MULTI-AGENT GUIDED POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Due to practical constraints such as partial observability and limited communication, Centralized Training with Decentralized Execution (CTDE) has become the dominant paradigm in cooperative Multi-Agent Reinforcement Learning (MARL). However, existing CTDE methods often underutilize centralized training or lack theoretical guarantees. We propose Multi-Agent Guided Policy Optimization (MAGPO), a novel framework that better leverages centralized training by integrating centralized guidance with decentralized execution. MAGPO uses an autoregressive joint policy for scalable, coordinated exploration and explicitly aligns it with decentralized policies to ensure deployability under partial observability. We provide theoretical guarantees of monotonic policy improvement and empirically evaluate MAGPO on 43 tasks across 6 diverse environments. Results show that MAGPO consistently outperforms strong CTDE baselines and matches or surpasses fully centralized approaches, offering a principled and practical solution for decentralized multi-agent learning.

1 INTRODUCTION

Cooperative Multi-Agent Reinforcement Learning (MARL) provides a powerful framework for solving complex real-world problems such as autonomous driving (Zhou et al., 2020), traffic management (Singh et al., 2020), and robot swarm coordination (Hüttenrauch et al., 2019; Zhang et al., 2021a). However, MARL faces two fundamental challenges: the exponential growth of the joint action space with the number of agents, which hinders scalability, and the requirement for decentralized execution under partial observability, which complicates policy learning.

A widely adopted solution is Centralized Training with Decentralized Execution (CTDE) (Oliehoek et al., 2008; Kraemer & Banerjee, 2016), where agents are trained using privileged global information but execute independently based on local observations. CTDE forms the foundation of many state-of-the-art MARL algorithms and typically incorporates a centralized value function to guide decentralized policies or utility functions during training. This setup allows algorithms to benefit from global context without violating the constraints of decentralized deployment.

Parallel efforts in single-agent RL have explored similar ideas in the context of Partially Observable Markov Decision Processes (POMDPs) (Oliehoek & Amato, 2016). Two main approaches have emerged: asymmetric actor-critic (Pinto et al., 2018), which uses full-state information in the critic but restricts the actor to partial observations; and teacher-student learning, where a teacher policy trained with privileged information supervises a student policy that learns to act under partial observability.

These insights have recently inspired the Centralized Teacher with Decentralized Student (CTDS) paradigm in MARL (Zhao et al., 2024). CTDS combines a centralized critic with a teacher policy that outputs joint actions based on the global state. These actions are then distilled into decentralized policies for execution. While CTDS shows promise—particularly in enabling coordinated exploration and better utilization of centralized training—it faces two key challenges. First, training a centralized teacher faces scalability problems due to the exponential size of the joint action space. Second, even with a strong teacher, the decentralized policies may suffer from the **imitation gap** (Weihs et al., 2021): student policies operating under partial observability may fail to replicate the teacher's behavior, leading to degraded performance. In MARL, this issue is exacerbated by *policy asymmetry*, where the space of decentralized behaviors cannot fully capture the teacher's joint strategy.

To overcome these limitations, we propose **Multi-Agent Guided Policy Optimization (MAGPO)**, a novel framework that bridges centralized training and decentralized execution through a principled

and MARL-specific design. MAGPO addresses the scalability and *policy asymmetry* problem by constraining a centralized, autoregressive guider policy to remain closely aligned with decentralized learners throughout training. The guider policy allows agents to act sequentially conditioned on previous actions, utilizing global information and coordinated data collection (Wen et al., 2022; Mahjoub et al., 2025). The alignment ensures that the coordination strategies developed under centralized supervision remain realizable by decentralized policies, thus mitigating the imitation gap that undermines prior CTDS approaches. Unlike a direct extension of single-agent GPO (Li et al., 2025), MAGPO introduces structural mechanisms tailored to multi-agent learning, including sequential joint action modeling and decentralization-aligned updates, while preserving scalability and parallelism. We provide theoretical guarantees of monotonic policy improvement and empirically evaluate MAGPO across 43 tasks in 6 diverse environments. Results show that MAGPO consistently outperforms strong CTDE baselines and even matches or exceeds fully centralized methods, establishing it as a theoretically grounded and practically deployable solution for MARL under partial observability.

2 BACKGROUND

2.1 FORMULATION

We consider Decentralized Partially Observable Markov Decision Process (Dec-POMDP) (Oliehoek & Amato, 2016) in modeling cooperative multi-agent tasks. The Dec-POMDP is characterized by the tuple $\langle \mathcal{N}, \mathcal{S}, \mathcal{A}, r, \mathcal{P}, \mathcal{O}, \mathcal{Z}, \gamma \rangle$, where $\mathcal{N}$ is the set of agents, $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, r is the reward function, $\mathcal{P}$ is the transition probability function, $\mathcal{Z}$ is the individual partial observation generated by the observation function $\mathcal{O}$, and γ is the discount factor. At each timestep, each agent $i \in \mathcal{N}$ receives a partial observation $o_i \in \mathcal{Z}$ according to $\mathcal{O}(s; i)$ at state $s \in \mathcal{S}$. Then, each agent selects an action $a_i \in \mathcal{A}$ according to its action-observation history $\tau_i \in (\mathcal{Z} \times \mathcal{A})^*$, collectively forming a joint action denoted as $\boldsymbol{a}$. The state s undergoes a transition to the next state s' in accordance with $\mathcal{P}(s'|s, \boldsymbol{a})$, and agents receive a shared reward r. Assuming an initial state distribution $\rho \in \Delta(\mathcal{S})$, the goal is to find a decentralized policy $\boldsymbol{\pi} = \{\pi_i\}_{i=1}^n$ that maximizes the expected cumulative return:

$$V_\rho(\pi) \triangleq \mathbb{E}_{s\sim\rho}[V_{\boldsymbol{\pi}}(s)] = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 \sim \rho]. \tag{1}$$

This work follows the Centralized Training with Decentralized Execution (CTDE) paradigm (Oliehoek et al., 2008; Kraemer & Banerjee, 2016). During training, CTDE allows access to global state to stabilize learning. However, during execution, each agent operates independently, relying solely on its local action-observation history.

In centralized training, we can therefore optimize a joint policy $\mu(\mathbf{a} \mid s)$ that coordinates all agents while enabling joint exploration. To update this joint policy, we will consider the policy mirror descent (PMD) objective (Shani et al., 2019):

$$\mu^{(k+1)} = \arg\max_{\mu} \left\{ \eta_k \langle \nabla V_\rho(\mu^{(k)}), \mu \rangle - \frac{1}{1-\gamma} D_{d_\rho(\mu^{(k)})}(\mu, \mu^{(k)}) \right\}, \tag{2}$$

where η_k is the step size, $\rho \in \Delta(\mathcal{S})$ is an arbitrary state distribution, $d_\rho(\mu^{(k)})$ is the discounted state-visitation distribution under $\mu^{(k)}$, and $D_{d_\rho(\mu^{(k)})}$ denotes the corresponding weighted Bregman divergence. Conceptually, PMD can be viewed as a class of preconditioned policy gradient methods. Choosing the Bregman divergence to be the Euclidean distance or the Kullback–Leibler (KL) divergence recovers projected policy gradient and natural policy gradient (NPG) (Kakade, 2001), respectively. Thus, PMD provides a unifying framework that encompasses many modern policy-based RL algorithms, including TRPO (Schulman et al., 2015a) and PPO (Schulman et al., 2017). We build upon this formulation to develop our theoretical results in Section 4.1.

2.2 RELATED WORKS

CTDE. CTDE methods can be broadly categorized into value-based and policy-based approaches. Value-based methods typically employ a joint value function conditioned on the global state and

joint action, alongside individual utility functions based on local observations and actions. These functions often satisfy the Individual-Global-Max (IGM) principle (Son et al., 2019), ensuring that the optimal joint policy decomposes into locally optimal policies. This line of work is known as value factorization, and includes methods such as VDN (Sunehag et al., 2017), QMIX (Rashid et al., 2020), QTRAN (Son et al., 2019), QPLEX (Wang et al., 2021a), and QATTEN (Yang et al., 2020). Policy-based methods, in contrast, typically use centralized value functions to guide decentralized policies, allowing for direct extensions of single-agent policy gradient methods to multi-agent settings. Notable examples include COMA (Foerster et al., 2018), MADDPG (Lowe et al., 2017), MAA2C (Papoudakis et al., 2020) and MAPPO (Yu et al., 2022). Additionally, hybrid methods that combine value factorization with policy-based training have been proposed, such as DOP (Wang et al., 2021b), FOP (Zhang et al., 2021b), and FACMAC (Peng et al., 2021). While CTDE has achieved strong empirical performance, most existing methods leverage global information only through the value function. We refer to these as **vanilla CTDE** methods, as they do not fully exploit the potential of centralized training.

CTDS. More recently, researchers have explored extending the teacher-student framework from single-agent settings to multi-agent systems, leading to the Centralized Teacher with Decentralized Students (CTDS) paradigm (Zhao et al., 2024; Chen et al., 2024; Zhou et al., 2025). In this framework, a centralized teacher policy—accessing global state and acting jointly—collects high-quality trajectories and facilitates more coordinated exploration. CTDS methods offer stronger supervision than vanilla CTDE methods. However, due to observation asymmetry (Weihs et al., 2021) and policy space mismatch, the learned decentralized policies may still suffer from suboptimal performance—issues that we explore further in the next section.

HARL. In contrast to vanilla CTDE and CTDS methods—many of which lack theoretical guarantees—another line of research focuses on Heterogeneous Agent Reinforcement Learning (HARL), where agents are updated sequentially during training (Zhong et al., 2023). This formulation underpins algorithms such as HATRPO and HAPPO (Kuba et al., 2022) and HASAC (Liu et al., 2025). While HARL provides better theoretical guarantees and stability, it requires agents to be heterogeneous and updated one at a time. As a result, these methods lack parallelism which is important in large-scale MARL tasks and cannot exploit parameter sharing, which has proven effective in many MARL applications (Gupta et al., 2017; Terry et al., 2020; Christianos et al., 2021a).

CTCE. Centralized Training with Centralized Execution (CTCE) approaches treat the multi-agent system as a single-agent problem with a combinatorially large action space. Beyond directly applying single-agent RL algorithms to MARL, a promising direction in CTCE has been to use transformers (Vaswani et al., 2017) to frame multi-agent trajectories as sequences (Chen et al., 2021). This has led to the development of powerful transformer-based methods such as Updet (Hu et al., 2021), Transfqmix (Gallici et al., 2023), and other offline methods (Meng et al., 2023; Tseng et al., 2022; Zhang et al., 2022). Two representative online methods are Multi-Agent Transformer (MAT) (Wen et al., 2022) and Sable (Mahjoub et al., 2025), which currently achieve state-of-the-art performance in cooperative MARL tasks. CTCE methods offer strong theoretical guarantees (Wen et al., 2022) and impressive empirical results. However, they fall short in practical settings that demand decentralized execution, where each agent must act based solely on its local observation and policy.

