Revisiting Some Common Practices in Cooperative Multi-Agent Reinforcement Learning

Wei Fu [1] **Chao Yu** [2] **Zelai Xu** [2] **Jiaqi Yang** [3] **Yi Wu** [1 4]

Abstract

Many advances in cooperative multi-agent reinforcement learning (MARL) are based on two common design principles: value decomposition and parameter sharing. A typical MARL algorithm of this fashion decomposes a centralized Q-function into local Q-networks with parameters shared across agents. Such an algorithmic paradigm enables centralized training and decentralized execution (CTDE) and leads to efficient learning in practice. Despite all the advantages, we revisit these two principles and show that in certain scenarios, e.g., environments with a highly multi-modal reward landscape, value decomposition, and parameter sharing can be problematic and lead to undesired outcomes. In contrast, policy gradient (PG) methods with individual policies provably converge to an optimal solution in these cases, which partially supports some recent empirical observations that PG can be effective in many MARL testbeds. Inspired by our theoretical analysis, we present practical suggestions on implementing multi-agent PG algorithms for either high rewards or diverse emergent behaviors and empirically validate our findings on a variety of domains, ranging from the simplified matrix and grid-world games to complex benchmarks such as StarCraft Multi-Agent Challenge and Google Research Football. We hope our insights could benefit the community towards developing more general and more powerful MARL algorithms. Check our project website at https://sites.google.com/view/revisiting-marl.

[1]Institute for Interdisciplinary Information Sciences, Tsinghua University, China [2]Department of Electronics Engineering, Tsinghua University, China [3]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, USA [4]Shanghai Qi Zhi Institute, China. Correspondence to: Wei Fu <fuwth17@gmail.com>, Yi Wu <jxwuyi@gmail.com>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

1. Introduction

Value decomposition has become the most popular paradigm for tackling cooperative multi-agent reinforcement learning (MARL) problems (Sunehag et al., 2017; Wang et al., 2020c; Rashid et al., 2018; Wang et al., 2020a; Hu et al., 2021). Under this paradigm, algorithms decompose the global Q-function into local Q-functions that satisfy the Individual Global Max (IGM) principle (Son et al., 2019). Then the optimal global policy can simply be derived via greedily selecting locally optimal actions for each agent. This process enables centralized training and decentralized execution (CTDE) (Foerster et al., 2016a; Gupta et al., 2017), and the factorized representation has also led to enormous benefits, including credit assignment (Zhang et al., 2020), emergent communication (Wang et al., 2019b), exploration (Liu et al., 2021), etc. In addition, the local Q-networks are typically implemented with shared parameters leading to a shared replay buffer across agents, which not only reduces the number of total parameters but also results in more stable training (Christianos et al., 2020).

Despite all the aforementioned advantages, in this paper, we revisit popular cooperative MARL methods and argue that some common practices including value-decomposition representation and parameter sharing, can be problematic in certain scenarios. We start our discussion by considering a simple 2-by-2 matrix game, the *XOR game*, which has two symmetric global optimal solutions. We prove that even for this particularly simple game, a value-decomposition-based algorithm cannot represent the underlying payoff structure, and thus may not learn the optimal policy via factorized local Q-networks. Interestingly, we notice that in this example, policy gradient (PG) with individual policies provably converges to either of the two optimal solutions. In addition, we also introduce another policy-based *auto-regressive* representation and show that PG with such a representation can learn *all* the strategy modes even on the N-player version of the XOR game, which has exponentially many optimal solutions. Our theoretical findings suggest that on certain multi-modal problems, policy gradient, which is often considered sample-inefficient as an on-policy method, can be preferable compared to popular value-based learning methods. This insight can be served as initial evidence to

partially explain the recent empirical observations that PG methods perform surprisingly well on many MARL benchmarks (Papoudakis et al., 2020; Yu et al., 2021).

We further extend our study to a grid-world game, *Bridge*, which is a temporal variant of the XOR game. Based on our theoretical insights, we present two practical suggestions for applying PG to cooperative Markov games concerning different evaluation targets. For the highest final rewards and the fastest convergence, it can be beneficial to adopt an individual or agent-specific policy. For a policy that can capture multi-modal behaviors, we propose an attention-based auto-regressive policy representation that has a minimal computation overhead while maximally retaining the expressiveness power of a joint policy. A few training techniques for efficiently learning multi-modal auto-regressive policies are also introduced in the Bridge game.

We also validate our two suggestions in more complex domains including the StarCraft Multi-Agent Challenge (SMAC) (Rashid et al., 2019) and Google Research Football (GRF) (Kurach et al., 2019). We show that in many scenarios, using individual or agent-specific policies can further improve the performance of the state-of-the-art multi-agent proximal policy optimization (PPO) method (Yu et al., 2021). Meanwhile, using our auto-regressive policy learning method, we can discover interesting emergent behaviors that are never discovered by existing fully decentralized PG or value-based methods.

We emphasize that our work is *NOT* claiming for a new algorithm that is universally applicable. Instead, **we attempt to provide concrete analysis to explain recent empirical evidence on PPO's effectiveness and present practical alternatives of interest in certain scenarios.** We hope our theoretical and empirical findings can bring useful insights to the community towards developing more general and more powerful MARL algorithms in the future.

2. Related Work

Centralized Training with Decentralized Execution (CTDE) (Lowe et al., 2017) is perhaps the most popular framework for MARL. The fundamental idea is to adopt global information for simplified training (i.e., by reducing a POMDP to an MDP) while maintaining policies that only take local information for producing actions. Value decomposition (VD) methods perform centralized Q-learning and represent the global Q-function as a combination of local Q-networks following the Individual-Global-Max (IGM) principle (Son et al., 2019), i.e., the optimal joint action should be equivalent to the collection of greedy local actions of each agent, which naturally enables CTDE. Representative VD methods, such as Value Decomposition Network (VDN) (Sunehag et al., 2017), QMIX (Rashid et al., 2018), QTRAN (Son et al., 2019), QPLEX (Wang et al., 2020a), and other variants (Rashid et al., 2020; Yang et al., 2020), have been providing increasing expressiveness power towards the function class satisfying the IGM principle. Other works also introduce policy networks into the VD framework (Su et al., 2020; Wang et al., 2020c; Zhang et al., 2021; Wu et al., 2021) or further adapt the decomposed Q-networks towards communication learning (Wang et al., 2019b), zero-shot adaptation (Hu et al., 2020b), credit assignment (Zhang et al., 2020), or structural emergent behavior (Wang et al., 2019a; 2020b). Despite the success of the IGM principle, we argue in this paper that in certain scenarios with a highly symmetric multi-modal reward structure, VD methods or even the IGM principle itself can be problematic and therefore suggest feasible alternative directions for future MARL research.

Another line of MARL methods, such as COMA (Foerster et al., 2018), adopt policy gradient (PG) and follow CTDE by learning local policies with a centralized value function as the critic. Due to the on-policy fashion, PG methods are typically believed to be less sample efficient and therefore less utilized in the academic literature with limited computation resources. Whereas, some recent empirical studies (de Witt et al., 2020; Papoudakis et al., 2020; Yu et al., 2021) demonstrate that with proper input representation and hyper-parameter tuning, multi-agent PPO can achieve surprisingly strong performance and sample efficiency in many cooperative MARL benchmarks compared to off-policy VD methods. Our theoretical analysis partially justifies these recent empirical findings and provides insights toward more effective implementation of PG methods.

