Learning Implicit Credit Assignment for Multi-Agent Actor-Critic

Meng Zhou[*] **Ziyu Liu**[*] **Pengwei Sui** **Yixuan Li** **Yuk Ying Chung**
School of Computer Science
The University of Sydney
{mzho7212,zliu6676,psui3905,yili2331}@uni.sydney.edu.au
vera.chung@sydney.edu.au

Abstract

We present a new policy-based multi-agent reinforcement learning algorithm that implicitly addresses the credit assignment problem under fully cooperative settings. Our key motivation is that credit assignment may not require an explicit formulation as long as (1) the policy gradients of a trained, centralized critic carry sufficient information for the decentralized agents to maximize the critic estimate through optimal cooperation and (2) a sustained level of agent exploration is enforced throughout training. In this work, we achieve the former by formulating the centralized critic as a hypernetwork such that the latent state representation is now fused into the policy gradients through its multiplicative association with the agent policies, and we show that this is key to learning optimal joint actions that may otherwise require explicit credit assignment. To achieve the latter, we further propose a practical technique called *adaptive entropy regularization* where magnitudes of the policy gradients from the entropy term are dynamically rescaled to sustain consistent levels of exploration throughout training. Our final algorithm, which we call LICA, is evaluated on several benchmarks including the multi-agent particle environments and a set of challenging StarCraft II micromanagement tasks, and we show that LICA significantly outperforms previous methods.

1 Introduction

Many complex real-world problems such as autonomous vehicle coordination [2], network routing [30], and robot swarm control [8] can naturally be formulated as multi-agent cooperative games, where reinforcement learning (RL) presents a powerful and general framework for training robust agents. Though single agent RL algorithms can be trivially applied to these environments by treating the multi-agent system as a single actor with a joint action space, the limited scalability and the inherent constraints on agent observability and communication in many multi-agent environments necessitate *decentralized* agents that act only on their local observations. A straightforward approach to learn decentralized policies is to train the agents independently, but the simultaneous exploration of the agents often results in non-stationary environments where learning becomes highly unstable. To this end, previous work relies on a standard paradigm known as *centralized training with decentralized execution* (CTDE) [18, 11, 20, 4, 14, 3], where the independent agents can access additional state information that is unavailable during policy inference.

However, one major challenge of CTDE in cooperative settings is *credit assignment*, which refers to the task of attributing a global reward from the environment to the individual agents' actions. Solving the credit assignment issue can give useful insights into which agents or agent actions were responsible for the collective reward signal and may thus substantially facilitate policy optimization, but it is often

[*]Equal Contribution

nontrivial since the interactions between the agents can be highly complex. A notable approach is to assess agent actions by calculating *difference rewards* against a certain reward baseline [27, 19, 4], but these methods may be inefficient as they either require a separate estimator for the baseline or extra simulations with the environment. They also become less effective with complex cooperation behaviors. Another line of approaches is to represent the global state-action value as a (deterministic or learned) aggregation of the individual state-action values [26, 20, 25, 14]. A representative method is QMIX [20], which achieves *implicit* credit assignment by learning a *mixing network* that conditions on the global state and combines the individual agent Q-values to estimate the joint action Q-value. While these methods allow more complex credit assignment, the capacity of the mixing network is still limited by the *monotonic* relationships between the joint and the individual Q-values. Moreover, since these approaches are value-based, extensions to continuous spaces may require additional strategies that can compromise performance and/or complexity.

In this work, we propose a new policy-based algorithm called LICA for learning implicit credit assignment that addresses the above limitations. LICA is closely related to multi-agent deterministic actor-critic methods [24, 13] where the decentralized policies are directly updated in the direction of the approximate joint-action Q-value gradient. Our key contribution is to extend the concept of *value mixing* [26, 20, 14, 25] to *policy mixing*, where the centralized critic is formulated as a *hypernetwork* [5] that maps the state information into a set of weights which, in turn, mixes the individual agent action probabilities into the joint action-value estimate. Compared to [13], this practical formulation allows the decentralized agents to learn optimal cooperative strategies by fusing the current state into the policy gradients while removing the need for explicitly formulating agent communication or policy approximation. It also trivially achieves higher expressiveness than value mixing methods as there are no inherent constraints on the centralized critic's mixing weights.

Another notable challenge for policy-based algorithms and particularly deterministic methods [24, 12, 13] is maintaining consistent levels of agent exploration for preventing premature convergence to sub-optimal policies [29, 15, 1]. Many existing methods [23, 7, 9, 13, 28] do so by introducing an additional entropy loss term that favors more stochastic actions. However, we argue that its vanilla form is ineffective for this purpose due to the undesirable curvature of the entropy function derivative with respect to the action probabilities. To this end, we further propose a simple technique called *adaptive entropy regularization* where the magnitudes of policy gradients from the entropy term are inversely adjusted based on the policy entropy itself. We empirically show that this allows easier tuning of regularization strength and more consistent levels of agent exploration throughout training.

