OPTIMAL JOINT-ACTION SCREENING FOR COOPERATIVE MULTI-AGENT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Value factorization enables centralized training with decentralized execution in cooperative multi-agent reinforcement learning, but the widely adopted monotonic mixing structure leads to the suppression of individually optimal actions when they are paired with inferior teammate actions. Existing weighted methods mitigate this interference by identifying and emphasizing potentially optimal joint actions. However, inaccurate optimality criteria may exclude truly optimal samples from the protected set while including non-optimal ones, undermining the intended effect of sample weighting. Instead of identifying optimal joint actions directly, we seek a screening threshold that retains all optimal joint actions while safely identifying non-optimal samples for downweighting. We show that the maximum value from the least-squares projection onto the QMIX function class lower-bounds the optimal joint-action value and therefore provides such a threshold. Based on this, we propose Optimal Joint-Action Screening (OJS), which uses an independently trained reference QMIX to construct the candidate set. Candidate samples receive unit weight, while the remaining samples are adaptively soft-weighted according to their potential interference with optimal-action learning. Experiments across diverse cooperative benchmarks show that OJS achieves competitive or superior performance over representative value-factorization baselines.

1 INTRODUCTION

Cooperative multi-agent reinforcement learning (MARL) has shown great potential in applications such as multi-robot coordination, autonomous driving, and distributed resource allocation (Howell et al., 2023; Wu et al., 2023; Li et al., 2023). A widely adopted paradigm is centralized training with decentralized execution (CTDE), which exploits global information during training while requiring each agent to act solely from its local action-observation history (Oroojlooy & Hajinezhad, 2023). Value factorization provides an effective realization of CTDE by decomposing a centralized joint action-value function into individual utilities (Sunehag et al., 2018; Rashid et al., 2020b; Xu et al., 2025). To ensure consistency between centralized training and decentralized execution, the individually greedy actions should jointly maximize the global action-value function, as formalized by the individual-global-max (IGM) principle (Son et al., 2019). QMIX enforces this consistency through a monotonic mixing function and has demonstrated strong empirical performance across cooperative MARL benchmarks (Hu et al., 2023).

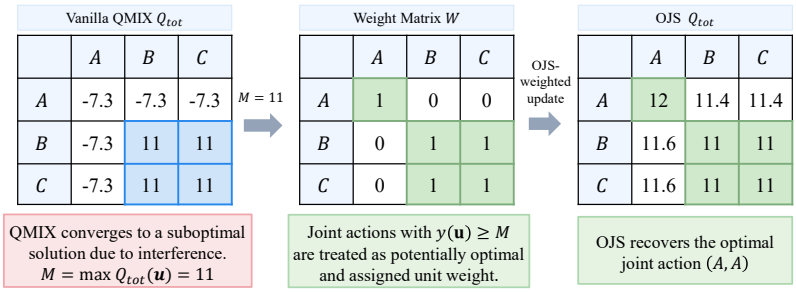
Despite its effectiveness, QMIX suffers from inherently coupled credit assignment (Gupta et al., 2021; Wang et al., 2021a; Li et al., 2025). Under monotonic value factorization, the utility assigned to an individual action is learned from joint-action samples containing that action and therefore depends on the actions selected by other agents (Zheng et al., 2021; Zhou et al., 2022). As a result, an action belonging to an optimal joint action may be undervalued or incorrectly penalized when paired with suboptimal teammate actions (Hou et al., 2026). This problem is particularly severe because replay buffers can contain many suboptimal or miscoordinated joint actions (Mahajan et al., 2019), whose gradients may overwhelm the relatively sparse signals from successful coordination and impair individual credit assignment. Figure 1(a) illustrates this interference in a simple matrix game.

Recent weighted value-factorization methods address this issue by modulating the contribution of different joint-action samples during training (Rashid et al., 2020a; Wan et al., 2022; Hou et al., 2026; Huang et al., 2026). Their common objective is to identify potentially optimal joint actions and assign

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

u_1 \ u_2	A	B	C
A	12	-12	-12
B	-12	11	11
C	-12	11	11

(a) Interference-prone payoff matrix



(b) OJS screening and weighted update

Figure 1: Illustration of OJS in an interference-prone matrix game. (a) The optimal joint action (A, A) shares its individual actions with inferior joint actions, making them susceptible to interference. (b) Vanilla QMIX consequently converges to a suboptimal solution. OJS uses the maximum projected value $M = \max_{\mathbf{u}} Q_{\text{tot}}(\mathbf{u})$ as a screening threshold. Samples with $y(\mathbf{u}) \geq M$ are retained with unit weight, while the remaining samples are downweighted, allowing OJS to recover (A, A).

them greater learning weight. However, existing methods rely largely on heuristic optimality criteria, such as optimistic value estimates or learned auxiliary recognizers. These criteria do not provide a reliable guarantee for distinguishing optimal from non-optimal joint actions. Consequently, truly optimal samples may be excluded from the protected set while non-optimal samples are incorrectly included, undermining the intended effect of sample weighting and hindering reliable optimal-policy recovery.

To overcome these limitations, we derive a theoretically justified screening criterion from the projection view of QMIX. Specifically, we show that the maximum of the projected value function lower-bounds the optimal joint-action value, which enables the construction of a candidate set guaranteed to retain all optimal joint actions. Building on this result, we propose Optimal Joint-Action Screening (OJS), a weighted value-factorization method that assigns unit weight to candidate samples and adaptively soft-weights the remaining samples according to the magnitude of their interference with optimal-action learning. Figure 1(b) illustrates this screening-and-weighting mechanism. This design provides principled protection for optimal joint actions while safely reducing interference from non-optimal samples.

Our main contributions are summarized as follows:

- We prove that the maximum value obtained from the least-squares projection onto the QMIX function class lower-bounds the true optimal joint-action value, and derive from this bound a screening criterion that provably retains all optimal joint actions.
- We propose OJS, a weighting framework that uses an independently trained reference QMIX for candidate screening and adaptively soft-weights non-candidate samples according to their interference magnitude with optimal-action learning.
- Extensive experiments across diverse cooperative MARL benchmarks show that OJS achieves competitive or superior performance over representative value-factorization baselines, while component ablations validate the contributions of decoupled screening and adaptive soft weighting.

2 Preliminaries

Dec-POMDP A cooperative multi-agent task is commonly modeled as a decentralized partially observable Markov decision process (Dec-POMDP), defined as $\mathcal{G} = \langle \mathcal{N}, \mathcal{S}, \{\mathcal{U}_i\}_{i=1}^N, P, r, \{\mathcal{O}_i\}_{i=1}^N, O, \gamma \rangle$ (Oliehoek & Amato, 2016), where $\mathcal{N} = \{1, \ldots, N\}$ is the agent set and $\mathcal{S}$ is the global state space. For each agent i, $\mathcal{U}_i$ and $\mathcal{O}_i$ denote its action and observation spaces, respectively, with joint spaces $\mathcal{U} = \prod_{i=1}^N \mathcal{U}_i$ and $\mathcal{O} = \prod_{i=1}^N \mathcal{O}_i$. At time t, agent

i selects $u_i^t \in \mathcal{U}_i$ based on its local action-observation history $\tau_i^t \in \mathcal{T}_i$, forming the joint action $\mathbf{u}^t = (u_1^t, \ldots, u_N^t) \in \mathcal{U}$. The environment transitions according to $P(s^{t+1} \mid s^t, \mathbf{u}^t)$ and provides all agents with a shared reward $r(s^t, \mathbf{u}^t)$. Finally, the observation function O generates a joint observation $\mathbf{o}^{t+1} \in \mathcal{O}$ according to $O(\cdot \mid s^{t+1}, \mathbf{u}^t)$, and each agent receives its local component o_i^{t+1}. Let $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_N)$ denote the joint policy. The objective is to maximize the expected discounted return $\mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s^t, \mathbf{u}^t)\right]$, where $\gamma \in [0, 1)$ is the discount factor. Under centralized training with decentralized execution, global information may be exploited during training, whereas each agent must make decisions using only its local action-observation history τ_i during execution.

Value Function Factorization Value factorization represents the centralized joint action-value function as a combination of individual utilities:

$$Q_{\text{tot}}(\boldsymbol{\tau}, \mathbf{u}) = f_{\text{mix}}\left(Q_1(\tau_1, u_1), \ldots, Q_N(\tau_N, u_N); s\right), \tag{1}$$

where $Q_i(\tau_i, u_i)$ is the utility of agent i and f_{mix} is a mixing function that may depend on the global state during training. VDN uses the additive factorization (Sunehag et al., 2018) $Q_{\text{tot}} = \sum_{i=1}^{N} Q_i$, whereas QMIX uses a nonlinear mixer constrained to be monotonic in every individual utility (Rashid et al., 2020b):

$$\frac{\partial Q_{\text{tot}}}{\partial Q_i} \geq 0, \qquad \forall i \in \mathcal{N}. \tag{2}$$

Individual-Global-Max Consistency A factorized value function satisfies individual-global-max consistency if individually greedy actions form a globally greedy joint action:

$$\arg\max_{\mathbf{u}} Q_{\text{tot}}(\boldsymbol{\tau}, \mathbf{u}) = \Big(\arg\max_{u_1} Q_1(\tau_1, u_1), \ldots, \arg\max_{u_N} Q_N(\tau_N, u_N)\Big). \tag{3}$$

The additive structure of VDN and the monotonicity constraint of QMIX are sufficient for IGM, enabling decentralized greedy action selection, though at the cost of restricting the representable class of joint value functions.