Multi-modality is a common phenomenon in many multi-agent games. For example, many works have shown that diverse emergent behaviors can be obtained under a simple reward function (Lowe et al., 2019; Baker et al., 2019; Tang et al., 2021). There is also a direction of research focusing on discovering diverse behaviors, e.g., by evolution methods (Cully et al., 2015; Pugh et al., 2016), population-based training (Vinyals et al., 2019; Parker-Holder et al., 2020; Lupu et al., 2021) or iterative policy optimization (Lanctot et al., 2017; Masood & Doshi-Velez, 2019; Zahavy et al., 2021; Zhou et al., 2022). In game theory, the multi-modality issue is closely related to the concept of equilibrium refinement (Kreps & Wilson, 1982), which studies selecting the desired Nash equilibrium. The XOR game can be also interpreted as a cooperative version of the chicken game, which corresponds to the concept of correlated equilibrium (Aumann, 1974). Our work points out that the existence of multiple optimal strategies can be an issue for existing MARL methods. We also develop an auto-regressive representation for representing a single multi-modal policy. Existing MARL literature primarily applies sequential execution for learning a consensus solution among agents by assuming all

optimal solutions are known in advance (Boutilier, 1996). By contrast, we focus on directly learning a multi-modal policy from scratch, which is more challenging. Developing more powerful algorithms for diverse behaviors or sophisticated equilibria is beyond the scope of this paper.

We remark that the auto-regressive representation is widely adopted in the language generation literature, from N-gram models (Damerau, 2018) to recent attention-based architectures (Vaswani et al., 2017; Devlin et al., 2018), which is the foundation of our proposed auto-regressive policy learning. There is a recent trend in developing non-auto-regressive text generation methods (Gu et al., 2017; Qian et al., 2020) for fast convergence and computational efficiency, which conceptually motivates our proposal of individual policy learning. In recent MARL literature, sequential policy update has been proposed to guarantee monotonic policy improvement (Kuba et al., 2022), which improves the performance of PG methods under CTDE. Instead of focusing on efficient training for decentralized execution, our proposed auto-regressive representation improves the policy modeling capacity and maintains the full expressiveness power of a centralized policy. Hence, we can learn diverse behaviors using a single auto-regressive policy. Finally, auto-regressive policy learning requires broadcasting each agent's action to all the following agents, which assumes a perfect communication channel on actions and is therefore related to multi-agent communication (Foerster et al., 2016b; Wang et al., 2019b).

3. Background

3.1. Notation

We study cooperative MARL under the framework of multi-agent Partially Observable Markov Decision Process (Boutilier, 1996; Kaelbling et al., 1998), which is defined as a tuple $\mathcal{M} = \langle n, \mathcal{S}, \mathcal{A}, \mathcal{O}, r, P, O, \gamma, H \rangle$. Here, $n \in \mathbb{N}$ is the number of agents, $\mathcal{S}$ is the state space, $\mathcal{A}$ and $\mathcal{O}$ are the action and observation space of each agent, $r : \mathcal{S} \times \mathcal{A}^n \to \mathbb{R}$ is the reward function, P is the transition model, $O : \mathcal{S} \to \mathcal{O}$ is the observation function, γ is the discount factor, and H is the horizon. For state $s, s' \in \mathcal{S}$ and a joint action $\mathbf{a} \in \mathcal{A}^n$, the transition probability of reaching state s' from state s by executing action $\mathbf{a}$ is $P(s' \mid s, \mathbf{a})$. At timestep t, s_t denotes the state and each agent $i \in [n]$ receives a observation $o_t^i = O(i, s_t)$. Then, given the joint observation $\mathbf{o}_t = (o_t^1, o_t^2, \dots, o_t^n)$, agents output a joint action $\mathbf{a} \in \mathcal{A}^n$ according to the joint policy $\pi : \mathcal{O}^n \to \triangle(\mathcal{A}^n)$. We aim to find the optimal joint policy $\pi^\star$ to maximize the expected return, i.e.,

$$\pi^\star = \arg\max_{\pi} \mathbb{E}_{(s_t, \mathbf{a}_t) \sim (P, \pi)} \left[\sum_{t=1}^{H} \gamma^{t-1} r(s_t, \mathbf{a}_t) \right].$$

3.2. Value Decomposition in MARL

Value-based methods in cooperative MARL tasks aim to learn a global Q-function $Q_{\text{tot}} : \mathcal{S} \times \mathcal{A}^n \to \mathbb{R}$ to estimate the future expected return given current state s_t and joint action $\mathbf{a}$. However, the global state s_t is typically unavailable during execution and the dimension of joint action space $\mathcal{A}^n$ grows exponentially w.r.t. agent number n. Value decomposition (VD) addresses this issue by decomposing the global Q-value $Q_{\text{tot}}(s, \mathbf{a})$ into local Q-values $Q_i(o^i, a^i)$. In particular, VD methods learn local Q-functions $Q_i : \mathcal{O} \times \mathcal{A} \to \mathbb{R}$ and a mixing function $f_{\text{mix}}(\cdot; s)$ conditioning on state s to represent the global Q-value $Q_{\text{tot}}(s, \mathbf{a})$ by

$$Q_{\text{tot}}(s, \mathbf{a}) = f_{\text{mix}}(Q_1(o^1, a^1), \dots, Q_n(o^n, a^n); s). \quad (1)$$

For a reduced model size and efficient training, the parameters for each local Q-function Q_i are shared, which is called *parameter sharing*. f_{mix} is enforced to satisfy the *Individual-Global-Max* (IGM) (Son et al., 2019) principle such that

$$\arg\max_{\mathbf{a} \in \mathcal{A}^n} Q_{\text{tot}}(s, \mathbf{a}) = \cup_{i=1}^{n} \{ \arg\max_{a^i \in \mathcal{A}} Q_i(o^i, a^i) \}. \quad (2)$$

Therefore, the optimal joint action can be easily derived by independently choosing a local optimal action from each local Q-function Q_i, which enables centralized training and decentralized execution (CTDE). Different VD methods propose different representations for the mixing function with various expressiveness capabilities. Notably, QPLEX (Wang et al., 2020a) proposed the first architecture that can represent any possible mixing function under the IGM principle.

3.3. Policy Learning in MARL

Policy gradient (PG) and its variants (Williams, 1992; Schulman et al., 2017; Foerster et al., 2018) directly optimize the policy π by gradient descent. In cooperative MARL tasks, popular PG methods follow CTDE by learning an individual actor $\pi_{\theta_i} : \mathcal{O} \to \triangle(\mathcal{A})$ for agent i parameterized by θ_i, and a centralized value function $V_{\psi_i} : \mathcal{S} \to \mathbb{R}$ (i.e., critic) parameterized by ψ_i. In order to leverage global information, the value function takes the global state s (Yu et al., 2021) or the combination of all the local observations $[o^1, \dots, o^n]$ (Lowe et al., 2017) as its input for an accurate global value estimate. The majority of existing multi-agent PG works adopt parameter sharing, i.e., $\theta_1 = \cdots = \theta_n = \theta$ and $\psi_1 = \cdots = \psi_n = \psi$, for reducing model size. Such a decentralized formulation naturally induces the following implicit joint policy with a *fully independent* factorization:

$$\pi(\mathbf{a} \mid \mathbf{o}) \approx \prod_{i=1}^{n} \pi_{\theta_i}(a^i \mid o^i). \quad (3)$$

With such a policy representation, each agent i simultaneously performs policy learning by running the single-agent policy gradient over its individual policy π_{θ_i} and assuming other policies unchanged.

4. A Motivating Example: XOR Game

We start our analysis by considering a cooperative game in the simplest form of a 2-by-2 matrix game called *XOR game* as shown in Table 1. In this 1-step stateless game, each agent has two possible actions and they will get a positive reward only if they output different actions. There are two symmetric and equally optimal strategies in this game. We will show that value-decomposition methods may fail to converge on this particularly simple game (Sec. 4.1) but policy gradient methods can be proved to converge (Sec. 4.2). Lastly, we will show a simple PG variant that can learn a policy covering all possible modes even in the n-player extension of the XOR game (Sec. 4.3).

