One by One, Continual Coordinating with Humans via Hyper-Teammate Identification

Cong Guan[1], Feng Chen[1], Ke Xue[1], Chunpeng Fan[2], Lichao Zhang[2], Ziqian Zhang[1], Pengyao Zhao[2], Zongzhang Zhang[1], Chao Qian[1], Lei Yuan[1,2], Yang Yu[1,2]*

[1] *National Key Laboratory for Novel Software Technology, Nanjing University*
School of Artificial Intelligence, Nanjing University
[2] *Polixir Technologies*

{guanc, chenf, xuek}@lamda.nju.edu.cn, {chunpeng.fan, lichao.zhang}@polixir.ai, zhangzq@lamda.nju.edu.cn, pengyao.zhao@polixir.ai, {zzzhang, qianc}@nju.edu.cn, yuanl@lamda.nju.edu.cn, yuy@nju.edu.cn

Reviewed on OpenReview: *https://openreview.net/forum?id=HVxumpoWBm*

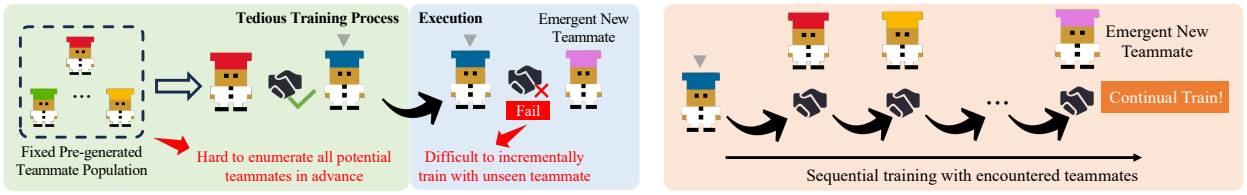
Abstract

One of the primary objectives in modern artificial intelligence researches is to empower agents to effectively coordinate with diverse teammates, particularly human teammates. Previous studies focused on training agents either with a fixed population of pre-generated teammates or through the co-evolution of distinct populations of agents and teammates. However, it is challenging to enumerate all possible teammates in advance, and it is costly, or even impractical to maintain such a sufficiently diverse population and repeatedly interact with previously encountered teammates. Additional design considerations, such as prioritized sampling, are also required to ensure efficient training. To address these challenges and obtain an efficient human-AI coordination paradigm, we propose a novel approach called **Concord**. Considering that human participants tend to occur in a sequential manner, we model the training process with different teammates as a continual learning framework, akin to how humans learn and adapt in the real world. We propose a mechanism based on hyper-teammate identification to prevent catastrophic forgetting while promoting forward knowledge transfer. Concretely, we introduce a teammate recognition module that captures the identification of corresponding teammates. Leveraging the identification, a well-coordinated AI policy can be generated via the hyper-network. The entire framework is trained in a decomposed policy gradient manner, allowing for effective credit assignment among agents. This approach enables us to train agents to cooperate with teammates one by one, ensuring that agents can coordinate effectively with concurrent teammates without forgetting previous knowledge. Our approach outperforms multiple baselines in various multi-agent benchmarks, either with generated human proxies or real human participants.

1 Introduction

Cooperative Multi-Agent Reinforcement Learning (MARL) has made significant progress in recent years, enabling multiple agents to work together towards a shared goal in diverse domains such as active voltage control (Wang et al., 2021a) and dynamic algorithm configuration (Xue et al., 2022b). However, building AI agents that can effectively coordinate with unseen teammates, especially human teammates, remains a significant challenge (Klien et al., 2004; Mutlu et al., 2013; Strouse et al., 2021; Koster et al., 2022; Yuan et al., 2023c). Previous approaches focused on building effective behavior models from human data, which

*Corresponding Author



(a) Typical scenarios of classic Human-AI coordination methods

(b) Scenario of continual coordination learning

Figure 1: This figure provides a visual comparison between our proposed setting of continual coordination learning and typical scenarios of classic Human-AI coordination methods.

is inefficient for complex problems (Kidd & Breazeal, 2008) and may raise privacy concerns (Pan et al., 2019). Recent approaches (Treutlein et al., 2021; Mirsky et al., 2022; Fosong et al., 2022) show remarkable coordination abilities in a wide range of tasks such as Overcooked (Carroll et al., 2019), but still suffer from over-fitting to their training teammates and struggle to coordinate effectively with unseen agents (Mahajan et al., 2022). Thus, the challenge of building AI agents capable of generalizing to unseen teammates in the open-world (Zhou, 2022) remains.

Training with different teammates is a promising approach to tackling the mentioned issue, involving the generation of diverse teammates and an efficient training paradigm. To achieve the former, one approach is to use hand-crafted policies (Xie et al., 2021; Papoudakis et al., 2021), special object regularizer (Derek & Isola, 2021; Lupu et al., 2021), or Population-Based Training (PBT) (Strouse et al., 2021; Xue et al., 2022a; Zhao et al., 2023). Regarding the training paradigm, a naive way is self-play (Tesauro, 1994; Silver et al., 2018), where the agent iteratively improves via playing against itself. Fictitious Co-Play (FCP) (Heinrich et al., 2015; Strouse et al., 2021) trains agents to be the best response to both the fully-trained agents and their checkpoints. These approaches have demonstrated significant progress in benchmarks such as Overcooked (Carroll et al., 2019) and Hanabi (Lupu et al., 2021), offering promising prospects for human-AI coordination and cooperation. However, the aforementioned methods pre-generate various teammates and necessitate access to all of them during training, which might be unfeasible in the real world. On the one hand, the task of enumerating all potential teammates in advance poses significant challenges. Maintaining a sufficiently diverse population has already placed a substantial demand on computing and storage resources, incurring considerable costs. Furthermore, if agents are required to cooperate with an unseen teammate, learning from scratch with all previous seen teammates (especially human participants) would be even more wasteful, considering the limitations of global time differences and economic costs. On the other hand, learning a generalized agent via interactions with a population of teammates requires meticulous design to ensure efficient training, such as prioritized sampling (Zhao et al., 2023).

