{\mathbf{a}' \sim \mathbf{u}_{tot}(\cdot|\mathbf{o}')} Q_{tot}(\mathbf{o}', \mathbf{a}') \right] \right. \\
&\quad \left. - \gamma \mathbb{E}_{\mathbf{o}' \sim P(\cdot|\mathbf{o}, \mathbf{a}_2)} \left[\max_{\mathbf{a}' \sim \mathbf{u}_{tot}(\cdot|\mathbf{o}')} Q_{tot}(\mathbf{o}', \mathbf{a}') \right] \right| \\
&\leq |r(\mathbf{o}, \mathbf{a}_1) - r(\mathbf{o}, \mathbf{a}_2)| + \gamma \left| \mathbb{E}_{\mathbf{o}' \sim P(\cdot|\mathbf{o}, \mathbf{a}_1)} \left[\max_{\mathbf{a}' \sim \mathbf{u}_{tot}(\cdot|\mathbf{o}')} Q_{tot}(\mathbf{o}', \mathbf{a}') \right] \right. \\
&\quad \left. - \mathbb{E}_{\mathbf{o}' \sim P(\cdot|\mathbf{o}, \mathbf{a}_2)} \left[\max_{\mathbf{a}' \sim \mathbf{u}_{tot}(\cdot|\mathbf{o}')} Q_{tot}(\mathbf{o}', \mathbf{a}') \right] \right| \\
&= |r(\mathbf{o}, \mathbf{a}_1) - r(\mathbf{o}, \mathbf{a}_2)| + \gamma \left| \sum_{\mathbf{o}'} (P(\mathbf{o}'|\mathbf{o}, \mathbf{a}_1) - P(\mathbf{o}'|\mathbf{o}, \mathbf{a}_2)) \max_{\mathbf{a}' \sim \mathbf{u}_{tot}(\cdot|\mathbf{o}')} Q_{tot}(\mathbf{o}', \mathbf{a}') \right| \\
&\leq |r(\mathbf{o}, \mathbf{a}_1) - r(\mathbf{o}, \mathbf{a}_2)| + \gamma \sum_{\mathbf{o}'} |P(\mathbf{o}'|\mathbf{o}, \mathbf{a}_1) - P(\mathbf{o}'|\mathbf{o}, \mathbf{a}_2)| \cdot \left| \max_{\mathbf{a}' \sim \mathbf{u}_{tot}(\cdot|\mathbf{o}')} Q_{tot}(\mathbf{o}', \mathbf{a}') \right| \\
&\leq K_r \|\mathbf{a}_1 - \mathbf{a}_2\| + \gamma \sum_{\mathbf{o}'} K_P \|\mathbf{a}_1 - \mathbf{a}_2\| Q_{\max} \\
&\leq (K_r + \gamma K_P |\mathcal{S}| Q_{\max}) \|\mathbf{a}_1 - \mathbf{a}_2\|.
\end{aligned}
$$

By letting $K_{\mathcal{T}} := K_r + \gamma K_P |\mathcal{S}| Q_{\max}$, we obtain the desired Lipschitz constant. □

We consider the setting in Theorem 4, where the parameters θ are updated to θ' via a single gradient step on $(\mathbf{o}, \mathbf{a}) \in \mathcal{D}$. This update induces a change in the Q value for an arbitrary joint action $(\mathbf{o}, \tilde{\mathbf{a}})$ in the local neighborhood. The parameter update is defined as:

$$
\theta' = \theta + \eta \left(\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}; \theta) - Q_{tot}(\mathbf{o}, \mathbf{a}; \theta) \right) \nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a}; \theta). \tag{15}
$$

Theorem 4. *Under Assumptions 2 to 6, for a sufficiently small learning rate η and a joint action $\tilde{\mathbf{a}}$ in the proximity of $\mathbf{a}$, the following holds:*

$$Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta') = Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) + C_1 \left(\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) - Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) + C_2 \|\tilde{\mathbf{a}} - \mathbf{a}\|\right) + O(\|\theta' - \theta\|^2). \tag{16}$$

Proof. Applying a Taylor expansion of the value function at θ, we have:

$$Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta') = Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) + \nabla_\theta Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta)^\top (\theta' - \theta) + O(\|\theta' - \theta\|^2). \tag{17}$$

By substituting the update rule from Eq. (15) into Eq. (17), we obtain:

$$\begin{aligned} Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta') = Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) + \eta \nabla_\theta Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta)^\top \nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a}; \theta) \\ \times \left(\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}; \theta) - Q_{tot}(\mathbf{o}, \mathbf{a}; \theta)\right) + O(\|\theta' - \theta\|^2). \end{aligned} \tag{18}$$

From the Lipschitz conditions stated in Assumption (2) and Lemma (1), it follows that:

$$|Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) - Q_{tot}(\mathbf{o}, \mathbf{a}; \theta)| \leq K_Q \|\tilde{\mathbf{a}} - \mathbf{a}\|, \tag{19}$$

$$|\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) - \mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}; \theta)| \leq K_\mathcal{T} \|\tilde{\mathbf{a}} - \mathbf{a}\|. \tag{20}$$