0	1
1	0

Table 1. Payoff matrix of XOR game.

4.1. Value Decomposition in XOR game

Let's assume each agent has two actions, 1 and 2. In this stateless setting, the global Q-value degenerates to $Q_{\text{tot}}(a^1, a^2)$ and can be further decomposed as

$$Q_{\text{tot}}(a^1, a^2) = f_{\text{mix}}(Q_1(a^1), Q_2(a^2)), \tag{4}$$

where f_{mix} needs to satisfy the IGM principle.

Theorem 4.1. *Value decomposition (Eq. (4)) cannot represent the underlying global Q-function in XOR game.*

Proof. In XOR game, the desired global Q-value is the payoff. Let $(\alpha_1, \alpha_2, \beta_1, \beta_2) = (Q_1(1), Q_1(2), Q_2(1), Q_2(2))$. Suppose for contradiction that it could represent. Then $f_{\text{mix}}(\alpha_1, \beta_2) = f_{\text{mix}}(\alpha_2, \beta_1) = 1$. Now if $\alpha_1 \geqslant \alpha_2$ then by monotonicity we have

$$0 = Q_{\text{tot}}(1,1) = f_{\text{mix}}(\alpha_1, \beta_1) \geqslant f_{\text{mix}}(\alpha_2, \beta_1) = 1, \tag{5}$$

contradiction. Otherwise, we have

$$0 = Q_{\text{tot}}(2,2) = f_{\text{mix}}(\alpha_2, \beta_2) \geqslant f_{\text{mix}}(\alpha_1, \beta_2) = 1, \tag{6}$$

contradiction. □

Remark: Theorem 4.1 suggests that *with VD, Q-learning will NOT converge on XOR game*. We also conduct empirical experiments by applying state-of-the-art VD methods, including VDN (Sunehag et al., 2017), QMIX (Rashid et al., 2018) and QPLEX (Wang et al., 2020a), by fitting the payoff matrix and average the results over 6 random seeds. Note that since the XOR game is stateless, Q-values are just separate learnable constants. Curves of loss are shown in Fig. 1. None of the VD methods achieve a zero regression loss as pointed out by Theorem 4.1, even for the most representative method QPLEX. We can also observe that QPLEX suffers from significant loss fluctuations. This implies that the eventual policy induced by the Q-functions learned by VD-based methods can be completely arbitrary due to the condition of the optimization process. We additionally emphasize that Theorem 4.1 holds whenever the local Q networks are shared or not — the failure is due to the limited representation power of VD methods.

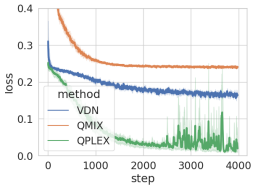

Figure 1. Loss in XOR game.

4.2. Policy Gradient in XOR game

We refer *shared policy learning* (PG-sh) to the setting of learning a single policy parameter θ for all the agents, i.e., $\pi_1 = \pi_2 = \pi_\theta$, and refer *individual policy learning* (PG-Ind) to the setting of learning a separate policy parameter θ_i for each agent's policy π_{θ_i}. We will show by the following theorems that a shared policy cannot solve the XOR game while individual policy learning can provably converge to an optimal solution.

Theorem 4.2. *Shared policy learning cannot learn an optimal policy for the XOR game.*

Proof. Let π_i be a shared policy and a^i denote the action by agent i. Let $\alpha = \mathbb{P}(a^i = 0)$. The expected return of π_i is

$$\begin{aligned}\mathbb{E}[R(\pi_i)] &= \mathbb{P}(a^1 = 0, a^2 = 1) \\ &\quad + \mathbb{P}(a^1 = 1, a^2 = 0) \qquad (7) \\ &= 2\alpha(1-\alpha) \leqslant 0.5 < 1, \qquad (8)\end{aligned}$$

but the optimal return is 1. □

Lemma 4.3. *Individual policies can represent the optimal policy for the XOR game.*

Proof. In the XOR game, there exists a deterministic optimal (joint) policy. Then, we can construct deterministic individual policies for each agent w.r.t. the global optimum. □

Theorem 4.4. *Individual policy learning via stochastic policy gradient can learn an optimal policy in the XOR game.*

Proof. Previous works have shown that SGD can escape saddle points and converge to local optima under mild assumptions (Kleinberg et al., 2018). For any individual policy π_i, if it has a positive reward but not the maximum reward (which is 1), then we can find a better policy in the neighborhood by increasing probability along the direction of the permutation where π_i puts maximum probability. Otherwise, if it has a zero reward, then we can find a better policy by increasing probability in all directions. □

Remark: Through the theoretical analysis in Sec. 4.1 and 4.2, we show that even a simple 2-by-2 matrix game, i.e., XOR game, can be a counterexample that VD-based MARL algorithms fundamentally fail to converge. By contrast, PG methods, as an SGD-based approach, can provably converge to an optimum. This fact suggests that, even though PG methods are much less utilized in MARL compared with VD methods, it can be preferable in certain cases, e.g., games with multiple strategic modalities, which can be common in real-world applications.

4.3. Auto-Regressive Policy Learning

Although we have proved in the previous discussions that individual PG can effectively learn an optimal mode in the XOR game, the strategy mode that it finally reaches can be highly dependent on the initialization of the policies. Thus, a natural question will be:

Can we learn a single policy that can cover all the optimal modes?

We remark that learning multi-modality policies is meaningful for a wide range of focuses, such as emergent behavior (Tang et al., 2021), exploration (Mahajan et al., 2019), learning to adapt (Lanctot et al., 2017; Balduzzi et al., 2019) or interacting with humans (Hu et al., 2020a).

Let's re-visit a global policy $\pi(a^1, a^2)$, which models the joint action probabilities for both agents. It would be trivial to construct a multi-modal global policy such that it has equal chances to output joint actions of either $(2, 1)$ or $(1, 2)$. However, following the decentralized policy gradient formulation (Eq. 3), the factorized representation $\pi(a^1, a^2) \leftarrow \pi(a^1)\pi(a^2)$ is only able to represent one particular mode. Therefore, for a multi-modal policy, we need to develop a policy representation with a stronger expressiveness of the joint policy and with minimal computation overhead compared with independent policies (Eq. 3).

We propose to represent the policy in an *auto-regressive* form, i.e., $\pi(a^1, a^2) = \pi(a^1)\pi(a^2|a^1)$ in XOR game.

Formally, let's consider a general *auto-regressive policy representation* for n agents under $X = \{x_1, \dots, x_n\}$, a permutation over 1 to n, denoting an *execution order* for the agents. Given any execution order X, we can factorize the joint policy π_θ into the form of

$$\pi_\theta(\mathbf{a} \mid \mathbf{o}) \approx \prod_{i=1}^{n} \pi_{\theta^{x_i}}(a^{x_i} \mid o^{x_i}, a^{x_1}, \dots, a^{x_{i-1}}), \qquad (9)$$

where the action produced by agent x_i depends on its observation o^{x_i} and all the actions from its previous agents $x_1, \dots, x_{i-1}$ under the execution order X. Note that the auto-regressive factorization is *equivalent to* the joint policy in the fully observable setting while maintaining a strong expressiveness in Dec-POMDPs. Moreover, by sequentially generating actions according to X, the output dimension of each agent's policy $\pi_{\theta_{x_i}}$ remains the same as individual policy representation, which results in a minimal policy computation overhead.