3 Interference from Suboptimal Joint Actions

For a sample b, let $Q_{\text{tot},b}$ denote the estimated value of its joint action and y_b the corresponding training target. Define $\delta_b = Q_{\text{tot},b} - y_b$. Under the squared loss $\mathcal{L}_b = \frac{1}{2}\delta_b^2$, the gradient with respect to the selected utility $Q_{i,b}$ is

$$\frac{\partial \mathcal{L}_b}{\partial Q_{i,b}} = \delta_b \frac{\partial Q_{\text{tot},b}}{\partial Q_{i,b}}. \tag{4}$$

The mixing derivative equals one for VDN and is non-negative for QMIX. Consequently, gradient descent decreases every selected utility with positive mixing sensitivity when $\delta_b > 0$ and increases it when $\delta_b < 0$. To characterize the resulting interference, let $Q(s, \mathbf{u})$ denote the true joint action value and let $\mathbf{u}^* = (u_1^*, \ldots, u_N^*) \in \arg\max_{\mathbf{u} \in \mathcal{U}} Q(s, \mathbf{u})$. For agent i, define

$$\mathcal{U}_i^-(\mathbf{u}^*) \coloneqq \{\mathbf{u} \in \mathcal{U} \mid u_i = u_i^*,\ Q(s, \mathbf{u}) < Q(s, \mathbf{u}^*)\}. \tag{5}$$

The set $\mathcal{U}_i^-(\mathbf{u}^*)$ contains suboptimal joint actions that share the optimal individual action u_i^*. When such a sample induces a downward update, it suppresses the utility of u_i^* using a return generated by suboptimal coordination. If these updates dominate the replay distribution, they may overwhelm the comparatively rare updates from successful coordination and prevent reliable credit assignment (Rashid et al., 2020a; Hou et al., 2026). We therefore seek a screening criterion that retains all optimal joint actions before selectively reducing the influence of the remaining samples.

4 METHOD

In this section, we present Optimal Joint-Action Screening (OJS). We first derive a projected-maximum criterion that retains all optimal joint actions under the projection assumptions. We then introduce an independently trained reference QMIX to decouple the screening reference from the primary QMIX, followed by a soft weighting rule and the corresponding learning objectives. The overall framework of OJS is illustrated in Figure 2.

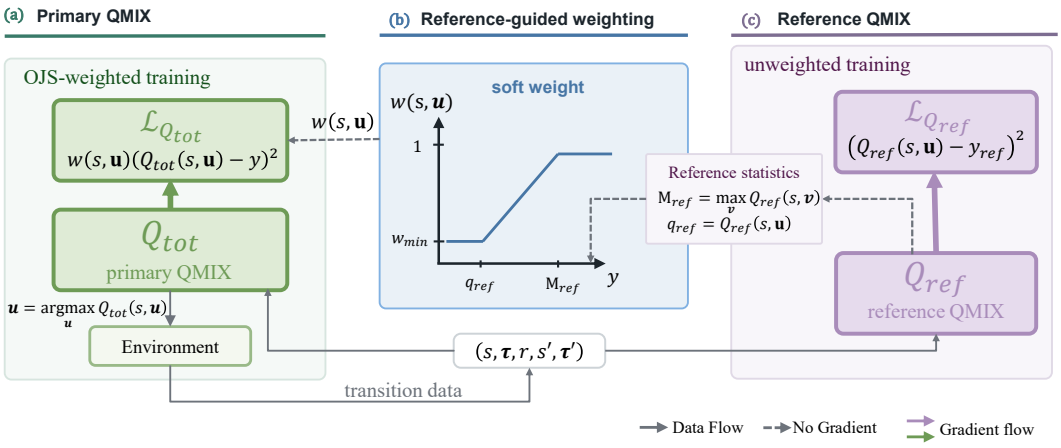


Figure 2: Overview of OJS. (a) The primary QMIX interacts with the environment and is trained with the OJS-weighted loss. (b) Reference-guided weighting compares the target y with the sampled-action reference value q_{ref} and screening threshold M_{ref}. Candidate samples and non-candidate samples whose updates do not decrease the selected utilities receive unit weight, while the remaining samples receive adaptive soft weights. (c) The reference QMIX is independently trained with the unweighted loss on the same transitions.

4.1 Optimal Joint-Action Screening

Following the projection framework of Weighted QMIX (Rashid et al., 2020a), let $Q : \mathcal{S} \times \mathcal{U} \to \mathbb{R}$ denote a joint action-value function that is not necessarily representable by QMIX. The QMIX-representable function class is defined as

$$\mathcal{Q}_{\text{mix}} := \left\{ q \;\middle|\; q(s,\mathbf{u}) = f_s(q_1(s,u_1),\ldots,q_N(s,u_N)),\, \frac{\partial f_s}{\partial q_i} \geq 0, \forall i \in \{1,\ldots,N\} \right\}, \tag{6}$$

where q_i is the individual utility function of agent i, and f_s is a state-dependent monotonic mixing function. For state $s \in \mathcal{S}$, the unweighted least-squares projection of Q onto $\mathcal{Q}_{\text{mix}}$ is defined as

$$\Pi_{\mathcal{Q}_{\text{mix}}} Q := \underset{q \in \mathcal{Q}_{\text{mix}}}{\arg\min} \sum_{\mathbf{u} \in \mathcal{U}} \left(Q(s,\mathbf{u}) - q(s,\mathbf{u})\right)^2. \tag{7}$$

Since the minimizer may not be unique, $\Pi_{\mathcal{Q}_{\text{mix}}} Q$ denotes the set of all minimizers. We use $Q_{\text{tot}} \in \Pi_{\mathcal{Q}_{\text{mix}}} Q$ to denote an arbitrary projected solution.

The following proposition specializes Propositions 1–2 of Rashid et al. (2020a) to the unweighted case and restates them in our notation.

Proposition 1. *Let $Q_{\text{tot}} \in \Pi_{\mathcal{Q}_{\text{mix}}} Q$. Then, for every $s \in \mathcal{S}$, the following properties hold:*

1. *For every $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u}} Q_{\text{tot}}(s,\mathbf{u})$, $Q_{\text{tot}}(s,\hat{\mathbf{u}}) \geq Q(s,\hat{\mathbf{u}})$. If, additionally, $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u}\in\mathcal{U}} Q(s,\mathbf{u})$, then $Q_{\text{tot}}(s,\hat{\mathbf{u}}) = Q(s,\hat{\mathbf{u}})$.*
2. *There exists at least one $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u}} Q_{\text{tot}}(s,\mathbf{u})$ such that $Q_{\text{tot}}(s,\hat{\mathbf{u}}) = Q(s,\hat{\mathbf{u}})$.*

We now use these projection properties to derive a screening criterion for optimal joint actions. Define the maximum values of Q_{tot} and Q as

$$M_{\text{tot}}(s) := \max_{\mathbf{u}\in\mathcal{U}} Q_{\text{tot}}(s,\mathbf{u}), \qquad M(s) := \max_{\mathbf{u}\in\mathcal{U}} Q(s,\mathbf{u}). \tag{8}$$

Proposition 1 immediately gives the following result.

Corollary 1. *Let $Q_{\text{tot}} \in \Pi_{\mathcal{Q}_{\text{mix}}} Q$. For every $s \in \mathcal{S}$, there exists at least one $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u}\in\mathcal{U}} Q_{\text{tot}}(s,\mathbf{u})$ such that $M_{\text{tot}}(s) = Q_{\text{tot}}(s,\hat{\mathbf{u}}) = Q(s,\hat{\mathbf{u}})$. Consequently, $M_{\text{tot}}(s) \leq M(s)$.*

Corollary 1 shows that the projected maximum matches the true value of at least one projected-greedy joint action, while lower-bounding the true optimal joint-action value. This motivates screening joint actions whose true values are no smaller than the projected maximum. We therefore define the candidate set as

$$\mathcal{C}(s) \coloneqq \{\mathbf{u} \in \mathcal{U} \mid Q(s, \mathbf{u}) \geq M_{\mathrm{tot}}(s)\}. \tag{9}$$

The following corollary establishes the resulting optimal joint-action containment guarantee.

Corollary 2. *Let $Q_{\mathrm{tot}} \in \Pi_{\mathcal{Q}_{\mathrm{mix}}} Q$. Then, for every $s \in \mathcal{S}$,* $\arg\max_{\mathbf{u} \in \mathcal{U}} Q(s, \mathbf{u}) \subseteq \mathcal{C}(s)$.

