Learning to Coordinate with Anyone

Lei Yuan
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University; Polixir Technologies
Nanjing, China
yuanl@lamda.nju.edu.cn

Lihe Li
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
lilh@lamda.nju.edu.cn

Ziqian Zhang
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
zhangzq@lamda.nju.edu.cn

Feng Chen
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
chenf@lamda.nju.edu.cn

Tianyi Zhang
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
zhangty@lamda.nju.edu.cn

Cong Guan
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
guanc@lamda.nju.edu.cn

Yang Yu*
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University; Polixir Technologies
Nanjing, China
yuy@nju.edu.cn

Zhi-Hua Zhou
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
zhouzh@lamda.nju.edu.cn

ABSTRACT

In open multi-agent environments, the agents may encounter unexpected teammates. Classical multi-agent learning approaches train agents that can only coordinate with seen teammates. Recent studies attempted to generate diverse teammates in order to enhance the generalizable coordination ability, but were restricted by pre-defined teammates. In this work, our aim is to train agents with strong coordination ability by generating teammates that fully cover the teammate policy space, so that agents can coordinate with any teammates. Since the teammate policy space is too huge to be enumerated, we find only *dissimilar* teammates that are *incompatible* with controllable agents, which highly reduces the number of teammates that needed to be trained with. However, it is hard to determine the number of such incompatible teammates beforehand. We therefore introduce a continual multi-agent learning process, in which the agent learns to coordinate with different teammates until no more incompatible teammates can be found. The above idea is implemented in the proposed Macop (**M**ulti-**a**gent **co**mpatible **p**olicy learning) algorithm. We conduct experiments in 8 scenarios from 4 environments that have distinct coordination patterns. Experiments show that Macop generates training teammates with much lower compatibility than previous methods. As a result, in all scenarios Macop achieves the best overall coordination ability while never significantly worse than the baselines, showing strong generalization ability.

CCS CONCEPTS

• **Single/Multi-agent Learning → Multi-agent learning**.

KEYWORDS

Multi-agent System, Coordination and Cooperation, Continual Learning, Reinforcement Learning

ACM Reference Format:
Lei Yuan, Lihe Li, Ziqian Zhang, Feng Chen, Tianyi Zhang, Cong Guan, Yang Yu, and Zhi-Hua Zhou. 2023. Learning to Coordinate with Anyone. In *The Fifth International Conference on Distributed Artificial Intelligence (DAI*

*Yang Yu is the corresponding author.

DAI '23, November 30–December 03, 2023, Singapore, Singapore

ACM ISBN 979-8-4007-0848-0/23/11.
https://doi.org/10.1145/3627676.3627678

'23), November 30–December 03, 2023, Singapore, Singapore. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3627676.3627678

1 INTRODUCTION

Cooperative Multi-Agent Reinforcement Learning (MARL) [15] has garnered significant attention due to its demonstrated potential in various real-world applications. Recent studies have showcased MARL's exceptional performance in tasks such as pathfinding[21], active voltage control [27], and dynamic algorithm configuration [32]. However, these achievements are typically made within closed environments where teammates are pre-defined. The system will suffer from coordination ability decline when deploying the trained policies in real-world scenarios, where agents may encounter unexpected teammates in such open environments [34, 36].

Training with diverse teammates presents a promising avenue for tackling the aforementioned challenge. Various methods have emerged in domains such as ad-hoc teamwork [13], zero-shot coordination [26], and few-shot teamwork [4]. Addressing this challenge involves two crucial factors. Firstly, to enhance generalization and avoid overfitting to specific partners, it is essential for agents to be exposed to diverse teammates during the training process. Diversity can be achieved through various techniques, such as hand-crafted policies [16], object regularizers designed among agents [1, 2, 11], or population-based training [23, 31]. Secondly, when dealing with multiple teammates, especially in the context of multi-modal scenarios, specialized consideration is necessary. Approaches like self-play [22, 25], Fictitious Co-Play [6, 23], or co-evolving agent and partner populations [31], have been explored (related work in App.A.1). Nevertheless, complex scenarios often present substantial challenges arising from both complexity and vastness of the teammate policy space. On one hand, enumerating all possible teammate groups is a daunting task, and training the agents can be time-consuming. On the other hand, even when we pre-define only representative and diverse teammates, some instances may still be accidentally omitted. The exact number of such teammates cannot be determined in advance. This prompts a crucial question: Can we design a more efficient training paradigm that ensures our controllable agents are trained alongside partners in a policy space that guarantees coverage, ultimately enabling high generalization and coordination ability with diverse teammates?

Inspired by the idea of rehearsal learning [37], we tackle the mentioned issue and propose a novel coordination paradigm known as Macop, with which we can obtain a multi-agent compatible policy via incompatible teammates evolution. The core principle of Macop is the adversarial generation of new teammate instances, which are strategically crafted to challenge and refine the ego-system's (the agents we control) coordination policy. However, the exact number of representative teammates can not be determined beforehand, and maintaining a sufficiently diverse population requires significant computing and storage resources. We therefore introduce Continual Teammate Dec-POMDP (CT-Dec-POMDP), wherein the ego-system is trained with groups of teammates generated sequentially until convergence is reached. Our approach is rooted in two crucial factors: instance diversity and incompatibility between the newly generated teammates and the ego-system. During the training process, we iteratively refine teammate generation and optimize the ego-system until convergence is reached. This approach empowers the ego-system, leading to a coordination policy capable of seamlessly handling a wide array of team compositions and promptly adapting to new teammates.

We conduct experiments on different MARL benchmarks that have distinct coordination patterns, including Level-based Foraging (LBF) [17], Predator-Prey (PP), Cooperative Navigation (CN) from MPE [10], and StarCraft Multi-agent Challenge (SMAC) [20]. Experimental results show that Macop exhibits remarkable improvement compared with existing methods, achieving nearly 20% average performance improvement in the conducted benchmarks, and more experiments reveal it from multiple aspects.

2 PROBLEM FORMULATION

As we aim to solve a continual coordination problem, where the controllable agents cooperate with diverse teammates arising sequentially, we formalize it as a Continual Teammate Dec-POMDP (CT-Dec-POMDP) by extending the Dec-POMDP [14]. It can be described as a tuple $\mathcal{M} = \langle \mathcal{N}, \mathcal{S}, \mathcal{A}, P, \{\pi^k_{\text{tm}}\}^{\infty}_{k=1}, m, \Omega, O, R, \gamma\rangle$, $\mathcal{N} = \{1, \ldots, n\}$, $\mathcal{S}$, $\mathcal{A} = \mathcal{A}^1 \times ... \times \mathcal{A}^n$ and Ω are the sets of agents, global state, joint action, observation. P is the transition function, $\{\pi^k_{\text{tm}}\}^{\infty}_{k=1}$ represents the k groups of teammates encountered by the m controllable agents sequentially during the training phase, and $\gamma \in [0, 1)$ is the discounted factor. At each time step, agent i receives the observation $o^i = O(s, i)$ and outputs action $a^i \in \mathcal{A}^i$.