Consequently, the difference between the TD-errors at $\tilde{\mathbf{a}}$ and $\mathbf{a}$ satisfies:

$$\begin{aligned} &|(\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}) - Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}})) - (\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}) - Q_{tot}(\mathbf{o}, \mathbf{a}))| \\ &\leq |\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}) - \mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a})| + |Q_{tot}(\mathbf{o}, \mathbf{a}) - Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}})| \\ &\leq (K_\mathcal{T} + K_Q) \|\tilde{\mathbf{a}} - \mathbf{a}\|. \end{aligned}$$

Thus, there exists a constant $C_2 \in [-(K_\mathcal{T} + K_Q), K_\mathcal{T} + K_Q]$ such that:

$$\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \mathbf{a}) - Q_{tot}(\mathbf{o}, \mathbf{a}) = \mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}) - Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}) + C_2 \|\tilde{\mathbf{a}} - \mathbf{a}\|. \tag{21}$$

Let $\mathbf{v} = \nabla_\theta Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}) - \nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a})$. By Assumption (3),

$$\|\mathbf{v}\| = \|\nabla_\theta Q_{\text{tot}}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) - \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \mathbf{a}; \theta)\| \leq K_g \|\tilde{\mathbf{a}} - \mathbf{a}\| \tag{22}$$

The inner product is then bounded as follows:

$$\begin{aligned} \nabla_\theta Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}})^\top \nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a}) &= (\nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a}) + \mathbf{v})^\top \nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a}) \\ &= \|\nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a})\|^2 + \mathbf{v}^\top \nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a}) \\ &\geq \|\nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a})\|^2 - \|\mathbf{v}\| \|\nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a})\| \\ &\geq \|\nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a})\|^2 - K_g \|\tilde{\mathbf{a}} - \mathbf{a}\| \|\nabla_\theta Q_{tot}(\mathbf{o}, \mathbf{a})\|. \end{aligned}$$

Therefore, when $\tilde{\mathbf{a}}$ and $\mathbf{a}$ are sufficiently close such that $\|\tilde{\mathbf{a}} - \mathbf{a}\| \leq \|\nabla_\theta Q_{\text{tot}}(\mathbf{o}, \mathbf{a}; \theta)\| / K_g$, it follows that $\eta \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \tilde{\mathbf{a}}; \theta)^\top \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \mathbf{a}; \theta) \geq 0$. Moreover, since the gradient norm is bounded by $g_{\max}$ according to Assumption 4, we have:

$$\eta \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \tilde{\mathbf{a}}; \theta)^\top \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \mathbf{a}; \theta) \leq \eta g_{\max}^2.$$

By selecting a sufficiently small learning rate η such that $\eta \leq 1/g_{\max}^2$, we ensure that:

$$\eta \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \tilde{\mathbf{a}}; \theta)^\top \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \mathbf{a}; \theta) \leq 1$$

In such cases, where $\tilde{\mathbf{a}}$ and $\mathbf{a}$ are sufficiently close and the learning rate η is sufficiently small, we let:

$$C_1 := \eta \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \tilde{\mathbf{a}}; \theta)^\top \nabla_\theta Q_{\text{tot}}(\mathbf{o}, \mathbf{a}; \theta), \tag{23}$$

which satisfies $C_1 \in [0, 1]$.

Substituting Eq. (21) and (23) into Equation (18) yields:

$$Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta') = Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) + C_1 \left(\mathcal{T}_{u_{tot}} Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) - Q_{tot}(\mathbf{o}, \tilde{\mathbf{a}}; \theta) + C_2 \|\tilde{\mathbf{a}} - \mathbf{a}\|\right) + O(\|\theta' - \theta\|^2), \tag{24}$$

where $C_1 \in [0, 1]$ acts as an effective learning rate, and the coefficient C_2 is bounded by $C_2 \in [-K_Q - K_\mathcal{T}, K_Q + K_\mathcal{T}]$ with $K_\mathcal{T} = K_r + \gamma K_P |\mathcal{S}| Q_{\max}$.

This completes the proof. □

A.2 Proof of Theorem 2 and Theorem 3

Theorem 5 establishes that the JNC operator is a γ-contraction and admits a unique fixed point. Theorem 6 shows that the induced optimal value is no lower than that of strict in-sample optimization.

We first introduce the fundamental definitions for the Joint-action Neighborhood Constraint (JNC).

