Adaptive TD-Lambda for Cooperative Multi-agent Reinforcement Learning

Yue Deng[1] , **Zirui Wang**[2] and **Yin Zhang**[2] *
[1]Zhongguancun Academy, Beijing, China
[2]College of Computer Science and Technology, Zhejiang University, Zhejiang, China
dengyue@zgci.ac.cn, {ziseoiwong, zhangyin98}@zju.edu.cn

Abstract

TD(λ) in value-based MARL algorithms or the Temporal Difference critic learning in Actor-Critic-based (AC-based) algorithms synergistically integrate elements from Monte-Carlo simulation and Q function bootstrapping via dynamic programming, which effectively addresses the inherent bias-variance trade-off in value estimation. Based on that, some recent works link the adaptive λ value to the policy distribution in the single-agent reinforcement learning area. However, because of the large joint action space from multiple agents and the limited transition data in Multi-agent Reinforcement Learning, the policy distribution is infeasible to calculate statistically. To solve the policy distribution calculation problem in MARL settings, we employ a parametric likelihood-free density ratio estimator with two replay buffers instead of calculating statistically. The two replay buffers of different sizes respectively store the historical trajectories that represent the data distribution of the past and current policies. Based on the estimator, we assign Adaptive TD(λ), **ATD(λ)**, values to state-action pairs based on their likelihood under the stationary distribution of the current policy. We apply the proposed method on two competitive baseline methods, QMIX for value-based algorithms, and MAPPO for AC-based algorithms, over SMAC benchmarks and Gfootball academy scenarios, and demonstrate consistently competitive or superior performance compared to other baseline approaches with static λ values.

1 Introduction

Recent advances in Multi-agent reinforcement learning (MARL) have led to significant progress in a wide range of applications, such as autonomous vehicle teams [Cao *et al.*, 2012] and sensor networks [Zhang and Lesser, 2011]. Within the MARL landscape, various value-based approaches target enhancements in either value decomposition [Sunehag *et al.*, 2017; Rashid *et al.*, 2018; Wang *et al.*, 2020a] or cooperative exploration [Mahajan *et al.*, 2019; Yang *et al.*, 2020; Wang *et al.*, 2020b]. These prevalent methodologies involve the utilization of temporal difference (TD) updates for training the Q value function. Additionally, actor-critic methods, including [Foerster *et al.*, 2018; Yu *et al.*, 2022; Wang *et al.*, 2023; Wu *et al.*, 2026], have also exhibited outstanding performance across challenging tasks, such as StarCraft II [Samvelyan *et al.*, 2019] and Google Football Research [Kurach *et al.*, 2020], and the critic network can also be updated by TD or GAE methods.

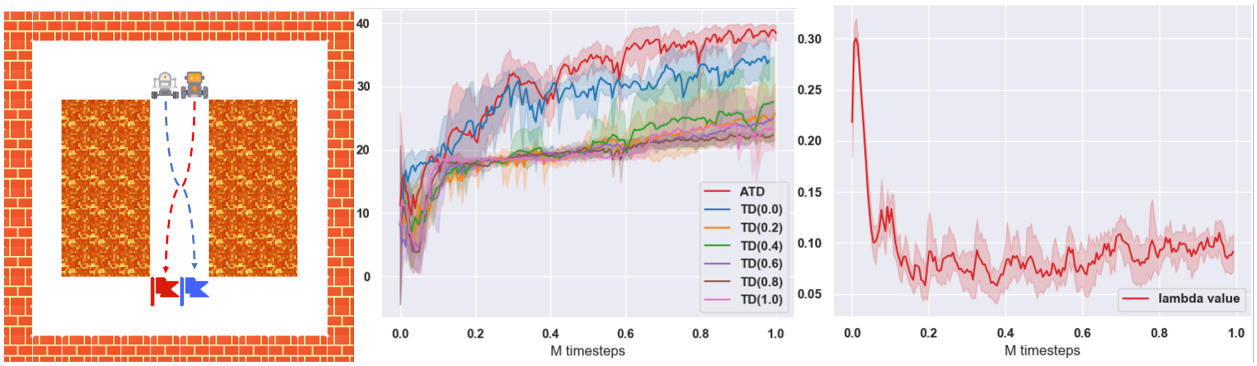
Nevertheless, TD learning confronts the challenge of overestimation bias stemming from function approximation [Cicek *et al.*, 2021], and Monte-Carlo methods introduce lower bias but exhibit larger variances [Sutton and Barto, 2018]. The large bias or the large variance may make the credit assignment process unstable in the training process. Therefore, the fundamental trade-off in MARL also lies in the definition of the update target: should one estimate Monte-Carlo returns or bootstrap from an existing Q-function [Seijen and Sutton, 2014]? To flexibly navigate this trade-off in value estimation, the incorporation of TD(λ) becomes crucial.

To demonstrate the significant impact of different λ values in the TD(λ) method on final performance, we conducted a toy experiment within a multi-agent lava-path scenario. Illustrated in Figure 1, the learning curve of the adaptive TD(λ) values surpasses those of fixed lambda values. This observation underscores the profound influence of well-chosen λ values in enhancing training performance while underscoring the potential detriment incurred by inappropriate λ values. This complexity in the relationship between λ values and performance makes the choice of λ values as hyper-parameters a challenging task.

Meanwhile, proper λ values vary intuitively based on the different training MARL processes. For example, higher λ values (almost MC) allow the target to better reflect real long-term returns, which accelerates the fitting and speeds up the stabilization of the joint value at the early training stage. In contrast, when the policy stabilizes, the replay samples from older policies become biased relative to the current policy from the middle to later training process. In such a way, smaller λ values reduce reliance on outdated MC returns and instead trust the increasingly accurate critic.

Based on that, we introduce the **ATD(λ)**, a novel approach

*Corresponding Author: Yin Zhang.