Concretely, when training to cooperate with a group of teammates π^k_{tm}, the agents do not have access to previous teammates groups $\pi^{k'}_{\text{tm}}, k' = 1, ..., k-1$. However, they are expected to remember how to cooperate with all previously encountered teammates groups. For simplicity, we denote a group of teammates as "teammate" when no ambiguity arises. The training phase of cooperating with teammate π^k_{tm} can be described as $\mathcal{M}_k = \langle \mathcal{N}, \mathcal{S}, \mathcal{A}, P, \boldsymbol{\pi}^k_{\text{tm}}, m, \Omega, O, R, \gamma\rangle$. The controllable agents $\boldsymbol{\pi}_{\text{ego}} = \{\pi^1_{\text{ego}}, ..., \pi^m_{\text{ego}}\} \in \Pi_{\text{ego}} = \otimes^m_{i=1}\Pi_i$ and the teammate $\boldsymbol{\pi}^k_{\text{tm}} = \{\pi^{k,m+1}_{\text{tm}}, ..., \pi^{k,n}_{\text{tm}}\} \in \Pi_{\text{tm}} = \otimes^n_{i=m+1}\Pi_i$ formulate a new joint policy $\langle \boldsymbol{\pi}_{\text{ego}}, \boldsymbol{\pi}^k_{\text{tm}}\rangle$. The joint action $\langle \boldsymbol{a}_{\text{ego}}, \boldsymbol{a}^k_{\text{tm}}\rangle = \langle \boldsymbol{\pi}_{\text{ego}}(\boldsymbol{\tau}_{\text{ego}}), \boldsymbol{\pi}^k_{\text{tm}}(\boldsymbol{\tau}^k_{\text{tm}})\rangle$ leads to the next state $s' \sim P(\cdot|s, \langle \boldsymbol{a}_{\text{ego}}, \boldsymbol{a}^k_{\text{tm}}\rangle)$ and the global reward $R(s, \langle \boldsymbol{a}_{\text{ego}}, \boldsymbol{a}^k_{\text{tm}}\rangle)$, where $\boldsymbol{\tau}_{\text{ego}} = \{\tau^i\}^m_{i=1}$, $\boldsymbol{\tau}^k_{\text{tm}} = \{\tau^i\}^n_{i=m+1}$. The controllable agents are optimized to maximize the expected return when cooperating with teammate $\boldsymbol{\pi}^k_{\text{tm}}$:

$$\max_{\boldsymbol{\pi}_{\text{ego}}} \mathcal{J}(\langle \boldsymbol{\pi}_{\text{ego}}, \boldsymbol{\pi}^k_{\text{tm}}\rangle) = \mathbb{E}_{\boldsymbol{\tau}\sim\rho(\langle \boldsymbol{\pi}_{\text{ego}}, \boldsymbol{\pi}^k_{\text{tm}}\rangle)}[G(\boldsymbol{\tau})], \tag{1}$$

where $G(\boldsymbol{\tau}) = \sum^T_{t=0} \gamma^t R(s_t, \boldsymbol{a}_t)$ is the return of a joint trajectory. At the same time, for a formal characterization of the relationship between the policy space of $\boldsymbol{\pi}_{\text{ego}}$ and $\boldsymbol{\pi}_{\text{tm}}$, we introduce the concept of complementary policy class:

Definition 2.1 (complementary policy class). For any sub policy $\boldsymbol{\pi} \in \Pi_{i:j} = \otimes^j_{h=i}\Pi_h, i \le j$, we define its complementary policy class as $\Pi^c_{\boldsymbol{\pi}} = \otimes^{i-1}_{h=1}\Pi_h \times \otimes^n_{h=j+1}\Pi_h$. We denote the complementary policy class of controllable agents and the teammate as Π^c_{ego} and Π^c_{tm} for simplicity. We also refer $\mathcal{J}_{\text{sp}}(\boldsymbol{\pi}_{\text{ego}}) = \max_{\bar{\boldsymbol{\pi}}_{\text{tm}}\in\Pi^c_{\text{ego}}} \mathcal{J}(\langle \boldsymbol{\pi}_{\text{ego}}, \bar{\boldsymbol{\pi}}_{\text{tm}}\rangle)$ and $\mathcal{J}_{\text{sp}}(\boldsymbol{\pi}_{\text{tm}}) = \max_{\bar{\boldsymbol{\pi}}_{\text{ego}}\in\Pi^c_{\text{tm}}} \mathcal{J}(\langle \bar{\boldsymbol{\pi}}_{\text{ego}}, \boldsymbol{\pi}_{\text{tm}}\rangle)$ as "self-play return" of $\boldsymbol{\pi}_{\text{ego}}$ and $\boldsymbol{\pi}_{\text{tm}}$, respectively.

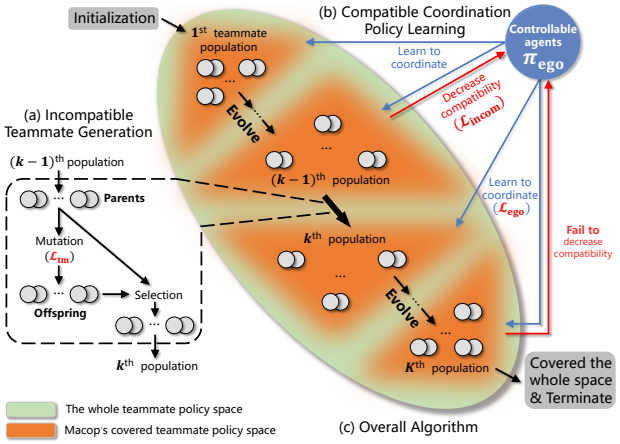


Figure 1: The overall workflow of Macop.

3 METHOD

In this section, we present the detailed design of our proposed method Macop (c.f. Fig. 1). First, we introduce a novel continual teammate generation module by combining population-based training and incompatible policy learning (Fig. 1(a)). Next, we outline the design of our continual coordination policy learning paradigm (Fig. 1(b)). These two phases proceed alternatively to train a robust multi-agent coordination policy that is capable of effectively cooperating with diverse teammates (Fig. 1(c)).

3.1 Incompatible Teammate Generation

The primary objective of Macop is to develop a joint policy that can effectively cooperate with diverse teammates. Since the policy space of teammate groups is too huge to be enumerated, we focus on identifying dissimilar teammate groups. To achieve this, we begin by establishing a complementary-policy-agnostic measure capable of effectively quantifying the similarity between two teammate groups, ensuring that it remains unaffected by complementary policies. In particular, we pair two teammate groups with any arbitrary complementary policy, as defined in Definition 2.1. These groups are considered similar if the probability of the trajectory produced by both groups surpasses a predefined threshold.

Definition 3.1 (ϵ-similar policies). We measure the similarity between two different teammates $\boldsymbol{\pi}^i_{\text{tm}}, \boldsymbol{\pi}^j_{\text{tm}}$ with the probability of the trajectory induced by them when paired with any complementary policies. Specifically, for any fixed complementary policy $\bar{\boldsymbol{\pi}} \in \Pi^c_{\text{tm}}$, the probability of the trajectory produced by the joint policy $P(\boldsymbol{\tau}|\langle\bar{\boldsymbol{\pi}}, \boldsymbol{\pi}_{\text{tm}}\rangle) = \prod_{t=0}^{T-1} \bar{\boldsymbol{\pi}}(\bar{\boldsymbol{a}}_t|\bar{\boldsymbol{\tau}}_t)\boldsymbol{\pi}_{\text{tm}}(\boldsymbol{a}_{tm,t}|\boldsymbol{\tau}_{tm,t})P(s_{t+1}|s_t, \langle\bar{\boldsymbol{a}}_t, \boldsymbol{a}_{tm,t}\rangle)$. Accordingly, we define the dissimilarity between the two teammates $d(\boldsymbol{\pi}^i_{\text{tm}}, \boldsymbol{\pi}^j_{\text{tm}}) = \max_{\boldsymbol{\tau}} |1 - \frac{P(\boldsymbol{\tau}|\langle\bar{\boldsymbol{\pi}}, \boldsymbol{\pi}^i_{\text{tm}}\rangle)}{P(\boldsymbol{\tau}|\langle\bar{\boldsymbol{\pi}}, \boldsymbol{\pi}^j_{\text{tm}}\rangle)}| = \max_{\boldsymbol{\tau}} |1 - \prod_{t=0}^{T-1} \frac{\boldsymbol{\pi}^i_{\text{tm}}(\boldsymbol{a}_{tm,t}|\boldsymbol{\tau}_{tm,t})}{\boldsymbol{\pi}^j_{\text{tm}}(\boldsymbol{a}_{tm,t}|\boldsymbol{\tau}_{tm,t})}|$. Teammates $\boldsymbol{\pi}^i_{\text{tm}}$ and $\boldsymbol{\pi}^j_{\text{tm}}$ are $\epsilon-$similar policies if and only if $d(\boldsymbol{\pi}^i_{\text{tm}}, \boldsymbol{\pi}^j_{\text{tm}}) \leq \epsilon, 0 \leq \epsilon \leq 1$, which implies that $1 - \epsilon \leq \frac{P(\boldsymbol{\tau}|\langle\bar{\boldsymbol{\pi}}, \boldsymbol{\pi}^i_{\text{tm}}\rangle)}{P(\boldsymbol{\tau}|\langle\bar{\boldsymbol{\pi}}, \boldsymbol{\pi}^j_{\text{tm}}\rangle)} \leq 1 + \epsilon, \forall\boldsymbol{\tau}$.