To address the aforementioned challenges, we draw inspiration from the manner in which humans learn to coordinate with diverse teammates. Human participants, with whom agents are trained to cooperate, typically emerge sequentially. Instead of learning to cooperate with all agents simultaneously, humans continually adapt to new teammates while retaining knowledge from previous interactions (Hadsell et al., 2020). In accordance with this idea, we propose a new coordination learning setting and formulate it as a Multi-Agent Continual Markov Decision Process (MACMDP), where the controlled agents are trained to coordinate effectively with different teammates that appear sequentially. Fig. 1 offers a visual comparison between this novel problem setting and the conventional scenarios found in previous works. We observe that naively applying single-agent continual learning methods to MARL is inefficient, and typical MARL approaches are inadequate for continual learning scenarios (RQ1 in Sec. 4.2). Therefore, we propose a general framework called **Concord** (abbreviation for **Con**tinual **coord**ination) to enable efficient continual training of AI agents and solve the MACMDP. Concretely, we construct a teammate representation space where the teammates are represented as latent embeddings, and we introduce a recognition module that enables recognizing teammates through few-shot interaction. We then utilize the learned teammate embeddings based on a hyper-network (Ha et al., 2017; Oswald et al., 2020) to generate the policy parameters of each controlled agent. Finally, we use value decomposition to facilitate credit assignment among agents, enabling

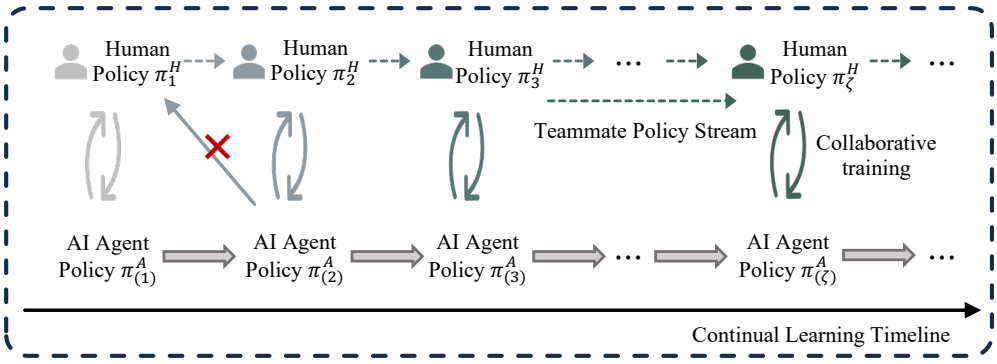


Figure 2: Workflow diagram of Multi-Agent Continual Markov Decision Process.

our framework to handle multiple controlled agents rather than only one. To evaluate the effectiveness of Concord, we conduct extensive experiments on various cooperative multi-agent benchmarks including Overcooked (Carroll et al., 2019) and StarCraft Multi-Agent Challenge benchmark (SMAC) (Samvelyan et al., 2019). We compare Concord against multiple methods in different scenarios, and the results demonstrate that Concord significantly outperforms these methods, achieving superiority across a range of evaluation metrics. Our main contributions are:

- To the best of our knowledge, this is the first time that human-AI coordination challenge is explored in a continual manner via our formulated MACMDP.
- We propose a continual MARL paradigm based on hyper-teammate identification, which is capable of effectively coordinating with diverse teammates and striking a balance between forward transferring and avoiding forgetting.
- Empirical studies on various benchmarks demonstrate the effectiveness of Concord in continuously coordinating with diverse teammates, including real human teammates.

2 Problem Formalization

The purpose of human-AI coordination is for AI agents to cooperate well with diverse human teammates. Previous works rely on a fixed policy population, involving extensive training of AI agents, but this approach is computationally expensive and does not align with real-world scenarios where humans are encountered sequentially. Currently, there is limited research on training AI agents to work well with humans in the continual setting. To address this gap, we introduce a problem formulation for continual multi-agent coordination learning, so that agents can learn to cooperate with unseen human teammates continually.

We formalize the continual multi-agent coordination learning problem as a Multi-Agent Continual Markov Decision Process (MACMDP) consisting of a tuple $\mathcal{M} := \{I, S, A, P, R, \Theta, \mu, \gamma, T\}$, where I is the set of AI agents, S and A are the state and action space, respectively. P is the transition function, R is the reward function, $\gamma \in [0, 1)$ is the discount factor and T is the continual learning length. Besides, Θ is the teammate type space, and μ is an indicator function which means that $\mu(\zeta) \in \Theta(1 \leq \zeta \leq T)$ denotes the teammate policy encountered at the ζ-th stage. For the sake of brevity, we denote the policy determined by μ in the ζ-th stage as π_ζ^H.