During learning, the true value $Q(s, \mathbf{u})$ is unavailable. Following QMIX and Weighted QMIX, we use the training target y as a noisy sample-level proxy for the value of the sampled joint action (Rashid et al., 2020b;a):

$$\widetilde{\mathcal{C}}(s) \coloneqq \{\mathbf{u} \in \mathcal{U} \mid y \geq M_{\mathrm{tot}}(s)\}. \tag{10}$$

Algorithm 1 Training Procedure of OJS

1: Initialize replay buffer $\mathcal{R}$, primary QMIX θ, reference QMIX ϕ, and their target networks
2: **for** $episode = 1$ to M **do**
3: Collect one episode with the ϵ-greedy primary policy and store it in $\mathcal{R}$
4: **if** $\mathcal{R}$ contains sufficient samples **then**
5: Sample a minibatch and compute the primary and reference QMIX values and targets
6: Compute $\delta = Q_{\mathrm{tot}}(s, \mathbf{u}) - y$ and $M_{\mathrm{ref}}(s)$ using equation 11
7: Identify candidate samples using equation 12
8: Set unit weights for candidate samples or samples with $\delta \leq 0$, and compute soft weights for the remaining samples using equation 13
9: Update θ and ϕ separately using equation 14
10: Periodically update both target networks
11: **end if**
12: **end for**

4.2 Decoupled Screening with a Reference QMIX

OJS maintains two identically structured but independently parameterized QMIX networks. The primary QMIX, $Q_{\mathrm{tot}}(\cdot; \theta)$, selects actions and is trained with the OJS-weighted objective, whereas the reference QMIX, $Q_{\mathrm{ref}}(\cdot; \phi)$, is trained with the standard unweighted objective and supplies the screening reference. This separation is important because if the primary QMIX both produced the screening threshold and were optimized using weights induced by that threshold, early approximation errors could alter the candidate set and subsequently reinforce themselves, increasing the risk of convergence to a poor local solution. The independently trained reference QMIX breaks this feedback loop and provides a more stable, less aggressive surrogate for the projected lower bound. For notational convenience, define the sampled-action reference value and reference maximum as

$$q_{\mathrm{ref}}(s, \mathbf{u}) \coloneqq Q_{\mathrm{ref}}(s, \mathbf{u}; \phi), \qquad M_{\mathrm{ref}}(s) \coloneqq \max_{\mathbf{u} \in \mathcal{U}} Q_{\mathrm{ref}}(s, \mathbf{u}; \phi). \tag{11}$$

By the IGM property of the reference QMIX, $M_{\mathrm{ref}}(s)$ can be computed without enumerating joint actions by evaluating Q_{ref} at the joint action formed by the agents' individually greedy actions. Using y as a proxy for the value of the sampled joint action, the practical candidate set is

$$\widehat{\mathcal{C}}(s) = \{\mathbf{u} \in \mathcal{U} \mid y \geq M_{\mathrm{ref}}(s)\}. \tag{12}$$

For a minibatch, we call a sample a candidate sample when its joint action belongs to $\widehat{\mathcal{C}}(s)$.

4.3 Soft Weighting and Dual-Network Optimization

Being classified as a non-candidate is not by itself sufficient for downweighting. Let $\delta = Q_{\mathrm{tot}}(s, \mathbf{u}) - y$ denote the prediction error of the primary QMIX. Under monotonic mixing, a non-candidate sample with $\delta > 0$ induces a downward update to the selected utilities and may suppress an individual action that also belongs to an optimal joint action, while samples with $\delta \leq 0$ retain unit weight because they do not trigger this interference. Since the target y is a noisy sample-based proxy for the underlying

joint-action value, a larger margin below the screening threshold indicates both higher confidence in the sample's non-optimality and stronger potential interference with optimal-action learning. To account for this uncertainty, we let the weight decrease smoothly with the margin below the threshold:

$$w(s,\mathbf{u}) = \begin{cases} 1, & y \geq M_{\text{ref}}(s), \\ w_{\min}, & y < q_{\text{ref}}(s,\mathbf{u}), \\ w_{\min} + (1-w_{\min})\dfrac{y - q_{\text{ref}}(s,\mathbf{u})}{M_{\text{ref}}(s) - q_{\text{ref}}(s,\mathbf{u})}, & \text{otherwise.} \end{cases} \tag{13}$$

Here $w_{\min} \in [0,1]$ is the minimum weight. For downward updates from non-candidate samples, the weight increases smoothly from $w_{\min}$ to 1 as y approaches $M_{\text{ref}}(s)$. The primary and reference networks are optimized separately. Let y^{ref} denote the training target of the reference network. Their objectives are

$$\begin{aligned} \mathcal{L}_{Q_{\text{tot}}} &= \mathbb{E}\Big[w(s,\mathbf{u})\left(Q_{\text{tot}}(s,\mathbf{u}) - y\right)^2\Big], \\ \mathcal{L}_{Q_{\text{ref}}} &= \mathbb{E}\Big[\left(Q_{\text{ref}}(s,\mathbf{u}) - y^{\text{ref}}\right)^2\Big]. \end{aligned} \tag{14}$$

The parameter sets θ and ϕ are updated independently. The overall training procedure is summarized in Algorithm 1.

5 Experiments

In this section, we evaluate OJS on a one-step matrix game, Predator–Prey, the Highway-Env Intersection task and SMAC (Samvelyan et al., 2019). We compare OJS with QMIX, QPLEX, CW-QMIX, OW-QMIX, and POW-QMIX (Rashid et al., 2020b; Wang et al., 2021b; Rashid et al., 2020a; Huang et al., 2026). QMIX provides a monotonic value-factorization baseline, QPLEX represents the complete IGM function class, and CW-QMIX, OW-QMIX, and POW-QMIX provide weighted baselines. For a fair comparison, we keep all shared hyperparameters and optimizer settings identical across methods, while method-specific hyperparameters follow their original implementations. Unless otherwise stated, results are averaged over five random seeds and reported with 95% confidence intervals.

Table 1: Ground-truth payoffs and learned values on the one-step matrix game. Parentheses report individual utilities and boldface denotes decentralized greedy actions. Blue highlights the optimal joint action, while red highlights suboptimal joint actions caused by interference. OJS's primary QMIX and CW-QMIX recover the optimal joint action. The centralized estimator in OW-QMIX fits the payoff matrix but fails to correct OW-QMIX's decentralized greedy actions.

u_1 \ u_2	A	B	C
A	12	-12	-12
B	-12	11	11
C	-12	11	11

(a) Payoff Matrix

Q_1 \ Q_2	**A(0.6)**	B(0.5)	C(0.5)
A(0.5)	12.0	11.4	11.4
B(0.4)	11.6	11.0	11.0
C(0.4)	11.6	11.0	11.0

(b) OJS Q_{tot}

Q_1 \ Q_2	A(-607.4)	**B(-0.3)**	**C(-0.3)**
A(-607.4)	-7.3	-7.3	-7.3
B(-0.3)	-7.3	11.0	11.0
C(-0.3)	-7.3	11.0	11.0

(c) OJS Q_{ref}

Q_1 \ Q_2	**A(0.8)**	B(-1.6)	C(-1.6)
A(0.7)	12.0	-0.2	-0.2
B(-1.6)	-0.2	-0.2	-0.2
C(-1.6)	-0.2	-0.2	-0.2

(d) CW-QMIX Q_{tot}

Q_1 \ Q_2	A(-0.2)	**B(0.1)**	**C(0.1)**
A(-0.2)	4.7	4.7	4.7
B(0.1)	4.7	11.0	11.0
C(0.1)	4.7	11.0	11.0

(e) OW-QMIX Q_{tot}

Q_1 \ Q_2	A(-1.7)	**B(0.8)**	**C(0.8)**
A(-1.7)	12.0	-12.0	-12.0
B(0.8)	-12.0	11.0	11.0
C(0.8)	-12.0	11.0	11.0

(f) OW-QMIX Centralized Q

Q_1 \ Q_2	A(-606.6)	**B(0.1)**	**C(0.1)**
A(-606.7)	-6.8	-6.8	-6.8
B(0.1)	-6.8	11.0	11.0
C(0.1)	-6.8	11.0	11.0

(g) QMIX Q_{tot}

Q_1 \ Q_2	A(2.3)	**B(5.1)**	C(4.4)
A(2.3)	10.9	-18.5	-18.5
B(5.1)	-18.5	10.9	10.9
C(4.4)	-18.5	10.9	10.9

(h) QPLEX Q_{tot}

Q_1 \ Q_2	A(-10.7)	**B(0.8)**	**C(0.8)**
A(-10.4)	-7.5	-7.5	-7.5
B(0.8)	-7.5	10.7	10.7
C(0.8)	-7.5	10.7	10.7

(i) POW-QMIX Q_{tot}

5.1 MATRIX GAME

We first consider the one-step matrix game in Figure 1(a), which isolates interference from suboptimal joint actions without temporal credit assignment. As shown in Table 1, OJS correctly recovers (A, A) as the greedy joint action, while QMIX, OW-QMIX, and POW-QMIX converge to suboptimal solutions. CW-QMIX also recovers the optimum in this simple setting. These results show that appropriate sample weighting can effectively mitigate the interference described in Section 3.

The learned values are also consistent with the projection behavior in Proposition 1. The maximum value of the reference QMIX, which is trained with the unweighted objective, matches the true value of one of its greedy joint actions. OJS exploits this property through the reference QMIX to construct the screening threshold, providing empirical support for the practical effectiveness of the proposed weighting mechanism.

5.2 PREDATOR–PREY

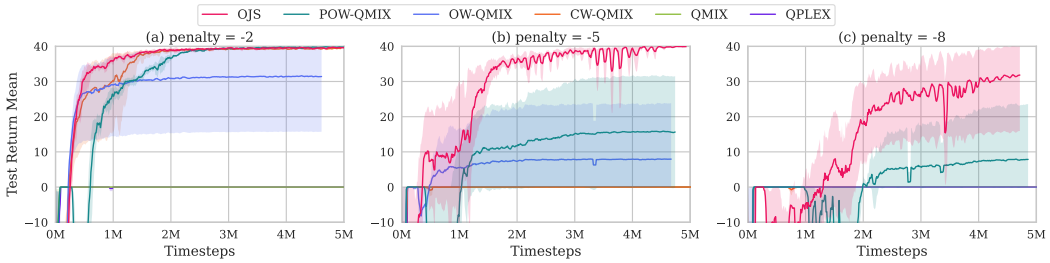


Figure 3: Performance comparison of OJS and baseline methods on Predator–Prey under different penalty settings. Subfigures (a)–(c) report the mean test returns for penalties -2, -5, and -8, respectively. Solid lines denote mean test returns, and shaded regions indicate 95% confidence intervals.