3 Problems of Current CTDS

In this section, we examine the limitations of using a centralized teacher to supervise decentralized student policies. Two key challenges arise in this setting: **asymmetric observation spaces** and **asymmetric policy spaces**.

The first challenge—*asymmetric observation spaces*—is shared with single-agent POMDPs involving privileged information and has been extensively studied in prior work (Warrington et al., 2020; Weihs et al., 2021; Shenfeld et al., 2023; Li et al., 2025). When the teacher relies on privileged information that is unavailable (and not inferable) to the student, the student cannot faithfully reproduce the teacher's behavior. Instead, it learns to approximate the conditional expectation of the teacher's action given the observable input (Weihs et al., 2021; Warrington et al., 2020). This typically results in an "averaged" behavior that can be significantly suboptimal.

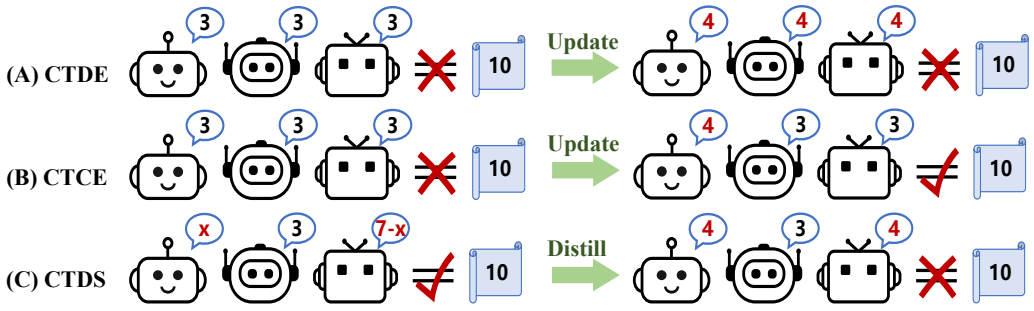


Figure 1: Illustrative example showing three different MARL settings.

The second challenge—*asymmetric policy spaces*—is unique to the multi-agent setting. It arises from the structural mismatch between the teacher's policy (typically joint and expressive) and the students' policies (factorized and decentralized). We illustrate this challenge through a simple example shown in Figure 1. Consider a cooperative task where three agents must each output an integer such that their sum equals a target value of 10. Each agent acts once, and the system succeeds only if the total sum is exactly 10. We compare three MARL frameworks:

(A) Vanilla CTDE. Agents share a centralized value function but act independently via decentralized policies. Suppose all three agents use the same deterministic policy $\pi^i(\cdot|10) = 3$, producing a total of 9—which fails the task. Because each agent observes the same global state and optimizes the same objective, they may all simultaneously increase their action to 4 in the next update, producing 12, still failing. Lacking inter-agent coordination signals, the agents struggle to determine which one should adjust its action. This leads to classic miscoordination, requiring random trial-and-error exploration and memorization of rare successful configurations to eventually coordinate.

(B) CTCE. Agents act sequentially, observing previous agents' actions before choosing their own. Suppose the first agent updates its action to 4 and the second still picks 3. The third agent, having observed both previous actions, selects 3 and achieves the correct total. Sequential execution effectively transforms the multi-agent coordination problem into a single-agent decision-making process over a joint policy. This makes coordination straightforward and stable, without repeated random exploration. However, this setting assumes centralized execution, which is often infeasible in real-world applications requiring decentralization.

(C) CTDS. Now consider distilling a successful CTCE policy from (B) into decentralized student policies. If the teacher's policy is deterministic and factorizable (e.g., always producing $[4, 3, 3]$), CTDS can recover an optimal decentralized solution. However, a CTCE teacher may exploit stochastic strategies. For instance, the first agent samples $x \in \{3, 4\}$ at random, the second agent always outputs 3, and the third agent—having seen the first two actions—outputs $7 - x$, ensuring the total is always 10. While this is optimal under CTCE, it cannot be factored into independent policies: CTDS would learn two independent stochastic policies for the first and third agents, leading to failures such as $[4, 3, 4]$. If the first CTCE agent selects 3 or 4 with equal probability, the distilled decentralized policy succeeds only 50% of the time.

This example highlights the core failure mode: coordination patterns encoded in the teacher's joint policy are lost when forced into a decentralized representation. To address these limitations, we propose a new approach that constrains the teacher's policy during training, preventing it from exploiting unrepresentable coordination strategies while still allowing it to guide decentralized learners effectively.

4 METHOD

We introduce **Multi-Agent Guided Policy Optimization (MAGPO)**, a framework that leverages a centralized, sequentially executed guider policy to supervise decentralized learners while keeping them closely aligned. MAGPO is designed to combine the coordination benefits of centralized training with the deployment constraints of decentralized execution.

We begin by presenting the theoretical formulation and guarantee of monotonic policy improvement in the tabular setting. For clarity, we initially assume full observability—i.e., all agents observe the global state s, reducing the setting to a cooperative Markov game (Littman, 1994). We will return to the partially observable case in the subsequent implementation section.

4.1 Multi-Agent Guided Policy Optimization

Our algorithm maintains a centralized guider policy with an autoregressive structure over agent actions: $\boldsymbol{\mu}(\boldsymbol{a}|s) = \mu^{i_1}(a^{i_1}|s)\mu^{i_2}(a^{i_2}|s, a^{i_1}) \dots \mu^{i_n}(a^{i_n}|s, \boldsymbol{a}^{i_{1:n-1}})$, where $i_{1:m}$ (with $m \leq n$) denotes an ordered subset $\{i_1, ..., i_m\}$ of the agent set $\mathcal{N}$, specifying the execution order. The decentralized learner policy is defined as: $\boldsymbol{\pi}(\boldsymbol{a}|s) = \prod_{j=1}^{n} \pi^{i_j}(a^{i_j}|s)$ for any ordering $i_{1:n}$, implying that all agents act independently.

Building on this structure, MAGPO optimizes the centralized guider and decentralized learner policies through an iterative four-step procedure inspired by the GPO framework (Li et al., 2025):

- **Data Collection**: Roll out the current guider policy $\boldsymbol{\mu}_k$ to collect trajectories.
- **Guider Training**: Update the guider $\boldsymbol{\mu}_k$ to $\hat{\boldsymbol{\mu}}_k$ by maximizing RL objective.
- **Learner Training**: Update the learner $\boldsymbol{\pi}_k$ to $\boldsymbol{\pi}_{k+1}$ by minimizing the KL distance $\mathrm{D}_{\mathrm{KL}}(\boldsymbol{\pi}, \hat{\boldsymbol{\mu}}_k)$.
- **Guider Backtracking**: Set $\boldsymbol{\mu}_{k+1} = \boldsymbol{\pi}_{k+1}$ for all states s.

The first step allows MAGPO to perform coordinated exploration using a joint policy. In the second step, the guider is updated using the Policy Mirror Descent (PMD) framework (Xiao, 2022), which solves the following optimization:

$$\hat{\boldsymbol{\mu}}_k = \arg\max_{\boldsymbol{\mu}} \left\{ \eta_k \langle Q^{\boldsymbol{\mu}_k}(s, \cdot), \boldsymbol{\mu}(\cdot|s) \rangle - \mathrm{D}_{\mathrm{KL}}(\boldsymbol{\mu}(\cdot|s), \boldsymbol{\mu}_k(\cdot|s)) \right\}, \tag{3}$$

where $Q^{\boldsymbol{\mu}_k}$ is the Q-function of guider and η_k is the learning rate. As discussed in Section 2.1, PMD is a general policy gradient framework that subsumes popular algorithms such as PPO and TRPO. Here, we adopt PMD for theoretical clarity and instantiate it using PPO-style updates in our practical implementation. Additional details are provided in Appendix A. In the final step, we perform guider backtracking, where the guider is reset to the current learner policy. Theoretically, this is always feasible since any decentralized policy $\boldsymbol{\pi}$ defines a valid autoregressive joint policy $\boldsymbol{\mu}$ by simply ignoring the conditioning on past actions.

Based on the framework introduced above, we have the following theorem for MAGPO.

Theorem 4.1 (Monotonic Improvement of MAGPO)**.** *Let $(\boldsymbol{\pi}_k)_{k=0}^{\infty}$ be the sequence of joint learner policies obtained by iteratively applying the four steps of MAGPO. Then,*

$$V_\rho(\boldsymbol{\pi}_{k+1}) \geq V_\rho(\boldsymbol{\pi}_k), \ \forall k, \tag{4}$$

where V_ρ is the expected return under initial state distribution ρ.

Proof. See Appendix A. □

In contrast to CTDS and standard CTDE methods like MAPPO, MAGPO provides a provable guarantee of policy improvement. This result can be understood intuitively: the guider identifies a policy that improves return in the full joint space using PMD. The learner then projects this policy into the decentralized policy space via KL minimization. Since the target was chosen via projected gradient, the resulting learner policy also improves return.