We remark that *auto-regressive policy* in Eq. (9) is based on a different factorization scheme from the classical decentralized learning paradigm in Eq. (3). To sufficiently distinguish these two factorization schemes in our paper, we call the representation in Eq. (3) *independent policy*.

In theory, we argue that an auto-regressive policy could learn substantially more diverse policies on the n-player variant of XOR game, called *permutation game*. Here, we measure the diversity of a policy π by the entropy of its trajectories, i.e., $\mathcal{H}(\pi) = -\mathbb{E}_{\tau\sim\pi}[\log p_\pi(\tau)]$, where τ denotes a trajectory and $p_\pi(\tau)$ is the probability of τ under π.

Definition 4.5 (Permutation game)**.** An n-agent permutation game is a stateless Dec-POMDP. Each agent i has n actions $\mathcal{A} = \{1, \cdots, n\}$. The reward function is $r(a^1, \cdots, a^n) = \mathbb{I}\{(a^1, \cdots, a^n) \text{ is a permutation}\}$.

Note that the XOR game is the 2-player permutation game.

Theorem 4.6. *For n-agent permutation game, the optimal independent policy has entropy $\mathcal{H}(\pi^\star_{\text{ind}}) = 0$, while the optimal auto-regressive policy could have entropy $\mathcal{H}(\pi^\star_{\text{auto}}) = \log(n!)$.*

Proof. The optimal independent policy is given by $\pi^\star_{\text{ind}} = (\pi_1, \cdots, \pi_n)$, where $\pi_1 = \cdots = \pi_{n-1} = 0$ and $\pi_n = 1$. Note that this is a deterministic policy, so $\mathcal{H}(\pi) = 0$.

The optimal auto-regressive policy is given by $\pi^\star_{\text{auto}} = (\pi_1, \cdots, \pi_n)$, where $\pi_i(a_i|a_1, \cdots, a_{i-1}, s_0) = \frac{1}{n-i}\mathbb{I}\{a_i \neq a_1, a_2, \cdots, a_{i-1}\}$. Note that the joint policy is $\pi = \text{Unif}(\{(a_1, \cdots, a_n) \mid a_i \text{ is a permutation}\})$, which is a uniform distribution over $n!$ permutations, so its entropy is $\mathcal{H}(\pi) = \log(n!)$. □

We empirically evaluate the effectiveness of auto-regressive policy learning in the 4-player permutation game. We plot the entropy of policies learned by auto-regressive (AR) learning and individual (Ind.) learning in Fig. 2. The results are averaged over 3 seeds with negligible variance. The AR policy consistently learns a multi-modal behavior while individual learning always converges to a single mode with a zero policy entropy. Moreover, we also pick policies trained on a particular trial and illustrate the distribution of joint actions

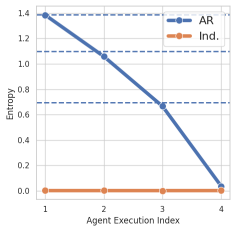

Figure 2. Entropy during action selection in the 4-player permutation game.

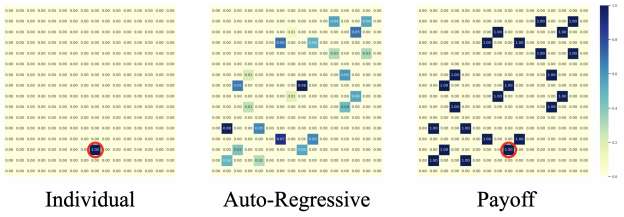

Figure 3. Heatmap demonstrating the frequency of every possible joint action during evaluation of 1000 episodes. X-axis indicates the joint action of the first two agents and Y-axis indicates the joint action of the last two agents, which forms a 16×16 matrix. The red circles indicate the single optimal mode discovered by individual PG. Auto-regressive PG can discover all the permutations that solve the game.

in Fig. 3. Note that there are a total of $4^4 = 256$ possible joint actions in the 4-player permutation game while only $4! = 24$ of them yield a positive reward. We can observe that the AR policy successfully covers all the optimal modes while independent policies only converge to a specific mode.

Remark: Auto-regressive policy learning is a minimal approximation of centralized learning, which is beyond the setting of decentralized learning on Dec-POMDP (Oliehoek et al., 2008). Most decentralized MARL methods formulated on Dec-POMDP are based on independent policy factorization (Eq. (1, 2, 3)), which has a fundamental limitation — the joint policy cannot represent *all* the optimal modes. Therefore, we propose Eq. (9) to overcome this expressiveness limitation via a mild additional assumption (communication of actions) and computation overhead.

5. Bridge: a Temporal XOR Game

We further extend our study to a grid-world Markov game, *Bridge*, as shown in Fig. 4. In this game, two agents spawn symmetrically at the two corners of the map. Each agent needs to get through the bridge to reach the spawn point of the other agent. At each timestep, each agent receives a penalty which is proportional to the distance between the current position and its goal. More environment details can be found in Appendix B.1.

In this game, each grid may contain only one agent. Swapping positions between two agents within a single timestep is not permitted. Therefore, two agents cannot pass the bridge simultaneously. As shown in the top row from Fig. 4, one agent must temporarily leave the bridge and wait until the other one passes by. By contrast, if both agents perform the same actions and both enter the bridge, it will result in a dead loop, and additional penalties may be incurred for both agents due to time waste. *Bridge* can be interpreted as a temporal version of the XOR game since the two agents need to perform different macro actions, i.e., either wait or move, to achieve the optimal reward. Likewise, there are symmetric optimal strategies.

5.1. Agent-Specific Policy Learning on Bridge

Based on our theoretical analysis in Sec. 4, individual policy learning with unshared parameters (PG-Ind) would be preferred. However, learning a separate policy for each agent introduces more model parameters and may challenge optimization. We consider an alternative to PG-Ind by learning an *agent-ID-conditioned policy* (PG-ID): parameters are still shared across agents but the observation o^i is concatenated with the one-hot agent ID as an additional policy input feature, which enables the policy to become *agent-specific*. The effectiveness of such an ID-conditioning technique was also studied by Yu et al. (2021).

Method	Reward
QMIX	-1.18(0.70)
QPLEX	-1.48(1.30)
PG-sh	-0.64(0.02)
PG-ID	**-0.48(0.00)**
PG-Ind	**-0.48(0.00)**
optimal	**-0.48**

Table 2. Evaluation results in Bridge over 3 seeds.

We remark that PG-Ind and PG-ID are two implementation choices to realize an individual policy π_i conditioning on agent index i. PG-Ind parameterizes each π_i as $\pi_{\theta_i}(a^i \mid o^i)$ (conditioning by using different parameters) while PG-ID adopts $\pi_\theta(a^i \mid o^i; i)$ (conditioning by different inputs). Both two choices are valid under the universal approximation theorem (Hornik et al., 1989). In practice, $\pi_{\theta_i}(a^i \mid o^i)$ has a stronger conditioning power than $\pi_\theta(a^i \mid o^i; i)$ but may be harder to train due to more parameters and fewer data for each π_{θ_i}.

We compare the empirical performances of agent-specific policy learning, including PG-Ind and PG-ID, with shared policy learning (PG-sh) as well as popular VD algorithms, including QMIX and QPLEX, on the 2-player Bridge game. All the algorithms use the same batch size and are properly trained with sufficient samples. The final evaluation rewards are shown in Table. 2. Both VD methods empirically fail to solve the game and produce particularly poor final rewards. Regarding PG methods, since the game is fully symmetric, a shared policy may not solve this game (PG-sh), while all the agent-specific PG methods, i.e., PG-ID and PG-Ind, can consistently learn the optimal behavior with negligible variance. Moreover, since the VD methods may not converge in this Bridge game, we can observe that their variances are substantially higher than PG methods.