(a) (b) (c)

Figure 1: (a): Two agents are asked to reach their opposite goals within 60 steps without collision. Agents can choose from moving in four directions and a 'staying' action. Agents receive 40 marks when both of them reach their goals and -10 marks when they step into the lava. Otherwise, agents receive the marks of their distance to their goals subtracted by 40 after the time step limit. (b): The performance curves of different commonly-used preset TD(λ) values with adaptive λ values. The x-axis is the training time steps (e6), and the y-axis is the final performance. (c) The average adaptive lambda values during the training process.

for determining the λ value based on sampled transitions during training. Inspired by recent studies in density ratio calculation [Sinha *et al.*, 2022] and off-policy policy evaluation [Grover *et al.*, 2019], we employ a likelihood-free parametric network to simulate the density ratio for each state-action pair of a batch of sampled trajectories, and scale this ratio to serve as the adaptive λ values. This approach utilizes a large replay buffer to store off-policy trajectories for sampling and a much smaller replay buffer to store on-policy trajectories and estimates the degree of on-policiness adherence for sampled off-policy trajectory data by calculating the f-divergences between the two replay buffers by the parametric network.

The main contributions of this work are: **1)** We propose an MARL-specific formulation of an adaptive λ calculation method for each transition based on parametric likelihood-free off-policy estimation. **2)** We propose a training mechanism using two replay buffers to approximate density ratios between on- and off-policy distributions without explicit policy modeling and adapt that to existing value-based approaches and value-based critic AC algorithms with minor changes to existing MARL codebases. **3)** We describe the feasibility of our framework from theoretical perspectives and validate our methods empirically by extensive experiments on SMAC benchmarks and Gfootball academy tasks. Experimental results indicate that existing MARL methods equipped with ATD can compete with or outperform original MARL methods in terms of the winning rates or accumulated rewards.

2 Related Work

Multi-agent Reinforcement Learning: In multi-agent value-based algorithms, the centralized value function, joint Q-function, is decomposed into local utility functions. Many methods have been proposed to meet the Individual-Global-Maximum (IGM) [Son *et al.*, 2019] assumption, which indicates the consistency between the local optimal actions and the optimal global joint action. VDN [Sunehag *et al.*, 2017] and QMIX [Rashid *et al.*, 2018] introduce additivity and monotonicity to Q-functions. QTRAN [Son *et al.*, 2019] transforms IGM into optimization constraints. QPLEX [Wang *et al.*, 2020a] uses a duplex dueling network architecture to guarantee IGM assumption. Instead of focusing on value decomposition, multi-agent policy gradient algorithms provide a centralized value function to evaluate current joint policy and guide the update of each local utility network. Most policy-based MARL methods extend single-agent RL ideas, including MADDPG [Lowe *et al.*, 2017], MAPPO [Yu *et al.*, 2022]. FOP [Zhang *et al.*, 2021] algorithm factorizes optimal joint policy by maximum entropy and MACPF [Wang *et al.*, 2023] mixes critic values of each agent.

TD(λ) in MARL: Recent works on TD(λ) in MARL have explored the application and enhancement of TD(λ), addressing the challenges in centralized value functions and policy gradients. SMIX(λ) [Yao *et al.*, 2021] uses an off-policy training to achieve a stable centralized value function by avoiding the greedy assumption and connects the SMIX(λ) to Q(λ) [Peng and Williams, 1994]. ETD(λ) [Jiang *et al.*, 2021] ensures the convergence in the linear case by appropriately weighting TD(λ) updates. [Wang *et al.*, 2020c] explores off-policy multi-agent learning with decomposed policy gradients, incorporating TD(λ) methods for estimating the decomposed critic. [Li *et al.*, 2025] introduces a λ annealing mechanism and a λ^* threshold according to the training episodes. Importance sampling is the simplest way to correct for the discrepancy between behavior policy and target policy [Precup, 2000; Geist *et al.*, 2014], but suffers from large variance. Retrace[Munos *et al.*, 2016] is an off-policy reinforcement learning algorithm that uses truncated importance sampling with eligibility traces to enable safe and efficient value function updates from off-policy data. $Q^*(\lambda)$[Harutyunyan *et al.*, 2016] introduces an off-policy correction based on the Q-baseline which avoids the blow-up of the variance but does not guarantee convergence for arbitrary π and μ. Tree-backup algorithm [Precup, 2000] corrects the discrepancy by multi-

plying each term of the sum of the product of target policy probabilities, however, it is not efficient in the near on-policy case as it unnecessarily cuts the traces. Motivated by [Hu *et al.*, 2021], our method introduces a λ value calculation method based on the likelihood that the sampled transitions occur in the current policy during the training process.

3 Background

MARL modeling A fully cooperative multi-agent task is described as a Dec-POMDP [Oliehoek *et al.*, 2016] task which consists of a tuple $G = \langle S, A, P, r, Z, O, N, \gamma \rangle$ in which $s \in S$ is the global state of the environment and N is the number of agents. At each time step, each agent $i \in N \equiv \{1, \ldots, n\}$ chooses an action $a_i \in A$ which forms the joint action $\mathbf{a} \in \mathbf{A} \equiv A^N$. The transition on the environment is according to the state transition function that $P(\cdot|s, \mathbf{a}) : S \times \mathbf{A} \times S \to [0, 1]$. The reward function, $r(s, \mathbf{a}) : S \times A \to \mathbb{R}$, is shared among all the agents, and $\gamma \in [0, 1)$ is the discount factor for future reward penalty. Partially observable scenarios are considered in this paper, in which each agent draws individual observations $z_i \in Z$ of the environment according to the observation functions $O(s, i) : S \times N \to Z$. Meanwhile, the action-observation history, $\tau_i \in H \equiv (Z \times A)^*$, is preserved for each agent and conditions the stochastic policy $\pi_i(a_i|\tau_i) : H \times A \to [0, 1]$. In the Centralized Training with Decentralized Execution (CTDE) settings, the state is provided during the centralized training phase and the agents can only acquire partial observations during the decentralized execution phase.