These teammates are encountered sequentially, as depicted in Fig. 2. In human-AI coordination tasks, cooperating with different humans can be defined as different tasks, and each human teammate policy π_ζ^H can be represented by a task indicator ζ. In MACMDP, AI agents can not interact with the previous human teammates, which corresponds to the assumption that the agent can not interact with previous environments in the common continual reinforcement learning setting (Khetarpal et al., 2022), but are expected to remember how to cooperate with all previous human teammates. When handling the task ζ,

the AI policy π^A and human policy π_ζ^H select actions a^A, a^H according to the current state s, respectively. The next environment state s' is obtained according to the environment transition function $P(s'|s, a^A, a^H)$ and a global reward is returned by the global reward function $R(s, a^A, a^H)$. We use the discounted return $J_\zeta(\pi^A, \pi_\zeta^H) = \mathbb{E}_{s,a^A,a^H}\left[\sum_t \gamma^t R(s, a^A, a^H)\right]$ as the objective on task ζ. A special note is that the AI agents may need to collaborate with multiple human teammates currently, not just one. Thus, the human policy π_ζ^H here can refer to several teammates, and we actually conduct continual coordination learning with human teammates one (team) by one (team). Here, the term "team" may consist of one single teammate like in Overcooked or multiple teammates in scenarios of more players.

A workflow diagram illustrating MACMDP is provided in Fig. 2 for better understanding. As depicted in this diagram, the AI agents are expected to be collaboratively trained with the teammate policies one by one, and interacting with the previous teammates is prohibited. Under this problem formalization, our goal is to find the AI agent policy π^A that can maximize the expected return with current human teammates after continual process, without forgetting the previous cooperation patterns, which can be formally represented as $\max_{\pi^A} \sum_{\zeta=1}^{T} J_\zeta(\pi^A, \pi_\zeta^H)$.

3 Method

In this section, we introduce the proposed Concord framework. We first introduce the teammate-adaptive AI framework in Sec. 3.1, which includes three main components, i.e., human-aware actor critic, teammate representation and recognition, and anti-forgetting mechanism. Then, we give the overview of our algorithm in Sec. 3.2, which includes the pseudo-code of the two phases and a brief diagram illustration.

3.1 Teammate-adaptive AI Framework

AI agents are anticipated to identify different teammates and adapt accordingly, enabling them to acquire the ability to coordinate continually. Additionally, the agents should be able to remember previous teammates while learning to coordinate with new ones. We here propose a comprehensive pipeline (depicted in Fig. 3 and App. A) that involves designing a human-aware actor-critic structure for policies training, building teammate representations for teammates capturing, and implementing an anti-forgetting mechanism to address the forgetting problem during continual learning.

Human-Aware Actor-Critic In a human-AI coordination scenario, the decision-making process of the overall multi-agent system is jointly determined by both the AI agents and the humans. To enable AI agents to identify different teammates, we introduce a teammate-adaptive structure, an extension and variant of hyper-network (Ha et al., 2017; Oswald et al., 2020). This structure is specifically designed for continual human-AI coordination tasks and can generate actor networks that can effectively coordinate with specific teammates by leveraging consistent teammate representations. Concretely, the parameters of the AI actor networks are obtained through $\theta_i^{A_\zeta} = f_\psi^{\text{hyper}}(z_\zeta)$, where f_ψ^{hyper} represents the hyper-network parameterized by ψ. In essence, we incorporate the ability to adapt to different teammates into the hyper-network, so that it can generate appropriate actor networks for cooperating with the human teammates based on the input of different teammate representations z_ζ. More details are reported in App. A.1.

To better assess the contribution of AI to the overall performance, we here design a linearly decomposed centralized critic like DOP (Wang et al., 2021b). Therefore, we can create individual critics for the uncontrollable human agents, and model the global Q function by combining the individual Q values of both human and AI agents. This approach enables the global Q function to more accurately reflect the joint decision-making process of human-AI coordination, and facilitate credit assignment between AI and humans, allowing us to isolate the contribution of the AI agents and provide more precise learning signals for updating π^A. The critic and actor networks are updated through minimizing the critic loss $\mathcal{L}^{\text{critic}} = \mathbb{E}_{\boldsymbol{\tau},a^A,a^H}\left[\left(Q_{\text{tot}}^\phi(\boldsymbol{\tau}, a^A, a^H) - y^{\boldsymbol{\tau}}\right)^2\right]$ and actor loss $\mathcal{L}^{\text{actor}} = \mathbb{E}_{\pi^A}\left[\sum_i k_i(\boldsymbol{\tau}) \log \pi_i^A\left(a_i^A|\tau_i^A; \theta_i^{A_\zeta}\right) Q_i^{\phi_i^A}(\boldsymbol{\tau}, a_i^A)\right]$, respectively, where $k_i(\boldsymbol{\tau})$ represents the coefficients of value decomposition generated by an additional hyper-network when inputting the historical information $\boldsymbol{\tau}$, the global value $Q_{\text{tot}}^\phi(\boldsymbol{\tau}, a^A, a^H)$ is obtained through