Based on the Def. 3.1 above, our approach involves the identification of representative teammate groups, ensuring that the dissimilarity between them surpasses the specified threshold ϵ. We continually generate such dissimilar teammate groups in order to gradually cover the space of teammate policies. Drawing inspiration from the proven efficacy of population-based training (PBT) [7] and evolutionary algorithms (EA) [38], we adopt an evolutionary process to formulate the teammate generation process by maintaining a population of teammates $\mathcal{P}_{\text{tm}} = \{\boldsymbol{\pi}^j_{\text{tm}}\}^{n_p}_{j=1}$ under the changing controllable agents $\boldsymbol{\pi}_{\text{ego}}$. By ensuring that the teammate groups exhibits dissimilarity between instances in not only the current population but also previous ones, our aim is to systematically explore and cover the entire teammate policy space over time.

Specifically, in each generation, the current population is first initialized through a customized parent selection mechanism (details provided later). We focus on promoting diversity within the teammate population, striving to enhance the dissimilarity between each individual, i.e., $\max \sum_{i\neq j} d(\boldsymbol{\pi}^i_{\text{tm}}, \boldsymbol{\pi}^j_{\text{tm}})$. To achieve the goal mentioned, we take Jensen-Shannon divergence (JSD) [5] as a reliable proxy to effectively measure the dissimilarity between teammates' policies as is introduced in [35]:

$$\mathcal{L}_{\text{div}} = \mathbb{E}_s[\frac{1}{n_p}\sum_{i=1}^{n_p} D_{KL}(\boldsymbol{\pi}^i_{\text{tm}}(\cdot|s)||\bar{\boldsymbol{\pi}}_{\text{tm}}(\cdot|s))], \tag{2}$$

where $\bar{\boldsymbol{\pi}}_{\text{tm}}(\cdot|s) = \frac{1}{n_p}\sum_{i=1}^{n_p} \boldsymbol{\pi}^i_{\text{tm}}(\cdot|s)$ is the average policy of the population, and D_{KL} is the Kullback-Leibler (KL) divergence between two distributions. We prove the JSD proxy is a certifiable lower bound of the original dissimilarity objective in App.A.2.

Despite the effectiveness of the population-based training with the $\mathcal{L}_{\text{div}}$ in Eqn. 2, the continual generation would still result in teammate groups with similar behaviors in different generations without other guarantees. Meanwhile, the size of the population n_p might also have a significant impact. Inspired by the relationship between similarity and compatibility proved in [1], we extend the theorem to our CT-Dec-POMDP:

Definition 3.2 (ϵ-compatible teammates). For the controllable agents $\boldsymbol{\pi}_{\text{ego}}$, let $\mathcal{J}_{\text{sp}}(\boldsymbol{\pi}_{\text{ego}}) = \alpha$. We refer $\boldsymbol{\pi}_{\text{tm}}$ as an ϵ-compatible teammate $\boldsymbol{\pi}_{\text{ego}}$ if and only if $\mathcal{J}(\langle\boldsymbol{\pi}_{\text{ego}}, \boldsymbol{\pi}_{\text{tm}}\rangle) \geq (1-\epsilon)\alpha$.

Theorem 3.3. *Given the controllable agents $\boldsymbol{\pi}_{ego}$ and teammate policies $\boldsymbol{\pi}_{tm}$ and $\forall\boldsymbol{\pi}'_{tm}$, $\boldsymbol{\pi}_{tm}$, $\boldsymbol{\pi}'_{tm}$ are $\epsilon-$similar policies. Then we have $(1-\epsilon)\mathcal{J}(\langle\boldsymbol{\pi}_{ego}, \boldsymbol{\pi}_{tm}\rangle) \leq \mathcal{J}(\langle\boldsymbol{\pi}_{ego}, \boldsymbol{\pi}'_{tm}\rangle) \leq (1+\epsilon)\mathcal{J}(\langle\boldsymbol{\pi}_{ego}, \boldsymbol{\pi}_{tm}\rangle)$.*

The underlying idea behind Thm. 3.3 is that controllable agents, when effectively collaborating with a specific teammate group, will also be compatible with the teammate group's ϵ-similar policies. Proofs are given in App.A.2. We thus have the following corollary:

Corollary 3.4. *Given the controllable agents $\boldsymbol{\pi}_{ego}$ and teammates $\boldsymbol{\pi}_{tm}$. If $\mathcal{J}(\langle\boldsymbol{\pi}_{ego}, \boldsymbol{\pi}'_{tm}\rangle) < (1-\epsilon)\mathcal{J}(\langle\boldsymbol{\pi}_{ego}, \boldsymbol{\pi}_{tm}\rangle)$, then $\boldsymbol{\pi}_{tm}$ and $\boldsymbol{\pi}'_{tm}$ are not ϵ-similar policies, i.e., $d(\boldsymbol{\pi}_{tm}, \boldsymbol{\pi}'_{tm}) > \epsilon$.*

The result from Cor. 3.4 shows that we can ensure that teammate groups generated in the current population are different from those before by decreasing its compatibility with the controllable agents $\boldsymbol{\pi}_{\text{ego}}$, which are trained to effectively collaborate with the teammates generated so far. Assuming that the controllable agents

are fixed during the teammate population evolving stage, the optimization objective can be written as:

$$\mathcal{L}_{\text{incom}} = -\frac{1}{n_p}\sum_{i=1}^{n_p}\mathcal{J}(\langle \boldsymbol{\pi}_{\text{ego}}, \boldsymbol{\pi}^i_{\text{tm}}\rangle). \tag{3}$$

To ensure the meaningful learning of teammate groups' policies, it is crucial for each individual in the population to be capable of cooperating with complementary policies. Thus, the optimization of teammate focuses on maximizing the following objective:

$$\mathcal{L}_{\text{sp}} = \frac{1}{n_p}\sum_{i=1}^{n_p}\mathcal{J}_{\text{sp}}(\boldsymbol{\pi}^i_{\text{tm}}),\ \textit{where } i = 1, 2, .., n_p. \tag{4}$$

Considering the specified objectives, the complete objective function for the teammate population is as follows:

$$\mathcal{L}_{\text{tm}} = \mathcal{L}_{\text{sp}} + \alpha_{\text{div}}\mathcal{L}_{\text{div}} + \alpha_{\text{incom}}\mathcal{L}_{\text{incom}}, \tag{5}$$

where, α_{div} and α_{incom} are adjustable hyper-parameters that control the balance between the three objectives.