Definition 5 (Joint-action Neighborhood Constraint)**.** *For a given observation $\mathbf{o}$, the joint action neighborhood $\mathcal{C}_N(\mathbf{o})$ is defined as the set of joint actions $\tilde{\mathbf{a}}$ that are within an ϵ-radius of any joint action in the dataset $\mathcal{D}$ for that observation:*

$$\mathcal{C}_N(\mathbf{o}) := \{\tilde{\mathbf{a}} \in \mathcal{A}^n \mid \|\tilde{\mathbf{a}} - \mathbf{a}\| \leq \epsilon, (\mathbf{o}, \mathbf{a}) \in \mathcal{D}\} \tag{25}$$

This set represents a generalized action region. By construction, it satisfies $\mathcal{C}_\mathcal{D}(\mathbf{o}) \subseteq \mathcal{C}_N(\mathbf{o})$, where $\mathcal{C}_\mathcal{D}(\mathbf{o}) := \{\mathbf{a} \in \mathcal{A}^n \mid (\mathbf{o}, \mathbf{a}) \in \mathcal{D}\}$ denotes the strictly in-sample actions.

Definition 6 (JNC Operator)**.** *The Joint Action Neighborhood Constraint (JNC) operator $\mathcal{T}_{JNC}$ is defined as:*

$$\mathcal{T}_{JNC}Q(\mathbf{o}, \mathbf{a}) := r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}' \sim P(\cdot|\mathbf{o},\mathbf{a})} \left[\max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} Q(\mathbf{o}', \tilde{\mathbf{a}}') \right] \tag{26}$$

where the maximization is performed over the ϵ-expanded neighborhood at the next time step, allowing for local generalization beyond recorded cooperative patterns.

Definition 7 (In-sample Q Learning Operator)**.** *The Joint-action In-sample operator $\mathcal{T}_{In}$ [24] is defined as:*

$$\mathcal{T}_{In}Q(\mathbf{o}, \mathbf{a}) := r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}' \in P(\cdot|\mathbf{o},\mathbf{a})} \left[\max_{\mathbf{a}' \in \mathcal{C}_\mathcal{D}(\mathbf{o}')} Q(\mathbf{o}', \mathbf{a}') \right] \tag{27}$$

where $\mathcal{C}_\mathcal{D}(\mathbf{o}')$ is the set of joint actions available in the dataset for observation $\mathbf{o}'$.

To prove Theorem 5, we first establish the non-expansiveness of maximization in Lemma 2 and use it to prove contraction of the In-sample operator in Lemma 3. The same argument then applies to the JNC operator.

Lemma 2. *For any two functions f_1, f_2 and any non-empty compact set $\mathcal{X}$, the following inequality holds:*

$$\left| \max_{x \in \mathcal{X}} f_1(x) - \max_{x \in \mathcal{X}} f_2(x) \right| \leq \max_{x \in \mathcal{X}} |f_1(x) - f_2(x)| \tag{28}$$

Proof. Define $x_1 := \arg\max_{x \in \mathcal{X}} f_1(x)$ and $x_2 := \arg\max_{x \in \mathcal{X}} f_2(x)$. Without loss of generality, assume $\max_{x \in \mathcal{X}} f_1(x) \geq \max_{x \in \mathcal{X}} f_2(x)$. Then:

$$\begin{aligned} 0 \leq \max_{x \in \mathcal{X}} f_1(x) - \max_{x \in \mathcal{X}} f_2(x) &= f_1(x_1) - f_2(x_2) \leq f_1(x_1) - f_2(x_1) \\ &\leq |f_1(x_1) - f_2(x_1)| \leq \max_{x \in \mathcal{X}} |f_1(x) - f_2(x)| \end{aligned}$$

Similarly, if $\max_{x \in \mathcal{X}} f_2(x) > \max_{x \in \mathcal{X}} f_1(x)$, we have:

$$\max_{x \in \mathcal{X}} f_2(x) - \max_{x \in \mathcal{X}} f_1(x) \leq f_2(x_2) - f_1(x_2) \leq \max_{x \in \mathcal{X}} |f_1(x) - f_2(x)|$$

Combining these two cases, we obtain:

$$\left| \max_{x \in \mathcal{X}} f_1(x) - \max_{x \in \mathcal{X}} f_2(x) \right| \leq \max_{x \in \mathcal{X}} |f_1(x) - f_2(x)|$$

This concludes the proof of Lemma 2. □

Lemma 3 (Contraction of In-sample Operator)**.** *The In-sample operator $\mathcal{T}_{In}$ is a γ-contraction mapping in the in-sample region $\mathcal{C}_\mathcal{D}$ under the L_∞ norm.*