We benchmark our methods on two sets of cooperative environments, the Multi-Agent Particle Environments [13] and the StarCraft Multi-Agent Challenge [21], and we observe significant performance improvements over previous state-of-the-art algorithms with faster convergence. We also conduct further component studies to demonstrate that (1) compared to difference reward based credit assignment approaches (e.g. COMA [4]), LICA has higher representational capacity and can readily handle environments where multiple global optima exist; (2) our adaptive entropy regularization is crucial for encouraging consistent exploration and faster policy convergence in complex scenarios.

2 Related Work

Explicit Credit Assignment. In general, explicit methods provide strategies for attributing agent contributions that are at least provably locally optimal [10]. COMA [4] is a representative method that uses a centralized critic to estimate the counterfactual advantage of an agent action, but it quickly becomes ineffective with complex cooperation behaviors. SQDDPG [28] provides a theoretical framework where credit assignment is based on an agent's approximate *marginal contribution* as it is sequentially added to the agent group. While theoretically justified, this formulation assumes an initial oracle scheduling the agents with global observation that is often unavailable in practice.

Implicit Credit Assignment. In contrast, implicit methods do not purposely assess the individual actions against a certain baseline, and most previous methods address credit assignment by directly learning a value decomposition from the shared reward signal into the individual component value functions [26, 20, 25, 14]. An earlier work is VDN [26], where the value decomposition is linear and the state information is ignored during training. QMIX [20] presents an improvement by conditioning a hypernetwork on the global state for a non-linear mixing of the individual action-values, but it is still limited by the monotonicity constraint of its mixing weights. QTRAN [25] attempts to address these limitations with a provably more general value factorization, but it imposes intractable constraints even for discrete state-action spaces [14]. Compared to these value-based methods, our policy-based LICA easily extends to continuous actions, provides higher expressiveness as no additive or monotonic constraints exist, and shows stronger empirical performance in complex environments.

Deterministic Policy Gradients (DPG). As our method learns the decentralized policies by maximizing a joint action value function, it is related to the family of DPG algorithms [24, 12, 13] which consider deterministic policies to more efficiently estimate the policy gradients of the action-value function. MADDPG [13] extends the framework to multi-agent settings, where each agent maintains a separate centralized critic for learning different reward structures. In contrast, our formulations have higher capacity for learning complex agent cooperation while being simpler and more efficient.

Entropy Regularization is a common technique for improving policy optimization by inducing more stochastic actions and a smoother objective landscape [1], where most existing methods [29, 6, 15, 22, 23, 7, 9, 13, 28] augment the training objective with a weighted entropy loss term. Other formulations [16, 1] introduce a decaying schedule for the regularization strength. As mentioned in Section 1, we argue that its current form is ineffective at sustaining a consistent agent exploration level that is often required for uncovering optimal joint actions in challenging environments.

3 Methods

3.1 Preliminaries

A fully cooperative multi-agent task with n agents $\mathcal{A} = \{1, ..., n\}$ can be formulated into a Dec-POMDP [17] as a tuple $G = (S, U, P, r, Z, O, n, \gamma)$. At each time step t, each agent a chooses an action u_t^a from its action space $U_a \in \{U_1, ..., U_n\} \equiv U$, forming a joint action $u_t \in (U_1 \times \cdots \times U_n) \equiv U^n$. $P(s_{t+1}|s_t, u_t)\colon S \times U^n \times S \to [0, 1]$ is the state transition function where S is the set of truth states and $s_t \in S$ is the state at time t. $r(s_t, u_t)\colon S \times U^n \to \mathbb{R}$ is the reward function yielding a shared reward r_t for u_t, and $\gamma \in [0, 1)$ is the discount factor. We consider partially observable settings, so each agent a acts on its local observation $z_t^a \in Z$ drawn from the observation function $O(s_t, a)\colon S \times \mathcal{A} \to Z$ where Z is the observation space. To select an action, each agent a follow its stochastic policy $\pi^a(u_t^a|z_t^a)\colon Z \times U_a \to [0, 1]$, and the agents' joint policy π induces a joint action value function $Q^\pi(s_t, u_t) = \mathbb{E}_{s_{t+1:\infty}, u_{t+1:\infty}}[R_t|s_t, u_t]$ where $R_t = \sum_{i=0}^{T} \gamma^i r_{t+i}$ is the discounted accumulated reward with time horizon T. Our goal is to find the optimal joint policy π^* such that $Q^{\pi^*}(s_t, u_t) \geq Q^\pi(s_t, u_t)$, for all π and $(s_t, u_t) \in S \times U^n$.