The Predator–Prey task requires agents to coordinate capture actions under partial observability. Two neighboring agents capture a prey by executing the capture action simultaneously and receive a reward of $+10$. In contrast, an isolated capture attempt incurs a penalty $p \in \{-2, -5, -8\}$. Larger $|p|$ increases the cost of miscoordination and makes coordinated exploration more difficult. As shown in Figure 3, QMIX and QPLEX fail to achieve positive returns across all settings, as frequent failed capture attempts can dominate the sparse successful ones. Weighted methods alleviate this issue, but their performance degrades as the penalty increases. In contrast, OJS preserves candidate actions through theoretically grounded screening and consistently maintains stronger performance as the miscoordination penalty increases.

5.3 HIGHWAY-ENV INTERSECTION

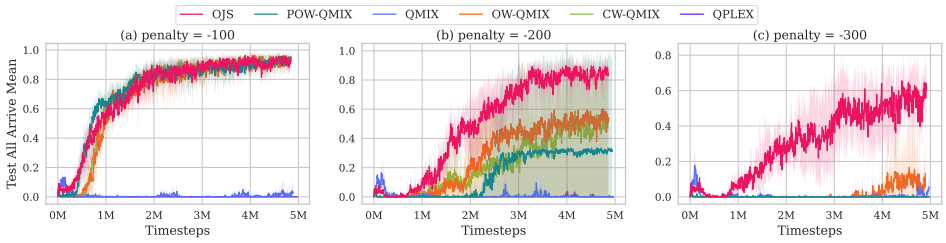


Figure 4: Performance comparison on the Highway-Env Intersection task with collision penalties of -100, -200, and -300.

We further evaluate OJS on the Highway-Env Intersection task, another representative setting of interference from suboptimal joint actions. The team receives a reward of -1 at each step, while a collision terminates the episode with an additional penalty and a successful passage yields a terminal

reward of $+100$. This setting is more challenging than Predator–Prey because positive feedback is only obtained at the end of a successful episode. We consider collision penalties of -100, -200, and -300. As shown in Figure 4, QMIX and QPLEX achieve near-zero success rates across all settings, whereas OJS maintains relatively stable performance as the penalty increases. This result further supports the effectiveness of OJS under sparse rewards and strong miscoordination penalties.

5.4 SMAC

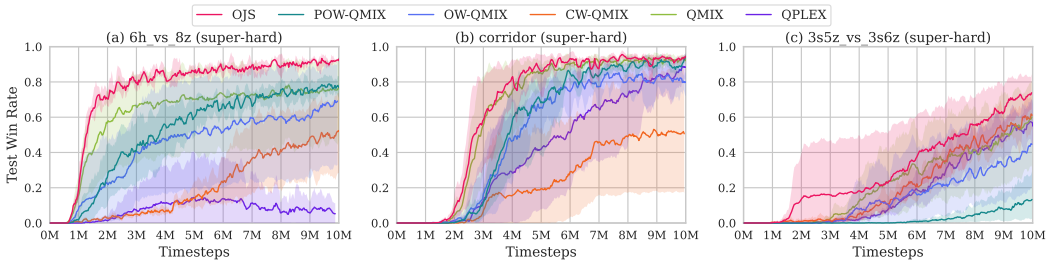


Figure 5: Performance comparison on three challenging SMAC scenarios.

The StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019) provides challenging cooperative tasks that require long-horizon coordination under partial observability. Unlike the preceding environments, these tasks are not specifically designed to exhibit severe interference from suboptimal joint actions, and standard QMIX can often match or outperform existing weighted variants. As shown in Figure 5, OJS remains competitive across the three challenging scenarios and achieves modest improvements on several tasks. Although successful coordination trajectories are relatively sparse among the large number of unsuccessful experiences, the proposed screening mechanism preserves their learning influence while reducing interference from less informative samples. These results suggest that OJS can provide additional benefits even when suboptimal-action interference is not the dominant source of difficulty. Additional SMAC and SMACv2 results are provided in Appendix E.

5.5 Hyperparameter Analysis

Table 2: Sensitivity of OJS to $w_{\min}$ on Predator–Prey and Highway-Env Intersection. Entries report the mean and the 95% confidence-interval half-width. Boldface marks the highest mean in each column.

(a) Predator–Prey mean test return

$w_{\min}$	$p=-2$	$p=-5$	$p=-8$
0	39.53 ± 0.56	$\mathbf{39.89 \pm 0.18}$	$\mathbf{31.67 \pm 21.99}$
0.02	39.28 ± 1.22	23.24 ± 26.37	16.00 ± 27.20
0.05	$\mathbf{39.57 \pm 0.58}$	7.89 ± 21.90	0.00 ± 0.00
0.1	31.82 ± 22.09	0.00 ± 0.00	0.00 ± 0.00

(b) Highway-Env Intersection success rate

$w_{\min}$	$p=-100$	$p=-200$	$p=-300$
0	0.778 ± 0.102	0.650 ± 0.164	$\mathbf{0.511 \pm 0.232}$
0.2	0.896 ± 0.108	$\mathbf{0.893 \pm 0.089}$	0.499 ± 0.276
0.5	0.915 ± 0.045	0.864 ± 0.102	0.380 ± 0.438
0.9	$\mathbf{0.932 \pm 0.038}$	0.798 ± 0.173	0.342 ± 0.328

The minimum weight $w_{\min}$ controls how strongly OJS suppresses samples rejected by the screening criterion. Table 2(a) shows that Predator–Prey is relatively insensitive to small values of $w_{\min}$ when $p = -2$, whereas stronger penalties increasingly favor $w_{\min} = 0$. Even a small positive floor substantially degrades performance for $p = -5$ and $p = -8$. In Intersection, a positive floor performs best under the milder penalties, with $w_{\min} = 0.9$ at -100 and $w_{\min} = 0.2$ at -200, while $w_{\min} = 0$ achieves the highest mean under the most severe penalty of -300. Overall, stronger interference favors more aggressive downweighting, whereas retaining part of the learning signal is beneficial when rejected samples may still provide useful corrections.

5.6 Ablation Study

Figure 6 validates both components of OJS. Without decoupled screening, learning collapses to zero return under every penalty, showing that an independently optimized reference QMIX is essential

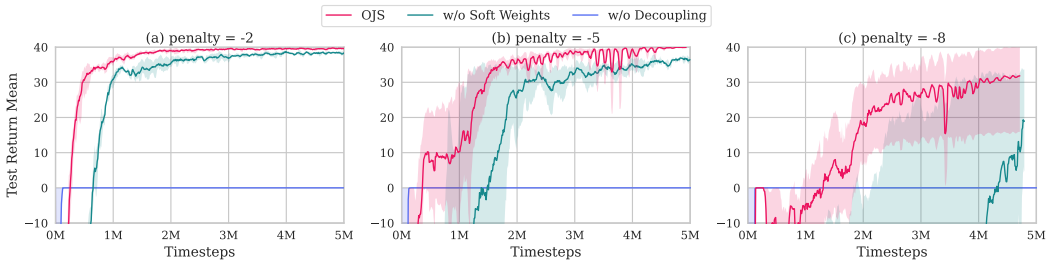


Figure 6: Component ablations on Predator–Prey with miscoordination penalties $p = -2$, $p = -5$, and $p = -8$. "w/o Soft Weights" assigns zero weight to samples rejected by screening, while "w/o Decoupling" uses the primary QMIX itself for screening. Solid lines show the mean test return, and shaded regions show 95% confidence intervals.

for preventing the screening criterion from being distorted by the weighted learner. Removing soft weighting remains competitive for $p = -2$ but increasingly degrades performance as the penalty grows, reducing the final mean return from 39.89 to 36.36 at $p = -5$ and from 31.67 to 15.37 at $p = -8$. Thus, soft weighting becomes particularly important under strong interference, where completely discarding rejected samples removes useful corrective updates. Final statistics are reported in Table 3.

6 Related Work

Value Function Factorization. Value function factorization decomposes a centralized joint value into individual utilities under CTDE. VDN uses an additive decomposition (Sunehag et al., 2018), whereas QMIX employs a monotonic nonlinear mixer (Rashid et al., 2020b) and Qatten models agent contributions with non-negative attention (Yang et al., 2020). These structures ensure IGM but restrict representational expressiveness. QTRAN, QPLEX, and ResQ improve expressiveness through auxiliary constraints, advantage decomposition, or residual learning (Son et al., 2019; Wang et al., 2021b; Shen et al., 2022). However, they do not prevent poorly coordinated joint actions from suppressing individual actions shared with optimal joint actions (Hou et al., 2026; Wan et al., 2022; Li et al., 2024), which motivates sample-dependent weighting.

Weighted Value Factorization. Weighted value factorization regulates the influence of joint-action samples during training. WQMIX formulates monotonic value learning as a projection and introduces CW-QMIX and OW-QMIX (Rashid et al., 2020a). OW-QMIX can emphasize underestimated yet suboptimal actions, whereas CW-QMIX depends on an unrestricted centralized estimator that may be inaccurate. GVR emphasizes superior experiences through target shaping and weighted replay (Wan et al., 2022), GPVD downweights interfering samples (Hou et al., 2026), and POW identifies potentially optimal actions with a learned recognizer (Huang et al., 2026). These criteria remain vulnerable to estimation or recognition errors. OJS instead derives its screening criterion from the projected maximum and guarantees optimal-action containment under the projection assumptions.

7 Conclusion

In this work, we investigated how abundant suboptimal or miscoordinated joint actions hinder coordination in monotonic value factorization. Although weighted value factorization methods attempt to alleviate this problem by emphasizing potentially optimal samples, their heuristic criteria may underweight optimal joint actions. To address this issue, we proposed OJS, motivated by a theoretically grounded screening threshold whose ideal candidate set is guaranteed to contain all optimal joint actions. OJS assigns unit weight to candidate samples and adaptively soft-weights the remaining samples according to their potential interference with optimal-action learning. Experiments across diverse cooperative benchmarks demonstrate that OJS achieves competitive or superior performance compared with representative value-factorization baselines, highlighting the importance of reliable optimal-action screening in cooperative multi-agent learning.