Proof. Let f_1 and f_2 be two arbitrary value functions. For any state-action pair $(\mathbf{o}, \mathbf{a})$ such that $\mathbf{a} \in \mathcal{C}_{\mathcal{D}}(\mathbf{o})$, we have:

$$
\begin{aligned}
&|\mathcal{T}_{In} f_1(\mathbf{o}, \mathbf{a}) - \mathcal{T}_{In} f_2(\mathbf{o}, \mathbf{a})| \\
&= \left| \left(r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}'} \left[\max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} f_1(\mathbf{o}', \mathbf{a}') \right] \right) - \left(r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}'} \left[\max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} f_2(\mathbf{o}', \mathbf{a}') \right] \right) \right| \\
&= \gamma \left| \mathbb{E}_{\mathbf{o}'} \left[\max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} f_1(\mathbf{o}', \mathbf{a}') - \max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} f_2(\mathbf{o}', \mathbf{a}') \right] \right| \\
&\leq \gamma \mathbb{E}_{\mathbf{o}'} \left[\left| \max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} f_1(\mathbf{o}', \mathbf{a}') - \max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} f_2(\mathbf{o}', \mathbf{a}') \right| \right] \\
&\leq \gamma \mathbb{E}_{\mathbf{o}'} \left[\max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} |f_1(\mathbf{o}', \mathbf{a}') - f_2(\mathbf{o}', \mathbf{a}')| \right] \leq \gamma \|f_1 - f_2\|_\infty
\end{aligned}
$$

where the third inequality holds by applying Lemma 2.

Therefore, $\mathcal{T}_{In}$ is a γ-contraction operator. By repeatedly applying $\mathcal{T}_{In}$, any initial Q function converges to a unique fixed point Q^*_{In}:

$$
Q^*_{In}(\mathbf{o}, \mathbf{a}) = r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}'} \left[\max_{\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')} Q^*_{In}(\mathbf{o}', \mathbf{a}') \right]. \tag{29}
$$

We denote its induced policy by π^*_{In}, which is defined as:

$$
\pi^*_{In}(\mathbf{o}) := \arg \max_{\mathbf{a} \in \mathcal{C}_{\mathcal{D}}(\mathbf{o})} Q^*_{In}(\mathbf{o}, \mathbf{a}). \tag{30}
$$

□

Theorem 5 (Contraction of JNC operator)**.** *The JNC operator $\mathcal{T}_{JNC}$ is a γ-contraction mapping in the generalized neighborhood region $\mathcal{C}_N$ under the L_∞ norm.*

Proof. Let f_1 and f_2 be two arbitrary value functions. For any $(\mathbf{o}, \mathbf{a})$ such that $\mathbf{a} \in \mathcal{C}_N(\mathbf{o})$:

$$
\begin{aligned}
&|\mathcal{T}_{JNC} f_1(\mathbf{o}, \mathbf{a}) - \mathcal{T}_{JNC} f_2(\mathbf{o}, \mathbf{a})| \\
&= \left| \left(r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}'} \left[\max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} f_1(\mathbf{o}', \tilde{\mathbf{a}}') \right] \right) - \left(r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}'} \left[\max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} f_2(\mathbf{o}', \tilde{\mathbf{a}}') \right] \right) \right| \\
&= \gamma \left| \mathbb{E}_{\mathbf{o}'} \left[\max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} f_1(\mathbf{o}', \tilde{\mathbf{a}}') - \max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} f_2(\mathbf{o}', \tilde{\mathbf{a}}') \right] \right| \\
&\leq \gamma \mathbb{E}_{\mathbf{o}'} \left[\left| \max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} f_1(\mathbf{o}', \tilde{\mathbf{a}}') - \max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} f_2(\mathbf{o}', \tilde{\mathbf{a}}') \right| \right] \\
&\leq \gamma \mathbb{E}_{\mathbf{o}'} \left[\max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} |f_1(\mathbf{o}', \tilde{\mathbf{a}}') - f_2(\mathbf{o}', \tilde{\mathbf{a}}')| \right] \leq \gamma \|f_1 - f_2\|_\infty
\end{aligned}
$$

By repeatedly applying $\mathcal{T}_{JNC}$, there exists a unique fixed point Q^*_{JNC} such that:

$$
Q^*_{JNC}(\mathbf{o}, \mathbf{a}) = r(\mathbf{o}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{o}' \in P(\cdot|\mathbf{o}, \mathbf{a})} \left[\max_{\tilde{\mathbf{a}}' \in \mathcal{C}_N(\mathbf{o}')} Q^*_{JNC}(\mathbf{o}', \tilde{\mathbf{a}}') \right] \tag{31}
$$

We define the induced JNC optimal policy π^*_{JNC} as:

$$
\pi^*_{JNC}(\mathbf{o}) := \arg \max_{\tilde{\mathbf{a}} \in \mathcal{C}_N(\mathbf{o})} Q^*_{JNC}(\mathbf{o}, \tilde{\mathbf{a}}) \tag{32}
$$

This concludes the proof. □

To prove Theorem 6, we first compare the two operators applied to the same value function in Lemma 4, using the inclusion between their feasible action sets. We then establish the monotonicity of the JNC operator in Lemma 5.