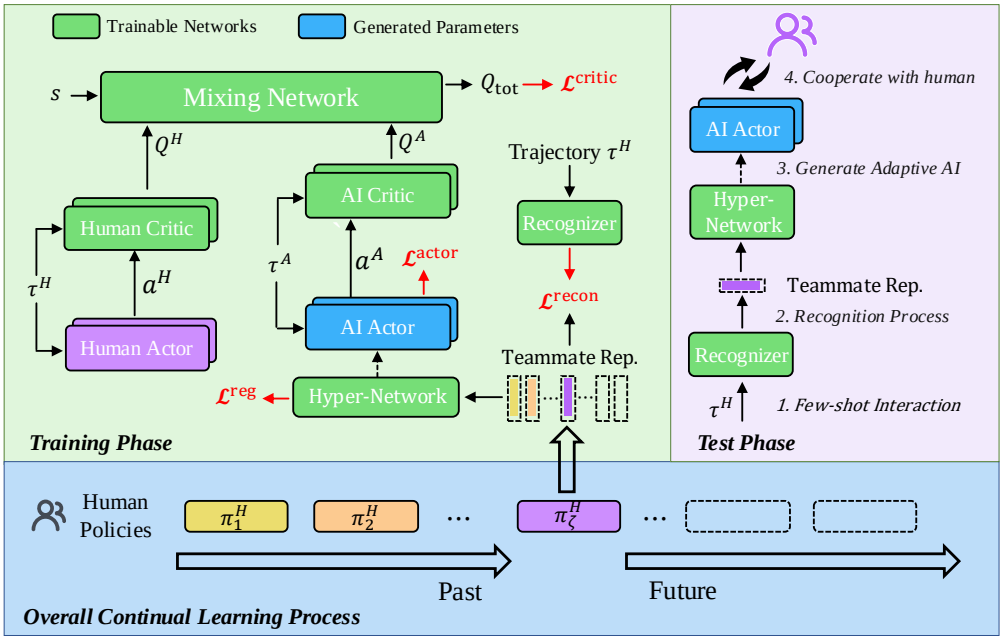


Figure 3: The overall framework of **Concord**. During the training phase, a human-aware actor-critic is trained for AI to coordinate with humans effectively, where the AI actors are generated via a hyper-network. Teammate representations are trained via backpropagation, while generated by the recognizer in the test phase. When testing, the AI first interacts with human to collect few trajectories, then obtains the corresponding teammates representation for decision-making.

$Q^{\phi}_{\text{tot}}(\boldsymbol{\tau}, a^A, a^H) = \sum_i k_i(\boldsymbol{\tau}) Q_i^{\phi_i^A}(\boldsymbol{\tau}, a_i^A) + \sum_j k_j(\boldsymbol{\tau}) Q_j^{\phi_j^H}(\boldsymbol{\tau}, a_j^H) + b(\boldsymbol{\tau})$, where $k_i(\boldsymbol{\tau})$ and $b(\boldsymbol{\tau})$ are generated by learnable networks whose inputs are global state, and $k_i(\boldsymbol{\tau})$ is restricted to be non-negative to ensure monotonicity, and $y^{\boldsymbol{\tau}}$ denotes the target value $y^{\boldsymbol{\tau}} = r + \gamma \mathbb{E}_{a_i'^A, a_j'^H}\left[Q^{\phi'}_{\text{tot}}(\boldsymbol{\tau}', a_i'^A, a_j'^H)\right]$. Note that we refer to the parameters of the critic network as ϕ, and those of the target network as ϕ'.

Teammate Representation Learning To enable our method to adapt to different teammates, as mentioned above, we use one hyper-network that can generate adaptive actor networks by inputting different teammate representations. Here, the teammate representations serve the crucial role of recognizing the behavior patterns of different human teammates. To adopt this teammate representation scheme, we have two necessary prerequisites: 1) the teammate representations can help the hyper-network generate AI actor that can cooperates well with human; 2) the teammate representations should be obtainable with a minimal amount of interaction with humans during the test phase.

To fulfill the first prerequisite, we design an end-to-end update scheme for teammate representations. In particular, we maintain a separate teammate representation for each human team encountered during the continual learning process This teammate representation z_ζ is then fed into the hyper-network to obtain the parameters of the AI actor network θ^{A_ζ} which is expected to coordinate effectively with π_ζ^H. Thus, we directly update the hyper-network and teammate representations with the gradients back-propagated from optimizing the actor loss $\mathcal{L}^{\text{actor}}$. However, we hold an assumption that the identity of the teammates is unknown during testing. Thus, although we have learned representations for each team, we do not know which one to utilize during testing, which leads to the second prerequisite above.

In terms of the second prerequisite, we additionally design a recognizer network that can reconstruct learned teammate representations from limited interaction data. This recognizer network is typically implemented as a trajectory-encoder that takes the trajectories of human teammates as input and outputs the prediction of the corresponding teammate representations. For training the recognizer network, we encode trajectories

of varying lengths and minimize one reconstruction loss denoted as $L_\zeta^{\text{recon}} = \sum_{t=0}^{T_\tau - 1} \lambda^{T_\tau - 1 - t} \|\hat{z}_\zeta^t - z_\zeta\|_2^2$. This loss function encourages the encoder to effectively reconstruct teammate representations using trajectories of different lengths. During the test phase, we roll out some trajectories (typically 2 in our experiments) with the teammate. Each entire trajectory undergoes encoding by the encoder, yielding a reconstructed representation for each trajectory. Then, they are averaged to yield the final recognition outcome, thereby achieving few-shot recognition of human teammates.