MARL algorithms Value-based MARL algorithm aims to find the optimal joint action-value function $Q^*(s, \mathbf{a}; \theta) = r(s, \mathbf{a}) + \gamma \mathbb{E}_{s'} \left[\max_{\mathbf{a}'} Q^*(s', \mathbf{a}'; \theta)\right]$ and parameters θ are learned by minimizing the expected TD error. VDN learns a joint action-value function $Q_{tot}(\tau, \mathbf{a})$ as the sum of individual value functions: $Q_{tot}^{\text{VDN}}(\tau, \mathbf{a}) = \sum_{i=1}^{n} Q_i(\tau_i, a_i)$. QMIX introduces a monotonic restriction $\forall i \in N, \frac{\partial Q_{tot}^{\text{QMIX}}(\tau, \mathbf{a})}{\partial Q_i(\tau_i, a_i)} > 0$ to the mixing network to meet the IGM assumption. In policy-based algorithms, agents use a policy $\pi_\theta(a_i|\tau_i)$ parameterized by θ to produce an action a_i from the local observation and jointly optimize the discounted accumulated reward $J(\theta) = \mathbb{E}_{a^t, s^t}[\sum_t \gamma^t r(s^t, a^t)]$ where a^t is the joint action at time step t. In the AC-based algorithm, MAPPO algorithm, the actor is updated by optimizing the target function $J_{\theta^k}(\theta) = \sum_{s^t, a^t} \min(\frac{\pi_\theta(a^t|s^t)}{\pi_{\theta^k}(a^t|s^t)} A_{\theta^k}(s^t, a^t), clip(\frac{\pi_\theta(a^t|s^t)}{\pi_{\theta^k}(a^t|s^t)}, 1 - \epsilon, 1 + \epsilon) A_{\theta^k}(s^t, a^t))$, where the ϵ is the clip parameter and $A_{\theta^k}(s^t, a^t)$ is the advantage function. The critic training is similar to value-based Q learning by calculating TD-error and TD targets and the target value is calculated by bootstrapping from the existing Q-function according to temporal difference methods or Monte-Carlo returns.

TD(λ) Temporal-difference algorithms are based on the fact that the value function should satisfy Bellman equations for all s and the target is formulated as the regression target:

$$T_{TD(0)}(s_t, \mathbf{a_t}) := r(s_t, \mathbf{a_t}) + \gamma \hat{Q}(s_{t+1}, \mathbf{a_{t+1}}) \quad (1)$$

in which the T is the target value, the $\hat{Q}(s_{t+1}, \mathbf{a_{t+1}})$ is the estimated Q value in $t + 1$ time step and these algorithms are referred as TD(0). The Monte-Carlo approach is based on the intuition that the discounted sum of rewards realized by the policy from a state s_t is an unbiased estimator of $Q(s_t, \mathbf{a_t})$. The target value is calculated by:

$$T_{MC}(s_t, \mathbf{a_t}) = \sum_{k=0}^{n_i - t - 1} \gamma^t r(s_{t+k}, \mathbf{a_{t+k}}) \quad (2)$$

where the n_i is the length of the trajectory τ_i and these algorithms are also referred as TD(1).

Between the two extremes of TD(0) and TD(1), TD(λ) integrates the Temporal Difference method with Monte-Carlo methods by parameter λ:

$$T_t^\lambda = r(s_t, \mathbf{a_t}) + \gamma[\lambda T_{t+1}^\lambda + (1 - \lambda) \max_{\mathbf{a}' \in A} Q(s_{t+1}, \mathbf{a}')]. \quad (3)$$

The TD(λ) calculation is also related to the λ-return extension [Sutton, 1988] which considers the exponentially weighted sum of n-step returns to calculate Q values. The general return-based operator $\mathcal{R}$ [Munos *et al.*, 2016] is defined as:

$$\begin{aligned} \mathcal{R}Q(s, \mathbf{a}) :=& Q(s, \mathbf{a}) + \mathbb{E}_\mu[\sum_{t \geq 0} \gamma^t (\prod_{i=1}^{t} c_i) \\ & (r_t + \gamma \mathbb{E}_\pi Q(s_{t+1}, .) - Q(s_t, \mathbf{a}_t))] \end{aligned} \quad (4)$$

for some non-negative coefficients c_i, traces, where $\mathbb{E}_\pi Q(s, .) := \sum_{\mathbf{a}} \pi(\mathbf{a}|s) Q(s, \mathbf{a})$ and $c_0 = 1$.

According to the Bellman Equation, for a policy π, the Bellman operator $\mathcal{T}^\pi$ is defined as $\mathcal{T}^\pi Q := r + \gamma P^\pi Q$, and the Bellman optimality operator is $\mathcal{T} Q := r + \gamma \max_\pi P^\pi Q$, where the P^π operator is defined as $P^\pi Q(s, \mathbf{a}) := \sum_{s' \in S} \sum_{\mathbf{a}' \in A} P(s'|s, \mathbf{a}) \pi(\mathbf{a}'|s') Q(s', \mathbf{a}')$. In the policy evaluation setting, a fixed policy π is given whose value Q^π we wish to estimate from sample trajectories drawn from the behavior policy μ. In the rollout process, the policy depends on the sequences of Q-functions, such as ϵ-greedy policies, and seeks to approximate Q^*. Based on the notations above, the calculation of the off-policyness measurement is shown in the Method and the convergence analysis of $\mathcal{R}$ operator is shown in Appendix A.

4 Method

In this section, we introduce the overall architecture of our framework and the training details of the likelihood-free density ratio network. Our framework generates two replay buffers of different sizes, a large buffer for off-policy trajectories and a small buffer for on-policy trajectories. The λ-predictor network is trained adversarially by the transitions sampled from the two buffers. The network output is then used as the λ value to calculate the target Q values. Additionally, we provide the convergence justification of the adaptive λ method in Appendix A.

4.1 Architecture

The importance sampling between the behavior policy and the target policy according to the stability condition of the off-policy TD learning is currently the best method to adjust the data distribution [Sutton *et al.*, 2016; Jiang *et al.*, 2021].