3.2 LICA: Learning Implicit Credit Assignment for Cooperative Environments

In this section, we present a new multi-agent actor-critic method called LICA that implicitly achieves credit assignment among agents. The key motivation of our method is that if the decentralized policy networks can be directly optimized to maximize the output of a *perfect* joint action value function $Q^*(s, u)$, then any converged joint policies should have acquired the optimal cooperative behaviors without *explicitly* formulating a credit assignment. In practice, if we assume that a trained centralized critic network $Q_{\theta_c}^\pi$ provides a close approximation of Q^* at least for the current parametrized joint policy $\pi_\theta = \{\pi_{\theta_1}^1, ..., \pi_{\theta_n}^n\}$, then finding an end-to-end differentiable optimization setting where the policy parameters $\theta = \{\theta_1, ..., \theta_n\}$ receive gradients with $Q_{\theta_c}^\pi$ as the criterion, should act as a proxy for finding optimal credit assignment strategies (e.g. studied in [4]). A straightforward approach taken by MADDPG [13, 9] is to formulate $Q_{\theta_c}^\pi$ as an MLP that maps the concatenation $[s \| \pi_{\theta_1}^1(\cdot|z^1) \| \cdots \| \pi_{\theta_n}^n(\cdot|z^n)]$ into the joint action Q-value estimate. However, the critical drawback is that the gradients of $Q_{\theta_c}^\pi$ w.r.t. θ will *not* carry any state information as vector concatenation implies that the respective linear transforms of s and $\pi_{\theta_a}^a(\cdot|z^a)$ for all a are associated through *addition*.

A better approach is to instead formulate $Q_{\theta_c}^\pi$ as a *hypernetwork* [5] that maps s into a set of weights $\mathbf{W}$ that, in turn, maps the concatenated agent action probabilities into the joint action Q-value estimate. The key insight is that $\nabla_\theta Q_{\theta_c}^\pi$ are now expressed in terms of $\mathbf{W}$, which are effectively latent state representations, due to their *multiplicative* association with the stochastic actions. The resulting centralized critic hypernetwork, which we call the *mixing critic*, can be optimized by directly

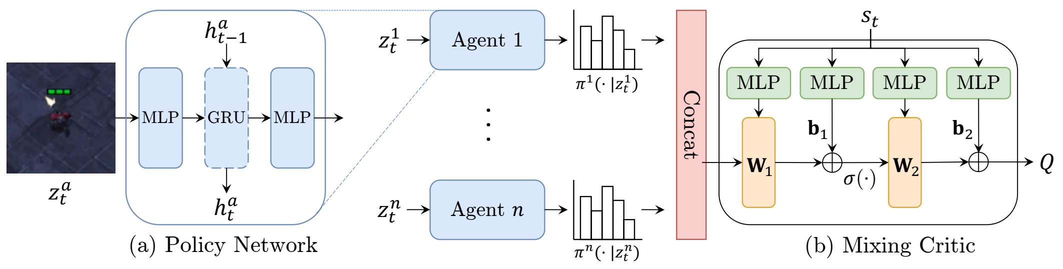

Figure 1: **Architecture for LICA**. (a) Architecture for the decentralized policy networks. (b) Architecture for the centralized critic network. Best viewed in color.

Algorithm 1 Optimization procedure for LICA

```
1: Randomly initialize $\theta$ and $\theta_c$ for the policy networks and the mixing critic respectively.
2: for number of training iterations do
3:     Sample $b$ trajectories $\mathcal{D}_1, ..., \mathcal{D}_b$ with $\mathcal{D}_i = \{z_{0,i}, s_{0,i}, u_{0,i}, r_{0,i}, ..., z_{T,i}, s_{T,i}, u_{T,i}, r_{T,i}\}$.
4:     Calculate accumulated rewards $\{R_{0,i}, ..., R_{T,i}\}$ for each trajectory $\mathcal{D}_i$.
5:     for $k$ iterations do
6:         Update the mixing critic by descending its gradient according to Eq. (1):
               $\nabla_{\theta_c} \sum_{i=1}^{b} \sum_{t=1}^{T} \left[R_{t,i} - Q^{\pi}_{\theta_c}\left(s_{t,i}, u^1_{t,i}, ..., u^n_{t,i}\right)\right]^2 \quad (3)$
7:     Update the decentralized policy networks by ascending their gradients according to Eq. (2):
               $\nabla_{\theta} \sum_{i=1}^{b} \sum_{t=1}^{T} \left[Q^{\pi}_{\theta_c}\left(s_{t,i}, \pi^1_{\theta_1}(\cdot|z^1_{t,i}), ..., \pi^n_{\theta_n}(\cdot|z^n_{t,i})\right) + \frac{1}{n}\sum_{a=1}^{n} \mathcal{H}\left(\pi^a_{\theta_a}(\cdot|z^a_{t,i})\right)\right] \quad (4)$
```

regressing to the discounted accumulated rewards R_t:

$$\min_{\theta_c} \mathbb{E}_{t,s_t,u_t^1,...,u_t^n} \left[R_t - Q^{\pi}_{\theta_c}(s_t, u^1_t, ..., u^n_t)\right]^2 \tag{1}$$