AI USE STATEMENT

Generative AI tools were used to aid and polish the writing of this paper, including improvements to clarity, grammar, organization, and presentation. The authors reviewed and revised all AI-assisted text and independently verified the technical claims, equations, experimental results, and citations. The authors take full responsibility for the final content of this work.

REFERENCES

Tarun Gupta, Anuj Mahajan, Bei Peng, Wendelin Boehmer, and Shimon Whiteson. Uneven: Universal value exploration for multi-agent reinforcement learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3930–3941. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/gupta21a.html.

Jie Hou, Haowen Dou, Lujuan Dang, Liangjun Chen, and Chenyang Ge. Gradient-protected value decomposition for cooperative multi-agent reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(26):21779–21787, Mar. 2026. doi: 10.1609/aaai.v40i26.39329. URL https://ojs.aaai.org/index.php/AAAI/article/view/39329.

Pierce Howell, Max Rudolph, Reza Joseph Torbati, Kevin Fu, and Harish Ravichandar. Generalization of heterogeneous multi-robot policies via awareness and communication of capabilities. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=N3VbFUpwaa.

Jian Hu, Siyang Jiang, Seth Austin Harding, Haibin Wu, and Shih wei Liao. Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning, 2023. URL https://arxiv.org/abs/2102.03479.

Chang Huang, Shatong Zhu, Junqiao Zhao, Hongtu Zhou, Hai Zhang, Di Zhang, Chen Ye, Ziqiao Wang, and Guang Chen. Potentially optimal joint actions recognition for cooperative multi-agent reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=YQ1muQBDV4.

Chao Li, Yupeng Zhang, Jianqi Wang, Yujing Hu, Shaokang Dong, Wenbin Li, Tangjie Lv, Changjie Fan, and Yang Gao. Optimistic value instructors for cooperative multi-agent reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 17453–17460, 2024.

Yihong Li, Xiaoxi Zhang, Tianyu Zeng, Jingpu Duan, Chuan Wu, Di Wu, and Xu Chen. Task placement and resource allocation for edge machine learning: A gnn-based multi-agent reinforcement learning paradigm. *IEEE Transactions on Parallel and Distributed Systems*, 34(12):3073–3089, 2023. doi: 10.1109/TPDS.2023.3313779.

Yueheng Li, Guangming Xie, and Zongqing Lu. Revisiting cooperative off-policy multi-agent reinforcement learning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=JPkJAyutW0.

Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. Maven: Multi-agent variational exploration. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/f816dc0acface7498e10496222e9db10-Paper.pdf.

Frans A Oliehoek and Christopher Amato. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

Afshin Oroojlooy and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, 53(11):13677–13722, 2023. doi: 10.1007/S10489-022-04105-Y. URL https://doi.org/10.1007/s10489-022-04105-y.

Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 10199–10210. Curran Associates, Inc., 2020a. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/73a427badebe0e32caa2e1fc7530b7f3-Paper.pdf.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020b. URL http://jmlr.org/papers/v21/20-081.html.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge, 2019. URL https://arxiv.org/abs/1902.04043.

Siqi Shen, Mengwei Qiu, Jun Liu, Weiquan Liu, Yongquan Fu, Xinwang Liu, and Cheng Wang. Resq: A residual q function-based approach for multi-agent reinforcement learning value factorization. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 5471–5483. Curran Associates, Inc., 2022. doi: 10.52202/068431-0395. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/2456a42386e445ba884511aa17ca4a30-Paper-Conference.pdf.

Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5887–5896. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/son19a.html.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius 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 *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, pp. 2085–2087, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.

Lipeng Wan, Zeyang Liu, Xingyu Chen, Xuguang Lan, and Nanning Zheng. Greedy based value representation for optimal coordination in multi-agent reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22512–22535. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/wan22c.html.

Jianhao Wang, Zhizhou Ren, Beining Han, Jianing Ye, and Chongjie Zhang. Towards understanding cooperative multi-agent q-learning with value factorization. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 29142–29155. Curran Associates, Inc., 2021a. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/f3f1fa1e4348bfbebdeee8c80a04c3b9-Paper.pdf.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. QPLEX: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=Rcmk0xxIQV.

Xiyang Wu, Rohan Chandra, Tianrui Guan, Amrit Bedi, and Dinesh Manocha. Intent-aware planning in heterogeneous traffic via distributed multi-agent reinforcement learning. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 446–477. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/wu23b.html.

Qinyu Xu, Yuanyang Zhu, Xuefei Wu, and Chunlin Chen. High-order interactions modeling for interpretable multi-agent q-learning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 156614–156646. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/e5a089875f7da17aa0d9f676bd5ef93f-Paper-Conference.pdf.

Yaodong Yang, Jianye Hao, Ben Liao, Kun Shao, Guangyong Chen, Wulong Liu, and Hongyao Tang. Qatten: A general framework for cooperative multiagent reinforcement learning, 2020. URL https://arxiv.org/abs/2002.03939.

Lulu Zheng, Jiarui Chen, Jianhao Wang, Jiamin He, Yujing Hu, Yingfeng Chen, Changjie Fan, Yang Gao, and Chongjie Zhang. Episodic multi-agent reinforcement learning with curiosity-driven exploration. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 3757–3769. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/1e8ca836c962598551882e689265c1c5-Paper.pdf.

Hanhan Zhou, Tian Lan, and Vaneet Aggarwal. Pac: Assisted value factorization with counterfactual predictions in multi-agent reinforcement learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 15757–15769. Curran Associates, Inc., 2022. doi: 10.52202/068431-1146. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/65338cfb603d4871a2c38e53a3e039c9-Paper-Conference.pdf.

A Detailed Analysis of Existing Weighting Methods

We construct a representative one-step matrix game that induces strong interference from suboptimal joint actions, whose ground-truth payoff matrix is shown in Figure 1(a). The converged joint-action values recovered by different methods are reported in Table 1. We next analyze why existing methods remain vulnerable to such interference.

As shown in Table 1(g), QMIX is highly susceptible to interference from suboptimal joint actions because all samples are treated uniformly. QPLEX decomposes the joint value as $Q_{\mathrm{tot}}(\boldsymbol{\tau}, \mathbf{u}) = \sum_{i=1}^{N} \big(Q_i(\boldsymbol{\tau}, u_i) + (\lambda_i(\boldsymbol{\tau}, \mathbf{u}) - 1) A_i(\boldsymbol{\tau}, u_i)\big)$, thereby improving its representational capacity. However, since the advantage terms are detached during backpropagation, the gradient of Q_{tot} with respect to each selected individual utility remains nonnegative. Therefore, QPLEX does not eliminate interference from suboptimal joint actions, as shown in Table 1(h).

Weighting can alleviate this problem, but its effectiveness critically depends on the accurate identification of potentially optimal samples. OW-QMIX assigns weights according to

$$w^{\mathrm{OW}} = \begin{cases} 1, & Q_{\mathrm{tot}} < y, \\ \alpha, & \text{otherwise}, \end{cases} \tag{15}$$

where $0 < \alpha < 1$. This rule gives unit weight to samples whose current predictions are below their targets and attenuates updates in the opposite direction. For a scalar prediction q and a random target Y, its asymmetric regression preference can be characterized through

$$\mathcal{J}_\alpha(q) = \mathbb{E}\left[\mathbb{I}(Y > q)(Y - q)^2 + \alpha\, \mathbb{I}(Y \le q)(Y - q)^2\right]. \tag{16}$$

Any stationary minimizer q_α satisfies

$$\mathbb{E}\left[(Y - q_\alpha)_+\right] = \alpha\, \mathbb{E}\left[(q_\alpha - Y)_+\right], \tag{17}$$

where $(x)_+ \triangleq \max(x, 0)$. Hence, q_α is the expectile associated with

$$\tau_\alpha = \frac{1}{1 + \alpha} > \frac{1}{2}. \tag{18}$$

OW-QMIX therefore regresses toward an upper expectile rather than the conditional mean of its targets. However, underestimation relative to a bootstrapped target does not imply joint-action

optimality. Underestimated suboptimal or miscoordinated joint actions may also receive unit weight, allowing their gradients to interfere with the utilities of potentially optimal actions. In Table 1(e), this behavior causes OW-QMIX to favor the joint actions formed by B and C rather than (A, A).

POW-QMIX uses a learned recognizer to identify potentially optimal joint actions, making its weighting vulnerable to recognition errors, especially when optimal experiences are sparse, as shown in Table 1(i). CW-QMIX recovers the optimal joint action in this example, but its weighting relies on an auxiliary critic and an approximate greedy-action reference, making it sensitive to estimation errors and unable to guarantee retention of all optimal joint actions.

These results indicate that weighting alone is insufficient. The key challenge is to determine which samples can be safely downweighted without suppressing optimal actions. OJS assigns unit weight to candidate samples and adaptive soft weights to the remaining samples according to their interference magnitude with optimal-action learning. Its screening threshold is derived from the least-squares QMIX projection, ensuring that all optimal joint actions are retained in the candidate set. As shown in Table 1(b), OJS recovers the optimal joint action.

B Relationship to Related Work

Representation and optimization. VDN and QMIX impose additive and monotonic factorizations, respectively, to support decentralized greedy action selection (Sunehag et al., 2018; Rashid et al., 2020b). QPLEX instead uses a duplex dueling decomposition and is claimed to represent the complete IGM function class (Wang et al., 2021b). OJS focuses on a different aspect of learning by controlling how strongly different sampled joint actions contribute to the regression objective. It retains the monotonic mixing structure of QMIX and modifies only the training weights without enlarging the representable function class.