To further clarify the structure of MAGPO, we show that its learner updates can be interpreted as sequential advantage-based updates—a procedure known to ensure monotonic improvement in multi-agent settings (Kuba et al., 2022). We begin with the following lemma:

Lemma 1 (Multi-Agent Advantage Decomposition (Kuba et al., 2022))**.** *In any cooperative Markov game, given a joint policy $\boldsymbol{\pi}$, for any state s, and any agent subset $i_{1:m}$, the following equations hold:*

$$A_{\boldsymbol{\pi}}^{i_{1:m}}(s, \boldsymbol{a}^{i_{1:m}}) = \sum_{j=1}^{m} A_{\boldsymbol{\pi}}^{i_j}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j}), \tag{5}$$

where

$$A^{i_{1:m}}_{\boldsymbol{\pi}}(s, \boldsymbol{a}^{j_{1:k}}, \boldsymbol{a}^{i_{1:m}}) \triangleq Q^{j_{1:k}, i_{1:m}}(s, , \boldsymbol{a}^{j_{1:k}}, \boldsymbol{a}^{i_{1:m}}) - Q^{j_{1:k}}(s, \boldsymbol{a}^{j_{1:k}}) \tag{6}$$

for disjoint sets $j_{1:k}$ and $i_{1:m}$. The state-action value function for a subset is defined as

$$Q^{i_{1:m}}(s, \boldsymbol{a}^{i_{1:m}}) \triangleq \mathbb{E}_{\boldsymbol{a}^{-i_{1:m}} \sim \boldsymbol{\pi}^{-i_{1:m}}} \left[Q(s, \boldsymbol{a}^{i_{1:m}}, \boldsymbol{a}^{-i_{1:m}}) \right]. \tag{7}$$

Using this, we derive the following:

Corollary 4.2 (Sequential Update of MAGPO)**.** *The update for any individual policy π^{i_j} with ordered subset $i_{1:j}$ can be written as:*

$$\pi^{i_j}_{k+1} = \arg\max_{\pi^{i_j}} \mathbb{E}_{a^{i_{1:j-1}} \sim \boldsymbol{\pi}^{i_{1:j-1}}_{k+1}, a^{i_j} \sim \pi^{i_j}} \left[A^{i_j}_{\boldsymbol{\pi}}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j}) \right] - \frac{1}{\eta_k} D_{KL}(\pi^{i_j}, \pi^{i_j}_k) \tag{8}$$

Proof. See Appendix A. □

This shows that MAGPO's learner updates are equivalent to performing sequential advantage-weighted policy updates. Importantly, unlike methods such as HARL which update agents one at a time, MAGPO allows for simultaneous updates of all agent policies. This enables parallel training and improves scalability to large agent populations. Moreover, HARL requires heterogeneous agents to guarantee policy improvement, while MAGPO works with either homogeneous or heterogeneous agents, allowing it to benefit from parameter sharing—a widely adopted practice that significantly improves efficiency and generalization in MARL (Gupta et al., 2017; Terry et al., 2020; Christianos et al., 2021a).

4.2 Practical Implementation

In this subsection, we describe the practical implementation of MAGPO. Our implementation is based on the original GPO-Clip framework, extended to the multi-agent setting. The key difference is that the guider in MAGPO is a sequential execution policy. Since MAGPO is compatible with any autoregressive CTCE method, we do not specify the exact encoder, decoder, or attention mechanisms used. Instead, we present general training objectives for both the guider and learner components.

Guider Update. As introduced in the previous section, the guider policy (parameterized by ϕ) is first optimized to maximize the RL objective, and then aligned with the learner policy. This is achieved via an RL update augmented with a KL constraint:

$$\mathcal{L}(\phi) = -\frac{1}{Tn} \sum_{j=1}^{n} \sum_{t=0}^{T-1} \Big[\min \Big(r_t^{i_j}(\phi) \hat{A}_t, \text{clip}(r_t^{i_j}(\phi), \epsilon, \delta) \hat{A}_t \Big) - m_t^{i_j} \mathrm{D}_{\mathrm{KL}} \big(\mu_\phi^{i_j}(\cdot | s_t, \boldsymbol{a}_t^{i_{1:j-1}}), \pi_\theta^{i_j}(\cdot | o_t^{i_j}) \big) \Big], \tag{9}$$

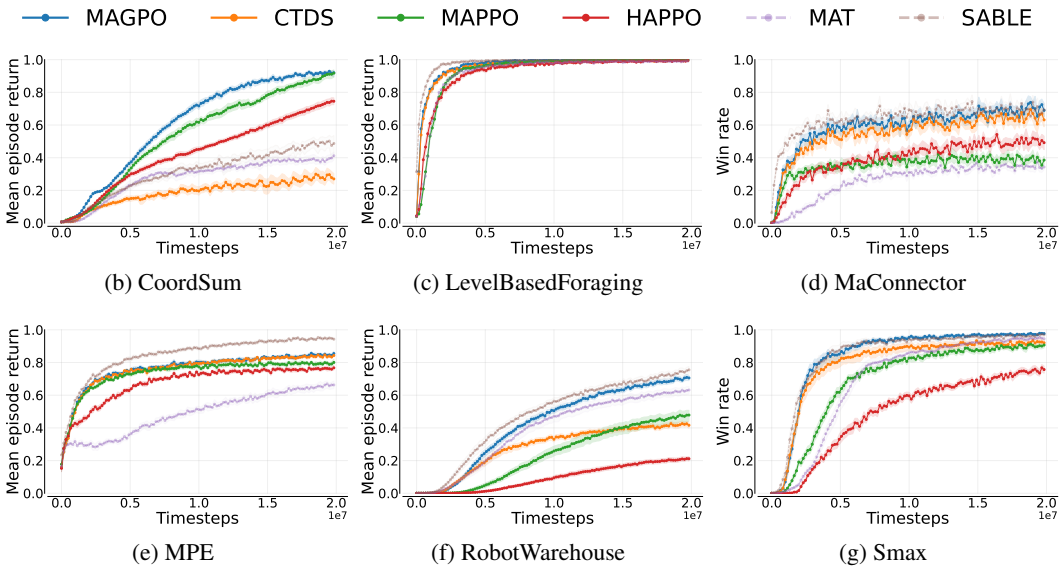
where

$$r_t^{i_j}(\phi) = \frac{\mu_\phi^{i_j}(a_t^{i_j} | s_t, \boldsymbol{a}_t^{i_{1:j-1}})}{\mu_{\phi_{\text{old}}}^{i_j}(a_t^{i_j} | s_t, \boldsymbol{a}_t^{i_{1:j-1}})}, \quad m_t^{i_j}(\delta) = \mathbb{I} \left(\frac{\mu_\phi^{i_j}(a_t^{i_j} | s_t, \boldsymbol{a}_t^{i_{1:j-1}})}{\pi_\theta^{i_j}(a_t^{i_j} | o_t^{i_j})} \notin \left(\frac{1}{\delta}, \delta \right) \right),$$

and

$$\text{clip}(r_t^{i_j}(\phi), \epsilon, \delta) = \text{clip} \left(\text{clip} \left(\frac{\mu_\phi^{i_j}(a_t^{i_j} | s_t, \boldsymbol{a}_t^{i_{1:j-1}})}{\pi_\theta^{i_j}(a_t^{i_j} | o_t^{i_j})}, \frac{1}{\delta}, \delta \right) \frac{\pi_\theta^{i_j}(a_t^{i_j} | o_t^{i_j})}{\mu_{\phi_{\text{old}}}^{i_j}(a_t^{i_j} | s_t, \boldsymbol{a}_t^{i_{1:j-1}})}, 1 - \epsilon, 1 + \epsilon \right). \tag{10}$$

This objective has two modifications compared to the standard one: a **double clipping function** $\text{clip}(\cdot, \epsilon, \delta)$ and a **mask function** $m^{i_j}(\delta)$, both controlled by a new hyperparameter $\delta > 1$, which bounds the ratio between guider and learner policies within $(\frac{1}{\delta}, \delta)$. The inner clip in the double clipping function stops the gradient when the advantage signal encourages the guider to drift too far from the learner. The mask function ensures the KL loss is only applied when this ratio constraint is violated. The advantage estimate $\hat{A}_t$ is computed via generalized advantage estimation (GAE) (Schulman et al., 2015b) with value functions.



(b) CoordSum (c) LevelBasedForaging (d) MaConnector

(e) MPE (f) RobotWarehouse (g) Smax

Figure 2: The sample efficiency curves aggregated per environment suite, where dashed lines represent the CTCE methods. For each environment, results are aggregated over all tasks and the min–max normalized inter-quartile mean with 95% stratified bootstrap confidence intervals are shown.

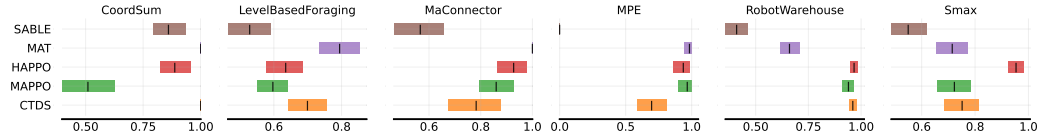


Figure 3: The overall aggregated probability of improvement for MAGPO compared to other baselines for that specific environment. A score of more than 0.5 where confidence intervals are also greater than 0.5 indicates statistically significant improvement over a baseline for a given environment (Agarwal et al., 2021).

Learner Update. The learner policy $\boldsymbol{\pi}$, parameterized by θ, is updated with two objectives: (i) behavior cloning toward the guider policy, and (ii) an RL auxiliary term to directly improve return from the collected trajectories.

$$\mathcal{L}(\theta) = \frac{1}{Tn}\sum_{j=1}^{n}\sum_{t=0}^{T-1}\Big[\mathrm{D}_{\mathrm{KL}}\big(\pi_\theta^{i_j}(\cdot|o_t^{i_j}), \mu_\phi^{i_j}(\cdot|s_t, \boldsymbol{a}_t^{i_{1:j-1}})\big) - \lambda \min\Big(r_t^{i_j}(\theta)\hat{A}_t, \mathrm{clip}(r_t^{i_j}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\Big)\Big], \tag{11}$$

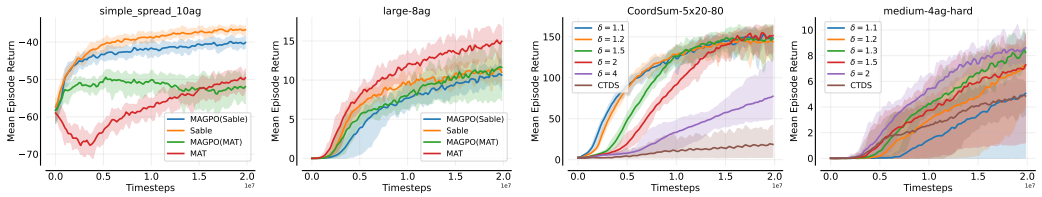
where

$$r_t^{i_j}(\theta) = \frac{\pi_\theta^{i_j}(a_t^{i_j}|o_t^{i_j})}{\mu_{\phi_{\text{old}}}^{i_j}(a_t^{i_j}|s_t, \boldsymbol{a}_t^{i_{1:j-1}})}. \tag{12}$$

The auxiliary RL objective helps maximize the utility of collected trajectories. Since the behavior policy (guider) is kept close to the learner, this term approximates an on-policy objective. In principle, we could apply sequential updates to each individual policy—analogous to HAPPO—to preserve the theoretical guarantees of monotonic improvement. However, this makes the learner updates non-parallelizable and incompatible with parameter sharing. Since no performance benefits are observed from using HAPPO, we instead adopt a MAPPO-style update: all learners share parameters and are updated jointly and in parallel. The auxiliary RL term can be treated as optional and controlled by λ.

5 Experiments

We evaluate MAGPO by comparing it against several SOTA baseline algorithms from the literature. Specifically, we consider two CTCE methods—Sable (Mahjoub et al., 2025) and MAT (Wen et al.,



(a) MAGPO with different CTCE methods as guider

(b) MAGPO with different δ

Figure 4: MAGPO performance varies with the choice of guider and the regularization ratio δ.