With a trained critic estimate, the decentralized policy networks can then be end-to-end optimized simultaneously to maximize $Q^{\pi}_{\theta_c}$ with the stochastic policies as inputs:

$$\max_{\theta} \mathbb{E}_{t,s_t,z_t^1,...,z_t^n} \left[Q^{\pi}_{\theta_c}\left(s_t, \pi^1_{\theta_1}(\cdot|z^1_t), ..., \pi^n_{\theta_n}(\cdot|z^n_t)\right) + \mathbb{E}_a\left[\mathcal{H}\left(\pi^a_{\theta_a}(\cdot|z^a_t)\right)\right]\right] \tag{2}$$

where $\mathcal{H}$ is the entropy regularization term (see Section 3.3). The overall setup and training procedure are summarized in Fig. 1 and Algorithm 1 respectively. During forward pass, the policy networks map their local observations z^a_t into the action probabilities $\pi^a_{\theta_a}(\cdot|z^a_t)$ that are then concatenated as the input to the mixing critic, which uses the state s_t to produce a set of weights $\mathbf{W}$ for transforming the action probabilities directly into the joint action Q-value estimate. As the architecture is end-to-end differentiable, we can directly train the agents with the backpropagated gradients. Note that we freeze the mixing critic during policy training and sample one-hot actions u^a from $\pi^a_{\theta_a}$ during critic training.

Discussion. We provide further analyses on LICA as follows. (1) A key distinction from MADDPG is that LICA isolates the state transformation from the action transformation through the use of a critic hypernetwork. This gives the advantage that during policy optimization, the direction of the policy gradients is *decoupled* from the direction of a state update (which is redundant). This is not the case with MADDPG since its critic implementation implies that the state and action vectors are associated through addition. (2) Through $\mathbf{W}$, the policy gradients in LICA carry the state information which is crucial for learning optimal cooperative behaviors without explicitly formulating a credit assignment. In contrast, the critic formulation of MADDPG implies a poor utilization of the state information made available from centralized training for implicitly addressing the credit assignment problem. (3) Because the policy gradients carry the state information, LICA also removes the need for agent communication, separate critic networks, or inferring other agents' policies under cooperative settings, significantly simplifying the training framework. (4) Compared to value mixing methods such as [26, 20, 14], LICA easily extends to continuous actions and has higher capacity for implicit credit assignment since there are no constraints on the mixing critic.

3.3 Adaptive Entropy Regularization

As LICA is policy-based, it shares a common limitation that insufficient agent experience may lead to premature convergence to poor policies. This is particularly notable when agents are acting in complex or real-world environments where sampling a large number of diverse trajectories can be expensive or impractical. To mitigate this issue, previous methods rely on an additional entropy term $\mathcal{H}\left(\pi^a(\cdot|z^a)\right) = \beta H\left(\pi^a(\cdot|z^a)\right) = \beta \mathbb{E}_{u^a \sim \pi^a}\left[-\log \pi^a(u^a|z^a)\right]$ for agent a weighted by some constant β during optimization. The rationale is that penalizing a low policy entropy resulting from under/over-confidence actions should encourage sustained exploration throughout training. However, in practice we observe that this is rarely the case: β often has high sensitivity in complex environments and a tiny nudge could lead to drastically different entropy trajectories; moreover, once policies start to converge (i.e. drop in entropy), the same regularization term hardly encourages further exploration.

We argue that these issues stem from the *curvature* of the entropy function derivative which in turn influences the magnitudes of its policy gradients on different action probabilities. Concretely, let us first consider a k-class action probability vector $p^a = \pi^a(\cdot|z^a)$ of an agent a with number of feasible actions $k > 2$. The derivative of $\mathcal{H}$ with respect to p^a is given by:

$$d\mathcal{H} = \left[\frac{\partial \mathcal{H}}{\partial p_1^a}, \cdots, \frac{\partial \mathcal{H}}{\partial p_k^a}\right] = \left[-\beta(\log p_1^a + 1), \cdots, -\beta(\log p_k^a + 1)\right] \tag{5}$$