Relationship to weighted QMIX. WQMIX introduces sample-dependent weighting to mitigate the limitations of monotonic value factorization (Rashid et al., 2020a). OW-QMIX gives unit weight to underestimated samples, but underestimation does not imply optimality. As shown in Appendix C, its asymmetric regression instead biases the solution toward an upper expectile, so suboptimal actions may still be overemphasized. CW-QMIX uses an unrestricted centralized estimator to assess potentially optimal actions, but its criterion remains sensitive to estimation errors during learning. In contrast, OJS uses the projected maximum of an independently trained reference QMIX with an unweighted objective as the screening reference. Corollary 1 establishes the projected maximum as a lower bound on the optimal joint-action value, providing the theoretical basis for using the reference maximum as a screening threshold.

Relationship to potentially optimal action recognition. POW learns an additional recognizer to identify potentially optimal joint actions and uses its predictions for sample weighting (Huang et al., 2026). Its effectiveness therefore depends on both the accuracy of the learned recognizer and an additional hyperparameter that determines which predictions are treated as potentially optimal. In contrast, OJS constructs the candidate set directly from the projected maximum of the reference QMIX, which is trained with an unweighted objective, avoiding a separate optimal-action recognition model. This yields a simpler and theoretically grounded screening criterion with guaranteed optimal-action containment under the projection assumptions.

C Expectile Characterization of OW-QMIX

This section characterizes the asymmetric regression objective induced by OW-QMIX from the perspective of expectile regression. Let Y denote a random bootstrapped target satisfying $\mathbb{E}[Y^2] < \infty$, and let q be a scalar prediction. The asymmetric objective induced by OW-QMIX is

$$\mathcal{J}_\alpha(q) = \mathbb{E}\left[\mathbb{I}(Y > q)(Y - q)^2 + \alpha\mathbb{I}(Y \le q)(Y - q)^2\right], \qquad 0 < \alpha < 1. \tag{19}$$

Differentiating the objective with respect to q gives

$$\frac{\partial \mathcal{J}_\alpha(q)}{\partial q} = -2\mathbb{E}\left[(Y - q)\mathbb{I}(Y > q)\right] + 2\alpha\mathbb{E}\left[(q - Y)\mathbb{I}(Y \le q)\right]. \tag{20}$$

Let q_α denote the minimizer of $\mathcal{J}_\alpha(q)$. It satisfies the stationary condition

$$\mathbb{E}\left[(Y - q_\alpha)\mathbb{I}(Y > q_\alpha)\right] = \alpha\mathbb{E}\left[(q_\alpha - Y)\mathbb{I}(Y \le q_\alpha)\right]. \tag{21}$$

Equivalently,

$$\mathbb{E}\left[(Y - q_\alpha)_+\right] = \alpha\mathbb{E}\left[(q_\alpha - Y)_+\right], \tag{22}$$

where $(x)_+ = \max(x, 0)$. The τ-expectile is defined as the minimizer of

$$\mathcal{J}_\tau^{\exp}(q) = \mathbb{E}\left[\tau\mathbb{I}(Y > q)(Y - q)^2 + (1 - \tau)\mathbb{I}(Y \le q)(Y - q)^2\right]. \tag{23}$$

Matching the relative weights of the two objectives yields

$$\alpha = \frac{1 - \tau}{\tau}, \tag{24}$$

and therefore

$$\tau_\alpha = \frac{1}{1 + \alpha}. \tag{25}$$

Since $0 < \alpha < 1$, we have $\tau_\alpha > 1/2$. Hence, OW-QMIX performs regression toward an upper expectile rather than the conditional mean of the target distribution.

D Proofs for Optimal Joint-Action Screening

The results in this section are direct special cases of Propositions 1 and 2 in Rashid et al. (2020a), obtained by setting the projection weight to $w(s, \mathbf{u}) \equiv 1$. We restate them in our notation because they are used in the subsequent analysis.

For a fixed state $s \in \mathcal{S}$, define the unweighted QMIX projection as

$$\Pi_{\mathcal{Q}_{\mathrm{mix}}} Q \coloneqq \arg\min_{q \in \mathcal{Q}_{\mathrm{mix}}} \sum_{\mathbf{u} \in \mathcal{U}} \left(Q(s, \mathbf{u}) - q(s, \mathbf{u})\right)^2. \tag{26}$$

Since the minimizer need not be unique, we interpret $\Pi_{\mathcal{Q}_{\mathrm{mix}}} Q$ as a set-valued projection and write $Q_{\mathrm{tot}} \in \Pi_{\mathcal{Q}_{\mathrm{mix}}} Q$ for an arbitrary minimizer.

Proposition 1. *Let $Q_{\mathrm{tot}} \in \Pi_{\mathcal{Q}_{\mathrm{mix}}} Q$. Then, for every $s \in \mathcal{S}$, the following properties hold:*

1. *For every $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u}} Q_{\mathrm{tot}}(s, \mathbf{u})$, $Q_{\mathrm{tot}}(s, \hat{\mathbf{u}}) \ge Q(s, \hat{\mathbf{u}})$. If, additionally, $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u} \in \mathcal{U}} Q(s, \mathbf{u})$, then $Q_{\mathrm{tot}}(s, \hat{\mathbf{u}}) = Q(s, \hat{\mathbf{u}})$.*
2. *There exists at least one $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u}} Q_{\mathrm{tot}}(s, \mathbf{u})$ such that $Q_{\mathrm{tot}}(s, \hat{\mathbf{u}}) = Q(s, \hat{\mathbf{u}})$.*

Proof. The first statement is Proposition 1 of Rashid et al. (2020a) specialized to the uniform weighting $w(s, \mathbf{u}) \equiv 1$. The second statement follows directly from Proposition 2 of Rashid et al. (2020a) under the same specialization. □

We now use these projection properties to derive a screening criterion for optimal joint actions. Define the maximum values of Q_{tot} and Q as

$$\begin{aligned} M_{\mathrm{tot}}(s) &\coloneqq \max_{\mathbf{u} \in \mathcal{U}} Q_{\mathrm{tot}}(s, \mathbf{u}), \\ M(s) &\coloneqq \max_{\mathbf{u} \in \mathcal{U}} Q(s, \mathbf{u}). \end{aligned} \tag{27}$$

Part 2 of Proposition 1 immediately yields the following result.

Corollary 1. *Let $Q_{\mathrm{tot}} \in \Pi_{\mathcal{Q}_{\mathrm{mix}}} Q$. For every $s \in \mathcal{S}$, there exists at least one $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u} \in \mathcal{U}} Q_{\mathrm{tot}}(s, \mathbf{u})$ such that $M_{\mathrm{tot}}(s) = Q_{\mathrm{tot}}(s, \hat{\mathbf{u}}) = Q(s, \hat{\mathbf{u}})$. Consequently, $M_{\mathrm{tot}}(s) \le M(s)$.*

Proof. Fix an arbitrary state $s \in \mathcal{S}$. By Part 2 of Proposition 1, there exists $\hat{\mathbf{u}} \in \arg\max_{\mathbf{u} \in \mathcal{U}} Q_{\mathrm{tot}}(s, \mathbf{u})$ such that $Q_{\mathrm{tot}}(s, \hat{\mathbf{u}}) = Q(s, \hat{\mathbf{u}})$. Since $\hat{\mathbf{u}}$ maximizes $Q_{\mathrm{tot}}(s, \cdot)$, we have

$$M_{\mathrm{tot}}(s) = Q_{\mathrm{tot}}(s, \hat{\mathbf{u}}) = Q(s, \hat{\mathbf{u}}).$$

Moreover, by the definition of $M(s)$,

$$Q(s, \hat{\mathbf{u}}) \leq \max_{\mathbf{u} \in \mathcal{U}} Q(s, \mathbf{u}) = M(s).$$

Therefore,

$$M_{\text{tot}}(s) \leq M(s).$$

□

By Corollary 1, the projected maximum is a lower bound on the true maximum. This motivates retaining joint actions whose true values are no smaller than the projected maximum. Accordingly, define

$$\mathcal{C}(s) \coloneqq \{\mathbf{u} \in \mathcal{U} \,|\, Q(s, \mathbf{u}) \geq M_{\text{tot}}(s)\}. \tag{28}$$

Corollary 2. *Let $Q_{\text{tot}} \in \Pi_{\mathcal{Q}_{\text{mix}}} Q$. Then, for every $s \in \mathcal{S}$, $\arg\max_{\mathbf{u} \in \mathcal{U}} Q(s, \mathbf{u}) \subseteq \mathcal{C}(s)$.*

Proof. Fix an arbitrary state $s \in \mathcal{S}$ and let $\mathbf{u}^* \in \arg\max_{\mathbf{u} \in \mathcal{U}} Q(s, \mathbf{u})$. By definition,

$$Q(s, \mathbf{u}^*) = M(s).$$

Corollary 1 gives

$$M_{\text{tot}}(s) \leq M(s).$$

Therefore,

$$Q(s, \mathbf{u}^*) = M(s) \geq M_{\text{tot}}(s),$$

which implies

$$\mathbf{u}^* \in \mathcal{C}(s).$$

Since $\mathbf{u}^*$ was chosen arbitrarily from $\arg\max_{\mathbf{u}} Q(s, \mathbf{u})$, the desired set inclusion follows. □

E Additional Experimental Results

E.1 Sensitivity to $w_{\min}$

Figures 7 and 8 show the training dynamics corresponding to the final statistics in Table 2.