3.2 Compatible Coordination Policy Learning

After generating a new teammate population that is diverse and incompatible with the controllable agents, we aim to train the controllable agents to effectively cooperate with newly generated teammate groups, as well as maintain the coordination ability with the trained ones. It requires the controllable agents to possess the continual learning ability, as introduced in Sec. 2, where teammate policies appear sequentially in CT-Dec-POMDP.

In the context of evolutionary-generated teammate groups appearing sequentially, employing a single generalized policy network poses challenges due to the existence of multi-modality and varying behaviors among teammate groups. Consequently, conflicts and degeneration in the controllable agents' policies may arise. To address this issue, recent approaches like MACPro [33] have adopted a solution where customized heads are learned for each specific task. Building upon this idea, our approach involves designing a policy network with a shared backbone denoted as f_ϕ, complemented by multiple output heads represented as $\{h_{\psi_i}\}_{i=1}^m$. The shared backbone is responsible for extracting relevant features, while each output head handles making the final decisions.

With the structured policy network, when paired with the new teammate group's policy $\boldsymbol{\pi}^{k+1}_{\text{tm}}$, we first instantiate a new output head $h_{\psi_{m+1}}$. Subsequently, our focus shifts to training the controllable agents to effectively cooperate with the new teammate group.

$$\mathcal{L}_{\text{com}} = \mathcal{J}(\langle \boldsymbol{\pi}_{\text{ego}}, \boldsymbol{\pi}^{k+1}_{\text{tm}}\rangle). \tag{6}$$

It is worth noting that once trained, the output heads $\{h_{\psi_i}\}_{i=1}^m$ remain fixed, and during the training process, the gradient $\mathcal{L}_{\text{com}}$ only propagates through the parameters ϕ and ψ_{m+1}.

Training the best response via $\mathcal{L}_{\text{com}}$ enables us to derive a policy that is capable of cooperating with the new teammate group $\boldsymbol{\pi}^{k+1}_{\text{tm}}$. However, the use of one shared backbone poses a challenge as it inevitably leads to forgetting previously learned cooperation. To mitigate this problem, we apply a regularization objective by constraining the parameters from changing abruptly while learning the new output head $h_{\psi_{m+1}}$:

$$\mathcal{L}_{\text{reg}} = -\frac{1}{m}\sum_{i=1}^{m}||\phi - \phi_i||_p, \tag{7}$$

where ϕ_i is the saved snapshot of the backbone ϕ after obtaining the i^{th} output head, and $||\cdot||_p$ is l_p norm. This regularization mechanism helps to retain previously learned knowledge and ensures that the shared backbone adapts to the new teammate. Striking a balance between adaptability and retaining relevant knowledge, we can effectively enhance the cooperative performance of the policy with diverse teammates. The overall objective of the controllable agents when encountering the $(k+1)^{\text{th}}$ teammate group is defined as:

$$\mathcal{L}_{\text{ego}} = \mathcal{L}_{\text{com}} + \alpha_{\text{reg}}\mathcal{L}_{\text{reg}}, \tag{8}$$

where α_{reg} is a tunable weight.

Despite the effectiveness of combining the proposed $\mathcal{L}_{\text{ego}}$ and the carefully designed policy network architecture, a major limitation lies in its poor scalability as the number of output heads increases linearly with the dynamically generated teammate groups. To achieve better scalability, we propose a resilient head expansion strategy that effectively reduces the number of output heads while maintaining the policy's compatibility. Upon completing the training of the output head $h_{\psi_{m+1}}$, we proceed to evaluate the coordination performance of this head and all the existing ones $\{h_{\psi_i}\}_{i=1}^{m+1}$ when paired with the new teammate group's policy $\boldsymbol{\pi}^{k+1}_{\text{tm}}$. The coordination performance is measured using the empirical average return $\{\hat{R}_i\}_{i=1}^{m+1}$, where $\hat{R}_i = \frac{1}{N}\sum_{j=1}^{N} G(\tau^i_j)$ represents the average return obtained by executing trajectories τ^i_j generated by applying the i^{th} output head. To manage the number of output heads and prevent uncontrolled growth, we choose to retain the newly trained head if its performance surpasses a certain threshold compared to the best-performing existing head. Formally, we keep the newly trained head if $\frac{\hat{R}_{m+1} - \max_i\{\hat{R}_i\}_{i=1}^m}{\max_i\{\hat{R}_i\}_{i=1}^m} \geq \lambda$. This approach ensures that we only expand the number of output heads when there is a substantial improvement in performance, indicating that the new teammate group's behavior requires a distinct policy. Otherwise, if the existing output heads are sufficiently generalized to cooperate effectively with the new teammate, no new head will be expanded.

By adopting this resilient head expand strategy, we strike a balance between reducing the number of output heads and maintaining the policy's adaptability, resulting in a more scalable and efficient approach to handling dynamic teammate groups under the continual coordination setting.

3.3 Overall Algorithm

In this section, we present a comprehensive overview of the Macop (Multi-agent Compatible Policy Learning) procedure. Macop aims to train controllable agents to effectively cooperate with various teammate groups. During the training phase, Macop employs an evolutionary method to generate diverse and incompatible teammate groups and trains the controllable agents to be compatible with the teammates under the continual setting. In each iteration (generation) $k(k > 1)$, we first select the $(k-1)^{\text{th}}$ teammate population

$\mathcal{P}_{\text{tm}}^{k-1}$ as the parent population. Then, the offspring population is derived by training the parent population with $\mathcal{L}_{\text{tm}}$ in Eqn. 5, i.e., mutation. The teammate groups are constructed based on value-based methods [18, 24], and $\boldsymbol{\pi}_{\text{tm}}(\cdot|s)$ is replaced with $softmax(Q_{\text{tm}}^{i}(\cdot|s))$ in $\mathcal{L}_{\text{div}}$ for practical use. With n_p teammate groups of the parent population and n_p teammate groups of the offspring population, we apply a carefully designed selection scheme as follows. To expedite the training of meaningful teammate groups, we first eliminate $\lfloor \frac{n_p}{2} \rfloor$ teammate groups with the lowest self-play return, i.e., $\max_{\bar{\boldsymbol{\pi}}_{\text{ego}}^{i} \in \Pi_{\text{tm}}^{c}} \mathcal{J}(\langle \bar{\boldsymbol{\pi}}_{\text{ego}}^{i}, \boldsymbol{\pi}_{\text{tm}}^{i} \rangle)$. Next, we proceed to eliminate $\lceil \frac{n_p}{2} \rceil$ teammate groups with the highest cross-play return under the controllable agents, i.e., $\mathcal{J}(\langle \boldsymbol{\pi}_{\text{ego}}, \boldsymbol{\pi}_{\text{tm}}^{i} \rangle)$, so as to improve incompatibility. Finally, we utilize the remaining n_p teammate groups as the new teammate population of iteration k, i.e., $\mathcal{P}_{\text{tm}}^{k}$.

With the teammate population $\mathcal{P}_{\text{tm}}^{k}$ in place, we construct n_p continual coordination processes in a sequential order and train controllable agents to learn compatible policies. The controllable agents are optimized using $\mathcal{L}_{\text{ego}}$ (defined in Eqn. 8), and the output head is expanded as introduced in Sec. 3.2.