Lemma 4 (Operator Dominance)**.** *For any value function f, the following inequality holds:*

$$
\mathcal{T}_{JNC} f(\mathbf{o}, \mathbf{a}) \geq \mathcal{T}_{In} f(\mathbf{o}, \mathbf{a}), \quad \forall \mathbf{o} \in \mathcal{D}, \forall \mathbf{a} \in \mathcal{C}_N(\mathbf{o}) \tag{33}
$$

Proof. By Definition 1, $\mathcal{C}_{\mathcal{D}}(\mathbf{o}) \subseteq \mathcal{C}_N(\mathbf{o})$.

$$\mathcal{T}_{JNC}f(\mathbf{o},\mathbf{a}) - \mathcal{T}_{In}f(\mathbf{o},\mathbf{a}) = \gamma\mathbb{E}_{\mathbf{o}'}\left[\max_{\tilde{\mathbf{a}}'\in\mathcal{C}_N(\mathbf{o}')} f(\mathbf{o}',\tilde{\mathbf{a}}') - \max_{\mathbf{a}'\in\mathcal{C}_{\mathcal{D}}(\mathbf{o}')} f(\mathbf{o}',\mathbf{a}')\right]$$

Since $\mathcal{C}_{\mathcal{D}}(\mathbf{o}') \subseteq \mathcal{C}_N(\mathbf{o}')$, the maximum over the larger set $\mathcal{C}_N$ is greater than or equal to that over $\mathcal{C}_{\mathcal{D}}$. Thus, $\mathcal{T}_{JNC}f(\mathbf{o},\mathbf{a}) \geq \mathcal{T}_{In}f(\mathbf{o},\mathbf{a})$. □

Lemma 5 (Monotonicity of JNC operator)**.** *For any two value functions f_1, f_2 satisfying $f_1(\mathbf{o},\mathbf{a}) \geq f_2(\mathbf{o},\mathbf{a})$ for all $\mathbf{o}$ and $\mathbf{a} \in \mathcal{C}_N(\mathbf{o})$:*

$$\mathcal{T}_{JNC}f_1(\mathbf{o},\mathbf{a}) \geq \mathcal{T}_{JNC}f_2(\mathbf{o},\mathbf{a}), \quad \forall \mathbf{o} \in \mathcal{D}, \forall \mathbf{a} \in \mathcal{C}_N(\mathbf{o}) \tag{34}$$

Proof.

$$\mathcal{T}_{JNC}f_1(\mathbf{o},\mathbf{a}) - \mathcal{T}_{JNC}f_2(\mathbf{o},\mathbf{a}) = \gamma\mathbb{E}_{\mathbf{o}'}\left[\max_{\tilde{\mathbf{a}}'\in\mathcal{C}_N(\mathbf{o}')} f_1(\mathbf{o}',\tilde{\mathbf{a}}') - \max_{\tilde{\mathbf{a}}'\in\mathcal{C}_N(\mathbf{o}')} f_2(\mathbf{o}',\tilde{\mathbf{a}}')\right]$$

Since $f_1 \geq f_2$, the difference in expectations is non-negative. □

Theorem 6 (Performance Superiority)**.** *The value functions of π^*_{JNC} and π^*_{In} satisfy:*

$$V^*_{JNC}(\mathbf{o}) \geq V^*_{In}(\mathbf{o}), \quad \forall \mathbf{o} \in \mathcal{D}. \tag{35}$$

Proof. We first prove by induction that for any $k \in \mathbb{Z}^+$:

$$(\mathcal{T}_{JNC})^k f(\mathbf{o},\mathbf{a}) \geq (\mathcal{T}_{In})^k f(\mathbf{o},\mathbf{a}), \quad \forall f, \mathbf{o} \in \mathcal{D}, \mathbf{a} \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}). \tag{36}$$

Base Case ($k = 1$): According to Lemma 4, it directly holds that:

$$(\mathcal{T}_{JNC})^1 f(\mathbf{o},\mathbf{a}) \geq (\mathcal{T}_{In})^1 f(\mathbf{o},\mathbf{a}), \quad \forall f, \mathbf{o} \in \mathcal{D}, \mathbf{a} \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}).$$

Inductive Step: Suppose that the inequality holds for $k = i$. Let $f_1 = (\mathcal{T}_{JNC})^i f$ and $f_2 = (\mathcal{T}_{In})^i f$. By Lemma 4 (Operator Dominance), we have:

$$\mathcal{T}_{JNC}(\mathcal{T}_{JNC})^i f(\mathbf{o},\mathbf{a}) \geq \mathcal{T}_{In}(\mathcal{T}_{JNC})^i f(\mathbf{o},\mathbf{a}), \quad \forall f,\ \mathbf{o} \in \mathcal{D},\ \mathbf{a} \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}). \tag{37}$$

The induction hypothesis gives $f_1(\mathbf{o}',\mathbf{a}') \geq f_2(\mathbf{o}',\mathbf{a}')$ for all relevant next observations $\mathbf{o}'$ and $\mathbf{a}' \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}')$. Since the In-sample operator maximizes only over $\mathcal{C}_{\mathcal{D}}(\mathbf{o}')$, it is monotone with respect to this ordering. Therefore,

$$\mathcal{T}_{In}(\mathcal{T}_{JNC})^i f(\mathbf{o},\mathbf{a}) \geq \mathcal{T}_{In}(\mathcal{T}_{In})^i f(\mathbf{o},\mathbf{a}), \quad \forall f,\ \mathbf{o} \in \mathcal{D},\ \mathbf{a} \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}). \tag{38}$$