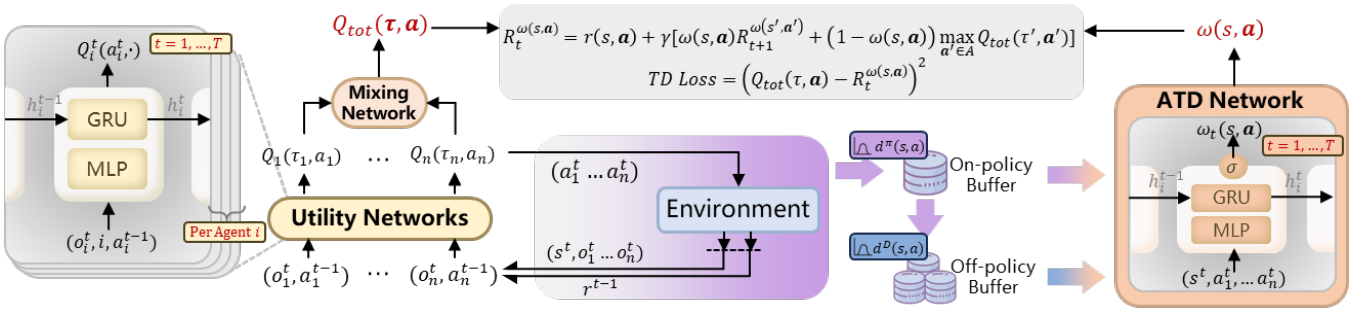


Figure 2: The utility networks and the mixing network are from the original MARL algorithms, QMIX in this paper. Interactive transitions are stored in two replay buffers. One of them is small (on-policy buffer) and the other one is large (off-policy buffer). Training data are sampled uniformly from these two buffers and used for calculating the likelihood-free density ratios. The density ratios are used as the λ values and importance weights during the training process.

Meanwhile, given a fixed target policy π and behavior policy μ, and a set of non-negative coefficients $c_t = \omega(\mathbf{a}_t, \tau_t)$ under the assumption that $0 \leq c_s \leq \frac{\pi(\mathbf{a}|s)}{\mu(\mathbf{a}|s)} \leq 1$, the use of importance sampling, the operator $\mathcal{R}$ is γ-contraction [Munos *et al.*, 2016]. Moreover, the coefficients of Equation 4 are state-action specific, so that the coefficients can be represented by a parametric network conditioned on state-action pairs.

The training process of value-based MARL algorithms is the Q value Temporal Difference (TD) updating of each agent's utility network. In QMIX and the algorithms derived from QMIX, TD updates are applied to the mixed Q_{tot} value. The utility network is composed of multi-layer perceptron (MLP) layers and Gate Recurrent Unit (GRU) cells in which h_i^t is the historical hidden state. Similar to the QMIX algorithm, the utility network at time step t of agent i takes the observation o_i^t and its chosen action a_i^t as an input and outputs the $Q_i(\tau_i, a_i)$ of each agent according to the encoded history state τ_i. Then, these Q values are used for subsequent mixing mechanisms, QMIX and QPLEX for example, and trained by TD learning.

As shown in Figure 2, the interaction trajectories with the environments are stored in the original replay buffer (off-policy buffer) and a small on-policy buffer. The on-policy buffer is refreshed faster than the off-policy buffer, thus the transitions are more of an on-policy property. The new interactive trajectories are inserted into the fast buffer and the slow buffer. We then sample trajectories from both the replay buffers and train a network that takes state-action $(s, \mathbf{a})$ as input to calculate the on-policy density ratio. The ATD network is also composed of a multi-layer perceptron (MLP) layer and Gate Recurrent Unit (GRU) cells. The results are then activated by a sigmoid layer to be scaled in the range between 0 and 1. During the centralized training process, the global states are available so the recurrent layer can be masked because of the Markov property. As for the algorithms without centralized learning, the recurrent layer encodes the history of observations and represents the latent state distribution. The ATD network takes the observation and the action of each agent as the input and provides the λ values for each agent. Finally, the λ values are used for calculating the eligibility trace and participating in the TD error calculation and the ATD network is updated with the same frequency as the target networks.

4.2 Measurement of on-policy transitions

We quantify the off-policy degree (1− on-policy) of a transition based on its age, aligning with the rationale that transitions generated by older policies, denoting a higher off-policy nature, are not accurate via MC simulation and should be assigned gradually decreasing TD(λ) values. The λ value depends on the likelihood that an off-policy transition is generated by the current policy. We define d as the distribution that the replay buffer D is sampled from and is supported on the entire state-action space, d^π as the stationary distribution of state-action pairs under the current policy, and $L_Q(\theta, d^\pi) = \|Q_{tot}(s, \mathbf{a}|\theta) - \mathcal{R}Q_{tot}(s, \mathbf{a}|\theta)\|_{d^\pi}^2$ as the loss function of the mixing network, in which the adaptive λ value is calculated by $\omega(s, \mathbf{a})$.

In practice, obtaining an accurate estimation of d^π requires on-policy samples from d^π and interactions with the environment. Moreover, when incorporating off-policy transitions from the replay buffer, calculating the on-policy degree $\omega(s, \mathbf{a}) := d^\pi(s, \mathbf{a})/d^D(s, \mathbf{a})$ becomes challenging due to the replay buffer D constituting a mixture of trajectories derived from policies at different time steps. In this paper, we adopt a variational representation of f-divergences between a set of older trajectories and a set of more recently generated trajectories to estimate the density ratios.

Theorem 1. *[Nguyen* et al.*, 2010] Assume that f has first-order derivatives f' at [0,+∞). $\forall P, Q \in \mathcal{P}(\mathcal{X})$ such that $P \ll Q$ and $\omega : \mathcal{X} \to \mathbb{R}^+$,*

$$D_f(P\|Q) \geq \mathbb{E}_P[f'(\omega(x))] - \mathbb{E}_Q[f^*(f'(\omega(x)))] \quad (5)$$

where f^* denotes the convex conjugate and the equality is achieved when $\omega = \frac{dP}{dQ}$.

According to Theorem 1, the density ratio $\omega(s, a) := d^\pi(s, a)/d^D(s, a)$ can be estimated by the samples from two sets of trajectories. One of the two sets of trajectories d^D can be sampled from the regular large replay buffer (off-policy buffer D_{off}) from original value-based MARL algorithms and the other one d^π from the small replay buffer (on-policy buffer D_{on}) which only contains recent trajectories. After each rollout process, the new trajectory is updated to D_{on}.