assuming natural logarithm. We denote $d\mathcal{H}_i := -\beta(\log p_i^a + 1)$ and note that $d\mathcal{H}_i$ depends only on the i-th input probability p_i^a when $k > 2$. In particular, note that since $d\mathcal{H}_i$ is log-shaped, the gradient magnitudes for dampening high-confidence actions (large p_i^a) are disproportionately small compared to that for encouraging low-confidence actions (small p_i^a). While favoring the latter should help achieve the former, we argue that in practice, insufficient penalties for over-confidence can still result in agents sticking to a subset of high-confidence actions while only updatng the action probabilities of other actions during stochastic optimization. On this basis, a reasonable approach is to manually induce a larger penalty (gradient magnitudes) for large p_i^a. To this end, we propose a practical technique called *adaptive entropy regularization* where we dynamically adjust the magnitudes of the policy gradients such that they are inversely proportional to the policy entropy itself during training:

$$d\mathcal{H}_i := -\lambda \cdot \frac{\log p_i^a + 1}{H(p^a)} \tag{6}$$

with constant λ controlling regularization strength. We visualize its values and curvature in Fig. 2 (a)[2], and note that large action probabilities now incur greater penalties from the entropy regularization. This formulation has a clean interpretation that the regularization strength previously controlled by the constant β is now *adaptive* based on the agent's exploration: if the policy exhibits deterministic behavior (i.e. low entropy), then the strength are scaled up by dividing a smaller term (the entropy); and if the policy is too stochastic, the strength is scaled down by dividing a larger term. Possible future work may also generalize Eq. 6 by replacing the entropy $H(p^a)$ with other measures of policy stochasticity, such as a multiplicative inverse of the L2 distance between p_a and a uniform action vector, that may lead to different empirical or theoretical outcomes.

4 Experiments

4.1 Multi-Agent Particle Environments

We first evaluate our algorithm against previous state-of-the-art methods on two common multi-agent particle environments [13]: Predator-Prey and Cooperative Navigation.

Predator-Prey. In this environment, 3 cooperating agents control 3 predators to chase a faster prey by controlling their velocities with actions [`up`,`down`,`left`,`right`,`stop`] within an area containing 2

[2]Note that $k = 2$ is a special case where the action probabilities are directly interdependent and Eq. 5 does not apply. For visualization purposes, Fig. 2 (a) follows Eq. 5 but note that our analysis applies to $k > 2$.

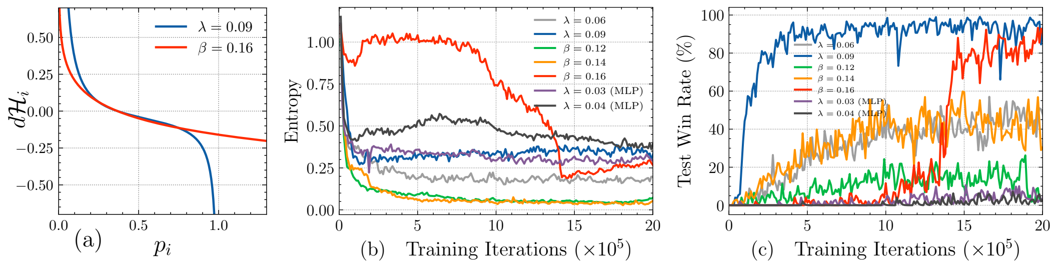

Figure 2: **Ablation experiments for *adaptive entropy regularization* and *mixing critic*.** (a) Plots of the entropy function derivative before (with β) and after (with λ) applying Eq. 6. (b) Entropy trajectories during training on StarCraft II `5m_vs_6m` (Hard). (c) Test win rates during training.

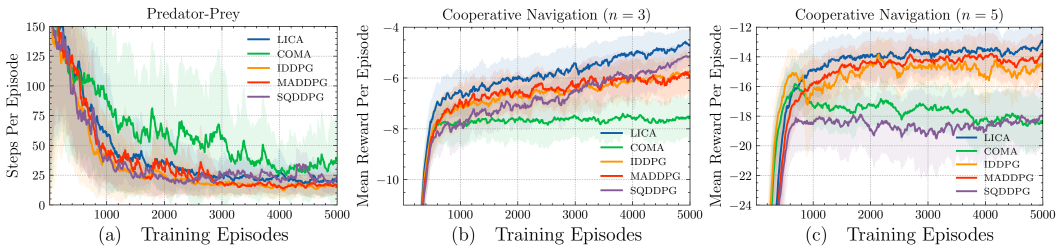

Figure 3: **Comparing performance during training in Particle Environments**. (a) Predator-Prey. (b, c) Cooperative Navigation with $n = 3$ and $n = 5$ respectively.

large obstacles at random locations. To focus on cooperative behavior, we follow [28] where the prey acts randomly. Each predator only observes its own velocity and position, and its displacement from the prey, obstacles, and other predators. The goal is to capture the prey with a minimal number of steps, and the shared reward is the minimum distance between the prey and any of the predators. The game terminates when the prey is captured and an additional positive shared reward is given.