Combining Eq. (37) and Eq. (38), we obtain the property for $i + 1$. Thus, Eq. (36) holds for all $k \in \mathbb{Z}^+$. Since both $\mathcal{T}_{JNC}$ and $\mathcal{T}_{In}$ are γ-contraction operators, their fixed points are attained as $k \to \infty$:

$$Q^*_{JNC}(\mathbf{o},\mathbf{a}) = \lim_{k\to\infty} (\mathcal{T}_{JNC})^k f(\mathbf{o},\mathbf{a}), \quad \forall f, \mathbf{o} \in \mathcal{D}, \mathbf{a} \in \mathcal{C}_N(\mathbf{o}). \tag{39}$$

$$Q^*_{In}(\mathbf{o},\mathbf{a}) = \lim_{k\to\infty} (\mathcal{T}_{In})^k f(\mathbf{o},\mathbf{a}), \quad \forall f, \mathbf{o} \in \mathcal{D}, \mathbf{a} \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}). \tag{40}$$

Given $\mathcal{C}_{\mathcal{D}} \subseteq \mathcal{C}_N$, combining Eq. (36) (as $k \to \infty$) with Eq. (39) and Eq. (40) yields:

$$Q^*_{JNC}(\mathbf{o},\mathbf{a}) \geq Q^*_{In}(\mathbf{o},\mathbf{a}), \quad \forall \mathbf{o} \in \mathcal{D}, \mathbf{a} \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}). \tag{41}$$

Finally, for any observation $\mathbf{o} \in \mathcal{D}$, the performance is compared as:

$$V^*_{JNC}(\mathbf{o}) = Q^*_{JNC}(\mathbf{o},\pi^*_{JNC}(\mathbf{o})) \geq Q^*_{JNC}(\mathbf{o},\pi^*_{In}(\mathbf{o})) \geq Q^*_{In}(\mathbf{o},\pi^*_{In}(\mathbf{o})) = V^*_{In}(\mathbf{o}). \tag{42}$$

where the first inequality follows from the optimality of π^*_{JNC} within $\mathcal{C}_N(\mathbf{o})$ and the fact that $\pi^*_{In}(\mathbf{o}) \in \mathcal{C}_{\mathcal{D}}(\mathbf{o}) \subseteq \mathcal{C}_N(\mathbf{o})$. The second inequality holds due to the pointwise dominance established in Eq. (41). This concludes the proof. □

Table 3: Summary of the offline datasets used in our experiments. Average returns are reported with the mean and standard deviation. State dimension refers to the global state, while observation and action dimensions refer to each agent.

Task	Dataset	Trajectories	Samples	Agents	State dim	Obs dim	Action dim	Average returns
HalfCheetah	expert	1.0K	1000K	6	138	23	1	2785.1±1053.1
	medium	1.0K	1000K	6	138	23	1	1425.7±20.1
	m-replay	1.0K	1000K	6	138	23	1	655.8±590.4
	m-expert	2.0K	2000K	6	138	23	1	2105.4±1073.2
Hopper	expert	1.5K	999K	3	42	14	1	2452.0±1097.9
	medium	4.0K	915K	3	42	14	1	723.6±211.7
	m-replay	4.2K	1311K	3	42	14	1	746.4±471.9
	m-expert	5.5K	1914K	3	42	14	1	1190.6±973.4
Ant	expert	1.0K	1000K	2	226	113	4	2055.1±22.1
	medium	1.0K	1000K	2	226	113	4	1418.7±37.0
	m-replay	1.8K	1750K	2	226	113	4	1029.5±141.3
	m-expert	2.0K	2000K	2	226	113	4	1736.9±319.6
Protoss_5_vs_5	expert	1.0K	31.5K	5	130	92	11	14.9±5.6
	medium	1.0K	32.6K	5	130	92	11	14.4±4.1
	m-expert	1.0K	32.2K	5	130	92	11	14.9±3.9
	random	1.0K	30.8K	5	130	92	11	10.7±3.4
Terran_5_vs_5	expert	1.0K	24.3K	5	120	82	11	15.2±7.2
	medium	1.0K	24.8K	5	120	82	11	10.9±4.7
	m-expert	1.0K	24.7K	5	120	82	11	12.9±5.5
	random	1.0K	26.5K	5	120	82	11	8.3±4.6
Zerg_5_vs_5	expert	1.0K	14.3K	5	120	82	11	13.2±6.0
	medium	1.0K	15.5K	5	120	82	11	10.6±5.0
	m-expert	1.0K	15.0K	5	120	82	11	12.1±5.7
	random	1.0K	20.3K	5	120	82	11	9.3±4.3

B Experimental Details

B.1 Offline Multi-Agent Datasets

The MaMujoco benchmark consists of Hopper-v2, Ant-v2, and HalfCheetah-v2, which involve 3, 2, and 6 agents, respectively. Following [17], we use datasets collected with HAPPO, each providing four data quality levels: expert, medium, medium-expert, and medium-replay. These datasets contain between 1.0K and 5.5K trajectories and between 0.9M and 2.0M samples. Table 3 summarizes their sizes, agent configurations, dimensions, and average returns.