Anti-forgetting Mechanism One of the key challenges in continual multi-agent coordination learning is to avoid catastrophic forgetting of previous human teammates as the AI policy learns and adapts to new ones. Specifically, within our framework, we aim to combat the forgetting problem in both the recognizer network f_η^{enc} and hyper-network f_ψ^{hyper} simultaneously, which correspond to the ability to recognize and coordinate with the previous human teammates, respectively. To tackle this challenge, we employ an anti-forgetting mechanism to enforce regularization on both f_η^{enc} and f_ψ^{hyper} networks during the continual learning process. For the recognizer network, we utilize a small replay buffer to store the teammate representations and a few of interaction trajectories of the previous human teammates. During cooperation learning with the ζ-th human team, the actual loss of the recognizer network is $\mathcal{L}^{\text{recon}} = \sum_{i=1}^{\zeta} \mathcal{L}_i^{\text{recon}}$. Similarly, when training the hyper-network at the ζ-th cooperation task, we additionally include one regularization loss $\mathcal{L}^{\text{reg}} = \frac{1}{\zeta-1} \sum_{i=1}^{\zeta-1} \|f_\psi^{\text{hyper}}(z_i) - f_{\tilde{\psi}}^{\text{hyper}}(z_i)\|_2^2$ in addition to the actor loss to avoid f_ψ^{hyper}'s forgetting the previous cooperation knowledge, where $f_{\tilde{\psi}}^{\text{hyper}}$ denotes the hyper-network obtained after the $(\zeta-1)$-th task. In summary, we propose an anti-forgetting mechanism that effectively safeguards against the forgetting of past knowledge in both the hyper-network and the recognizer network through the preservation of the updated hyper-network from the previous round and a small memory pool containing few-shot interactions and teammate representations of previous human teammates.

3.2 Overall Algorithm

Finally, we provide the overall description of the procedure of our approach under a continual coordination learning setting. The main parts of our approach Concord can be decomposed into the training phase and the test phase. The pseudo-codes for these two phases are respectively shown in Alg. 1 and Alg. 2. A brief diagram illustrating the main idea of our approach can also be found in App. A.2.

During the training phase, we conduct cooperation learning with human teammates one (team) by one (team) from π_1^H to π_T^H, where we learn the hyper-network and the recognizer network together. As detailed from lines 5 to 13 of Alg. 1, during the collaborative training with the ζ-th human team, we firstly optimize the human-aware actor-critic architecture by minimizing the $\mathcal{L}^{\text{critic}}$ and $\mathcal{L}^{\text{actor}}$ losses, where the teammate representation z_ζ is concurrently updated in an end-to-end manner with the back-propagated gradient $\nabla_{z_\zeta} \mathcal{L}^{\text{actor}}$. While for the hyper-network f_ψ^{hyper}, we update it with the actor loss $\mathcal{L}^{\text{actor}}$ plus one regularization loss $\mathcal{L}^{\text{reg}}$, which means that the actual loss function for the hyper-network is

$$\mathcal{L}^{\text{hyper}} = \mathcal{L}^{\text{actor}} + \beta^{\text{reg}} \mathcal{L}^{\text{reg}} = \mathcal{L}^{\text{actor}} + \frac{\beta^{\text{reg}}}{\zeta - 1} \sum_{i=1}^{\zeta-1} \|f_\psi^{\text{hyper}}(z_i) - f_{\tilde{\psi}}^{\text{hyper}}(z_i)\|_2^2, \tag{1}$$

where β^{reg} is the coefficient balancing these two loss terms. We optimize the critic network, hyper-network and teammate representation in lines 11, 12 and 13 of Alg. 1, respectively. Following the training process of hyper-network and teammate representation, in line 15 we update $\tilde{\psi}$ with the latest hyper-network, and in line 17 we retain a certain amount of trajectories to buffer $\mathcal{B}$. Subsequently, from lines 18 to 21, we further optimize the recognizer network via reconstructing the representations of all previously encountered human teammates, which means that we update the f_η^{enc} via minimizing the reconstruction loss $\sum_{i=1}^{\zeta} \mathcal{L}_i^{\text{recon}}$, where $\mathcal{L}_i^{\text{recon}}$ indicates the reconstruction loss for the teammate representation of the i-th human team. After completing these steps, we proceed to the collaborative training with the next human team. In brief, the collaborative training with each group of human teammates is decomposed of two separate stages, where we train the hyper-network and teammate representations at the first stage and update the recognizer network at the second stage.

Algorithm 1 Training phase

Require: Parameters η, ψ, ϕ, $\{z_\zeta\}_{\zeta=1}^{T}$, Small buffer $\mathcal{B}$
Explanation:
f_η^{enc}: Recognizer network parameterized by η
$\{\pi_\zeta^H\}_{\zeta=1}^{T}$: Teammate policy stream
τ_ζ^H: Few interaction data obtained by interacting with π_ζ^H
z_ζ: Learned teammate representation for teammate policy π_ζ^H
f_ψ^{hyper}: Hyper-network parameterized by ψ
$\mathcal{B}$: small buffer that stores a few of human trajectories and teammate representations for all previous seen teammates.
Ensure: Update the parameters $\eta, \psi, \phi, \{z_\zeta\}_{\zeta=1}^{T}$
1: Set $\tilde{\psi}$ as None
2: **for** π_ζ^H in the teammate policy stream **do**
3: // Training critic network, hyper-network and teammate representation
4: **for** episode in RL training period **do**
5: Compute the critic loss $\mathcal{L}^{\text{critic}}$, actor loss $\mathcal{L}^{\text{actor}}$
6: **if** $\tilde{\psi}$ is None **then**
7: Let $\mathcal{L}^{\text{hyper}} := \mathcal{L}^{\text{actor}}$
8: **else**
9: Let $\mathcal{L}^{\text{hyper}} := \mathcal{L}^{\text{actor}} + \beta^{\text{reg}}\mathcal{L}^{\text{reg}}$
10: **end if**
11: Update ϕ via gradient descend method with gradient $\nabla_\phi \mathcal{L}^{\text{critic}}$
12: Update ψ via gradient descend method with gradient $\nabla_\psi \mathcal{L}^{\text{hyper}}$
13: Update z_ζ via gradient descend method with gradient $\nabla_{z_\zeta} \mathcal{L}^{\text{actor}}$
14: **end for**
15: Let $\tilde{\psi} := \psi$ // Store the hyper-network parameters at the last round
16: // Training recognizer network
17: Select multiple human trajectories τ_ζ^H from the RL training process, and store each (τ_ζ^H, z_ζ) into the buffer $\mathcal{B}$.
18: **for** iteration in Recognizer training period **do**
19: Compute $\mathcal{L}^{\text{recon}}$ via sampling data from $\mathcal{B}$
20: Update η via gradient descend method with gradient $\nabla_\eta \mathcal{L}^{\text{recon}}$
21: **end for**
22: **end for**