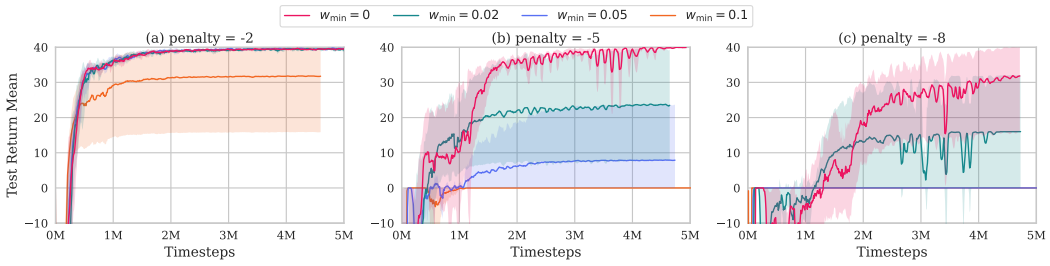


Figure 7: Training dynamics for different values of $w_{\min}$ on Predator–Prey with miscoordination penalties $p = -2$, $p = -5$, and $p = -8$. Solid lines show the mean test return, and shaded regions show 95% confidence intervals over the available random seeds.

The curves clarify that the preferred weight floor depends on the severity and structure of interference. In Predator–Prey, more aggressive downweighting is increasingly favored as the miscoordination penalty grows. Several values are competitive at $p = -2$, whereas $w_{\min} = 0$ is clearly favored at $p = -5$ and $p = -8$. Intersection instead benefits from retaining more rejected samples under penalties -100 and -200, while $w_{\min} = 0$ attains the highest final mean under the most severe penalty of -300. Variability also increases substantially in the hardest settings.

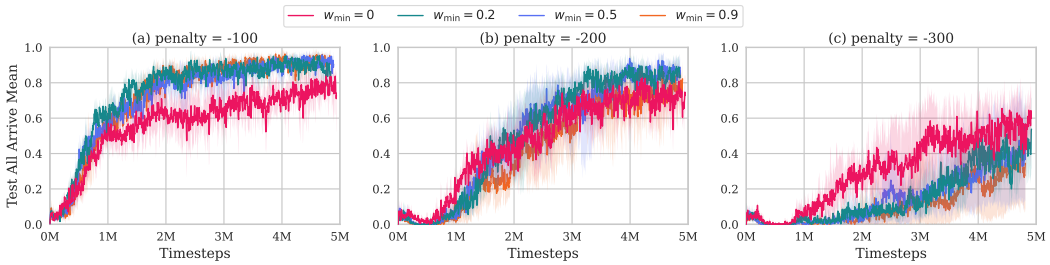


Figure 8: Training dynamics for different values of $w_{\min}$ on Highway-Env Intersection with collision penalties of -100, -200, and -300. Solid lines show the mean success rate, and shaded regions show 95% confidence intervals.

Table 3: Component ablations and baseline comparisons on Predator–Prey. Entries report the mean test return and the 95% confidence-interval half-width. Boldface marks the highest mean in each column.

Method	$p = -2$	$p = -5$	$p = -8$
OJS	39.57 ± 0.58	$\mathbf{39.89 \pm 0.18}$	$\mathbf{31.67 \pm 21.99}$
w/o Soft Weights	38.27 ± 0.87	36.36 ± 1.75	15.37 ± 32.05
w/o Decoupling	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
QMIX	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
QPLEX	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
CW-QMIX	39.42 ± 0.13	0.00 ± 0.00	0.00 ± 0.00
OW-QMIX	31.47 ± 21.84	7.94 ± 22.04	0.00 ± 0.00
POW-QMIX	$\mathbf{39.78 \pm 0.10}$	15.64 ± 26.95	7.72 ± 21.44

E.2 Component Ablations

Table 3 further separates the contributions of the two OJS components. Removing decoupled screening yields zero return under all three penalties, confirming that an independently trained reference QMIX is necessary to prevent the weighted primary network from reinforcing its own screening errors. Removing soft weighting causes a smaller degradation at $p = -2$, but the gap widens as the penalty increases. This result indicates that completely discarding non-candidate samples removes useful corrective information, particularly under severe miscoordination. Among the baselines, POW-QMIX and CW-QMIX remain competitive at $p = -2$, whereas OJS achieves the highest mean at $p = -5$ and $p = -8$. Although the confidence intervals are large in the hardest setting, the overall trend supports the complementary roles of decoupled screening and adaptive soft weighting.

E.3 Additional SMAC and SMACv2 Results

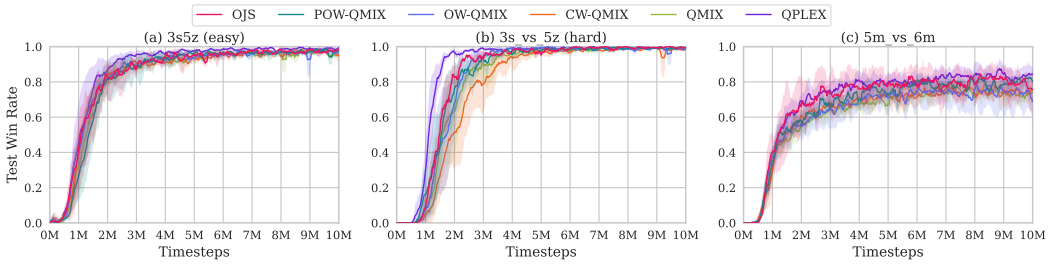


Figure 9: Performance comparison on three additional SMAC scenarios. The panels report the mean test win rate on `3s5z`, `3s_vs_5z`, and `5m_vs_6m`. Shaded regions show 95% confidence intervals over five random seeds.

Figure 9 complements the challenging scenarios reported in the main text. All methods approach near-perfect win rates on `3s5z` and `3s_vs_5z`, while performance differences are more visible on

5m_vs_6m. OJS remains competitive across all three scenarios and performs among the strongest methods on 5m_vs_6m.

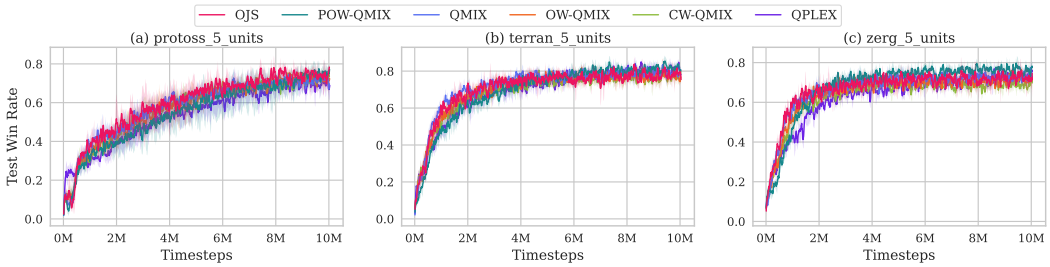


Figure 10: Performance comparison on the protoss_5_units, terran_5_units, and zerg_5_units SMACv2 distributions. Solid lines show the mean test win rate, and shaded regions show 95% confidence intervals over five random seeds.

SMACv2 evaluates coordination under procedurally generated unit compositions and initial configurations. Figure 10 shows broadly comparable performance across methods on all three race-specific distributions. OJS remains competitive throughout training without consistent performance degradation relative to the unweighted and weighted baselines. Together with the SMAC results, this finding indicates that OJS remains effective even when suboptimal-action interference is not the dominant source of difficulty.

F Implementation Details and Hyperparameters

F.1 Training Protocol

All methods use parameter sharing across agents and recurrent individual utility networks with hidden dimension 64. Unless stated otherwise, the mixing network has hidden dimension 32, and its hypernetworks have hidden dimension 64. The discount factor is 0.99, the learning rate is 10^{-3}, and gradients are clipped to a maximum norm of 10. Training uses eight parallel environments, a replay buffer of 5,000 episodes, and minibatches of 128 episodes. Target networks are updated every 200 collected episodes. Individual utilities condition on the local action–observation history, the previous action, and the agent identity. The previous-action input is omitted in the one-step matrix game.

Exploration follows a linear ϵ-greedy schedule from 1.0 to 0.05 over 5×10^5 environment steps. For Predator–Prey, the annealing horizon is shortened to 5×10^4 steps for all algorithms. Following the protocols of the original methods, ϵ remains at 1 throughout matrix-game training so that all nine joint actions continue to be sampled. Evaluation is greedy and masks unavailable actions.

F.2 Baseline-Specific Settings

The baseline hyperparameters follow the settings recommended in their original papers. All methods are trained using TD(λ) targets with $\lambda = 0.6$. In the one-step matrix game, every transition is terminal, so the target reduces to the observed shared payoff. In Predator–Prey, the miscoordination penalty p is varied across the conditions evaluated in Section 5 and is kept identical across methods within each condition.

QMIX uses a monotonic mixer with a 32-dimensional hidden layer and two-layer hypernetworks of width 64. It has no additional environment-specific hyperparameters beyond the common training protocol.

CW-QMIX and OW-QMIX use the same recurrent utilities and monotonic mixer as QMIX, together with their auxiliary centralized estimator. The central recurrent representation has dimension 64, and the centralized mixer has hidden dimension 256. The retained samples receive unit weight. Following their original settings, the default low weights are 0.75 for CW-QMIX and 0.5 for OW-QMIX. Both weights are set to 0.1 in the matrix game and Predator–Prey.

Table 4: Shared training hyperparameters. Replay and minibatch sizes are measured in episodes.

Parameter	Value
Discount factor γ	0.99
Learning rate	10^{-3}
Gradient norm clipping	10
Parallel rollout environments	8
Replay capacity	5,000 episodes
Training batch size	128 episodes
Recurrent hidden dimension	64
Mixing embedding dimension	32
Hypernetwork hidden dimension	64, where applicable
Exploration	Linear ϵ-greedy, $1.0 \rightarrow 0.05$
Default exploration annealing	500,000 environment steps
Hard target-network update interval	200 collected episodes

Table 5: Environment-dependent training and evaluation defaults. Counts refer to environment steps. The budget may be exceeded slightly because training stops at rollout boundaries.