5.2. Learning Multi-Modal Behavior with Auto-Regressive Policy

We further apply auto-regressive policy to learn multi-modal behaviors in *Bridge* game. To effectively discover multi-modal behavior in general Markov games, we propose the

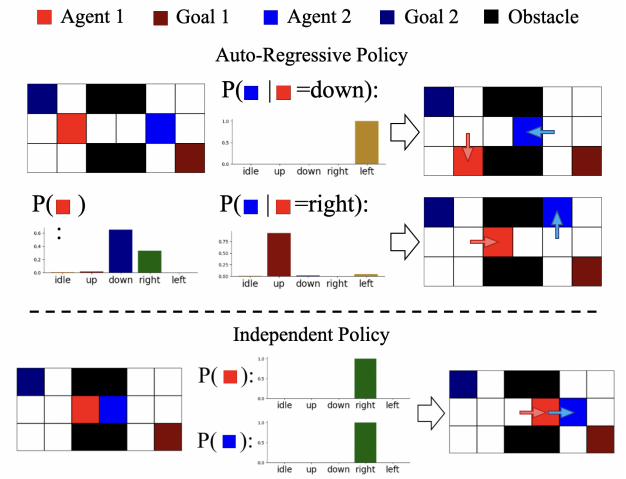

Figure 4. (top) Multi-modal behavior by PG-AR in Bridge game. Depending on the action of agent 1 (red), agent 2 (blue) makes different decisions. (bottom) Uni-modal behavior by PG-Ind Agent 1 (red) always passes the bridge first.

following training paradigm.

Attention-based policy: We propose to use an attention-based architecture (Vaswani et al., 2017). We convert local observations and input actions into separate embeddings and use a self-attention mechanism to effectively derive a combined input representation. Such a representation is size-invariant w.r.t. the number of input actions, which further enables parameter sharing. Self-attention also ensures the output properly conditions on the input actions, which helps learn a more coordinated joint policy.

Multi-step optimization: Given an execution order X, each agent x_i will output an action from its own conditioned policy $\pi^{x_i}(a^{x_i}|o^{x_i}, a^{x_{1:i-1}})$. Note that since all the output actions are made within the same timestep, every policy π^{x_i} has the same return R. Therefore, rather than solely optimize the joint policy π, we can optimize all these n "partial" policies by policy gradient as follows:

$$\nabla J(\pi) = \sum_{j=1}^{n} \mathbb{E}\left[-R \cdot \sum_{i=1}^{j} \nabla \log \pi(a^{x_i}|o^{x_i}, a^{x_{1:i-1}})\right]. \tag{10}$$

Randomized execution order: Any execution order over n should result in the same factorized distribution for the joint policy $\pi_\theta(\mathbf{a} \mid \mathbf{o})$. Therefore, to prevent the parameterized policy from overfitting a particular order, we randomized the execution order X at each timestep during training.

Results: We compared the entropy of learned AR policy (PG-AR) with PG-Ind and perform ablation studies over the techniques, i.e., self-attention (Attn.), multi-step optimization (MO), and randomized execution order (RO). The evaluation results are shown in Table 3. We can observe that individual learning (PG-Ind) converges to a specific mode with zero entropy while AR policies (PG-AR) can effectively produce high-entropy strategies. Note that all three proposed techniques are critical. We also illustrate the learned AR policy in the top part of Fig. 4, where agent 1 (red) captures two modes when entering the bridge, i.e., move right to pass the bridge and move down to wait. Based on the action of agent 1 (red), agent 2 (blue) can output the corresponding optimal action subsequently.

PG-Ind	PG-AR	w.o.MO	w.o.RO	w.o.Attn.
0.01(0.00)	0.74(0.03)	0.02(0.01)	0.71(0.28)	0.60(0.44)

Table 3. Policy entropy (standard deviation) at the state where both agents are at the bridge ends with ablation studies in Bridge.

Map	PG-ID	PG-sh.	PG-Ind.	RODE
1c3s5z	**100.0(0.0)**	97.4(1.0)	99.1(0.7)	**100.0(0.0)**
2s3z	**100.0(0.7)**	99.0(0.5)	99.1(0.9)	**100.0(0.0)**
3s_vs_5z	**100.0(0.6)**	96.7(1.8)	93.8(1.8)	78.9(4.2)
3s5z	**96.9(0.7)**	95.2(1.5)	80.4(3.3)	93.8(2.0)
3s5z_vs_3s6z	84.4(34.0)	42.3(4.0)	37.8(5.6)	**96.8(25.1)**
5m_vs_6m	**89.1(2.5)**	35.3(2.1)	44.4(2.9)	71.1(9.2)
6h_vs_8z	**88.3(3.7)**	79.9(4.8)	11.4(2.5)	78.1(37.0)
10m_vs_11m	**96.9(4.8)**	86.5(2.3)	78.4(2.7)	95.3(2.2)
corridor	**100.0(1.2)**	92.6(2.4)	82.2(1.8)	65.6(32.1)
MMM2	90.6(2.8)	**92.3(1.9)**	13.0(3.7)	89.8(6.7)

Table 4. Median evaluation winning rate (standard deviation) on selected SMAC maps over 6 random seeds.

6. Experiments on Popular MARL Testbeds

Based on the evidence from the Bridge game, we conduct experiments on popular MARL benchmarks, including StarCraft Multi-Agent Challenge (SMAC) (Rashid et al., 2019) and Google Research Football (GRF) (Kurach et al., 2019). We use the dense reward setting in both games. These two environments are intuitively multi-modal since professional football teams or video game players usually have different styles leading to a diverse collection of winning strategies. In this section, we want to show that 1) in complex multi-modal environments, *value decomposition and policy sharing can be outperformed by agent-specific policy learning methods*, and 2) *auto-regressive policy learning can discover interesting emergent behaviors requiring strong intra-agent coordination*, which are never discovered by existing multi-agent PG methods. Our implementation is based on the MAPPO project (Yu et al., 2021) with more details in Appendix C.

6.1. Learning Policies with Higher Rewards

We compare agent-specific policy learning, i.e., agent-ID-conditioned policy (PG-ID) and individual policy with unshared parameters (PG-Ind), with state-of-the-art VD-based

Scenario	PG-ID	PG-Ind.	CDS(QMIX)	CDS(QPLEX)
3v1	**90.7(1.5)**	**90.2(1.7)**	73.7(3.4)	83.6(4.0)
CA(Easy)	79.5(6.7)	**92.4(2.6)**	43.0(5.6)	40.4(4.8)
CA(Hard)	**68.2(2.0)**	**67.8(3.4)**	35.4(2.9)	34.8(3.4)
Corner	**27.3(1.3)**	21.6(2.4)	1.8(0.3)	20.8(1.7)
PS	43.4(8.8)	53.3(3.5)	83.5(4.0)	**86.8(1.9)**
RPS	66.6(3.1)	**78.8(1.5)**	65.5(7.0)	75.1(2.4)

Table 5. Median evaluation winning rate (standard deviation) in GRF academy scenarios over 6 random seeds. (CA=counter attack; PS=pass and shoot; RSP=run, pass and shoot).

algorithms, including RODE (Wang et al., 2020b) for SMAC and CDS (Li et al., 2021) for GRF. We also include the performance of shared policy learning (PG-sh).

Evaluation results are shown in Table 4 and Table 5 for SMAC and GRF respectively.

In SMAC, PG methods outperform VD methods on 9 out of 10 selected maps as reported by Yu et al. (2021). The agent ID can be critical and agent-conditioned policy (PG-ID) outperforms policies without agent ID input (PG-sh) in almost all the maps except *MMM2*, where the performance of PG-ID and PG-sh is comparable. For individual policies, it can achieve comparable performances with PG-ID on maps with a small number of agents, while on maps with a large number of agents, using unshared parameters may hurt performance. We believe this is due to the issue of model size.