4.3 Training pipeline

Based on the two samples from the off-policy replay buffer and the on-policy replay buffer, the $\omega(s, \mathbf{a})$ can be estimated by a network $\omega_\phi(s, \mathbf{a})$ parametrized by ϕ. To estimate the density ratio $\omega(s, a) = \frac{dP}{dQ}(s, a)$, the problem can be framed as an optimization task of maximizing the lower bound based on the inequality. Maximizing the right-hand side of the inequality:

$$\max_{\omega_\phi}(\mathbb{E}_P[f'(\omega_\phi(s, \mathbf{a}))] - \mathbb{E}_Q[f^* f'(\omega_\phi(s, \mathbf{a}))]) \quad (6)$$

is equivalent to minimizing the negative lower bound of:

$$L_\omega(\phi) := \mathbb{E}_{D_{on}}[f^*(f'(\omega_\phi(s, \mathbf{a})))] - \mathbb{E}_{D_{off}}[f'(\omega_\phi(s, \mathbf{a}))] \quad (7)$$

From Theorem 1, the estimate of the density ratio can be recovered from the ω_ϕ by minimizing the $L_\omega(\phi)$. Additionally, the output of the network ω_ϕ is scaled to the range of $(0, 1)$ by the sigmoid activation function. The f function chosen in this paper is symmetric divergence related to Jensen-Shannon (JS) divergence, which results in a binary cross-entropy loss:

$$L_\omega(\phi) := BCE_{s,\mathbf{a}\sim D_{off}}(\omega_\phi(s, \mathbf{a}), 0) + BCE_{s,\mathbf{a}\sim D_{on}}(\omega_\phi(s, \mathbf{a}), 1) \quad (8)$$

Therefore, the final objective for TD learning over Q is:

$$\begin{aligned} L_Q(\theta; d^\pi) &\approx L_Q(\theta; D_{off}, \omega_\phi) \\ &= \mathbb{E}_{(s,\mathbf{a})\sim D_{off}}[(Q_{tot}(s, \mathbf{a}|\theta) - R_t^{\omega(s,\mathbf{a})})^2] \end{aligned} \quad (9)$$

$$\begin{aligned} R_t^{\omega(s,\mathbf{a})} &= r(s, \mathbf{a}) + \gamma[\omega(s, \mathbf{a}) R_{t+1}^{\omega(s',\mathbf{a}')} \\ &+ (1 - \omega(s, \mathbf{a})) \max_{\mathbf{a}'\in A} Q_{tot}(s', \mathbf{a}'|\theta^-)] \end{aligned} \quad (10)$$

where the θ^- is the mixing network parameter which is maintained and updated frequently. The $R_t^{\omega(s,a)}$ in this formula is the target value at time step t calculated by TD(λ) in which the λ value is calculated by ω network and conditioned on the state-action $(s, \mathbf{a})$ pairs.

In basic value-based MARL algorithms, the parameter of the utility network or the mixing network θ is updated frequently and the lag parameter θ^- is employed during the calculation of the target mixed value. Consequently, for a given state-action pair, the target value remains unchanged between update intervals, providing a stable supervised signal to $Q_{tot}(s, \mathbf{a}|\theta)$. Therefore, we cache the λ value for each sampled transition based on $\omega(s, \mathbf{a})$ for subsequent use, and refresh these cached values to 0 after the update of θ^- to ensure the stability.

For MAPPO, the two buffers are also maintained. The first is the standard on-policy rollout buffer, whose capacity corresponds to the number of rollout threads defined by MAPPO, which serves as the original trajectory buffer. To incorporate additional historical experience, a second off-policy buffer with a size 50x larger than the on-policy buffer is introduced. In the standard MAPPO framework, the actor is optimized using GAE computed from the on-policy data, while the critic is trained by minimizing the mean-squared error between its value estimates and the cumulated (MC) return. In our modification, we replace the MC return with the ATD-based return, which is computed using samples drawn from both the on-policy and off-policy buffers. This changes the critic learning procedure accordingly while leaving the actor update unchanged, aside from the addition of the off-policy buffer used for ATD estimation.

5 Experiment

We evaluate the performance of our method via the fully cooperative StarCraftII micro-management challenges by the mean winning rate in each scenario and the average scoring results in Google Football Research. In this section, we mainly compare our ATD(λ) method to those commonly-used baselines. We also show the performance enhancement by presenting 6 out of 23 scenarios with 2 levels of difficulty from SMAC in this paper and 6 multi-player academy tasks from Gfootball. Additionally, ablation studies are also conducted to show the adaptability of our approach to other algorithms and the influence of fast buffer size. More discussions can be found in Appendix.

5.1 Experiment Settings

SMAC We verify the adaptive λ methods on 6 subtasks of two difficulties, **a)** hard tasks including 5m_vs_6m, 10m_vs_11m, and **b)** super-hard scenarios 3s5z_vs_3s6z, corridor, MMM2, and 6h_vs_8z. The winning rates of battles are calculated by the mean of 32 evaluation processes among 10 times with different seeds and smoothed by 0.6 for better visualization within 2M time steps. The shading area is the variance that represents the stability of the generated policies. In each experiment, the x-axis represents the time steps (e6) being evaluated and the y-axis is the mean of the winning rate among 10 seeds of 32 evaluation rollout rounds.

Gfootball We also verify our proposed dynamic λ method on 6 academy scenarios of Google Football Research of which the reward settings are sparse. Since value-based MARL algorithms underperform on sparse reward settings, we mainly show the improvement of ATD method on AC-based methods within 50M time steps.

Baseline We adapt our method to QMIX and MAPPO and compare our methods to the value-based W-QMIX and QPLEX, popular policy-based algorithm MAPPO, and a recent AC-based algorithm MACPF with their officially-provided default parameter settings. The QMIX, QPLEX, and W-QMIX in this paper are from the pymarl codebase [Rashid *et al.*, 2020]. The MACPF is from the codebase [Zhang *et al.*, 2021; Wang *et al.*, 2023] and the MAPPO is provided by [Yu *et al.*, 2022].