The SMACv2 benchmark consists of three scenarios, Protoss_5_vs_5, Terran_5_vs_5, and Zerg_5_vs_5, each involving 5 agents. Following [38], we use datasets collected with MAPPO, each providing four data quality levels: expert, medium, medium-expert, and random. The expert and medium datasets contain 1,000 trajectories collected from agents trained for 10 million and 5 million steps, respectively; the medium-expert dataset combines 500 randomly selected trajectories from each of the two, and the random dataset contains 1,000 trajectories collected using randomly initialized models. These datasets contain between 14.3K and 32.6K samples. Table 3 summarizes their sizes, agent configurations, dimensions, and average returns.

The average returns in Table 3 characterize the collected datasets. For SMACv2, these return statistics are distinct from the win rates reported in the main experimental results.

B.2 Implementation for Discrete Action Spaces

In environments with discrete action spaces, we implement a discrete variant of the framework. We define the neighborhood under an independence assumption. Specifically, let $\pi_i(a_i|o_i)$ denote the individual policy of agent i. Given the factorized joint behavior policy $\pi_{\text{BC}}(\mathbf{a}|\mathbf{o}) = \prod_i \pi_{\text{BC},i}(a_i|o_i)$,

the discrete joint neighborhood constraint is defined as:

$$\mathcal{C}_{DN}(\mathbf{o}) := \left\{ \prod_{i=1}^{n} \pi_{\delta,i} \;\middle|\; \sum_{i=1}^{n} D_{\mathrm{KL}}\big(\pi_{\delta,i} \,\|\, \pi_{\mathrm{BC},i}\big) \le \epsilon \right\}, \tag{43}$$

where $\pi_{\delta,i}$ represents the perturbed action distribution for agent i. This formulation ensures that the joint policy remains within the neighborhood of the dataset behavior policy. The learning flow is summarized as follows.

B.2.1 Adaptive Neighborhood Q learning

Based on $\mathcal{C}_{DN}(\mathbf{o})$, we solve for an improved joint policy $\boldsymbol{\pi}_\delta \in \mathcal{C}_{DN}(\mathbf{o})$ that maximizes the expected global Q value while considering the adaptive constraint:

$$\begin{aligned} \max_{\boldsymbol{\pi}_\delta} \quad & \mathbb{E}_{\tilde{\mathbf{a}}\sim\boldsymbol{\pi}_\delta(\cdot|\mathbf{o})}[Q_{\mathrm{tot}}(\mathbf{o},\tilde{\mathbf{a}})] \\ \text{s.t.} \quad & \sum_{i=1}^{n} D_{\mathrm{KL}}\big(\pi_{\delta,i} \,\|\, \pi_{\mathrm{BC},i}\big) e^{\alpha A_{\mathrm{tot}}(\mathbf{o},\mathbf{a})} \le \epsilon, \forall(\mathbf{o},\mathbf{a}) \in \mathcal{D}. \end{aligned} \tag{44}$$

In practice, this is solved via a dual formulation using a Lagrange multiplier $\lambda > 0$:

$$\begin{aligned} \max_{\boldsymbol{\pi}_\delta} \mathbb{E}_{(\mathbf{o},\mathbf{a})\sim\mathcal{D}} \Big[& \mathbb{E}_{\tilde{\mathbf{a}}\sim\boldsymbol{\pi}_\delta(\cdot|\mathbf{o})}[Q_{\mathrm{tot}}(\mathbf{o},\tilde{\mathbf{a}})] \\ & - \lambda \exp\big(\alpha(\bar{Q}_{\mathrm{tot}}(\mathbf{o},\mathbf{a}) - V_{\mathrm{tot}}(\mathbf{o}))\big) \cdot \sum_{i=1}^{n} D_{\mathrm{KL}}\big(\pi_{\delta,i} \,\|\, \pi_{\mathrm{BC},i}\big) \Big]. \end{aligned} \tag{45}$$

where $\bar{Q}_{tot}$ is the target global Q function.