When it comes to the test phase, in line 1 of Alg. 2, we first allow a few interaction with the humans. The, as detailed in lines 2 and 3 respectively, the sampled human trajectories are fed into the recognizer network f_η^{enc} to help attain the predicted teammate representation, which is then inputted into the hyper-network f_ψ^{hyper} to obtain the actor network for the AI policy. The final actor network here is employed to coordinate with the humans.

4 Experiments

In this section, we first conduct experiments on the widely-used Overcooked (Carroll et al., 2019) to verify whether Concord can effectively coordinate with human teammates under the continual setting. Then, we evaluate Concord's ability to coordinate with multiple teammates through experiments on SMAC (Samvelyan et al., 2019). The human teammates involved in these experiments consist of both human-like models and real human participants.

To comprehensively analyze the performance of various algorithms in the context of continual human-AI coordination, we introduce the Single Coordination Performance $P_i(t)$, where $t \in \{1, 2, ...T\}$ and $P_i(t)$ represents the episode reward when coordinating with the i-th human team after completing training with

Algorithm 2 Test phase

Require: Parameters η, ψ

Explanation:

f_η^{enc}: Recognizer network parameterized by η

π_ζ^H: Collaborative policy of the teammates

τ_ζ^H: Few interaction data obtained by interacting with the teammates

$\hat{z}_\zeta$: Predicted teammate representation obtained by inputting τ_ζ^H to f_η^{enc}

f_ψ^{hyper}: Hyper-network parameterized by ψ

AI_Actor: AI actor network generated by inputting $\hat{z}_\zeta$ to f_ψ^{hyper}

Ensure: Deployed AI policy in coordination with teammates

1: Interact with teammates using π_ζ^H to obtain τ_ζ^H

2: $\hat{z}_\zeta \leftarrow f_\eta^{\text{enc}}(\tau_\zeta^H)$

3: AI_Actor $\leftarrow f_\psi^{\text{hyper}}(\hat{z}_\zeta)$

4: **return** Deploy AI_Actor as AI policy in coordination with the teammates

the t-th team. For a principled assessment of the continual learning ability, we introduce followings metrics inspired by the concepts applied in (Wołczyk et al., 2021; Lopez-Paz & Ranzato, 2017):

- **Average coordination performance.** $P = \frac{1}{N}\sum_{i=1}^{N} P_i(T)$, which is the average episode reward when coordinating with all the human teams after the training process.
- **Forgetting.** $\frac{1}{N-1}\sum_{i=1}^{N-1} P_i(T) - P_i(i)$, which is used to measure the average performance loss with seen human teams (i.e., forget) after training with the new team.
- **Forward transfer.** $\frac{1}{N-1}\sum_{i=2}^{N} P_i(i-1)$, which is used to measure the coordination ability with unseen human teams.

We compare the proposed Concord with the multiple baselines. All experimental results are averaged over five random seeds, accompanied by standard deviation. More details including network architecture and hyper-parameters are provided in App. A.1 and B.1.

4.1 Baselines and Environments

To completely evaluate the performance of Concord, we introduce the following baselines for comparison. First of all, we adapt EWC (Kirkpatrick et al., 2017) and CLEAR (Rolnick et al., 2018), which are two classic continual learning algorithms, to our experiment setting. Then, we additionally design two other baselines: Vanilla and Oracle. Vanilla does not do any extra design for the continual learning process, while Oracle is always allowed for collaborative training with all human teammates. Here, to address the potential unfamiliarity of some readers with the field of continual learning, we have provided additional descriptions for the EWC and CLEAR algorithms to enhance the readers' understanding of the specific processes of EWC and CLEAR, as well as how we adapt them in our context.

EWC Elastic Weight Consolidation (EWC) is a regularization-based technique that selectively reduces the plasticity of weights to alleviate catastrophic forgetting. The technique Fisher information matrix evaluates the importance of all weights and uses that for gradient updates. There are two variants of EWC: Offline EWC keeps a model with a Fisher matrix for every previous task, which causes costly storage. Online EWC uses only one Fisher matrix that is computed based on all the previous tasks. Here, we choose Online EWC as our benchmark.