5.2 Experiment Results

We commence the evaluation of our proposed dynamic λ method on the QMIX algorithm (TD(λ)) across six benchmarks within the SMAC framework which encompass two hard tasks and four super-hard tasks. The average test winning rate, computed across 10 seeds for each of the 6 scenarios, is depicted in Figure 3 to provide a comprehensive overview of the algorithms' overall performance. In the tasks

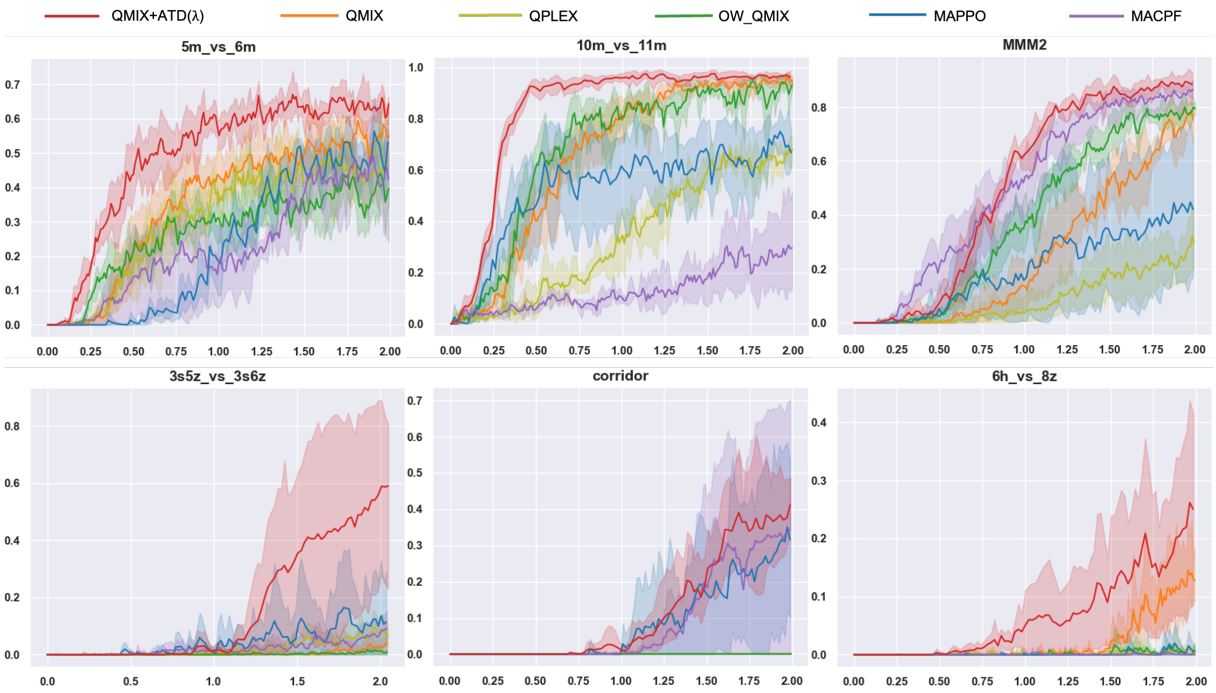


Figure 3: The winning rate curves evaluated on the 6 SMAC tasks with two major difficulties among our ATD+QMIX and other baselines.

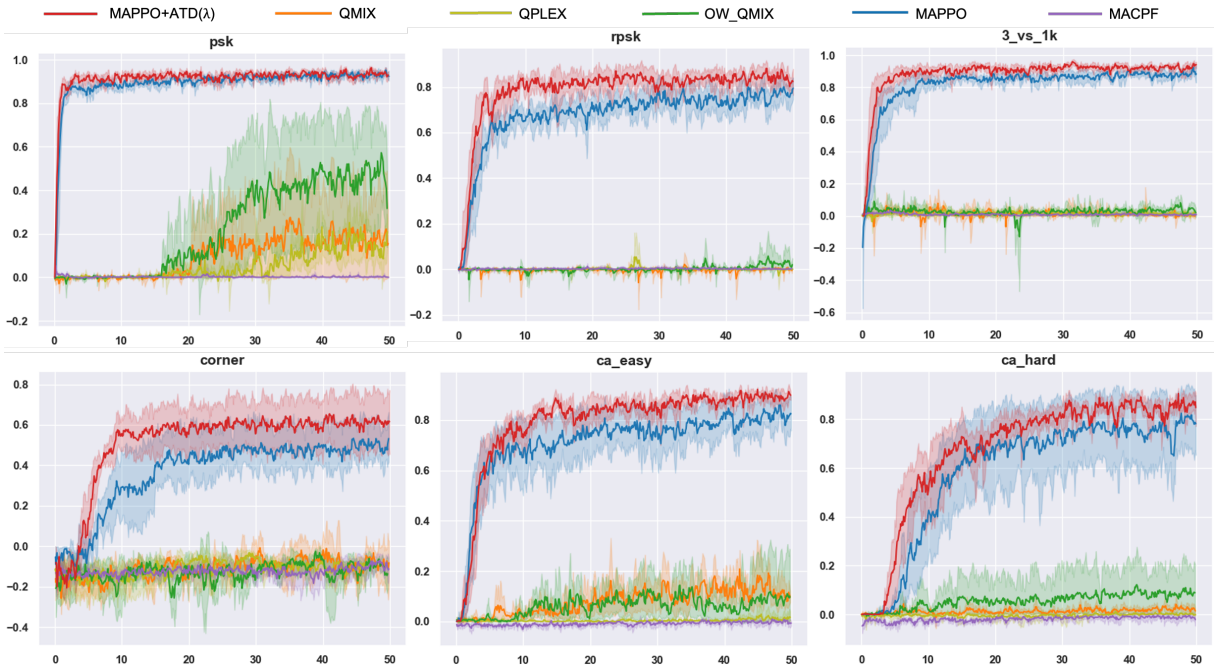


Figure 4: The winning rate curves evaluated on the 6 academy Gfootball tasks among our ATD+MAPPO and other baselines.

such as 5m_vs_6m and 10m_vs_11m, MMM2, our proposed method competes favorably with or outperforms other baseline algorithms. In the 3s5z_vs_3s6z, 6h_vs_8z, and the corridor task, where not all baseline algorithms exhibit winning rates, our method achieves commendable results. Notably, for the 3s5z_vs_3s6z task, we fine-tune the parameter of the mixing network size in QMIX and apply both the original setting and the adjusted setting to other baselines. The graph reflects the superior performance of the two settings. Other hyper-parameters are detailed in Appendix G.