To determine when the continual process should be terminated, a carefully designed stopping criterion is employed. The training phase terminates at the k^{th} iteration if the minimum cross-play return of $\mathcal{P}_{\text{tm}}^{k+1}$ and the controllable agents in iteration k exceeds a certain value, i.e., $C = \frac{\min_i \mathcal{J}(\langle \boldsymbol{\pi}_{\text{ego}}, \boldsymbol{\pi}_{\text{tm}}^{i} \rangle)}{\frac{1}{n_p} \sum_{i=1}^{n_p} \mathcal{J}_{\text{sp}}(\boldsymbol{\pi}_{\text{tm}}^{i})} \geq \xi,\ \boldsymbol{\pi}_{\text{tm}}^{i} \in \mathcal{P}_{\text{tm}}^{k+1}$. It indicates that the controllable agents at the k^{th} iteration can effectively cooperate with the $(k+1)^{\text{th}}$ teammate population even they have been trained to decrease the compatibility, and the teammate policy space is covered for a given environment.

During the testing phase, a meta-testing paradigm is employed to determine which output head is selected to pair with an unknown teammate group. Initially, all output heads are allowed to interact with the teammate group to collect a few trajectories, and their cooperation abilities are evaluated based on empirical returns. The output head with the highest performance is then chosen for testing. The pseudo-codes for both the training and testing phases of our Macop procedure are provided in App.A.3.

4 EXPERIMENTS

We conduct experiments to answer these questions: 1) Can Macop generate controllable agents capable of effectively collaborating with diverse teammates in different scenarios? 2) Does the evolutionary generation of teammates bring about a noticeable increase in diversity? 3) What is the detailed training process of Macop? 4) How does each component and hyperparameter influence Macop?

4.1 Environments and Baselines

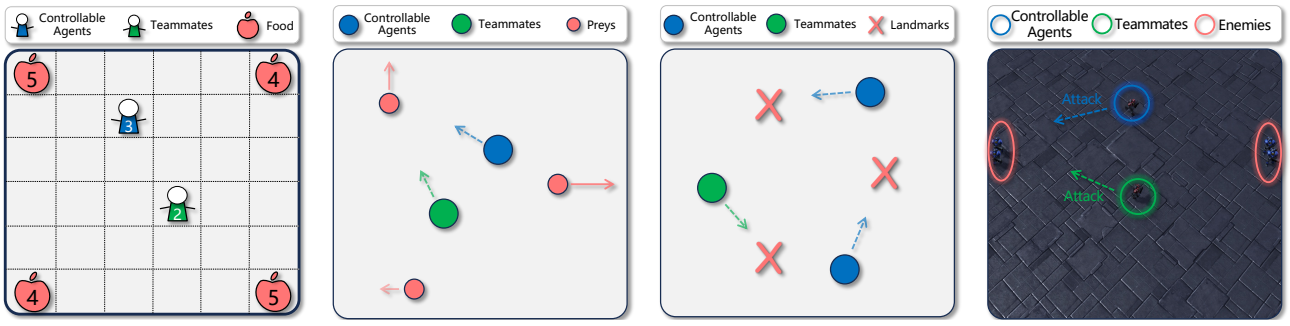
We select four multi-agent coordination environments and design eight scenarios as evaluation benchmarks (Fig. 2). Level-based Foraging (LBF) [17] is a challenging multi-agent cooperative game, where agents with varying levels navigate through a grid world, collaboratively striving to collect food with different levels. The successful collection occurs when the sum of levels of participating agents matches or exceeds the level of the food item. Predator Prey (PP) and Cooperative Navigation (CN) are two benchmarks coming from the popular MPE environment [10]. In the PP scenario, agents (predators) must together pursue the moving prey. In CN, multiple agents receive rewards when they navigate toward landmarks while ensuring they avoid collisions with one another. We also conduct experiments in the widely used StarCraft II environment, SMAC [20], which involves unit micromanagement tasks. In this setting, ally units are trained to beat enemy units controlled by the built-in AI. We specifically design two scenarios for each mentioned benchmark (e.g., PP1 and PP2), and details are in App.A.4.

To investigate whether Macop is capable of coordinating with diverse seen/unseen teammates, we implement Macop on the popular value-based methods VDN [24] and QMIX [18], and compare it with multiple baselines. First, to assess the impact of the teammate generation process on the coordination ability of the controllable agents, we compare Macop with FCP [23], which initially generates a set of teammate policies independently and then trains the controllable agents to be the best response to the set of teammates. The diversity among teammate polices is achieved solely through network random initialization. Additionally, we examine another population-based training mechanism that trains the teammate population using both $\mathcal{L}_{\text{sp}}$ and $\mathcal{L}_{\text{div}}$, aiming to generate teammates with enhanced diversity. This approach, which aligns with existing literature [3, 11], is referred to as TrajeDi for convenience. On the other hand, LIPO [1] induces teammate diversity by reducing the compatibility between the teammate policies in the population. Concretely, it trains the teammate population with an auxiliary objective $\mathcal{J}_{\text{LIPO}} = -\sum_{i \neq j} \mathcal{J}(\langle \boldsymbol{\pi}_{\text{tm}}^{i}, \boldsymbol{\pi}_{\text{tm}}^{j} \rangle)$, where the indices i, j refer to two randomly sampled teammates in the population. Furthermore, with the teammate generation module held constant, we proceed to compare Macop with Finetune. Finetune directly tunes all the parameters of the controllable agents to coordinate with the currently paired teammate group. We also investigate two other approaches: Single Head, which applies regularization $\mathcal{L}_{\text{reg}}$ to the backbone but does not utilize the multi-head architecture, and Random Head, which randomly selects an existing head during evaluation, thus verifying the necessity of Macop's testing paradigm. Finally, we employ the popular continual learning method EWC [8] to learn to coordinate with the teammates generated by TrajeDi, thereby providing an overall validation of the effectiveness of Macop. More details are illustrated in App.A.4.

4.2 Competitive Results

In this section, we analyze the effectiveness of the controllable agents learned from different methods from two aspects: coordination performance with diverse seen/unseen teammates, and continual learning ability on a sequence of incoming teammates.

Overall Coordination Performance: To ensure a fair comparison of coordination performance, we aggregate all the teammate groups generated by Macop and baselines into an *evaluation set*. For each method, we pair the learned controllable agents with teammate groups in this *evaluation set* to run 32 episodes for each pairing. The average episodic return over all episodes when pairing with different teammate groups is calculated as the evaluation metric. This metric serves as a comprehensive measure of the overall coordination performance and generalization ability of the controllable agents. We run each method for five distinct random seeds.



(a) Level-based Foraging (LBF) (b) Predator Prey (PP) (c) Cooperative Navigation (CN) (d) Starcraft Multi-agent Challenge (SMAC)

Figure 2: Environments used in this paper, all details could be seen in App.A.4.

Table 1: Average test return ± std when paired with teammate groups from *evaluation set* in different scenarios. We re-scale the value by taking the result of Finetune as an anchor and present average performance improvement w.r.t Finetune. The best result of each column is highlighted in bold. The symbols '+', '≈', and '-' indicate that the result is significantly inferior to, almost equivalent to, and superior to Macop, respectively, based on the Wilcoxon rank-sum test [12] with confidence level 0.05.