CLEAR When training, CLEAR samples a mini-batch from the current task and another replay mini-batch. The samples in the replay mini-batch are selected with the same probability as all previous tasks. CLEAR leverages behavioral cloning from replay mini-batch to enhance stability. Specifically, two additional

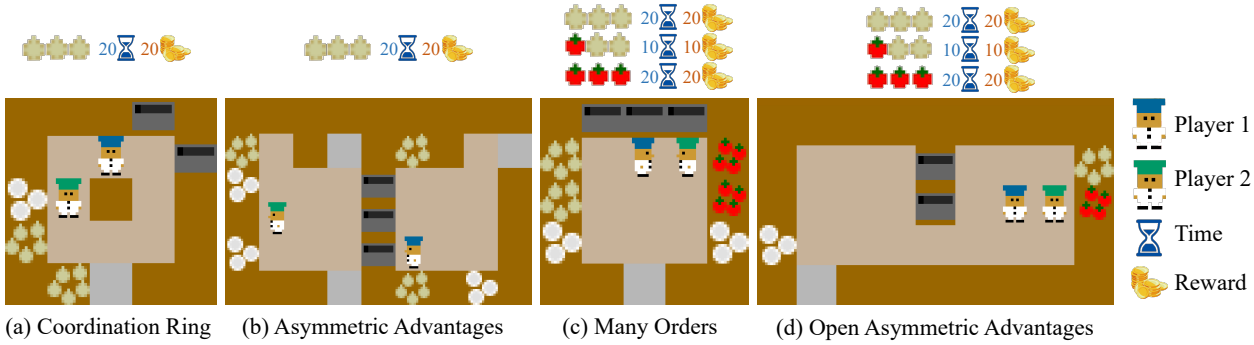


Figure 4: Four layouts of the Overcooked environment, with each one emphasizing different coordination strategies.

loss terms are added to induce behavioral cloning between the network and its past self. (1) the KL divergence between the historical policy distribution and the present policy distribution, (2) the ℓ_2 norm of the difference between the historical and present value functions. Unlike the original paper, we use Cross Entropy loss instead of KL loss to guide the agent to choose the same action as its past self under the same state because we find it more effective. Additionally, we don't add any constraints to criticism for easier learning.

Oracle Here, we would like to explain why the multi-task learning approach can be seen as Oracle. 1) Multi-task learning (Oracle) can access all the previous models in the training process. However, Concord (and other continual learning methods) can only be trained with the previous teammates once in the process. 2) Multi-task learning (Oracle) knows which human team it is coordinating with in the test phase. However, Concord (and other continual learning methods) has to identify the teammates by some specific mechanisms.

In the preceding discussion, we introduced the baseline algorithms compared in our study. In the following part, we will proceed to discuss the environments utilized in our experiments. Specifically, our experiments are conducted in two widely recognized multi-agent cooperation environments: Overcooked (Carroll et al., 2019) and SMAC (Samvelyan et al., 2019).

Overcooked Overcooked is an environment proposed for coordination challenges. In this environment, two players control one kitchen chef to cook and serve soup. Specifically, they are required to put specific ingredients in one pot, make soup and deliver it, for example, in Coord. Ring layout, the players need to put three onions in a pot, collect an onion soup from the pot after 20-time steps, and deliver the dish to a counter. The agents will receive 20 points for each dish served, and the goal is to serve as many dishes as possible within the allotted time. Moreover, we select 4 different layouts in our experiments, which are shown in Fig. 4.

StarCraft Multi-Agent Challenge (SMAC) SMAC is a popular benchmark for cooperative multi-agent reinforcement learning, which contains multiple combat scenarios. In this environment, the AI agents and human teammates control specific ally units and are expected to coordinate to defeat the enemy units controlled by the built-in AI. Specifically, at the beginning of each episode, the ally units are spawned on the fixed regions on the map. At each timestep, the ally agents can select actions from the action set `{no-op, move[right, left, up, down], stop, attack[enemy_id]}`, and then a global reward will be given according to the damage caused to enemy units. Extra rewards of 10 and 200 will be given if one enemy unit is killed or the ally units win the combat. We conduct experiments on three SMAC maps in this paper as shown in Fig. 5, which are 3m, 2z1s, and 4m, respectively. On all these three maps, there exist the same number and type of units on both ally and enemy sides, where 3m and 4m signify the presence of 3 marines and 4 marines, respectively, and 2z1s indicates 2 zealots and 1 stalkers.

(a) 3m (b) 2z1s (c) 4m

Figure 5: Three SMAC maps that are selected in our experiments.

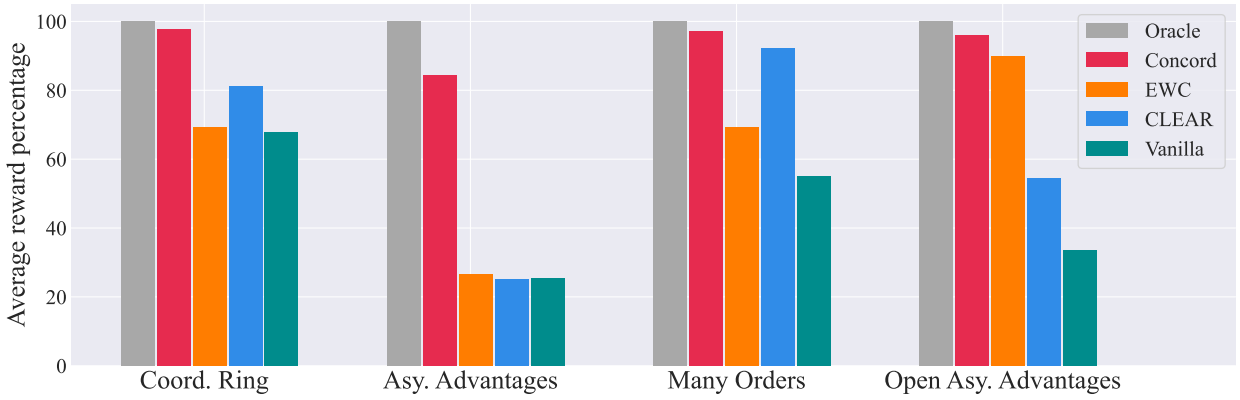


Figure 6: The overall performance of distinct algorithms, where Concord achieves comparable coordination ability with Oracle and outperforms other baselines.