2022)—two CTDE baselines—MAPPO (Yu et al., 2022) and HAPPO (Kuba et al., 2022)—and a vanilla implementation of on-policy CTDS, which can be viewed as MAGPO without double clipping, masking, and the RL auxiliary loss. For the joint policy in both MAGPO and CTDS, we use Sable as the default backbone. All algorithms are implemented using the JAX-based MARL library Mava (de Kock et al., 2023).

Evaluation & Hyperparameters. We follow the evaluation protocol from Mahjoub et al. (2025). Each algorithm is trained with 10 independent seeds per task. Training is conducted for 20 million environment steps, with 122 evenly spaced evaluation checkpoints. At each checkpoint, we record the task-specific metrics (e.g., mean episode return and win rate) over 32 evaluation episodes. For task-level results, we report the mean and 95% confidence intervals. For aggregate performance across entire environment suites, we report the min-max normalized interquartile mean (IQM) with 95% stratified bootstrap confidence intervals. The hyperparameters are tuned on each task for each algorithm, which are detailed in Appendix C.3.

Environments. We evaluate MAGPO on a diverse suite of JAX-based multi-agent benchmarks, including 4 tasks in CoordSum (introduced in this paper), 7 tasks in Level-based foraging (LBF) (Christianos et al., 2021b), 4 tasks in Connector (Bonnet et al., 2024), 3 tasks in the Multi-agent Particle Environment (MPE) (Lowe et al., 2017), 15 tasks in Robotic Warehouse (RWARE) (Papoudakis et al., 2020), and 10 tasks in The StarCraft Multi-Agent Challenge in JAX (SMAX) (Rutherford et al., 2024). The CoordSum environment, introduced in this paper, reflects the didactic examples discussed in Section 3, where agents must coordinate to output integers that sum to a given target without using fixed strategies. A detailed description is provided in Appendix B.

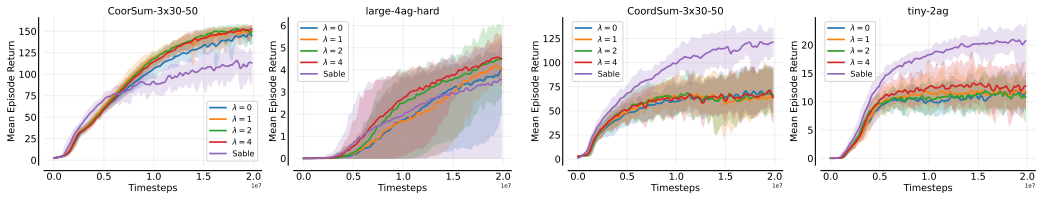
5.1 Main Results

Figure 2 presents the per-environment aggregated sample-efficiency curves. Our results show that MAGPO achieves state-of-the-art performance across all CTDE methods and even outperforms CTCE methods on a subset of tasks. Specifically, MAGPO surpasses all CTDE baselines on 32 out of 43 tasks, and outperforms all baselines on 20 out of 43 tasks. Figure 3 reports the probability of improvement of MAGPO over other baselines. MAGPO emerges as the most competitive CTDE method and performs comparably to the SOTA CTCE method Sable in three benchmark environments. Comparing MAGPO to CTDS reveals a significant performance gap in the CoordSum and RWARE domains, suggesting that in these environments, the CTCE teacher may learn policies that are not decentralizable—rendering direct policy distillation ineffective. Additional tabular results and environment/task-level aggregation plots are provided in Appendix C.2.

5.2 Ablations and Discussions

In this subsection, we discuss several key aspects and design choices of MAGPO.

Bridging CTCE and CTDE. MAGPO's performance intuitively depends heavily on the capability of the guider, which corresponds to the performance of the underlying CTCE method. In Figure 4(a), we evaluate MAGPO on two tasks where Sable and MAT exhibit different performance. In *simple_spread_10ag*, MAT performs significantly worse, resulting in poor performance of MAGPO



(a) MAGPO with different λ (b) CTDS with different auxiliary loss weighted by λ

Figure 5: The effect of RL auxiliary loss.

when using MAT as the guider. In contrast, on *large-8ag*, MAT outperforms Sable, leading to better performance of MAGPO with MAT. This dependency could be seen as a limitation, but it actuall serves as a core feature: MAGPO effectively bridges CTCE and CTDE. In many practical applications that require decentralized policies, MAGPO enables advances in CTCE methods to directly benefit CTDE methods as well—facilitating the co-development of both paradigms.

Effect of the Ratio δ. MAGPO introduces a hyperparameter δ to regulate the guider's deviation from the learner, which most strongly influences its performance. A smaller δ enforces a stricter constraint, keeping the guider closer to the learner policy; a larger δ allows the guider more freedom, potentially enabling it to explore regions of the policy space that are difficult or even unreachable under decentralized constraints. In Figure 4(b), we assess MAGPO's performance under varying δ values on two tasks. In *CoordSum-5x20-80*, a smaller δ yields better performance because the centralized guider tends to learn a policy that is not decentralizable, which must be restricted to improve imitability. Conversely, in *medium-4ag-hard*, the guider policy is more directly imitable, and restricting it too tightly hinders learning. These observations show the importance of tuning δ based on the task's structure and imitation feasibility.

Effect of RL Auxiliary Loss. MAGPO incorporates an RL auxiliary loss in the learner update to better utilize collected data and stabilize learning. This component is not as important as the δ, but a properly tuned λ can also improve performance, as shown in Figure 8(a). To understand this, consider the guider's RL objective is towards an undecentralizable direction and the learner pulls it backward (due to the imitation constraint), then this back-and-forth may repeatedly stall progress. By incorporating RL updates, the learner can "counter-supervise" the guider, helping it discover more decentralizable update directions. Furthermore, in Figure 8(b), we test applying the same RL auxiliary loss to a CTDS method. The results show limited benefit. This is because in CTDS, the behavioral policy is the teacher, which is not enforced to align with the student. If the teacher-student gap is too large, the collected data is off-policy, thus on-policy RL loss on the student provides little benefit.

Observation asymmetry. While most of our analysis has focused on asymmetries in the policy and action spaces, observation asymmetry is equally critical. In the current framework, CTCE methods like MAT and Sable condition on the union of agents' partial observations, whereas individual policies are limited to their own local views. This mismatch creates an imitation gap, making direct imitation methods like CTDS fail, even when the underlying joint policy is decentralizable. MAGPO addresses this issue similar to the single-agent setting (Li et al., 2025), by controlling the divergence between the guider and the learner through the parameter δ. In addition, privileged information—beyond the union of partial observations—is often available during centralized training (e.g., the true global state), although we do not explore it in this paper. Providing such privileged signals to the guider could further enhance its ability to supervise decentralized policies under partial observability.

Model capacity. In addition to the policy and observation asymmetry we emphasize in the main text, asymmetry can also arise from differences in model capacity between a teacher used at training time and a student used at deployment. It is common in practice to train with higher-capacity centralized models or teachers and then distill them into compact students for deployment (e.g., LLMs trained offline and distilled for latency-sensitive inference; VLA paired with lightweight controllers that require high-frequency control). MAGPO provides a promising paradigm for this

setting by constraining the teacher model imitable. Appendix C.4 contains a tentative experiment that simulates the distillation/deployment constraint and shows that MAGPO degrades more gracefully than CTDS when the deployed actor is compact. Further investigation is left to future work.

6 Conclusion

We presented MAGPO, a novel framework that bridges the gap between CTCE and CTDE in cooperative MARL. MAGPO leverages a sequentially executed guider for coordinated exploration while constraining it to remain close to the decentralized learner policies. This design enables stable and effective guidance without sacrificing deployability. Our approach builds upon the principles of GPO and introduces a practical training algorithm with provable monotonic improvement. Empirical results across 43 tasks in 6 diverse environments demonstrate that MAGPO consistently outperforms state-of-the-art CTDE methods and is competitive with CTCE methods, despite relying on decentralized execution.

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631, 2019.

Clément Bonnet, Daniel Luo, Donal Byrne, Shikha Surana, Sasha Abramowitz, Paul Duckworth, Vincent Coyette, Laurence I. Midgley, Elshadai Tegegn, Tristan Kalloniatis, Omayma Mahjoub, Matthew Macfarlane, Andries P. Smit, Nathan Grinsztajn, Raphael Boige, Cemlyn N. Waters, Mohamed A. Mimouni, Ulrich A. Mbou Sob, Ruan de Kock, Siddarth Singh, Daniel Furelos-Blanco, Victor Le, Arnu Pretorius, and Alexandre Laterre. Jumanji: a diverse suite of scalable reinforcement learning environments in jax, 2024. URL https://arxiv.org/abs/2306.09884.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Yiqun Chen, Hangyu Mao, Jiaxin Mao, Shiguang Wu, Tianle Zhang, Bin Zhang, Wei Yang, and Hongxing Chang. Ptde: Personalized training with distilled execution for multi-agent reinforcement learning, 2024. URL https://arxiv.org/abs/2210.08872.

Filippos Christianos, Georgios Papoudakis, Arrasy Rahman, and Stefano V. Albrecht. Scaling multi-agent reinforcement learning with selective parameter sharing, 2021a. URL https://arxiv.org/abs/2102.07475.

Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. Shared experience actor-critic for multi-agent reinforcement learning, 2021b. URL https://arxiv.org/abs/2006.07169.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. Gep-pg: Decoupling exploration and exploitation in deep reinforcement learning algorithms, 2018. URL https://arxiv.org/abs/1802.05054.

Ruan de Kock, Omayma Mahjoub, Sasha Abramowitz, Wiem Khlifi, Callum Rhys Tilbury, Claude Formanek, Andries P. Smit, and Arnu Pretorius. Mava: a research library for distributed multi-agent reinforcement learning in jax. *arXiv preprint arXiv:2107.01460*, 2023. URL https://arxiv.org/pdf/2107.01460.pdf.

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

Matteo Gallici, Mario Martin, and Ivan Masmitja. Transfqmix: Transformers for leveraging the graph structure of multi-agent reinforcement learning problems. *arXiv preprint arXiv:2301.05334*, 2023.

Rihab Gorsane, Omayma Mahjoub, Ruan de Kock, Roland Dubb, Siddarth Singh, and Arnu Pretorius. Towards a standardised performance evaluation protocol for cooperative marl, 2022. URL https://arxiv.org/abs/2209.10485.

Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *Autonomous Agents and Multiagent Systems: AAMAS 2017 Workshops, Best Papers, São Paulo, Brazil, May 8-12, 2017, Revised Selected Papers 16*, pp. 66–83. Springer, 2017.

Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. Updet: Universal multi-agent reinforcement learning via policy decoupling with transformers. *arXiv preprint arXiv:2101.08001*, 2021.