Method	LBF		PP		CN		SMAC		Avg. Performance
	LBF1	LBF4	PP1	PP2	CN2	CN3	SMAC1	SMAC2	Improvement (%)
Macop (ours)	1.14 ± 0.02	**1.64 ± 0.03**	**1.73 ± 0.11**	**2.14 ± 0.53**	**1.66 ± 0.03**	**1.70 ± 0.06**	**1.26 ± 0.42**	1.56 ± 0.17	**60.44**
Single Head	0.98 ± 0.07	1.10 ± 0.32	0.87 ± 0.58	1.44 ± 0.52	1.01 ± 0.49	0.99 ± 0.24	1.06 ± 0.14	1.25 ± 0.40	8.92
Random Head	0.92 ± 0.05	0.85 ± 0.10	0.88 ± 0.17	1.18 ± 0.39	0.98 ± 0.23	0.92 ± 0.11	0.97 ± 0.14	1.28 ± 0.21	−0.25
LIPO [1]	1.07 ± 0.09	1.53 ± 0.14	1.64 ± 0.21	1.93 ± 0.52	1.13 ± 0.41	1.33 ± 0.25	1.19 ± 0.18	1.08 ± 0.21	36.27
FCP [23]	**1.16 ± 0.02**	1.33 ± 0.06	1.17 ± 0.85	1.34 ± 0.12	0.90 ± 0.48	1.41 ± 0.23	0.97 ± 0.19	1.54 ± 0.10	25.82
TrajeDi [11]	1.16 ± 0.06	1.34 ± 0.11	1.68 ± 0.33	1.56 ± 0.52	1.29 ± 0.23	1.53 ± 0.11	1.25 ± 0.12	**1.57 ± 0.16**	42.26
EWC [8]	0.97 ± 0.08	0.99 ± 0.16	0.83 ± 0.48	0.77 ± 0.43	0.57 ± 0.37	0.71 ± 0.27	1.03 ± 0.13	0.61 ± 0.09	−18.82
Finetune	1.00 ± 0.16	1.00 ± 0.27	1.00 ± 0.58	1.00 ± 0.68	1.00 ± 0.31	1.00 ± 0.24	1.00 ± 0.17	1.00 ± 0.23	/
+/ ≈ /−	3/4/0	6/1/0	2/5/0	6/1/0	7/0/0	7/0/0	5/2/0	4/3/0	7/0/0

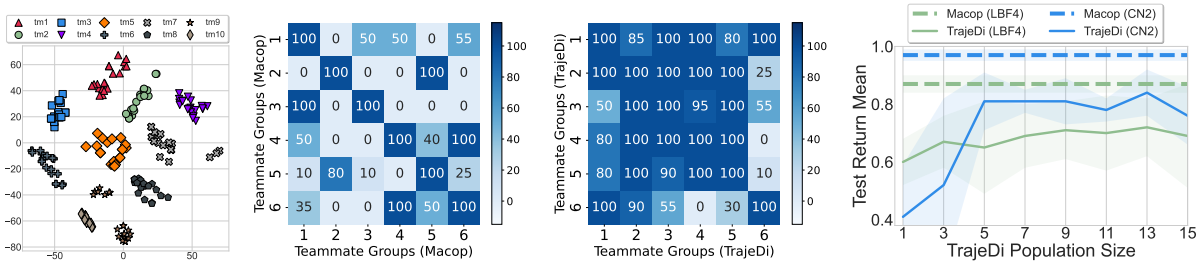
As depicted in Tab. 1, we observe that FCP, TrajeDi, and LIPO exhibit limited coordination ability in different scenarios. This highlights the need for ample coverage in teammate policy space to establish a robust coordination policy. Intriguingly, the three mentioned methods have similar performance, indicating that certain design elements, such as instance diversity among teammates, fail to fundamentally address this challenge. In contrast, when using generated teammates, simply finetuning the multi-agent policy or employing widely-used continual approaches like EWC exhibits inferior coordination performance, as confirmed by our experiments and in line with the findings in MACPro [33]. It proves the necessity of specialized designs tailored for multi-agent continual learning settings. Furthermore, Macop exhibits a remarkable performance advantage over nearly all baselines across various scenarios, demonstrating that controllable agents trained by Macop possess robust coordination abilities. Furthermore, we discovered that the Single Head architecture struggles due to the presence of multi-modality in teammate behavior, underscoring the necessity of a multi-head architecture. An effectively designed testing paradigm, utilizing multiple available learned heads, proves indispensable. It is worth noting that Random Head fails to select the optimal head for evaluation, resulting in a degradation in performance. Our pipeline relies on efficient design for continual learning, and more comprehensive results on the necessity of each component can be found in Sec. 4.5.

Continual Learning Ability: To investigate the continual learning ability of different methods, we utilize all teammate groups generated by Macop to construct a fixed teammate sequence. Four continual learning methods are applied to train the controllable agents to coordinate with this teammate sequence in a continual manner, including Macop, a replay-based method CLEAR [19], EWC [8] and Finetune. We introduce two metrics inspired by the continual learning setting [29, 30]: (1) Backward Transfer BWT= $\frac{1}{K-1}(\sum_{k=2}^{K}\frac{1}{k-1}\sum_{j=1}^{k-1}(\alpha_k^j-\alpha_j^j))$ evaluates the average influence of learning to cooperate with the newest teammate group on coordination with previously encountered teammates. (2) Forward Transfer FWT= $\frac{1}{K-1}(\sum_{k=2}^{K}\frac{1}{k-1}\sum_{j=2}^{k}(\alpha_j^j-\tilde{\alpha}_j))$ assesses the average influence of all previously encountered teammate groups on coordination with the new teammate. Here, α_k^j is the episodic return of the controllable agents paired with the j^{th} teammate group after completing training to cooperate with the k^{th} teammate group. Additionally, $\tilde{\alpha}_j$ is the episodic return of a randomly initialized complementary policy trained with the j^{th} teammate group.

We record experimental results in Tab. 2. Finetune demonstrates the worst BWT among all methods, validating the necessity of algorithm design to prevent catastrophic forgetting. However, even popular continual learning methods, CLEAR and EWC, grapple with forgetting to some degree. In contrast, Macop achieves the best BWT in all evaluated environments. As for FWT, Macop obtains a competitive result compared with other methods. Taking both BWT and FWT into consideration, Macop demonstrates a strong continual learning ability, empowering controllable agents to progressively acquire coordination ability with diverse teammates, along with the expanding coverage of teammate policy space.

Table 2: Continual learning ability. Average BWT/FWT ± std of four different methods.

Method	LBF4		PP1		CN3		SMAC1	
	BWT	FWT	BWT	FWT	BWT	FWT	BWT	FWT
Macop	**−0.01 ± 0.02**	**0.07 ± 0.07**	**0.03 ± 0.04**	−0.16 ± 0.18	**0.04 ± 0.06**	**0.10 ± 0.09**	**−0.02 ± 0.11**	0.07 ± 0.19
CLEAR	−0.05 ± 0.07	**0.07 ± 0.06**	0.01 ± 0.08	**−0.05 ± 0.11**	−0.16 ± 0.15	0.00 ± 0.20	−0.50 ± 0.32	0.04 ± 0.35
EWC	−0.30 ± 0.08	0.05 ± 0.07	−0.34 ± 0.08	−0.05 ± 0.13	−0.20 ± 0.11	0.03 ± 0.11	−1.02 ± 0.47	0.05 ± 0.31
Finetune	−0.34 ± 0.07	0.04 ± 0.06	−0.37 ± 0.07	−0.05 ± 0.22	−0.31 ± 0.11	0.05 ± 0.10	−1.24 ± 0.51	**0.33 ± 0.35**



(a) (b) (c) (d)

Figure 3: Teammate policy space analysis. (a) The t-SNE projections of the self-play trajectory features of Macop's generated teammate groups in CN2. (b)(c) The cross-play returns of Macop's and TrajeDi's generated teammate groups in LBF4. (d) The change in TrajeDi's coordination ability with varying population sizes in LBF4 and CN2, compared with Macop.