Figure 4 presents the results of our proposed dynamic λ method applied to the critic training of MAPPO

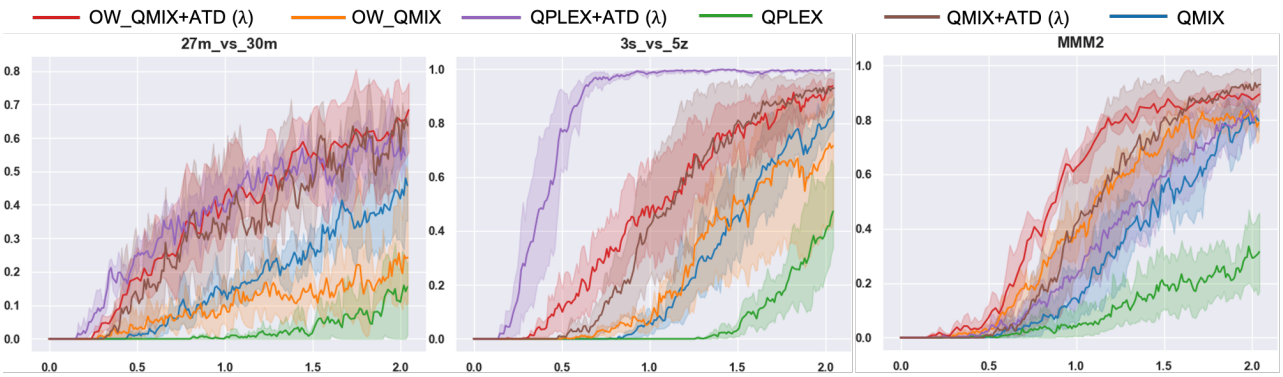


Figure 5: The winning rate curves evaluated on 27m_vs_30m, 3s_vs_5z, and MMM2 scenarios.

algorithm across six academy scenarios within the Google Football Research framework. In the scenarios 'pass_shoot_with_keeper' (abbreviated as psk) and '3_vs_1_with_keeper' (3_vs_1k), our method competes effectively with the MAPPO algorithm. Impressively, in the remaining four scenarios, our method significantly outperforms other baseline algorithms. It is noteworthy that Figure 4 also highlights a trend where the majority of value-based algorithms and an Actor-Critic-based algorithm, MACPF, struggle to perform well on these tasks. Despite applying our dynamic TD(λ) method to the QMIX algorithm, the observed performance improvement is marginal and falls short of matching the performance achieved by MAPPO.

5.3 Discussion

In this section, we mainly show the performance enhancement and the compatibility that our method can be adapted to other value-based methods. We also show the influence of on/off-policy buffer size and the λ value cache mechanism.

Compatibility We implement two replay buffers of different sizes and calculate the λ value according to the state-action density ratios, so our ATD(λ) module, based on TD updates, can be regarded as a plugin that can be adapted to other value-based MARL methods with minor changes. To test the compatibility of our work, we apply our method on QPLEX, and OW_QMIX algorithms in 27m_vs_30m, 3s_vs_5z, and MMM2 scenarios correspondingly.

According to Figure 5, in the three scenarios, all of the algorithms with our proposed ATD(λ) method outperform the algorithms with static λ values. By adding a large replay buffer and updating the target critic value, we also apply the idea of our approach to the MAPPO algorithm to assist the critic network training. Figure 4 also shows that our method can also improve the policy-based MARL algorithms with critic networks.

On/Off-policy Buffer Size According to the default settings of code-base pymarl, we set the size of the off-policy replay buffer as 5000. To test the influence of the on-policy replay buffer size, we choose four ratios, 10x, 25x, 50x, and 100x, compared with the off-policy replay buffer on the 10m_vs_11m scenario. As for the MAPPO algorithm, the on-policy buffer size is the parallel rollout size and the off-policy buffer size is calculated by multiplying the ratios.

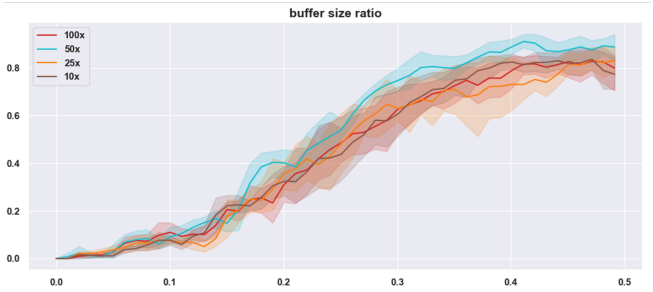


Figure 6: The winning rate curves evaluated on 10m_vs_11m with different ratios between the on/off-policy replay buffer size.

As shown in Figure 6, the 50x buffer size distinctly achieves the highest performance and faster convergence speed. Empirically with the policy improvement progress, a large on-policy buffer may be mixed into some off-policy data because old trajectories are refreshed slowly. In contrast, a small on-policy buffer may not be able to contain enough on-policy data due to the variance initial state. Thus, this paper chose a proper ratio of 50x and used it as the default setting among the experiments.

6 Conclusion and Future Work

In this work, we consider the challenge of choosing the hyperparameter of the λ value, which should be properly selected before training. We propose our ATD method that consists of two replay buffers and an extra network to calculate the f-divergence as the density ratio. Most MARL code bases have either provided the off-policy buffer (value-based methods) or on-policy buffer (policy-based methods), so our ATD can be easily implemented. The SMAC and Gfootball experimental results indicate that our method can be adapted to both value-based and AC-based methods. However, our method may not be suitable to the environment with too much randomness, the transitions in the replay buffer are non-repeating, such that the density ratio, ω, are always quickly decayed to 0. More discussions about SMACv2 are in Appendix. In the future, we might be concerned about how to solve the problem taken by large randomness such as the SMACv2 environment.