Cooperative Navigation. In this environment, n agents and n landmarks are initialized with random locations within an area, and the agents must cooperate to cover all landmarks by controlling their velocities with actions [`up`,`down`,`left`,`right`,`stop`]. Each agent only observes its own velocity and displacement from other agents and the landmarks, and the shared reward is the negative sum of displacements between each landmark and its nearest agent. Agents must also avoid collisions, which incur negative shared rewards. $n = 3$ by default, but we also extend the difficulty with $n = 5$.

Training Settings. For all environments, agents are trained for 5000 episodes, each has a maximum of 200 steps and may end early for Predator-Prey. We follow the open-source implementation from [28] for the baseline algorithms, including COMA [4], Independent DDPG [12] (IDDPG), MADDPG [13], and SQDDPG [28]. Each algorithm is trained for 5 times and the mean and standard deviation of the goal metric (steps per episode and mean episode reward, respectively) throughout training are reported. See Supplementary for further settings.

Results. Fig. 3 reports the results for the Particle Environments. We can first observe that LICA performs on par with or better than all previous state-of-the-art methods, verifying its effectiveness. We can also see that LICA consistently outperforms COMA and MADDPG, confirming its capacity for credit assignment. Moreover, $n = 5$ for Cooperative Navigation increases the task difficulty since now SQDDPG underperforms and MADDPG yields better relative performance due to the growth of its complexity with n. For Predator-Prey, LICA achieves similar convergence speed and performance compared to other methods, suggesting a saturation with the environment likely due to the random (instead of learned) prey. Despite achieving state-of-the-art results, we argue that the particle environments may not be sufficiently challenging to verify the superiority of LICA, given that IDDPG, which completely ignores cooperation and/or credit assignment, is already competitive.

4.2 StarCraft II Micromanagement

Configurations. StarCraft II provides a rich set of heterogeneous units each with diverse actions, allowing extremely complex cooperative behaviors among agents. We thus further evaluate LICA

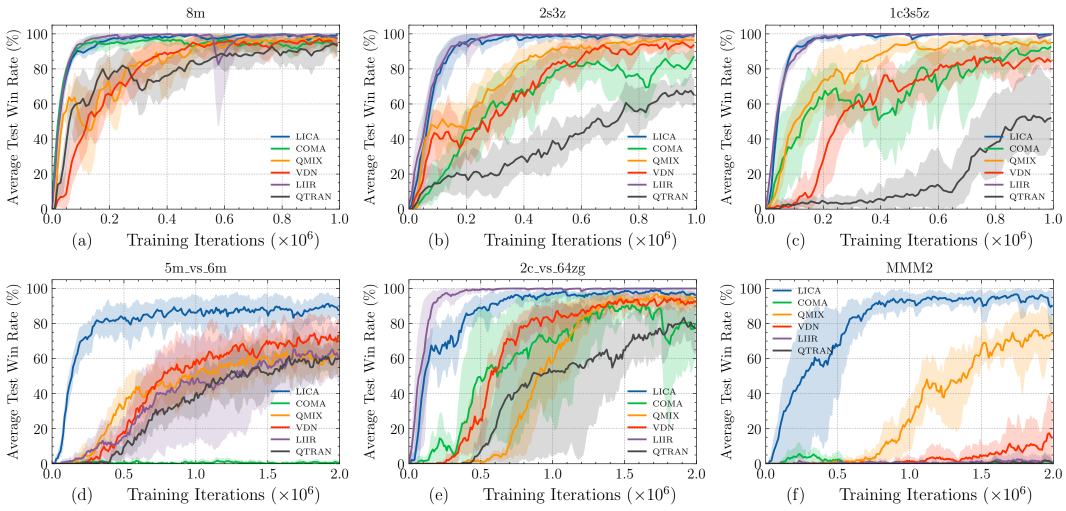

Figure 4: **Comparing performance across various scenarios in StarCraft II**. Top: **Easy** scenarios (`8m`, `2s3z`, `1c3s5z`). Bottom: **Hard** and **Super Hard** scenarios (`5m_vs_6m`, `2c_vs_64zg`, `MMM2`).

on several StarCraft II micromanagement tasks from the SMAC [21] benchmark, where a group of mixed-typed units controlled by decentralized agents needs to cooperate to defeat another group of mixed-typed enemy units controlled by built-in heuristic rules with "difficult" setting; the battles can be both symmetric (same units in both groups) or asymmetric. Same as previous work [4, 20, 26, 3], we used the default environment settings from SMAC. Each agent observes its own status and, within its field of view, it also observes other units' statistics such as health, location, and unit type; agents can only attack enemies within their shooting range. A shared reward is received on battle victory as well as damaging or killing enemy units. Each battle lasts for at most 250 steps and may end early.