In GRF, individual policies (PG-Ind) without parameter sharing achieve comparable or even higher results compared with agent-conditioned policies (PG-ID) in a total of 5 scenarios. Compared with the state-of-the-art algorithm, CDS, which combines a basic VD method with exploration rewards, PG-based methods, even without any intrinsic rewards, outperform CDS on a total of 5 scenarios except a simple scenario *pass-and-shoot*, which only has two agents. We believe this is due to PG converging towards a poor local optimum due to the deceptive distance-based rewards in GRF, which can be possibly addressed by leveraging more advanced exploration techniques in future work.

Practical Suggestion: Always include agent-specific information in the policy input; if there are not many agents in the game, individual policy learning may be worth trying.

6.2. Emergent Behavior by Auto-Regressive Modeling

We similarly apply auto-regressive policy learning (PG-AR) in SMAC and GRF. Due to an extremely high degree of freedom and the setting of dense reward, it is non-trivial to visually observe human interpretable multi-modal behaviors. Nevertheless, we still found interesting emergent behaviors discovered by auto-regressive policies: these emergent strategies require particularly strong coordination and are never discovered by other PG variants in our experiments.

Figure 5. Emergent behavior on the *2m_vs_1z* map of SMAC.

Figure 6. Emergent "*coordinated-spread-out*" behavior on the *3s_vs_5z* map of SMAC by the AR policy.

In SMAC, Fig. 5 visualizes the learned AR policy on the *2m-vs-1z* map. There are two agents (marines) and one environment-controlled enemy. In this strategy, *the two marines keep standing still throughout the game*. They perform attacks alternately while ensuring there is only one attacking marine at each timestep. Since the enemy will by default move towards the agent who attacks it, as a consequence of such an alternative attacking scheme, the enemy turns out to be constantly jittering between the two marines until death without getting any chance to even get close to any of the marines at all. By contrast, marines represented by independent policies will keep moving within the map to ensure a safe distance from the enemy. Fig. 6 visualizes another learned AR policy on the *3s-vs-5z* map where 3 agents are trained to fight against 5 enemies. The agents trained by PG-AR learn to spread out on the map and move towards different corners to keep distance from each other so that every single agent can take charge of just 1 or 2 enemies. By contrast, agents by independent policies will never coordinate to spread out. We illustrate the heatmap of agent positions produced by different policies in Fig. 7, where a significant visitation difference can be observed. AR policies are more coordinated than independent policies.

In GRF, we present the learned strategy by PG-AR for the *3 vs 1 with keeper* scenario in Fig. 8, where the AR policy learns a neat "Tiki-Taka" style behavior: each of the 3 controlled players keeps passing the ball to their teammates, from outside the penalty to the right side and then to the box, before finally scoring a goal. We remark that only the AR policy can discover such a strategy that involves interactions among all the 3 controlled players. In the strategy by independent policies, the active agent will directly perform long shots outside the penalty area without performing short

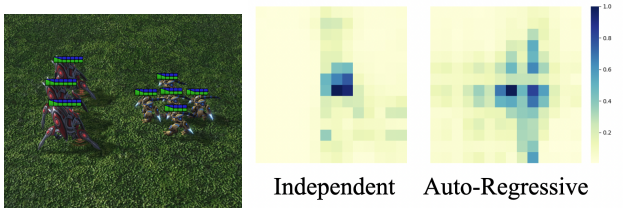

Figure 7. Heatmap of agent positions from the strategies learned by independent policies (middle) and AR policies (right). AR policies are more coordinated and can control the agents to well spread out to different corners of the map.

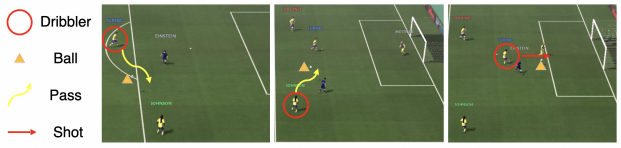

Figure 8. Emergent "Tiki-Taka" behavior in the GRF *3 vs 1 with keeper* scenario. Red circles indicate the dribbling player.

passes to teammates.

Remark: We admit that with additional training techniques (Sec. 5.2), auto-regressive policy learning will converge slower than individual policy learning, particularly in games with a lot of agents. So there is a trade-off between expressiveness capability and sample efficiency. More experiments on auto-regressive policy learning can be found in Appendix D.1. In general, when optimizing the final reward is not the only goal of a research project, we would suggest adopting auto-regressive modeling for diverse emergent behaviors.

7. Conclusion

In this paper, we provide a concrete analysis of the two common practices of MARL algorithms: value decomposition and policy sharing. Theoretical results show that under highly multi-modal scenarios, both two techniques can lead to unsatisfying behaviors, while policy gradient methods can be preferable for learning the optimal solution as well as for learning multi-modal behaviors. We propose practical enhancements for implementing effective policy gradient algorithms in general multi-agent Markov games and achieve strong performances in challenging MARL testbeds including StarCraft MultiAgent Challenge and Google Research Football. We hope our empirical suggestions can benefit the practitioners while our theoretical analysis could serve as a starting point toward more general and more powerful MARL algorithms.

Acknowledgement

Yi Wu is supported by 2030 Innovation Megaprojects of China (Programme on New Generation Artificial Intelligence) Grant No. 2021AAA0150000.

References

Aumann, R. J. Subjectivity and correlation in randomized strategies. *Journal of mathematical Economics*, 1(1): 67–96, 1974.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. Emergent tool use from multi-agent autocurricula. *arXiv preprint arXiv:1909.07528*, 2019.

Balduzzi, D., Garnelo, M., Bachrach, Y., Czarnecki, W., Perolat, J., Jaderberg, M., and Graepel, T. Open-ended learning in symmetric zero-sum games. In *International Conference on Machine Learning*, pp. 434–443. PMLR, 2019.

Boutilier, C. Planning, learning and coordination in multiagent decision processes. In Shoham, Y. (ed.), *Proceedings of the Sixth Conference on Theoretical Aspects of Rationality and Knowledge, De Zeeuwse Stromen, The Netherlands, March 17-20 1996*, pp. 195–210. Morgan Kaufmann, 1996.

Christianos, F., Schäfer, L., and Albrecht, S. V. Shared experience actor-critic for multi-agent reinforcement learning. *arXiv preprint arXiv:2006.07169*, 2020.

Cully, A., Clune, J., Tarapore, D., and Mouret, J.-B. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

Damerau, F. J. *Markov models and linguistic theory*. De Gruyter Mouton, 2018.

de Witt, C. S., Gupta, T., Makoviichuk, D., Makoviychuk, V., Torr, P. H., Sun, M., and Whiteson, S. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Foerster, J. N., Assael, Y. M., De Freitas, N., and Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. *arXiv preprint arXiv:1605.06676*, 2016a.

Foerster, J. N., Assael, Y. M., De Freitas, N., and Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. *arXiv preprint arXiv:1605.06676*, 2016b.

Gu, J., Bradbury, J., Xiong, C., Li, V. O., and Socher, R. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*, 2017.

Gupta, J. K., Egorov, M., and Kochenderfer, M. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems*, pp. 66–83. Springer, 2017.

Hornik, K., Stinchcombe, M., and White, H. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Hu, H., Lerer, A., Peysakhovich, A., and Foerster, J. "other-play" for zero-shot coordination. In *International Conference on Machine Learning*, pp. 4399–4410. PMLR, 2020a.