4.3 Teammate Policy Space Analysis

To investigate whether Macop is capable of generating teammate groups with diverse behaviors, a straightforward method involves comparing the self-play trajectories of different teammate groups. Concretely, we first learn a transformer-based encoder to map trajectories into a low-dimensional feature space (details will be provided in App.A.4.3). We subsequently encode the teammates' self-play trajectories generated by Macop into the feature space. For visualization, we sample 10 teammate groups in the CN2 scenario and extract their trajectory features, as shown in Fig. 3(a). The projection displays a notable dispersion, validating that teammate groups generated by Macop exhibit diverse behaviors as expected.

Furthermore, we conduct experiments to assess the compatibility among the generated teammate groups. In accordance with Def. 3.2, we paired different teammate groups in LBF4. The cross-play returns of Macop and TrajeDi are presented in Fig. 3(b)(c). It is evident that when pairing two distinct groups from Macop, there is a noticeable drop in returns outside the main diagonal, indicating the incompatibility among the teammate groups generated by Macop. Conversely, the cross-play returns of TrajeDi's teammate groups are nearly identical to their self-play returns, suggesting a significantly lower level of incompatibility among teammate groups because of the poorer coverage of teammate policy space.

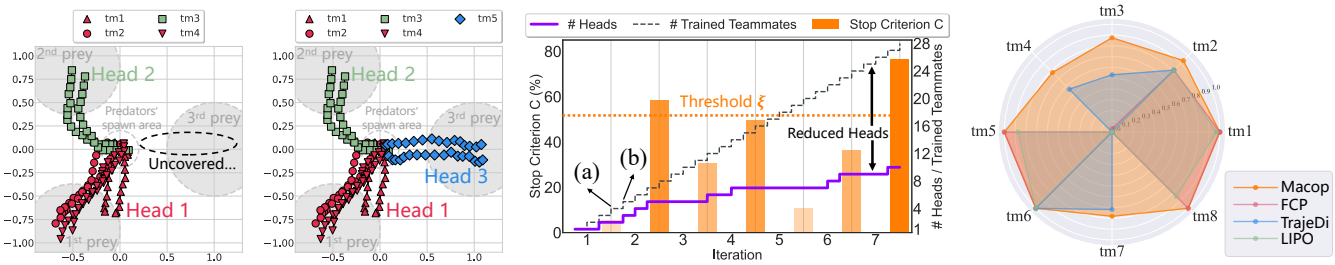
To further explore whether methods without dynamic teammate generation can realize policy space coverage by increasing the population size, we obtain teammates using TrajeDi, varying in population size from 1 to 15, to train the controllable agents. Subsequently, we evaluate their coordination ability using the *evaluation set*, as depicted in Fig. 3(d). The results illustrate that coordination ability improves as the population size increases until convergence. However, a considerable performance gap between TrajeDi and Macop persists. It proves that vanilla methods that lack dynamic teammate generation struggle with new and unfamiliar teammates due to inadequate coverage of the teammates' policy space. On the contrary, Macop's deliberate generation of incompatible teammates contributes to a more comprehensive coverage of the teammate policy space, ultimately enhancing its coordination ability.

4.4 Learning Process Analysis

To gain a comprehensive understanding of Macop's functioning, it's essential to delve into its learning process, which involves generating incompatible teammates and refining controllable agents until convergence. Fig. 4 illustrates the process in PP1, showcasing key aspects, including the number of generated teammate groups, the number of policy heads, and the stop criterion C, on each iteration (Fig. 4(c)). In the first iteration, the teammate generation module produces a population of four distinct teammate groups, with three specializing in capturing the first prey and one focused on the second prey (Fig. 4(a)). However, the population lacks desired diversity, as none of the groups learn to catch the remaining third prey. Controllable agents acquire the ability to collaborate with their teammates: Head 1 coordinates to capture the first prey, while Head 2 interacts with the group targeting the second prey.

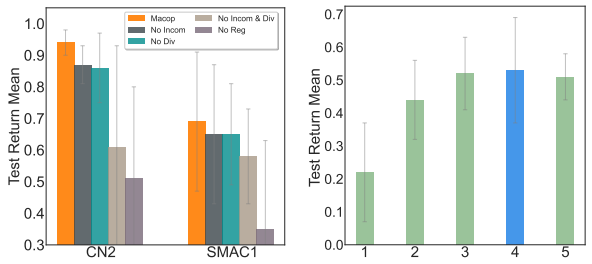
During the second iteration, the teammate generation module generates new teammates incompatible with the controllable agents, expanding the coverage of the teammate policy space. As shown in Fig. 4(b), a new teammate group ("tm5" in blue) successfully learns to capture the last prey, showcasing a novel behavior. Consequently, when the controllable agents complete their training with this new group, they establish a new head for better coordination.

The dynamic interplay between the adversarial teammate generation module and the training of controllable agents persists until the seventh iteration, resulting in an increased number of teammate groups and policy heads. In this final iteration, the teammate generation module endeavors to generate seemingly "incompatible" teammates as it has throughout the training process, but it encounters failure. The generated teammate groups up to this point have already effectively covered a wide range of the teammate policy



(a) (b) (c) (d)

Figure 4: Macop's learning process analysis. (a)(b) The self-play trajectories of the first four/five teammate groups. (c) The change of the number of trained teammate groups, the number of existing heads, and the stop criterion C on each iteration. (d) Coordination performance comparison with different teammate groups in the *evaluation set*.



(a) Ablation Study (b) Sensitivity of population size on PP2

Figure 5: Ablation and sensitivity studies.

space. The controllable agents have successfully learnt to coordinate with a sufficiently diverse array of teammates. The newly generated teammate groups do not exhibit enough incompatibility, as indicated by the stop criterion surpassing the specified threshold ξ. This signifies that the cross-play performance between the controllable agents and these new "incompatible" teammates is comparable to the self-play performance of the teammate groups. It's worth noting that the C value of the second iteration also exceeds the threshold, yet a minimum iteration count of 4 is enforced to ensure thorough exploration of the teammate policy space. This automated learning process within Macop terminates after the seventh iteration. As a result, Macop obtains controllable agents that possess 10 heads and strong coordination ability, as illustrated in Fig. 4(d).

4.5 Ablation and Sensitivity Studies

We here conduct ablation studies on CN2 and SMAC1 to comprehensively assess the impacts of multiple modules. *No Incom*, *No Div*, and *No Incom & Div*, are derived by setting $\alpha_{\text{incom}} = 0$, $\alpha_{\text{div}} = 0$, and $\alpha_{\text{incom}} = \alpha_{\text{div}} = 0$, respectively. Furthermore, we examine the impact of $\mathcal{L}_{\text{reg}}$, and designate this variant as *No Reg* to explore the effects of regularization on the backbone network ϕ. To ensure a fair comparison, we incorporate the teammate groups generated by the four ablations into the *evaluation set*. The results, as illustrated in Fig. 5(a), reveal essential insights into the functioning of Macop. Removing $\mathcal{L}_{\text{incom}}$ or $\mathcal{L}_{\text{div}}$ leads to a performance degradation compared to the complete Macop, highlighting the significant contributions to the teammate diversity. Moreover, No Incom & Div exhibits a substantial performance degradation, verifying the necessity of actively generating diverse teammates, instead of relying solely on random network initialization. Furthermore, No Reg demonstrates the poorest performance among all the variants. The absence of regularization on the backbone network undermines the controllable agents' continual learning ability, weakening their coordination capability with diverse teammates. These findings emphasize that each module plays an indispensable role in Macop.

As Macop includes multiple hyperparameters, we conduct experiments to investigate their sensitivity. The teammate groups generated by different hyperparameter settings are also incorporated into the *evaluation set* for a fair comparison. One important hyperparameter is the population size n_p. On one hand, with a very small population, Macop cannot cover the teammate policy space in an efficient manner. On the other hand, setting the population size to an excessively large number will unnecessarily increase the running time of Macop, reducing the overall efficiency. As shown in Fig. 5(b), we can find that when $n_p \leq 4$, the performance of Macop does improve with increasing population size. However, there is no further improvement as we continue to increase n_p, proving that $n_p = 4$ is the best setting in scenario PP2. More detailed analysis of other important hyperparameters is provided in App.A.5.