We consider 6 battle maps grouped [21] into **Easy** (`8m`, `2s3z`, `1c3s5z`), **Hard** (`5m_vs_6m`, `2c_vs_64zg`), and **Super Hard** (`MMM2`) scenarios against 5 baseline methods using their open-source implementations based on PyMARL [21]: COMA [4], LIIR [3], VDN [26], QMIX [20], and QTRAN [25]. COMA and LIIR are policy-based while VDN, QMIX, and QTRAN are value-based. We follow [3] where all methods use the same batch size (32 episodes) and same number of training iterations, which is 1 million for **Easy** (due to fast convergence) and 2 million for **Hard** and **Super Hard** scenarios. All methods also use GRU modules (e.g. see Fig. 1), and they share the parameters of the individual agent networks. Performance is evaluated every 10000 training iterations using 32 test episodes where agents act deterministically, and we report the mean, min, and max average win rate of the test episodes across three repeated runs for every method in every scenario. See Supplementary for further environment and training details and battle visualizations.

Results. Fig. 4 reports the results across all 6 scenarios. We first observe that LICA achieves competitive and consistent test win rates across homogeneous and heterogeneous groups (e.g. `8m` and `2s3z`), symmetric and asymmetric battles (e.g. `1c3s3z` and `5m_vs_6m`), and varying number of units (e.g. 2 units in `2c_vs_64zg` and 10 units in `MMM2`), underpinning its robustness. In contrast, other algorithms have varying performance, e.g. QMIX does not always outperform VDN and LIIR is only competitive on certain scenarios. LICA also has fast convergence partly due to our proposed adaptive entropy regularization, verifying its ease of training. For easy scenarios, all methods solve `8m` reasonably well due to the symmetry and homogeneity of the agent groups, but the heterogeneous agents in `2s3z` and `1c3s5z` introduce more diverse joint actions and complex cooperation behaviors that slow down convergence for most methods, while LICA is unaffected due to its representational capacity. For the asymmetric `5m_vs_6m`, basic agent coordination alone such as "focus firing" [4, 20, 21, 3] no longer suffices and consistent success requires complex cooperative strategies such as pulling back units with low health during combat. In `2c_vs_64zg`, we observe that while LICA reaches nearly perfect win rate, it underperforms LIIR in convergence speed. We argue that this is because an optimal control of only 2 Colossus units may prioritize individual performance over cooperation (also suggested by the competitive performance of VDN over QMIX) where LIIR may benefit due to its explicit formulation of an individual reward. Nevertheless, on the most challenging `MMM2` scenario involving 10 units of 3 types, LICA's significant advantage confirms that its high representational complexity for credit assignment can lead to more superior policies.

4.3 Component Studies

4.3.1 Mixing Critic

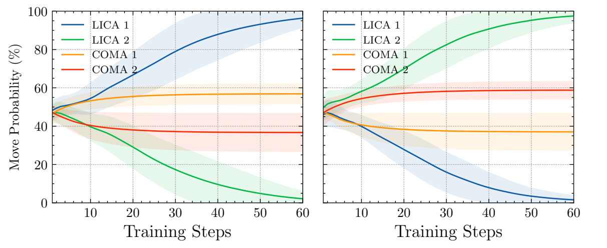

Figure 5: 1-Step Traffic Junction agent `move` probabilities throughout training for the two optimal outcomes. LICA agents learn to pass in order much faster than COMA agents.

1-Step Traffic Junction. To verify the credit assignment capacity of LICA, we introduce a simple 1-step environment where multiple optimal joint actions exist. The motivation is that explicit assignment methods based on difference rewards have limited capacity for complex credit assignment, and having multiple optima is one such scenario that is also common in the real world (e.g. traffic management). The 1-step game involves two vehicles controlled by two separate agents attempting to pass a traffic junction in order. Each agent can either `pass` or `wait`; if both try to `pass` or `wait` at the same time, they receive a shared reward of 0; otherwise they receive a shared reward of 1, resulting in two optimal joint actions. Taking COMA [4] as a representative example, difference reward based agents will expect a counterfactual advantage of 0 over the four possible joint actions; that is, if the centralized critic captures the true joint action value function, then COMA agents perform as good as *random agents* (though in practice, imperfect value functions and/or random exploration may allow agents to converge to optimal policies). To experiment, we train LICA and COMA agents for 500 random initializations each with 60 training iterations and show the mean and standard deviation of the agent action probabilities for the two optimal outcomes in Fig. 5. The fast convergence speed and low policy entropy verify the effectiveness of LICA's implicit credit assignment.

Ablations. We further perform an ablation study on StarCraft II `5m_vs_6m` (Hard) where the mixing critic is replaced with an MLP which takes the concatenated state and actions as the input, yielding a single-critic variant of MADDPG [13]. The results of two runs ($\lambda \in \{0.03, 0.04\}$, labeled with "(MLP)") are reported in Fig. 2 (b,c). Given similar levels of exploration as LICA ($\lambda = 0.09$), the poor performance clearly indicates the necessity of our mixing critic to learn optimal joint actions.