Hu, J., Jiang, S., Harding, S. A., Wu, H., and Liao, S.-w. Riit: Rethinking the importance of implementation tricks in multi-agent reinforcement learning. *arXiv preprint arXiv:2102.03479*, 2021.

Hu, S., Zhu, F., Chang, X., and Liang, X. Updet: Universal multi-agent rl via policy decoupling with transformers. In *International Conference on Learning Representations*, 2020b.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101(1-2):99–134, 1998. doi: 10.1016/S0004-3702(98)00023-X. URL https://doi.org/10.1016/S0004-3702(98)00023-X.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kleinberg, B., Li, Y., and Yuan, Y. An alternative view: When does sgd escape local minima? In *International Conference on Machine Learning*, pp. 2698–2707. PMLR, 2018.

Kreps, D. M. and Wilson, R. Sequential equilibria. *Econometrica: Journal of the Econometric Society*, pp. 863–894, 1982.

Kuba, J. G., Chen, R., Wen, M., Wen, Y., Sun, F., Wang, J., and Yang, Y. Trust region policy optimisation in multi-agent reinforcement learning. 2022.

Kurach, K., Raichuk, A., Stańczyk, P., Zając, M., Bachem, O., Espeholt, L., Riquelme, C., Vincent, D., Michalski, M., Bousquet, O., et al. Google research football: A novel reinforcement learning environment. *arXiv preprint arXiv:1907.11180*, 2019.

Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Pérolat, J., Silver, D., and Graepel, T. A unified game-theoretic approach to multiagent reinforcement learning. In *NIPS*, 2017.

Li, C., Wu, C., Wang, T., Yang, J., Zhao, Q., and Zhang, C. Celebrating diversity in shared multi-agent reinforcement learning. *arXiv preprint arXiv:2106.02195*, 2021.

Liu, I.-J., Jain, U., Yeh, R. A., and Schwing, A. Cooperative exploration for multi-agent deep reinforcement learning. In *International Conference on Machine Learning*, pp. 6826–6836. PMLR, 2021.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*, 2017.

Lowe, R., Foerster, J., Boureau, Y.-L., Pineau, J., and Dauphin, Y. On the pitfalls of measuring emergent communication. *arXiv preprint arXiv:1903.05168*, 2019.

Lupu, A., Cui, B., Hu, H., and Foerster, J. Trajectory diversity for zero-shot coordination. In *International Conference on Machine Learning*, pp. 7204–7213. PMLR, 2021.

Mahajan, A., Rashid, T., Samvelyan, M., and Whiteson, S. Maven: Multi-agent variational exploration. *arXiv preprint arXiv:1910.07483*, 2019.

Masood, M. A. and Doshi-Velez, F. Diversity-inducing policy gradient: Using maximum mean discrepancy to find a set of diverse policies. *arXiv preprint arXiv:1906.00088*, 2019.

Oliehoek, F. A., Spaan, M. T., and Vlassis, N. Optimal and approximate q-value functions for decentralized pomdps. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

Papoudakis, G., Christianos, F., Schäfer, L., and Albrecht, S. V. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. *arXiv preprint arXiv:2006.07869*, 2020.

Parker-Holder, J., Pacchiano, A., Choromanski, K., and Roberts, S. Effective diversity in population-based reinforcement learning. *NeurIPS*, 2020.

Pugh, J. K., Soros, L. B., and Stanley, K. Quality diversity: A new frontier for evolutionary computation. *Frontiers Robotics AI*, 3:40, 2016.

Qian, L., Zhou, H., Bao, Y., Wang, M., Qiu, L., Zhang, W., Yu, Y., and Li, L. Glancing transformer for non-autoregressive neural machine translation. *arXiv preprint arXiv:2008.07905*, 2020.

Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., and Whiteson, S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 4295–4304. PMLR, 2018.

Rashid, T., Torr, P. H., Farquhar, G., Hung, C.-M., Rudner, T. G., Nardelli, N., Whiteson, S., de Witt, C. S., Foerster, J., and Samvelyan, M. The Starcraft multi-agent challenge. volume 4, pp. 2186–2188. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

Rashid, T., Farquhar, G., Peng, B., and Whiteson, S. Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. *arXiv preprint arXiv:2006.10800*, 2020.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 5887–5896. PMLR, 2019.

Su, J., Adams, S., and Beling, P. A. Value-decomposition multi-agent actor-critics. *arXiv preprint arXiv:2007.12306*, 2020.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*, 2017.

Tang, Z., Yu, C., Chen, B., Xu, H., Wang, X., Fang, F., Du, S., Wang, Y., and Wu, Y. Discovering diverse multi-agent strategic behavior via reward randomization. In *International Conference on Learning Representations*, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wang, J., Ren, Z., Liu, T., Yu, Y., and Zhang, C. Qplex: Duplex dueling multi-agent q-learning. *arXiv preprint arXiv:2008.01062*, 2020a.

Wang, T., Wang, J., Wu, Y., and Zhang, C. Influence-based multi-agent exploration. *arXiv preprint arXiv:1910.05512*, 2019a.

Wang, T., Wang, J., Zheng, C., and Zhang, C. Learning nearly decomposable value functions via communication minimization. *arXiv preprint arXiv:1910.05366*, 2019b.

Wang, T., Gupta, T., Mahajan, A., Peng, B., Whiteson, S., and Zhang, C. Rode: Learning roles to decompose multi-agent tasks. *arXiv preprint arXiv:2010.01523*, 2020b.

Wang, Y., Han, B., Wang, T., Dong, H., and Zhang, C. Off-policy multi-agent decomposed policy gradients. *arXiv preprint arXiv:2007.12322*, 2020c.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Wu, Z., Yu, C., Ye, D., Zhang, J., Zhuo, H. H., et al. Coordinated proximal policy optimization. *Advances in Neural Information Processing Systems*, 34, 2021.

Yang, Y., Hao, J., Liao, B., Shao, K., Chen, G., Liu, W., and Tang, H. Qatten: A general framework for cooperative multiagent reinforcement learning. *arXiv preprint arXiv:2002.03939*, 2020.

Yu, C., Velu, A., Vinitsky, E., Wang, Y., Bayen, A., and Wu, Y. The surprising effectiveness of mappo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

Zahavy, T., O'Donoghue, B., Barreto, A., Flennerhag, S., Mnih, V., and Singh, S. Discovering diverse nearly optimal policies with successor features. In *ICML 2021 Workshop on Unsupervised Reinforcement Learning*, 2021.

Zhang, T., Xu, H., Wang, X., Wu, Y., Keutzer, K., Gonzalez, J. E., and Tian, Y. Multi-agent collaboration via reward attribution decomposition. *arXiv preprint arXiv:2010.08531*, 2020.

Zhang, T., Li, Y., Wang, C., Xie, G., and Lu, Z. Fop: Factorizing optimal joint policy of maximum-entropy multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 12491–12500. PMLR, 2021.

Zhou, Z., Fu, W., Zhang, B., and Wu, Y. Continuously discovering novel strategies via reward-switching policy optimization. In *International Conference on Learning Representations*, 2022.

A. Paper Website

Please check our project website https://sites.google.com/view/revisiting-marl for more information, including visualization of learned strategies.

B. Environment Details

B.1. *Bridge* game

The observation of *Bridge* game is a 6-dim vector representation combined with [self-position, goal position, ally position], and the action space is a Categorical distribution over [idle, moving up, moving down, moving left, moving right]. The observation is processed to be symmetric to both sides of the bridge. After one agent reaches its own goal, it will be marked as "dead", and the [ally position] part of the other agent's observation will be masked.

B.2. SMAC

We follow to use the SMAC environment and evaluation protocol in MAPPO (Yu et al., 2021).