Maximilian Hüttenrauch, Adrian Šošić, and Gerhard Neumann. Deep reinforcement learning for swarm systems. *J. Mach. Learn. Res.*, 20(1):1966–1996, January 2019. ISSN 1532-4435.

Sham Kakade. A natural policy gradient. In *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pp. 1531–1538, Cambridge, MA, USA, 2001. MIT Press.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, pp. 267–274, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. ISBN 1558608737.

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

Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning, 2022. URL https://arxiv.org/abs/2109.11251.

Yueheng Li, Guangming Xie, and Zongqing Lu. Guided policy optimization under partial observability, 2025. URL https://arxiv.org/abs/2505.15418.

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994.

Jiarong Liu, Yifan Zhong, Siyi Hu, Haobo Fu, Qiang Fu, Xiaojun Chang, and Yaodong Yang. Maximum entropy heterogeneous-agent reinforcement learning, 2025. URL https://arxiv.org/abs/2306.10715.

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

Omayma Mahjoub, Sasha Abramowitz, Ruan John de Kock, Wiem Khlifi, Simon Verster Du Toit, Jemma Daniel, Louay Ben Nessir, Louise Beyers, Juan Claude Formanek, Liam Clark, et al. Sable: a performant, efficient and scalable sequence model for marl. In *Forty-second International Conference on Machine Learning*, 2025.

Linghui Meng, Muning Wen, Chenyang Le, Xiyun Li, Dengpeng Xing, Weinan Zhang, Ying Wen, Haifeng Zhang, Jun Wang, Yaodong Yang, et al. Offline pre-trained multi-agent decision transformer. *Machine Intelligence Research*, 20(2):233–248, 2023.

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

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.

Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. *arXiv preprint arXiv:2006.07869*, 2020.

Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. Facmac: Factored multi-agent centralised policy gradients. *Advances in Neural Information Processing Systems*, 34:12208–12221, 2021.

Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning. *RSS*, 2018.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *The Journal of Machine Learning Research*, 21(1):7234–7284, 2020.

Alexander Rutherford, Benjamin Ellis, Matteo Gallici, Jonathan Cook, Andrei Lupu, Gardar Ingvarsson, Timon Willi, Ravi Hammond, Akbir Khan, Christian Schroeder de Witt, Alexandra Souly, Saptarashmi Bandyopadhyay, Mikayel Samvelyan, Minqi Jiang, Robert Tjarko Lange, Shimon Whiteson, Bruno Lacerda, Nick Hawes, Tim Rocktaschel, Chris Lu, and Jakob Nicolaus Foerster. Jaxmarl: Multi-agent rl environments and algorithms in jax, 2024. URL https://arxiv.org/abs/2311.10090.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization. *CoRR*, abs/1502.05477, 2015a. URL http://arxiv.org/abs/1502.05477.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *CoRR*, abs/1506.02438, 2015b. URL https://api.semanticscholar.org/CorpusID:3075448.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Lior Shani, Yonathan Efroni, and Shie Mannor. Adaptive trust region policy optimization: Global convergence and faster rates for regularized mdps. *CoRR*, abs/1909.02769, 2019. URL http://arxiv.org/abs/1909.02769.

Idan Shenfeld, Zhang-Wei Hong, Aviv Tamar, and Pulkit Agrawal. TGRL: An algorithm for teacher guided reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 31077–31093. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/shenfeld23a.html.

Arambam James Singh, Akshat Kumar, and Hoong Chuin Lau. Hierarchical multiagent reinforcement learning for maritime traffic management. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, pp. 1278–1286. International Foundation for Autonomous Agents and Multiagent Systems, 2020. doi: 10.5555/3398761.3398909.

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*, pp. 5887–5896. PMLR, 2019.

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

Justin K Terry, Nathaniel Grammel, Sanghyun Son, Benjamin Black, and Aakriti Agrawal. Revisiting parameter sharing in multi-agent deep reinforcement learning. *arXiv preprint arXiv:2005.13625*, 2020.

Wei-Cheng Tseng, Tsun-Hsuan Johnson Wang, Yen-Chen Lin, and Phillip Isola. Offline multi-agent reinforcement learning with knowledge distillation. *Advances in Neural Information Processing Systems*, 35:226–237, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. QPLEX: duplex dueling multi-agent q-learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021a.

Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. DOP: off-policy multi-agent decomposed policy gradients. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021b.

Andrew Warrington, Jonathan Wilder Lavington, A. Scibior, Mark W. Schmidt, and Frank D. Wood. Robust asymmetric learning in pomdps. In *International Conference on Machine Learning*, 2020. URL https://api.semanticscholar.org/CorpusID:229923742.

Luca Weihs, Unnat Jain, Iou-Jen Liu, Jordi Salvador, Svetlana Lazebnik, Aniruddha Kembhavi, and Alexander Schwing. Bridging the imitation gap by adaptive insubordination. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

Muning Wen, Jakub Grudzien Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Lin Xiao. On the convergence rates of policy gradient methods. *Journal of Machine Learning Research*, 23(282):1–36, 2022. URL http://jmlr.org/papers/v23/22-0056.html.

Yaodong Yang, Jianye Hao, Ben Liao, Kun Shao, Guangyong Chen, Wulong Liu, and Hongyao Tang. Qatten: A general framework for cooperative multiagent reinforcement learning. *arXiv preprint arXiv:2002.03939*, 2020.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Fuxiang Zhang, Chengxing Jia, Yi-Chen Li, Lei Yuan, Yang Yu, and Zongzhang Zhang. Discovering generalizable multi-agent coordination skills from multi-task offline data. In *The Eleventh International Conference on Learning Representations*, 2022.

Tianhao Zhang, Yueheng Li, Shuai Li, Qiwei Ye, Chen Wang, and Guangming Xie. Decentralized circle formation control for fish-like robots in the real-world via reinforcement learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8814–8820. IEEE, 2021a.

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

Jian Zhao, Xunhan Hu, Mingyu Yang, Wengang Zhou, Jiangcheng Zhu, and Houqiang Li. Ctds: Centralized teacher with decentralized student for multiagent reinforcement learning. *IEEE Transactions on Games*, 16(1):140–150, 2024. doi: 10.1109/TG.2022.3232390.

Yifan Zhong, Jakub Grudzien Kuba, Xidong Feng, Siyi Hu, Jiaming Ji, and Yaodong Yang. Heterogeneous-agent reinforcement learning, 2023. URL https://arxiv.org/abs/2304.09870.

Ming Zhou, Jun Luo, Julian Villela, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, Aurora Chongxi Huang, Ying Wen, Kimia Hassanzadeh, Daniel Graves, Dong Chen, Zhengbang Zhu, Nhat M. Nguyen, Mohamed Elsayed, Kun Shao, Sanjeevan Ahilan, Baokuan Zhang, Jiannan Wu, Zhengang Fu, Kasra Rezaee, Peyman Yadmellat, Mohsen Rohani, Nicolas Perez Nieves, Yihan Ni, Seyedershad Banijamali, Alexander Imani Cowen-Rivers, Zheng Tian, Daniel Palenicek, Haitham Bou-Ammar, Hongbo Zhang, Wulong Liu, Jianye Hao, and Jun Wang. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. In *Conference on Robot Learning*, 2020.

Yihe Zhou, Shunyu Liu, Yunpeng Qing, Kaixuan Chen, Tongya Zheng, Jie Song, and Mingli Song. Is centralized training with decentralized execution framework centralized enough for marl?, 2025. URL https://arxiv.org/abs/2305.17352.

THE USE OF LARGE LANGUAGE MODELS (LLMS)

LLMs are used to polish the paper writing.

A PROOFS

We will consider policy mirror descent (PMD) objective (Shani et al., 2019):

$$\mu^{(k+1)} = \arg\max_{\mu} \left\{ \eta_k \langle \nabla V_\rho(\mu^{(k)}), \mu \rangle - \frac{1}{1-\gamma} D_{d_\rho(\mu^{(k)})}(\mu, \mu^{(k)}) \right\}, \tag{13}$$

where η_k is the step size, $\rho \in \Delta(\mathcal{S})$ is an arbitrary state distribution and $d_\rho(\mu^{(k)})$ is the discounted state-visitation distribution under the policy $\mu^{(k)}$, $D_{d_\rho(\mu^{(k)})}$ is the weighted Bregman divergence. Considering that

$$\langle \nabla V_\rho(\mu^{(k)}), \mu \rangle = \sum_{s \in \mathcal{S}} \langle \nabla_s V_\rho(\mu^{(k)}), \mu(\cdot|s) \rangle, \tag{14}$$

we obtain

$$\begin{aligned} \mu^{(k+1)} &= \arg\max_{\mu} \left\{ \frac{1}{1-\gamma} \sum_{s \in \mathcal{S}} d_\rho(\mu^{(k)}) (\eta_k \langle Q^{\mu^{(k)}}(s, \cdot), \mu(\cdot|s) \rangle - D(\mu(\cdot|s), \mu^{(k)}(\cdot|s))) \right\}, \\ &= \arg\max_{\mu} \left\{ \sum_{s \in \mathcal{S}} (\eta_k \langle Q^{\mu^{(k)}}(s, \cdot), \mu(\cdot|s) \rangle - D(\mu(\cdot|s), \mu^{(k)}(\cdot|s))) \right\}. \end{aligned} \tag{15}$$

In this paper, we use KL divergence as a special case of Bregman divergence.

Theorem A.1 (Monotonic Improvement of MAGPO)**.** *A sequence $(\boldsymbol{\pi}_k)_{k=0}^{\infty}$ of joint policies updated by the four step of MAGPO has the monotonic property:*

$$V_\rho(\boldsymbol{\pi}_{k+1}) \geq V_\rho(\boldsymbol{\pi}_k), \ \forall k. \tag{16}$$

Proof. Following the derivation from Xiao (2022), the PMD objective is

$$\hat{\boldsymbol{\mu}}_k = \arg\max_{\boldsymbol{\mu}} \left\{ \eta_k \langle Q^{\boldsymbol{\mu}_k}(s, \cdot), \boldsymbol{\mu}(\cdot|s) \rangle - \mathrm{D}_{\mathrm{KL}}(\boldsymbol{\mu}(\cdot|s), \boldsymbol{\mu}_k(\cdot|s)) \right\}, \tag{17}$$

which admits the closed-form solution

$$\begin{aligned} \hat{\boldsymbol{\mu}}_k &= \boldsymbol{\mu}_k(\boldsymbol{a}|s) \frac{\exp\left(\eta_k Q^{\boldsymbol{\mu}_k}(s, \boldsymbol{a})\right)}{z_k(s)} \\ &= \boldsymbol{\pi}_k(\boldsymbol{a}|s) \frac{\exp\left(\eta_k Q^{\boldsymbol{\pi}_k}(s, \boldsymbol{a})\right)}{z_k(s)}. \end{aligned} \tag{18}$$

where we replace $\boldsymbol{\mu}_k$ with $\boldsymbol{\pi}_k$ due to the backtracking step.