Acknowledgements

This work was supported by the NSFC project (No. 62072399), the Zhejiang Provincial Natural Science Foundation of China under Grant No. LZ23F020009, the Zhongguancun Academy (Grant No.XTS0041), the Fundamental Research Funds for the Central Universities, MoE Engineering Research Center of Digital Library, China Research Centre on Data and Knowledge for Engineering Sciences and Technology. We also express our sincere gratitude to anonymous reviewers for their invaluable feedback and constructive comments.

References

[Cao *et al.*, 2012] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial informatics*, 9(1):427–438, 2012.

[Cicek *et al.*, 2021] Dogan C Cicek, Enes Duran, Baturay Saglam, Kagan Kaya, Furkan Mutlu, and Suleyman S Kozat. Awd3: Dynamic reduction of the estimation bias. In *2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 775–779. IEEE, 2021.

[Foerster *et al.*, 2018] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[Geist *et al.*, 2014] Matthieu Geist, Bruno Scherrer, et al. Off-policy learning with eligibility traces: a survey. *J. Mach. Learn. Res.*, 15(1):289–333, 2014.

[Grover *et al.*, 2019] Aditya Grover, Jiaming Song, Ashish Kapoor, Kenneth Tran, Alekh Agarwal, Eric J Horvitz, and Stefano Ermon. Bias correction of learned generative models using likelihood-free importance weighting. *Advances in neural information processing systems*, 32, 2019.

[Harutyunyan *et al.*, 2016] Anna Harutyunyan, Marc G Bellemare, Tom Stepleton, and Rémi Munos. Q (λ) with off-policy corrections. In *International Conference on Algorithmic Learning Theory*, pages 305–320. Springer, 2016.

[Hu *et al.*, 2021] 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. *arXiv preprint arXiv:2102.03479*, 2021.

[Jiang *et al.*, 2021] Ray Jiang, Tom Zahavy, Zhongwen Xu, Adam White, Matteo Hessel, Charles Blundell, and Hado Van Hasselt. Emphatic algorithms for deep reinforcement learning. In *International Conference on Machine Learning*, pages 5023–5033. PMLR, 2021.

[Kurach *et al.*, 2020] Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, et al. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 4501–4510, 2020.

[Li *et al.*, 2025] Yueheng Li, Guangming Xie, and Zongqing Lu. Revisiting cooperative off-policy multi-agent reinforcement learning. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net, 2025.

[Lowe *et al.*, 2017] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

[Mahajan *et al.*, 2019] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. Maven: Multi-agent variational exploration. *Advances in Neural Information Processing Systems*, 32, 2019.

[Munos *et al.*, 2016] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

[Nguyen *et al.*, 2010] XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, 2010.

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

[Peng and Williams, 1994] Jing Peng and Ronald J Williams. Incremental multi-step q-learning. In *Machine Learning Proceedings 1994*, pages 226–232. Elsevier, 1994.

[Precup, 2000] Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

[Rashid *et al.*, 2018] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder 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*, pages 4292–4301, 2018.

[Rashid *et al.*, 2020] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. *Advances in neural information processing systems*, 33:10199–10210, 2020.

[Samvelyan *et al.*, 2019] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philiph H. S. Torr, Jakob Foerster, and Shimon Whiteson. The StarCraft Multi-Agent Challenge. *CoRR*, abs/1902.04043, 2019.

[Seijen and Sutton, 2014] Harm Seijen and Rich Sutton. True online td (lambda). In *International Conference on Machine Learning*, pages 692–700. PMLR, 2014.

[Sinha *et al.*, 2022] Samarth Sinha, Jiaming Song, Animesh Garg, and Stefano Ermon. Experience replay with likelihood-free importance weights. In *Learning for Dynamics and Control Conference*, pages 110–123. PMLR, 2022.

[Son *et al.*, 2019] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, pages 5887–5896. PMLR, 2019.

[Sunehag *et al.*, 2017] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*, 2017.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Sutton *et al.*, 2016] Richard S Sutton, A Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research*, 17(73):1–29, 2016.

[Sutton, 1988] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3:9–44, 1988.

[Wang *et al.*, 2020a] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning. *arXiv preprint arXiv:2008.01062*, 2020.

[Wang *et al.*, 2020b] Tonghan Wang, Heng Dong, Victor Lesser, and Chongjie Zhang. Roma: Multi-agent reinforcement learning with emergent roles. *arXiv preprint arXiv:2003.08039*, 2020.

[Wang *et al.*, 2020c] Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. Dop: Off-policy multi-agent decomposed policy gradients. In *International conference on learning representations*, 2020.

[Wang *et al.*, 2023] Jiangxing Wang, Deheng Ye, and Zongqing Lu. More centralized training, still decentralized execution: Multi-agent conditional policy factorization. In *International Conference on Learning Representations (ICLR)*, 2023.

[Wu *et al.*, 2026] Cuiling Wu, Yaozhong Gan, Junliang Xing, and Ying Fu. Marpo: A reflective policy optimization for multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 29740–29748, 2026.

[Yang *et al.*, 2020] Yaodong Yang, Ying Wen, Jun Wang, Liheng Chen, Kun Shao, David Mguni, and Weinan Zhang. Multi-agent determinantal q-learning. In *International Conference on Machine Learning*, pages 10757–10766. PMLR, 2020.

[Yao *et al.*, 2021] Xinghu Yao, Chao Wen, Yuhui Wang, and Xiaoyang Tan. Smix (λ): Enhancing centralized value functions for cooperative multiagent reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[Yu *et al.*, 2022] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624, 2022.

[Zhang and Lesser, 2011] Chongjie Zhang and Victor Lesser. Coordinated multi-agent reinforcement learning in networked distributed pomdps. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[Zhang *et al.*, 2021] Tianhao Zhang, Yueheng Li, Chen Wang, Guangming Xie, and Zongqing Lu. Fop: Factorizing optimal joint policy of maximum-entropy multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 12491–12500. PMLR, 2021.