4.3.2 Adaptive Entropy Regularization

We further perform an ablation study to verify the effectiveness of our adaptive entropy regularization and report the results in Fig. 2. We first observe that the adaptive control of regularization strength (with λ) leads to sustained levels of exploration throughout training, which is not the case with the vanilla entropy regularization (with β). This property also makes tuning regularization strengths easier: increasing the strength from the sub-optimal $\lambda = 0.06$ where the policy converges prematurely to $\lambda = 0.09$ causes a *predictable* change in the entropy trajectory (Fig. 2 (b)); this is not the case when tuning the default entropy regularization ($\beta \in \{0.12, 0.14, 0.16\}$), where slight changes in initial strength can lead to similar or vastly different entropy trajectories. We also see that despite $\beta = 0.16$ gives reasonable win rates (already outperforming all other methods in Fig. 4 (d)), the agent exploration levels are drastically different before and after converging to exploitable policies (i.e. leap in win rates); in contrast, adaptive regularization encourages a consistent exploration level where the policy steadily improves ($\lambda = 0.09$). We believe that given a large β and sufficient training, the vanilla formulation can ultimately converge to similar performance due to LICA's inherent high capacity, though the empirical results justify the need for our proposed adaptive entropy regularization.

5 Conclusion

We present LICA, a new policy-based multi-agent actor-critic method that aims to implicitly address the credit assignment problem by learning the decentralized policies directly from their joint action Q-values. Compared to previous methods, LICA has a simpler and more general formulation that also has a larger capacity for complex credit assignment as verified by extensive experiments. To mitigate early convergence to sub-optimal policies, we further introduce a practical technique to dynamically adjust the strength of entropy regularization for encouraging sustained agent exploration throughout training. For immediate future work, LICA can be extended to continuous action domains such as physical control since there are no inherent constraints on the inputs to the mixing critic. It is also worth exploring further theoretical properties of our adaptive entropy regularization scheme.

Broader Impact

As many complex real-world problems can be formulated as multi-agent cooperative games, this work provides an effective approach to these problems. For example, decentralized agents can be applied to network routing optimization to speed up transmission, traffic management with autonomous vehicles to maximize traffic flow, and efficient package delivery with swarms of drones to reduce delivery costs. However, since our method relies on deep neural networks to implicitly attribute shared outcomes of the agent group to the individual agents, it faces the "black box problem" where behaviors of the individual agents may not be interpretable or rational from the human perspective. Furthermore, when maximizing a shared reward in multi-agent cooperative settings without considering the status of the individual agents, ethical issues may arise when the optimal joint action requires sacrificing certain agents. Using the task of traffic management with autonomous vehicles as an example, maximizing the total traffic volume could lead to indefinite delays for a subset of the vehicles.

References

[1] Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning*, pages 151–160, 2019.

[2] 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.

[3] Yali Du, Lei Han, Meng Fang, Ji Liu, Tianhong Dai, and Dacheng Tao. Liir: Learning individual intrinsic reward in multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 4405–4416, 2019.

[4] Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[5] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

[6] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1352–1361. JMLR. org, 2017.

[7] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[8] Maximilian Hüttenrauch, Adrian Šošić, and Gerhard Neumann. Guided deep reinforcement learning for swarm systems. *arXiv preprint arXiv:1709.06011*, 2017.

[9] Shariq Iqbal and Fei Sha. Actor-attention-critic for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 2961–2970, 2019.

[10] Kenneth E Kinnear, William B Langdon, Lee Spector, Peter J Angeline, and Una-May O'Reilly. *Advances in genetic programming*, volume 3. MIT press, 1994.

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

[12] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[13] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*, pages 6379–6390, 2017.

[14] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. Maven: Multi-agent variational exploration. In *Advances in Neural Information Processing Systems*, pages 7611–7622, 2019.

[15] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[16] Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2775–2785, 2017.

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

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

[19] Scott Proper and Kagan Tumer. Modeling difference rewards for multiagent learning. In *AAMAS*, pages 1397–1398, 2012.

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

[21] 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.

[22] John Schulman, Xi Chen, and Pieter Abbeel. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017.

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[24] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, 2014.

[25] 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*, 2019.

[26] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th international conference on autonomous agents and multiagent systems*, pages 2085–2087. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[27] Kagan Tumer and Adrian Agogino. Distributed agent-based air traffic flow management. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pages 1–8, 2007.

[28] Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. Shapley Q-value: A Local Reward Approach to Solve Global Reward Games. In *Thirty-fourth AAAI conference on artificial intelligence*, 2020.

[29] Ronald J Williams and Jing Peng. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

[30] Dayong Ye, Minjie Zhang, and Yun Yang. A multi-agent framework for packet routing in wireless sensor networks. *sensors*, 15(5):10026–10047, 2015.