Next, the learner update is defined as

$$\boldsymbol{\pi}_{k+1}(\cdot|s) = \arg\min_{\boldsymbol{\pi}} \mathrm{D}_{\mathrm{KL}}(\boldsymbol{\pi}(\cdot|s), \hat{\boldsymbol{\mu}}(\cdot|s)), \tag{19}$$

which guarantees the KL divergence decreases:

$$\mathrm{D}_{\mathrm{KL}}(\boldsymbol{\pi}^k(\cdot|s), \hat{\boldsymbol{\mu}}(\cdot|s)) \geq \mathrm{D}_{\mathrm{KL}}(\boldsymbol{\pi}^{k+1}(\cdot|s), \hat{\boldsymbol{\mu}}(\cdot|s)) \tag{20}$$

$$\mathbb{E}_{\boldsymbol{\pi}^k}\left[\log \boldsymbol{\pi}^k - \log \boldsymbol{\pi}^k - \eta_k Q^{\boldsymbol{\pi}^k}(s, \boldsymbol{a})\right] \geq \mathbb{E}_{\boldsymbol{\pi}^{k+1}}\left[\log \boldsymbol{\pi}^{k+1} - \log \boldsymbol{\pi}^k - \eta_k Q^{\boldsymbol{\pi}^k}(s, \boldsymbol{a})\right] \tag{21}$$

$$\eta_k \mathbb{E}_{\boldsymbol{\pi}^{k+1}}\left[Q^{\boldsymbol{\pi}^k}(s, \boldsymbol{a})\right] - \eta_k \mathbb{E}_{\boldsymbol{\pi}^k}\left[Q^{\boldsymbol{\pi}^k}(s, \boldsymbol{a})\right] \geq \mathrm{D}_{\mathrm{KL}}(\boldsymbol{\pi}^{k+1}(\cdot|s), \boldsymbol{\pi}^k(\cdot|s)) \tag{22}$$

Then, by the performance difference lemma (Kakade & Langford, 2002), we obtain:

$$\begin{aligned} V_\rho(\boldsymbol{\pi}_{k+1}) - V_\rho(\boldsymbol{\pi}_k) &= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\rho(\boldsymbol{\pi_{k+1}})}\left[\mathbb{E}_{\boldsymbol{\pi}^{k+1}}\left[Q^{\boldsymbol{\pi}^k}(s, \boldsymbol{a})\right] - \mathbb{E}_{\boldsymbol{\pi}^k}\left[Q^{\boldsymbol{\pi}^k}(s, \boldsymbol{a})\right]\right] \\ &\geq \frac{1}{1-\gamma}\frac{1}{\eta_k}\mathbb{E}_{\boldsymbol{\pi}^{k+1}}\left[\mathrm{D}_{\mathrm{KL}}(\boldsymbol{\pi}^{k+1}(\cdot|s), \boldsymbol{\pi}^k(\cdot|s))\right] \\ &\geq 0. \end{aligned} \tag{23}$$

□

Corollary A.2 (Sequential Update of MAGPO)**.** *The update of any individual policy π^{i_j} with any ordered subset $i_{1:j}$ can be written as:*

$$\pi^{i_j}_{k+1} = \arg\max_{\pi^{i_j}} \mathbb{E}_{a^{i_{1:j-1}}\sim\boldsymbol{\pi}^{i_{1:j-1}}_{k+1}, a^{i_j}\sim\pi^{i_j}}\left[A^{i_j}_{\boldsymbol{\pi}}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j})\right] - \frac{1}{\eta_k} D_{KL}(\pi^{i_j}, \pi^{i_j}_k) \tag{24}$$

Proof. We first decompose the guider policy in equation 18

$$\begin{aligned} \hat{\boldsymbol{\mu}}_k &= \boldsymbol{\pi}_k(\boldsymbol{a}|s)\frac{\exp\left(\eta_k Q^{\boldsymbol{\pi}_k}(s, \boldsymbol{a})\right)}{z_k(s)} \\ &= \boldsymbol{\pi}_k(\boldsymbol{a}|s)\exp\left(\eta_k Q^{\boldsymbol{\pi}_k}(s, \boldsymbol{a}) - \eta_k V^{\boldsymbol{\pi}_k}(s)\right)\frac{\exp\left(\eta_k V^{\boldsymbol{\pi}_k}(s)\right)}{z_k(s)} \\ &= \boldsymbol{\pi}_k(\boldsymbol{a}|s)\exp\left(\eta_k A^{\boldsymbol{\pi}_k}(s, \boldsymbol{a})\right)/\bar{z}_k(s) \\ &= \left(\prod_{j=1}^n \pi^{i_j}_k(a^{i_j}|s)\right)\exp\left(\eta_k \sum_{j=1}^n A^{i_j}_{\boldsymbol{\pi}}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j})\right)/\bar{z}_k(s) \\ &= \prod_{j=1}^n \pi^{i_j}_k(a^{i_j}|s)\frac{\exp\left(\eta_k A^{i_j}_{\boldsymbol{\pi}}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j})\right)}{z^i_k(s, \boldsymbol{a}^{i_{1:j-1}})}. \end{aligned} \tag{25}$$

This implies that the marginal guider policy for agent i_j is:

$$\hat{\mu}^{i_j}(a^{i_j}|s, \boldsymbol{a}^{i_{1:j-1}}) = \pi^{i_j}_k(a^{i_j}|s)\frac{\exp\left(\eta_k A^{i_j}_{\boldsymbol{\pi}}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j})\right)}{z^i_k(s, \boldsymbol{a}^{i_{1:j-1}})}. \tag{26}$$

Next, we decompose the KL divergence:

$$\begin{aligned} &\mathrm{D}_{\mathrm{KL}}(\boldsymbol{\pi}(\cdot|s), \hat{\boldsymbol{\mu}}(\cdot|s)) = \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}}\left[\log \boldsymbol{\pi}(\boldsymbol{a}|s) - \log \hat{\boldsymbol{\mu}}(\boldsymbol{a}|s)\right] \\ &= \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}}\left[\log\left(\prod_{j=1}^n \pi^{i_j}(a^{i_j}|s)\right) - \log\left(\prod_{j=1}^n \hat{\mu}^{i_j}(a^{i_j}|s, a^{i_{1:j-1}})\right)\right] \\ &= \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}}\left[\sum_{j=1}^n \log \pi^{i_j}(a^{i_j}|s) - \sum_{j=1}^n \log \hat{\mu}^{i_j}(a^{i_j}|s, a^{i_{1:j-1}})\right] \\ &= \sum_{j=1}^n \mathbb{E}_{\boldsymbol{a}^{i_{1:j-1}}\sim\boldsymbol{\pi}^{i_{1:j-1}}, a^{i_j}\sim\pi^{i_j}}\left[\log \pi^{i_j}(a^{i_j}|s) - \log \hat{\mu}^{i_j}(a^{i_j}|s, a^{i_{1:j-1}})\right]. \end{aligned} \tag{27}$$

Although the objective is not directly decoupled, we observe that each policy π^{i_j} is conditionally independent of the subsequent agents given the prior ones. Therefore, we can sequentially optimize:

$$\begin{aligned}
\pi_{k+1}^{i_1} &= \arg\min_{\pi^{i_1}} \mathbb{E}_{a^{i_1}\sim\pi^{i_1}} \left[\log \pi^{i_1}(a^{i_1}|s) - \log \hat{\mu}^{i_1}(a^{i_1}|s)\right] \\
\pi_{k+1}^{i_2} &= \arg\min_{\pi^{i_2}} \mathbb{E}_{a^{i_1}\sim\pi_{k+1}^{i_1}, a^{i_2}\sim\pi^{i_2}} \left[\log \pi^{i_2}(a^{i_2}|s) - \log \hat{\mu}^{i_2}(a^{i_2}|s, a^{i_1})\right] \\
&\cdots\cdots \\
\pi_{k+1}^{i_j} &= \arg\min_{\pi^{i_j}} \mathbb{E}_{a^{i_{1:j-1}}\sim\boldsymbol{\pi}_{k+1}^{i_{1:j-1}}, a^{i_j}\sim\pi^{i_j}} \left[\log \pi^{i_j}(a^{i_j}|s) - \log \hat{\mu}^{i_j}(a^{i_j}|s, a^{i_{1:j-1}})\right]
\end{aligned}$$

Substituting the expression for $\hat{\mu}^{i_j}$ yields:

$$\begin{aligned}
\pi_{k+1}^{i_j} &= \arg\min_{\pi^{i_j}} \mathbb{E}_{a^{i_{1:j-1}}\sim\boldsymbol{\pi}_{k+1}^{i_{1:j-1}}, a^{i_j}\sim\pi^{i_j}} \left[\log \pi^{i_j}(a^{i_j}|s) - \log \hat{\mu}^{i_j}(a^{i_j}|s, a^{i_{1:j-1}})\right] \\
&= \arg\min_{\pi^{i_j}} \mathbb{E}_{a^{i_{1:j-1}}\sim\boldsymbol{\pi}_{k+1}^{i_{1:j-1}}, a^{i_j}\sim\pi^{i_j}} \left[\log \pi^{i_j}(a^{i_j}|s) - \log \pi_k^{i_j}(a^{i_j}|s)) - \eta_k A_{\boldsymbol{\pi}}^{i_j}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j})\right] \\
&= \arg\max_{\pi^{i_j}} \mathbb{E}_{a^{i_{1:j-1}}\sim\boldsymbol{\pi}_{k+1}^{i_{1:j-1}}, a^{i_j}\sim\pi^{i_j}} \left[A_{\boldsymbol{\pi}}^{i_j}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j})\right] - \frac{1}{\eta_k}\mathrm{D}_{\mathrm{KL}}(\pi^{i_j}, \pi_k^{i_j}),
\end{aligned} \tag{28}$$

which completes the proof. □

B COORDSUM DETAILS

We introduce the **CoordSum** environment, a cooperative multi-agent benchmark designed to demonstrate the flaw of CTDS and evaluate the performance of existing paradigm. In this environment, a team of agents is tasked with selecting individual integers such that their sum matches a shared target, while avoiding repeated patterns that can be exploited by an adversarial guesser.

Naming Convention Each task in the CoordSum environment is denoted as:

`CoordSum-<num_agents>` × `<num_actions>` − `<max_target>`

where `<num_agents>` is the number of agents in the team, `<num_actions>` specifies the size of each agent's discrete action space, and `<max_target>` is the maximum possible target sum.