B.3. GRF

We use separate dense rewards for all algorithms, i.e., agents obtain independent "scoring" and "checkpoints" rewards at each timestep. We use the full action set and the "simple115v2" vector representation as the input of both policy and value inputs.

C. Implementation Details

C.1. Fitting the XOR Game

We use 2×2 trainable parameters to represent the local Q-values for each agent, and additional trainable parameters for the mixing network. Specifically, we use 2 trainable parameters as the weight of VDN, a one-layer neural network with 64 hidden units as the hyper-net for QMIX, and 4-head attention with 64 hidden units for QPLEX. All the methods apply stochastic gradient descent with a learning rate of 0.1.

C.2. Attention-Based Auto-regressive Backbone

We split the observation into agent-wise slots indicating different observable semantic information. Specifically, slots in the Bridge game contain "self", "goal", and "ally"; slots in SMAC contain "self", "move", "ally", and "enemy"; slots in GRF contain "self", "ball", "ally", and "enemy". We embed different slots using different embedding layers, and different entries in the same slot share the same embedding layer, which is similar to the architecture adopted in Baker et al. (2019). One-hot agent actions are embedded with another embedding layer and added onto the corresponding "ally" observation embeddings. Then, the embeddings are passed to a self-attention layer and a feed-forward layer, and the first output slot, which corresponds to the observation of the agent itself, is passed to the policy head to output the action distribution. We also mask the unavailable information in the self-attention layer, e.g., ally actions and positions will be masked if it is not visible. All embedding and self-attention layers have a hidden dimension of 64.

C.3. Hyperparameters and Other Details

Hyperparameters of VD methods (except for CDS and RODE) and PG methods are shown in Table 6 and Table 7. For all the networks and embedding layers, we use 64 hidden units. The backbone of policy, value, and Q network is a 2-hidden-layer MLP for Bridge, with an additional GRU layer for SMAC and GRF. We use 4 attention heads for QPLEX and the attention-based backbone of auto-regressive policy. We also add layer norm after each linear layer. Value normalization is applied to PG methods. The batch size is 3200 for PG methods in Bridge and SMAC, and 10000 in GRF. The PPO epoch is 5 in Bridge and 15 across all GRF scenarios. PG methods are trained for 50M environment frames on the counterattack-hard and corner scenario, and 25M frames on other scenarios in GRF. In SMAC, we adopt the same PPO epoch and total environment frames as Yu et al. (2021). For CDS and RODE, we use the hyperparameters and implementation adopted from the original paper.

For PG-sh, we use a single actor and a single critic for all the agents. For PG-ID, we additionally concatenate observation

Name	value
γ	0.99
GAE (Schulman et al., 2015) λ	0.95
PPO clip	0.2
value clip	0.2
value loss	huber
huber δ	10.0
entropy coefficient	0.01
optimizer	Adam (Kingma & Ba, 2014)
learning rate	5e-4
gradient norm	10.0

Table 6. PG hyperparameters.

Name	Value
γ	0.99
hard update interval	50
gradient norm	10.0
optimizer	Adam (Kingma & Ba, 2014)
learning rate	5e-4
value loss	MSE

Table 7. VD hyperparameters.

with one-hot agent IDs as the input of actor and critic. For PG-ind, we train n separate actors and critics for all agents. For PG-AR, we train a single actor with the attention-based auto-regressive representation and a single critic which is the same as PG-sh for each agent. For auto-regressive policy, we use an entropy coefficient of 0.05 and omit agent ID.

D. Additional Results

D.1. Learning Auto-Regressive Policies with Higher Rewards

Kuba et al. (2022) proposes Heterogeneous-Agent Proximal Policy Optimization (HAPPO) to solve cooperative MARL problems. In HAPPO, policies are trained sequentially by accumulating importance ratios across agents (see Kuba et al. (2022) for more details). However, HAPPO still adopts independent policy factorization in Eq. (3) and learns an individual policy for each agent, which restricts the expressiveness power. To unlock the full potential, it is then natural to extend HAPPO with our auto-regressive (AR) representation, such that *training and inference are both conducted sequentially.* We name this extension *HAPPO-AR*. By implementing such an algorithm, we can investigate to what extent expressiveness power affects sample efficiency and the performance of the state-of-the-art algorithm. Hopefully, HAPPO-AR can obtain the benefits of both — it can maintain high sample efficiency while learning interesting emergent behavior that requires strong intra-coordination.

We remark that the focus of HAPPO-AR is different from the focus of learning multi-modal behavior as presented in Sec. 5.2 and 6.2. In the main body, we want to show the effectiveness of the AR representation, while sample efficiency and algorithm performance are not the most imperative topics. By contrast, in this subsection, we aim at investigating the effect of the AR representation on algorithm performance. Learning multi-modal behavior is not the ultimate goal. Instead, we hope that the AR representation can aid the algorithm to converge to a better optimum. Therefore, instead of applying techniques introduced in Sec. 5.2, we only employ a minimal modification on HAPPO: the MLP policy takes one-hot actions of other agents as an additional input. Besides, we align the inference and training order to better utilize auto-regressive learning.

We evaluate HAPPO and HAPPO-AR on the 6 academy scenarios in Google Research Football following the same evaluation protocol as illustrated in Appendix C.3. Results are presented in Table 8. HAPPO-AR outperforms HAPPO in 5 out of 6 academy scenarios. This result indicates that the AR representation can be combined with the state-of-the-art PG method

	PG-ID	PG-Ind	HAPPO	HAPPO-AR
3v1	90.7(1.5)	90.2(1.7)	**95.1 (5.8)**	*94.9 (4.4)**
CA-E	79.5(6.7)	**92.4(2.6)**	54.1 (36.5)	*60.5 (35.9)**
CA-H	**68.2(2.0)**	**67.8(3.4)**	29.2 (43.9)	16.7 (39.5)
Corner	27.3(1.3)	21.6(2.4)	**35.4 (42.5)**	*39.1 (40.0)**
PS	43.4(8.8)	53.3(3.5)	**93.5 (6.0)**	*95.0 (4.5)**
RPS	66.6(3.1)	78.8(1.5)	**98.7 (4.2)**	*97.7 (4.2)**

Table 8. Median evaluation winning rate (standard deviation) on GRF academy scenarios over 6 random seeds. Bold numbers are the state-of-the-art results under CTDE. * indicates that HAPPO-AR performs at least as well as HAPPO. (CA-E&H=counter attack easy&hard; PS=pass and shoot; RSP=run, pass and shoot)

Scenario	Winning Rate
2m_vs_1z	100.0(0.0)
3s_vs_5z	100.0(0.9)
GRF 3v1	92.7(3.7)

Table 9. Median evaluation winning rate of auto-regressive policy in SMAC and GRF.

to improve algorithm performance. We also visualize the behavior of HAPPO and HAPPO-AR in the 3-vs-1 scenario. While HAPPO learns plain pass-and-shoot, HAPPO-AR can still learn "Tiki-Taka" behavior as shown in Sec. 6.2 (see our project website for GIF demonstration). However, in harder scenarios like counterattack-hard and corner, we do not observe distinguishable behavioral differences.

Finally, we remark that auto-regressive policy learning can be implemented in different ways depending on the purpose. We recommend applying techniques introduced in Sec. 5.2 if the aim is multi-modal behavior. If the aim is a higher reward, we recommend the implementation introduced in this section.

D.2. Evaluation Performance of Multi-Modal Behavior

In Sec. 6.2, we have shown emergent behavior on the *2m_vs_1z* and *3s_vs_5z* map in SMAC and on the academy-3-vs-1-with-keeper scenario in GRF. We additionally show the evaluation performance of these learned policies in Table 9. The results show that PG-AR can successfully learn *winning* multi-modal strategies.