B.2.2 Multi-Agent Value Function Learning

The local state-value functions $V_i(o_i)$ are updated to capture the maximum value attainable within the distribution-based neighborhood $\mathcal{C}_{DN}$:

$$\min_{V_i} \mathbb{E}_{o_i\sim\mathcal{D}} \left[\exp\left(\frac{w_i(\mathbf{o})}{\beta} \left(\mathbb{E}_{\tilde{a}_i\sim\bar{\pi}_{\delta,i}}[\bar{Q}_i(o_i,\tilde{a}_i)] - V_i(o_i) \right) \right) + \frac{w_i(\mathbf{o})}{\beta} V_i(o_i) \right]. \tag{46}$$

where $\bar{Q}_i$ is the target local Q function and $\bar{\pi}_{\delta,i}$ is the target local auxiliary policy. Based on the learned V-network, the Q learning objective for the global network is defined as:

$$\min_{\substack{Q_i,w_i,b \\ i=1,\ldots,n}} \mathbb{E}_{(\mathbf{o},\mathbf{a},\mathbf{o}')\sim\mathcal{D}} \left[\left(Q_{\mathrm{tot}}(\mathbf{o},\mathbf{a}) - (r(\mathbf{o},\mathbf{a}) + \gamma V_{\mathrm{tot}}(\mathbf{o}')) \right)^2 \right], \tag{47}$$

where $V_{\mathrm{tot}}(\mathbf{o}') = \sum_i w_i(\mathbf{o}') V_i(o_i') + b(\mathbf{o}')$ represents the maximum neighborhood value of the next state, ensuring Bellman consistency within the trusted region.

B.2.3 Multi-Agent Policy Learning

Finally, the individual policy networks π_i are trained to distill the improved cooperative strategies $\boldsymbol{\pi}_\delta$. By targeting the distributions identified during the neighborhood search, the individual policy is optimized by minimizing the following weighted KL-divergence objective:

$$\max_{\pi_i} \mathbb{E}_{o_i\sim\mathcal{D}} \left[\exp\left(\frac{w_i(\mathbf{o})}{\beta} \left(\mathbb{E}_{\tilde{a}_i\sim\pi_{\delta,i}}[\bar{Q}_i(o_i,\tilde{a}_i)] - V_i(o_i) \right) \right) \cdot \sum_{a_i\in\mathcal{A}} \pi_{\delta,i}(a_i|o_i) \log \pi_i(a_i|o_i) \right]. \tag{48}$$

This formulation aligns the policy π_i with the optimized neighborhood distribution $\pi_{\delta,i}$. By incorporating advantage-based weights, it encourages the policy to favor high-value actions discovered in the neighborhood while remaining consistent with the learned value functions.

B.3 Comparison with Independent ANQ

We compare JANC with Independent ANQ, a per-agent extension of ANQ [27] implemented in this work. Both methods use the same datasets, training budget, and backbone network architectures. The complete results are reported in Table 4.

Table 4: Average returns of Independent ANQ and JANC with the mean and standard deviation on MaMujoco tasks.

Task	Dataset	Independent ANQ	JANC(ours)
HalfCheetah	expert	3142.38±445.16	**4781.4±132.4**
	m-expert	2902.49±870.58	**4667.1±278.4**
	medium	3615.53±230.49	**4833.3±181.7**
	m-replay	3321.41±455.56	**3941.7±93.6**
Hopper	expert	1839.13±151.90	**2664.4±578.1**
	m-expert	1594.97±1162.10	**2392.8±450.4**
	medium	1273.73±376.80	**2243.8±267.6**
	m-replay	624.65±347.59	**1084.9±255.7**
Ant	expert	2050.69±15.91	**2108.3±27.0**
	m-expert	1939.90±82.71	**2094.6±48.2**
	medium	1430.27±7.13	**1454.0±37.9**
	m-replay	1005.76±29.78	**1114.8±81.2**

B.4 Hyperparameters

Our method introduces an auxiliary network for action refinement, whose architecture is identical to the policy network. The framework incorporates two critical hyperparameters: the lagrange multiplier λ and the inverse temperature α. For the multi-agent Mujoco (MaMujoco) [36] environments, we set $\lambda = 0.5$ for Hopper and Ant, and $\lambda = 0.01$ for HalfCheetah, while α is configured to 1.0. For the SMACv2 [37] environments, we maintain a consistent setting of $\lambda = 0.5$ and $\alpha = 1.0$ across all tasks. Detailed network architectures and training hyperparameters are summarized in Table 5.

Table 5: Hyperparameters of JANC

Hyperparameter	Value
Optimizer	Adam
Hidden dimension	256
Weight network hidden dimension	64
Batch size	128
State-value network learning rate	1e-4
Q value network learning rate	1e-4
Policy network learning rate	1e-4
Target update rate	0.005
Discount factor	0.99
Temperature parameter β	10
Temperature parameter α	1
Lagrange multiplier λ	{0.01, 0.5}

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to Section 6

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Please refer to Appendix A

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please refer to Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Please refer to the code in the supplemental material.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Please refer to Appendix B

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The results in the paper are accompanied by standard deviations across multiple seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please refer to Subsection 5.1

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please refer to Section 7

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The creators or original owners of assets used in the paper are properly credited and the license and terms of use are explicitly mentioned and properly respected.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code is well documented and anonymized.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.