Observation Space At each timestep $t \in [1, 100]$, all agents receive the same observation: the current target value $target[t] \sim U(0, \texttt{<max\_target>})$. The observation also includes the current step count.

Action Space Each agent selects an integer action from a discrete set:

$$\mathcal{A}_i = \{0, 1, \ldots, \texttt{<num\_actions>} - 1\}$$

for $i = 1, \ldots, \texttt{<num\_agents>}$. The joint action is the vector of all agents' selected integers.

Reward Function To encourage agents to coordinate without relying on fixed or easily predictable strategies, the environment incorporates an opponent that attempts to guess the first agent's action using a majority vote over historical data. Specifically, for each target value, the environment records the first agent's past actions and uses the most frequent one as its guess. The reward at each timestep is defined as follows:

- If the sum of all agents' actions equals the current target:
  - A reward of **2.0** is given if the opponent's guess does *not* match the first agent's action.
  - A reward of **1.0** is given if the opponent's guess *does* match the first agent's action.
- If the sum does not match the target, a reward of **0.0** is given.

The same reward is distributed uniformly to all agents.

C Further Experimental Results

C.1 Per-task sample efficiency results

In Figure 7, we give all task-level aggregated results. In all cases, we report the mean with 95% bootstrap confidence intervals over 10 independent runs.

C.2 Per-task tabular results

In Table 1, we provide absolute episode metric (Colas et al., 2018; Gorsane et al., 2022) over training averaged across 10 seeds with std reported. The bolded value means the best performance across all methods, while highlighted value represents the best among CTDE methods.

C.3 Hyperparameters

All algorithms were tuned on each task with a tuning budget of 40 trials using the Tree-structured Parzen Estimator (TPE) implemented in the Optuna library (Akiba et al., 2019). Since some of the tuned hyperparameters are provided in Mava (de Kock et al., 2023), we directly adopt them and only tune the additional algorithms and tasks. Specifically, MAGPO and HAPPO in all tasks, all algorithms in the newly introduced CoordSum tasks.

The default hyperparameters for all methods are listed in Table 2 and Table 3. The full hyperparameter search spaces are provided in Table 4, Table 5, Table 6, and Table 7.

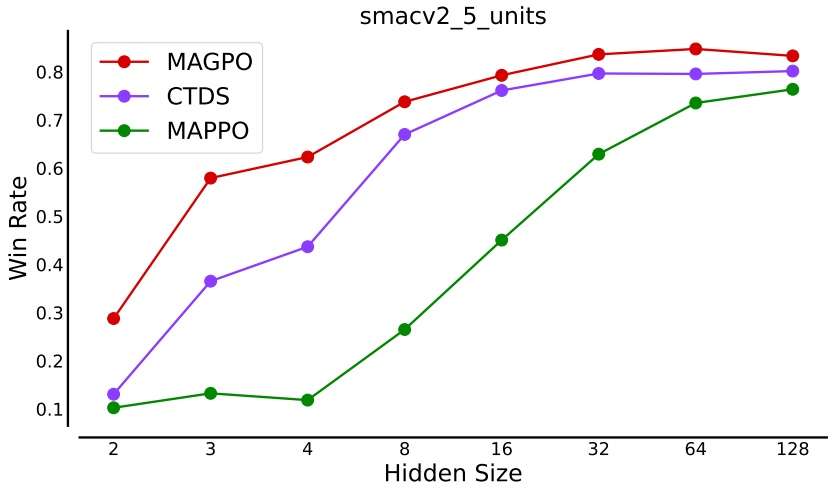


Figure 6: Effect of model capacity on performance.

C.4 Additional Experiment on Model Capacity

We evaluate MAGPO, CTDS, and MAPPO on *smacv2_5_units* to study the effect of model capacity mismatches between training and deployment. Specifically, we keep the size of all networks used only during training (e.g., critics, centralized policies) fixed at their full-capacity configuration, and vary only the hidden dimension of the deployable decentralized actors at evaluation time. This setting mimics a common practical scenario where a large teacher is trained but must be distilled into a smaller student for deployment due to compute or latency constraints.

As shown in Figure C.4, MAGPO consistently achieves higher returns than CTDS across all evaluated actor capacities, and the performance gap grows as the deployed actor becomes smaller. When compared to MAPPO, both MAGPO and CTDS — as teacher–student frameworks — better transfer knowledge from a large teacher to a small student. These results suggest that directly distilling from a large policy model is more effective than relying only on a value function for supervision,

and that MAGPO provides a principled approach for mitigating the performance loss caused by capacity-limited deployment.

Table 1: Performance comparison across tasks. Best overall value is bolded. Best among CTDE methods are highlighted.

	Task	MAGPO	CTDS	HAPPO	MAPPO	MAT	SABLE
CoordSum	3x10-30	153.13 ± 1.41	111.98 ± 11.15	153.32 ± 1.8	$\mathbf{156.37 \pm 3.56}$	68.29 ± 16.80	153.70 ± 3.77
	3x30-50	156.62 ± 1.59	76.27 ± 14.10	129.35 ± 5.2	$\mathbf{158.05 \pm 4.40}$	87.79 ± 8.79	125.04 ± 7.18
	5x20-80	$\mathbf{157.61 \pm 3.89}$	19.62 ± 11.79	119.19 ± 10.05	142.51 ± 4.87	86.86 ± 8.01	48.35 ± 15.76
	8x15-100	$\mathbf{129.94 \pm 8.47}$	23.96 ± 15.92	77.60 ± 5.09	127.32 ± 12.76	57.22 ± 4.97	28.91 ± 18.75
LevelBasedForaging	15x15-3p-5f	$\mathbf{0.99 \pm 0.00}$	0.96 ± 0.02	0.91 ± 0.02	0.97 ± 0.02	0.91 ± 0.02	0.96 ± 0.01
	15x15-4p-3f	$\mathbf{1.00 \pm 0.00}$	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.01	1.00 ± 0.00
	15x15-4p-5f	$\mathbf{0.99 \pm 0.00}$	0.99 ± 0.00	0.89 ± 0.02	0.98 ± 0.01	0.97 ± 0.01	0.99 ± 0.00
	2s-8x8-2p-2f-coop	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	$\mathbf{1.00 \pm 0.00}$	1.00 ± 0.00	$\mathbf{1.00 \pm 0.00}$
	2s-10x10-3p-3f	0.97 ± 0.01	0.87 ± 0.02	0.99 ± 0.01	$\mathbf{1.00 \pm 0.00}$	0.97 ± 0.01	0.99 ± 0.00
	8x8-2p-2f-coop	$\mathbf{1.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	10x10-3p-3f	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	$\mathbf{1.00 \pm 0.00}$	0.99 ± 0.00	1.00 ± 0.00
MaConnector	con-5x5x3a	$\mathbf{0.94 \pm 0.01}$	0.93 ± 0.02	0.93 ± 0.02	0.87 ± 0.02	0.85 ± 0.02	0.92 ± 0.02
	con-7x7x5a	$\mathbf{0.76 \pm 0.03}$	0.71 ± 0.05	0.67 ± 0.02	0.63 ± 0.02	0.62 ± 0.04	0.74 ± 0.03
	con-10x10x10a	$\mathbf{0.42 \pm 0.03}$	0.37 ± 0.04	0.21 ± 0.03	0.30 ± 0.01	0.22 ± 0.05	0.39 ± 0.03
	con-15x15x23a	0.02 ± 0.01	0.02 ± 0.01	0.00 ± 0.00	0.02 ± 0.01	0.00 ± 0.00	$\mathbf{0.08 \pm 0.01}$
MPE	spread_3ag	-6.10 ± 0.21	-6.34 ± 0.29	-6.63 ± 0.49	-6.72 ± 0.22	-6.59 ± 0.23	$\mathbf{-4.92 \pm 0.30}$
	spread_5ag	-18.67 ± 0.41	-20.38 ± 0.39	-23.42 ± 0.53	-22.84 ± 0.23	-25.30 ± 1.74	$\mathbf{-12.75 \pm 0.91}$
	spread_10ag	-40.51 ± 0.8	-40.09 ± 0.73	-43.68 ± 0.55	-41.83 ± 0.52	-50.07 ± 1.72	$\mathbf{-36.93 \pm 0.32}$
RobotWarehouse	large-4ag	$\mathbf{7.63 \pm 0.98}$	5.00 ± 0.54	0.61 ± 0.54	3.02 ± 2.26	4.61 ± 0.25	6.22 ± 1.73
	large-4ag-hard	$\mathbf{4.56 \pm 0.64}$	2.25 ± 1.50	0.00 ± 0.00	0.00 ± 0.01	2.28 ± 1.88	3.46 ± 1.80
	large-8ag	10.40 ± 0.52	7.68 ± 0.69	0.00 ± 0.00	8.35 ± 0.66	$\mathbf{14.72 \pm 0.79}$	11.01 ± 0.51
	large-8ag-hard	8.66 ± 0.61	4.32 ± 2.86	0.00 ± 0.00	3.38 ± 3.10	9.07 ± 0.71	$\mathbf{9.22 \pm 0.48}$
	medium-4ag	11.46 ± 1.16	8.22 ± 0.56	4.04 ± 0.63	7.82 ± 3.24	7.62 ± 3.83	$\mathbf{12.74 \pm 1.44}$
	medium-4ag-hard	$\mathbf{8.49 \pm 0.54}$	4.81 ± 0.79	3.75 ± 0.63	2.80 ± 2.77	4.64 ± 2.54	6.79 ± 1.34
	medium-6ag	$\mathbf{14.78 \pm 3.28}$	8.49 ± 1.07	4.99 ± 0.82	12.13 ± 0.53	13.32 ± 0.63	12.97 ± 1.03
	small-4ag	15.09 ± 0.71	11.07 ± 0.81	9.64 ± 2.43	10.52 ± 0.75	$\mathbf{18.27 \pm 0.53}$	16.47 ± 8.26
	small-4ag-hard	11.48 ± 0.41	7.56 ± 1.02	6.50 ± 1.09	9.44 ± 0.35	9.68 ± 3.19	$\mathbf{12.02 \pm 1.20}$
	tiny-2ag	19.70 ± 1.16	13.70 ± 1.96	10.93 ± 2.30	12.28 ± 5.86	17.06 ± 1.61	$\mathbf{21.17 \pm 1.24}$
	tiny-2ag-hard	$\mathbf{16.61 \pm 0.99}$	9.23 ± 0.88	9.61 ± 3.09	13.60 ± 0.73	13.44 ± 2.41	15.93 ± 0.74
	tiny-4ag	31.02 ± 1.95	14.42 ± 1.16	17.84 ± 2.17	26.29 ± 2.88	28.19 ± 1.02	$\mathbf{43.56 \pm 2.69}$
	tiny-4ag-hard	20.49 ± 2.61	11.31 ± 0.88	16.66 ± 2.50	19.01 ± 1.31	20.54 ± 11.99	$\mathbf{30.97 \pm 1.65}$
	xlarge-4ag	$\mathbf{5.74 \pm 0.71}$	3.02 ± 1.25	0.00 ± 0.00	3.73 ± 1.19	4.71 ± 0.43	3.76 ± 2.30
	xlarge-4ag-hard	0.31 ± 0.64	0.12 ± 0.34	0.00 ± 0.00	0.00 ± 0.00	0.39 ± 1.01	$\mathbf{0.70 \pm 1.39}$
Smax	2s3z	$\mathbf{1.00 \pm 0.00}$	0.99 ± 0.01	0.99 ± 0.01	1.00 ± 0.00	0.99 ± 0.00	1.00 ± 0.00
	3s5z	0.99 ± 0.01	0.98 ± 0.01	0.98 ± 0.01	1.00 ± 0.00	$\mathbf{1.00 \pm 0.00}$	1.00 ± 0.01
	3s5z_vs_3s6z	$\mathbf{0.95 \pm 0.02}$	0.89 ± 0.04	0.51 ± 0.11	0.91 ± 0.05	0.93 ± 0.03	0.94 ± 0.02
	3s_vs_5z	0.99 ± 0.01	0.97 ± 0.01	0.95 ± 0.01	$\mathbf{0.99 \pm 0.01}$	0.96 ± 0.01	0.99 ± 0.01
	6h_vs_8z	$\mathbf{1.00 \pm 0.00}$	0.99 ± 0.01	0.99 ± 0.00	1.00 ± 0.00	0.99 ± 0.01	1.00 ± 0.00
	10m_vs_11m	$\mathbf{0.98 \pm 0.02}$	0.80 ± 0.21	0.14 ± 0.03	0.39 ± 0.07	0.88 ± 0.19	0.83 ± 0.24
	5m_vs_6m	0.34 ± 0.35	0.15 ± 0.23	0.03 ± 0.01	0.28 ± 0.37	$\mathbf{0.68 \pm 0.36}$	0.59 ± 0.41
	smacv2_5_units	$\mathbf{0.83 \pm 0.02}$	0.80 ± 0.03	0.68 ± 0.02	0.76 ± 0.02	0.82 ± 0.02	0.78 ± 0.03
	smacv2_10_units	0.76 ± 0.05	0.65 ± 0.06	0.47 ± 0.06	$\mathbf{0.76 \pm 0.02}$	0.71 ± 0.04	0.65 ± 0.04
	27m_vs_30m	0.99 ± 0.01	0.99 ± 0.01	0.77 ± 0.05	0.83 ± 0.10	0.88 ± 0.09	$\mathbf{1.00 \pm 0.00}$