Environment	Training budget	Test interval	Test episodes
SMAC	10,050,000	10,000	32
SMACv2 (each race)	10,050,000	10,000	32
Highway-Env Intersection	5,050,000	10,000	16
Predator–Prey	5,050,000	10,000	16
Matrix game	1,000,000	1,000	1

QPLEX uses an attention-based duplex dueling architecture with four advantage kernels, a one-layer advantage hypernetwork of width 64, four attention heads, state-dependent bias, and an attention regularization coefficient of 10^{-3}. It has no additional environment-specific hyperparameters.

POW-QMIX begins with an unweighted warm-up of 10^5 environment steps and then applies potentially optimal action recognition. Recognized and unrecognized samples receive weights 1 and 0, respectively. The recognition mixer has hidden dimension 64, its hypernetwork and value network have dimension 128, and its advantage module uses four kernels with a three-layer hypernetwork of width 64. The recognition threshold is 0.05 in the matrix game, SMAC, and SMACv2, 1 in Predator–Prey, and 0.1 in Highway-Env Intersection.

F.3 OJS-Specific Training Settings

The primary and reference QMIX networks use the same recurrent utility and mixing architectures but have independent parameters. Both networks are trained from the same replayed transitions, which are generated by the behavior policy of the primary network, and both use TD(λ) with $\lambda = 0.6$. The reference network constructs its own TD(λ) targets from its utility and mixing networks and their target copies, rather than reusing the targets of the primary network.

At the current state, the reference maximum is computed by the online reference QMIX. The reference network is optimized only with the unweighted TD objective, keeping the screening criterion independent of the soft weights applied to the primary network. We select $w_{\min}$ separately for each environment configuration using separate tuning runs and report the selected value in the main comparison. Specifically, we use $w_{\min} = 0$ in the matrix game. For Predator–Prey, we use $w_{\min} = 0.05$ for $p = -2$ and $w_{\min} = 0$ for $p = -5$ and $p = -8$. For Highway-Env Intersection, we use $w_{\min} = 0.9$, 0.2, and 0 for collision penalties of -100, -200, and -300, respectively. The evaluated values and sensitivity results for these two environments are reported in Table 2 and Appendix E. The selected value is $w_{\min} = 0.9$ for all SMAC scenarios and all three SMACv2 race distributions.

G Environment Details

G.1 Matrix Game

We use a one-step cooperative matrix game with two agents and three actions, $\{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$, per agent. The shared payoff is

	A	B	C
A	12	-12	-12
B	-12	11	11
C	-12	11	11

where rows and columns correspond to the actions of the first and second agents, respectively. All actions are available, both agents act simultaneously, and the episode terminates after this joint action. The local observations and centralized state are two-dimensional indicators of the single decision step and therefore contain no information that distinguishes the agents' actions before execution. The unique optimum is (A, A) with payoff 12, whereas joint actions composed of B and C yield 11. Pairing A with either B or C yields -12, so the individually optimal action A also occurs in strongly suboptimal joint actions. This structure directly isolates the interference studied in this work.

G.2 StarCraft Multi-Agent Challenge

SMAC evaluates decentralized control of allied combat units against the built-in StarCraft II opponent (Samvelyan et al., 2019). Each allied unit is an agent. Available actions include movement in four directions, stopping, and target-specific combat actions. Unavailable actions are masked, and dead units use a no-op. Agents observe local movement information and visible ally/enemy features, including health and relative geometry. The centralized state includes unit features and the agents' previous actions.

We use StarCraft II version 4.6.2 with opponent difficulty 7. Each agent decision advances eight game frames, and a movement action covers two spatial units. Rewards provide dense positive feedback for damage and unit deaths, with a death value of 10, a win bonus of 200, and no additional defeat penalty. Returns are normalized by the maximum attainable reward on each map and scaled by 20. Observations include own and visible units' health. Win rate is the principal combat evaluation measure.

These scenarios vary both team composition and numerical disadvantage. For example, `3s_vs_5z` pits three stalkers against five zealots, `6h_vs_8z` pits six hydralisks against eight zealots, and `corridor` pits six zealots against 24 zerglings. They expose different demands on coordinated targeting, movement, and survival.

G.3 SMACv2

SMACv2 introduces procedurally generated combat scenarios to test adaptation to changing teams and initial configurations. We evaluate three race-specific distributions, as listed in Table 7. Every episode contains five allied units and five enemies, whose unit types are sampled independently from the corresponding race distribution.

Initial positions are drawn from an equal mixture of surrounded and reflected formations on a 32×32 map. The episode limit is 400 decisions. Team capabilities, the agent's own position, and standard StarCraft unit features are observable. We do not provide the full global state as a local observation and do not impose a conic field of view. The remaining combat dynamics, reward design, and health features follow the SMAC setting above.

G.4 Highway-Env Intersection

The Highway-Env Intersection task controls four vehicles entering a four-way intersection, one from each approach. Initial longitudinal positions are randomized, and each vehicle receives a route to another approach. No background vehicles are spawned. Agents must coordinate their speeds to complete their routes without collisions. Steering follows the assigned route rather than a learned lateral action.

Table 6: SMAC scenarios used in our experiments.

Map	Agents	Enemies	Decision limit
3s_vs_5z	3	5	250
6h_vs_8z	6	8	150
corridor	6	24	400
3s5z_vs_3s6z	8	9	170
3s5z	8	8	150
5m_vs_6m	5	6	70

Table 7: SMACv2 unit-type sampling distributions. The weights specify sampling probabilities rather than fixed per-episode proportions. The exception type denotes the low-probability unit type sampled with probability 0.10 for each race.

Race	Unit types (sampling weights)	Exception type
Protoss	stalker (0.45), zealot (0.45), colossus (0.10)	colossus
Terran	marine (0.45), marauder (0.45), medivac (0.10)	medivac
Zerg	zergling (0.45), hydralisk (0.45), baneling (0.10)	baneling

Each agent selects one of three longitudinal meta-actions consisting of decelerate, maintain speed, and accelerate. These actions decrease, preserve, or increase a target-speed index over $\{0, 2, 4, 6, 8, 10\}$ m/s. Thus, the six target speeds define the vehicle dynamics rather than the action-set size. Both the simulator and the policies operate at 5 Hz, and an episode lasts at most 20 seconds, corresponding to 100 decisions.

Table 8: Observation, state, and reward specification of the local Highway-Env Intersection task. Dimensions exclude controller-added previous actions and agent identifiers.

Component	Specification
Local observation	Four nearby vehicles represented by $(x, y, v_x, v_y, \text{heading})$ and five route waypoints represented by (x, y), for a total of $4{\times}5{+}5{\times}2 = 30$ dimensions.
Feature processing	Coordinates are expressed relative to the observing vehicle. Vehicles are ordered by lane distance without normalization or clipping.
Central state	Each vehicle's 5 ego features and 10 route coordinates, giving $4(5 + 10) = 60$ dimensions.
Step reward	-1 until a terminal reward condition occurs.
Successful completion	$-1 + 100 = 99$ on the step when all vehicles have arrived.
Collision	A penalty in $\{-100, -200, -300\}$ if any controlled vehicle crashes. This reward overrides the arrival bonus.
Termination	Any collision, all vehicles arriving, or duration exhaustion.

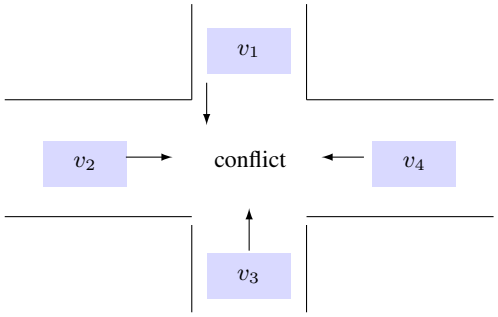
G.5 PREDATOR–PREY

This grid-based predator–prey task uses a bounded 10×10 grid with eight learning hunters and eight stags. Agents and prey are randomly initialized. Each hunter has six actions comprising four cardinal moves, staying still, and an explicit catch action. Movement into cells occupied by agents or prey is blocked. Prey move at every step according to the environment dynamics.

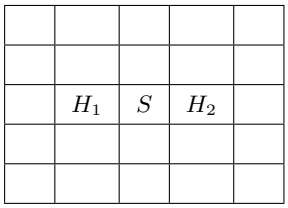
A stag is caught when at least two orthogonally adjacent hunters execute the catch action simultaneously. A lone attempt incurs the miscoordination penalty p. Each successful capture yields a shared reward of 10. There is no time penalty or agent-collision reward. Capturing hunters are frozen and removed, and the prey is removed. A single capture does not terminate the episode. Termination occurs when all prey are gone, all hunters are frozen, or the 200-step limit is reached.

Each hunter observes a centered 5×5 region with three entity channels, giving 75 raw observation features. The centralized state is the three-channel full grid, giving 300 features. Global observations at the agent level are disabled. Entity and agent identifiers are not appended to the observations.

We vary p as described in the main text and use the same value for all methods within each experimental condition. Negative p makes an uncoordinated catch costly even though the same individual catch action can participate in a successful joint capture, exposing the interference mechanism studied in this paper.



(a) Speed coordination at the intersection



(b) Local Predator–Prey capture example

Figure 11: Schematic illustrations of coordination requirements, not simulator screenshots. (a) Four vehicles adjust longitudinal speed before traversing the shared conflict region. Assigned routes may turn or go straight. (b) Two hunters adjacent to one stag must synchronize catch actions. The 5×5 crop illustrates the local spatial scale, not the complete 10×10 world.