5 FINAL REMARKS

We propose a novel approach to multi-agent policy learning called Macop, which is designed to enhance the coordination abilities of controllable agents when working with diverse teammates. Our approach starts by framing the problem as an CT-Dec-POMDP. This framework entails training the ego-system with sequentially generated groups of teammates until convergence is achieved. Empirical results obtained across various environments, compared against multiple baseline methods, provide strong evidence of its effectiveness. Looking ahead, in scenarios where we operate under a few-shot setting and need to collect some trajectories for selectingan the optimal head during policy deployment, developing mechanisms such as context-based recognition could be a potential future solution. Additionally, an intriguing direction for future research involves harnessing the capabilities of large language models [28] like ChatGPT [9] to expedite the learning process and further enhance the generalization capabilities of our approach.

ACKNOWLEDGMENTS

This work is supported by National Key Research and Development Program of China (2020AAA0107200), the National Science Foundation of China (61921006, 62022039, 62276124). We would like to thank Yichen Li and, Ke Xue for their valuable support.

REFERENCES

[1] Rujikorn Charakorn, Poramate Manoonpong, and Nat Dilokthanakul. 2023. Generating Diverse Cooperative Agents by Learning Incompatible Policies. In *ICLR*.

[2] Kenneth Derek and Phillip Isola. 2021. Adaptable agent populations via a generative model of policies. In *NeurIPS*. 3902–3913.

[3] Hao Ding, Chengxing Jia, Cong Guan, Feng Chen, Lei Yuan, Zongzhang Zhang, and Yang Yu. 2023. Coordination Scheme Probing for Generalizable Multi-Agent Reinforcement Learning. https://openreview.net/forum?id=PAKkOriJBd

[4] Elliot Fosong, Arrasy Rahman, Ignacio Carlucho, and Stefano V Albrecht. 2022. Few-shot teamwork. *preprint arXiv:2207.09300* (2022).

[5] Bent Fuglede and Flemming Topsøe. 2004. Jensen-Shannon divergence and Hilbert space embedding. In *ISIT*.

[6] Johannes Heinrich, Marc Lanctot, and David Silver. 2015. Fictitious self-play in extensive-form games. In *ICML*. 805–813.

[7] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. 2019. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science* 364, 6443 (2019), 859–865.

[8] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences* 114, 13 (2017), 3521–3526.

[9] Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, et al. 2023. Summary of chatgpt/gpt-4 research and perspective towards the future of large language models. *preprint arXiv:2304.01852* (2023).

[10] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NIPS*. 6379–6390.

[11] Andrei Lupu, Brandon Cui, Hengyuan Hu, and Jakob Foerster. 2021. Trajectory diversity for zero-shot coordination. In *ICML*. 7204–7213.

[12] Henry B Mann and Donald R Whitney. 1947. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics* (1947), 50–60.

[13] Reuth Mirsky, Ignacio Carlucho, Arrasy Rahman, Elliot Fosong, William Macke, Mohan Sridharan, Peter Stone, and Stefano V Albrecht. 2022. A Survey of Ad Hoc Teamwork: Definitions, Methods, and Open Problems. *preprint arXiv:2202.10450* (2022).

[14] Frans A Oliehoek and Christopher Amato. 2016. *A Concise Introduction to Decentralized POMDPs*. Springer.

[15] Afshin Oroojlooy and Davood Hajinezhad. 2023. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence* 53, 11 (2023), 13677–13722.

[16] Georgios Papoudakis, Filippos Christianos, and Stefano Albrecht. 2021. Agent modelling under partial observability for deep reinforcement learning. In *NeurIPS*. 19210–19222.

[17] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. 2021. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *NeurIPS*.

[18] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*. 4295–4304.

[19] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Greg Wayne. 2018. Experience Replay for Continual Learning. In *NeurIPS*. 348–358.

[20] Mikayel Samvelyan, Tabish Rashid, Christian Schröder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob N. Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. In *AAMAS*. 2186–2188.

[21] Guillaume Sartoretti, Justin Kerr, Yunfei Shi, Glenn Wagner, TK Satish Kumar, Sven Koenig, and Howie Choset. 2019. Primal: Pathfinding via reinforcement and imitation multi-agent learning. *IEEE Robotics and Automation Letters* 4, 3 (2019), 2378–2385.

[22] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* 362, 6419 (2018), 1140–1144.

[23] DJ Strouse, Kevin McKee, Matt Botvinick, Edward Hughes, and Richard Everett. 2021. Collaborating with humans without human data. In *NeurIPS*. 14502–14515.

[24] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. 2018. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *AAMAS*. 2085–2087.

[25] Gerald Tesauro. 1994. TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural computation* 6, 2 (1994), 215–219.

[26] Johannes Treutlein, Michael Dennis, Caspar Oesterheld, and Jakob Foerster. 2021. A new formalism, method and open issues for zero-shot coordination. In *ICML*. 10413–10423.

[27] Jianhong Wang, Wangkun Xu, Yunjie Gu, Wenbin Song, and Tim C. Green. 2021. Multi-Agent Reinforcement Learning for Active Voltage Control on Power Distribution Networks. In *NeurIPS*. 3271–3284.

[28] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2023. A Survey on Large Language Model based Autonomous Agents. *preprint arXiv:2308.11432* (2023).

[29] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. 2023. A comprehensive survey of continual learning: Theory, method and application. *preprint arXiv:2302.00487* (2023).

[30] Maciej Wołczyk, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. 2021. Continual world: A robotic benchmark for continual reinforcement learning. In *NeurIPS*. 28496–28510.

[31] Ke Xue, Yutong Wang, Lei Yuan, Cong Guan, Chao Qian, and Yang Yu. 2022. Heterogeneous Multi-agent Zero-Shot Coordination by Coevolution. *preprint arXiv:2208.04957* (2022).

[32] Ke Xue, Jiacheng Xu, Lei Yuan, Miqing Li, Chao Qian, Zongzhang Zhang, and Yang Yu. 2022. Multi-agent dynamic algorithm configuration. In *NeurIPS*. 20147–20161.

[33] Lei Yuan, Lihe Li, Ziqian Zhang, Fuxiang Zhang, Cong Guan, and Yang Yu. 2023. Multi-agent Continual Coordination via Progressive Task Contextualization. *preprint arXiv:2305.13937* (2023).

[34] Lei Yuan, Ziqian Zhang, Lihe Li, Cong Guan, and Yang Yu. 2023. A Survey of Progress on Cooperative Multi-agent Reinforcement Learning in Open Environment. *preprint arXiv:2312.01058* (2023).

[35] Lei Yuan, Ziqian Zhang, Ke Xue, Hao Yin, Feng Chen, Cong Guan, Lihe Li, Chao Qian, and Yang Yu. 2023. Robust multi-agent coordination via evolutionary generation of auxiliary adversarial attackers. In *AAAI*. 11753–11762.

[36] Zhi-Hua Zhou. 2022. Open-environment machine learning. *National Science Review* 9, 8 (2022), nwac123.

[37] Zhi-Hua Zhou. 2022. Rehearsal: Learning from prediction to decision. *Frontiers of Computer Science* 16, 4 (2022), 164352.

[38] Zhi-Hua Zhou, Yang Yu, and Chao Qian. 2019. *Evolutionary learning: Advances in theories and algorithms*. Springer.