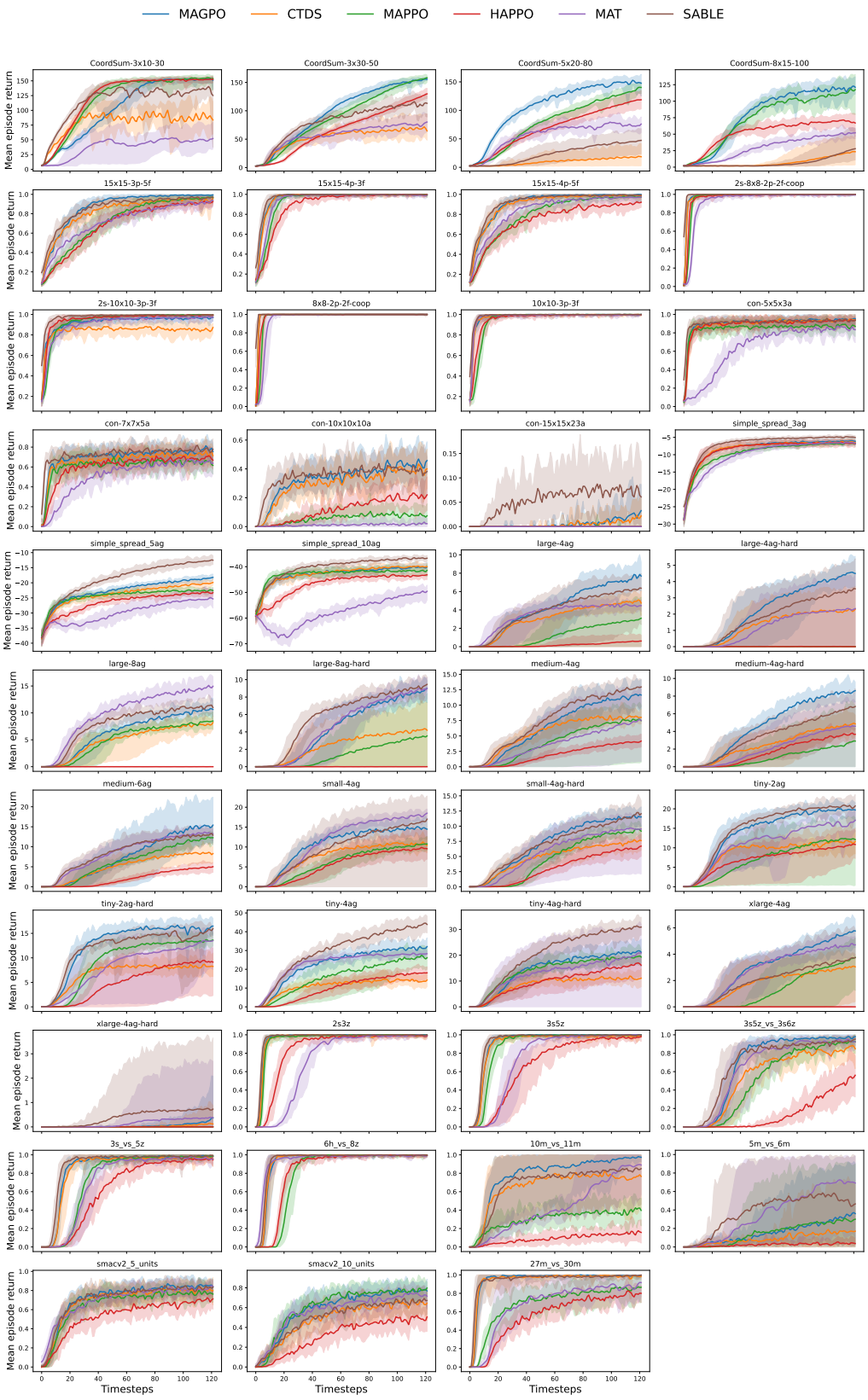


Figure 7: Mean episode return with 95% bootstrap confidence intervals on all tasks.

Table 2: Default hyperparameters for MAT and Sable.

Parameter	Value
Activation function	GeLU
Normalize Advantage	True
Value function coefficient	0.1
Discount	0.99
GAE	0.9
Rollout length	128
Add one-hot agent ID	True

Table 3: Default hyperparameters for HAPPO and MAPPO.

Parameter	Value
Value network layer sizes	[128,128]
Policy network layer sizes	[128,128]
Number of recurrent layers	1
Size of recurrent layer	128
Activation function	ReLU
Normalize Advantage	True
Value function coefficient	0.1
Discount	0.99
GAE	0.9
Rollout length	128
Add one-hot agent ID	True

Table 4: Hyperparameter Search Space for MAT.

Parameter	Value
PPO epochs	$\{2, 5, 10, 15\}$
Number of minibatches	$\{1, 2, 4, 8\}$
Entropy coefficient	$\{0.1, 0.01, 0.001, 1\}$
PPO clip ϵ	$\{0.05, 0.1, 0.2\}$
Maximum gradient norm	$\{0.5, 5, 10\}$
Learning rate	{1e-3, 5e-4, 2.5e-4, 1e-4, 1e-5}
Model embedding dimension	$\{32, 64, 128\}$
Number of transformer heads	$\{1, 2, 4\}$
Number of transformer blocks	$\{1, 2, 3\}$

Table 5: Hyperparameter Search Space for Sable.

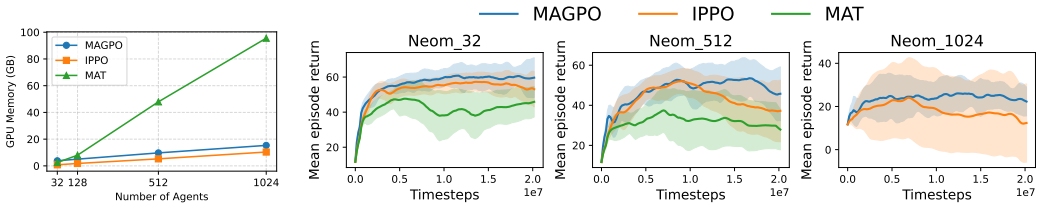
Parameter	Value
PPO epochs	$\{2, 5, 10, 15\}$
Number of minibatches	$\{1, 2, 4, 8\}$
Entropy coefficient	$\{0.1, 0.01, 0.001, 1\}$
PPO clip ϵ	$\{0.05, 0.1, 0.2\}$
Maximum gradient norm	$\{0.5, 5, 10\}$
Learning rate	{1e-3, 5e-4, 2.5e-4, 1e-4, 1e-5}
Model embedding dimension	$\{32, 64, 128\}$
Number retention heads	$\{1, 2, 4\}$
Number retention blocks	$\{1, 2, 3\}$
Retention heads scaling parameter	$\{0.3, 0.5, 0.8, 1\}$

Table 6: Hyperparameter Search Space for HAPPO and MAPPO.

Parameter	Value
PPO epochs	$\{2, 4, 8\}$
Number of minibatches	$\{2, 4, 8\}$
Entropy coefficient	$\{0, 0.01, 0.00001\}$
PPO clip ϵ	$\{0.05, 0.1, 0.2\}$
Maximum gradient norm	$\{0.5, 5, 10\}$
Value Learning rate	{1e-4, 2.5e-4, 5e-4}
Policy Learning rate	{1e-4, 2.5e-4, 5e-4}
Recurrent chunk size	$\{8, 16, 32, 64, 128\}$

Table 7: Hyperparameter Search Space for MAGPO.

Parameter	Value
Double clip δ	$\{1.1, 1.2, 1.3, 1.5, 2, 3\}$
RL auxiliary loss λ	$\{0, 1, 2, 4, 8\}$



(a) Memory usage

(b) Performance comparison

Figure 8: Memory usage and agent scalability experiment.