4.2 Continual Coordination with Human-like teammates

First, we evaluate continual coordination ability with human-like teammates on Overcooked (Carroll et al., 2019), where different layouts present a unique challenge that can be overcome through effective coordination. The players are required to put three onions in a pot, collect an onion soup from the pot after 20 time steps and deliver the dish to a counter. The agents will receive 20 points for each dish served, and the goal is to serve as many dishes as possible within the allotted time. We run each algorithm for 5 random seeds and each seed was tested 32 times. The final results were obtained after averaging these results. We investigate the following research questions (RQs).

RQ1: How does Concord perform compared to other methods? As can be seen from Fig. 6, Vanilla achieves the most inferior coordination ability in almost all scenarios, indicating a specific consideration for the tasks where teammates appear sequentially. Other successful approaches for single agent continual learning, like EWC, and CLEAR, also suffer from performance degradation in the involved benchmarks, demonstrating the necessity of specific design for MARL. The Oracle method, where we train all the tasks simultaneously, can be seen as an upper bound of performance on the related benchmarks, acquiring superiority over all baselines. Our approach Concord, obtains comparable performance to Oracle, indicating the efficiency of all the designed modules. Besides, according to Wilcoxon rank-sum test with significance level 0.05, Concord is significantly better than EWC, CLEAR and Vanilla, except for CLEAR on Many Orders and EWC on Open Asy. Advantages, which further strengthen our results.

Furthermore, we also show the comparisons of different methods on forgetting and forward transfer in Tab. 1. Concord achieves the minimum forgetting on all the four layouts, validating the effectiveness of the proposed anti-forgetting mechanism. Besides, the forward transfer values of Concord on all the layouts are best,

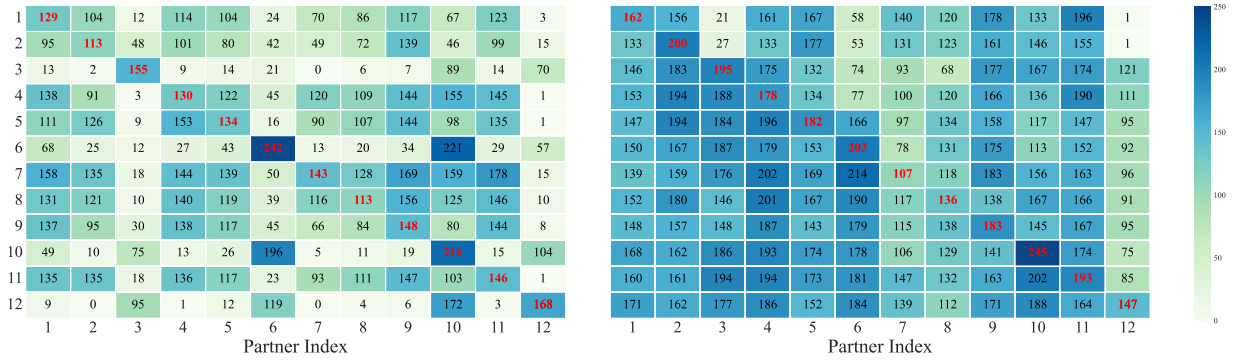


Figure 7: Single coordination performance of Vanilla (left) and Concord (right) on the Asym. Adv. layout. The value at i-th row and j-th in the matrix is $P_i(j)$.

Table 1: Comparisons of distinct algorithms in all 4 layouts on forgetting and forward transfer metrics. The best and second best values on each layout are colored in **red** and **blue**, respectively. The symbols '+', '−' and '≈' indicate that the result is significantly superior to, inferior to, and almost equivalent to Concord, respectively, according to the Wilcoxon rank-sum test with significance level 0.05.

Algorithm	Coord. Ring		Asy. Advantages		Many Orders		Open Asy. Advantages	
	Forg.	F. Transfer	Forg.	F. Transfer	Forg.	F. Transfer	Forg.	F. Transfer
Concord	**-1.31 ± 4.23**	**103.09 ± 13.02**	**-16.11 ± 6.85**	**119.59 ± 9.41**	**-1.27 ± 6.69**	**100.85 ± 3.14**	**-6.83 ± 4.59**	**162.29 ± 6.94**
EWC	-16.47 ± 4.18 −	**92.03 ± 8.61** −	**-54.14 ± 13.95** −	54.98 ± 8.01 −	-20.52 ± 1.41 −	82.85 ± 18.41 −	**-7.69 ± 3.38** ≈	**153.85 ± 3.68** ≈
CLEAR	**-10.79 ± 2.62** −	89.51 ± 2.46 −	-120.61 ± 2.52 −	**72.65 ± 1.73** −	**-2.33 ± 3.85** −	**99.58 ± 5.91** ≈	-63.14 ± 55.75 −	117.14 ± 14.92 −
Vanilla	-46.92 ± 11.57 −	80.94 ± 9.19 −	-109.83 ± 20.48 −	60.82 ± 7.33 −	-68.36 ± 10.14 −	67.61 ± 4.16 −	-123.89 ± 17.91 −	109.94 ± 32.31 −

indicating that Concord can coordinate well with unseen teammates. Vanilla is worst, demonstrating the necessity of mechanism for continual learning. Specifically, we compare the single coordination performance of Concord and Vanilla on the Asym. Adv. layout of Overcooked environment in Fig. 7. As expected, Vanilla exhibits a considerable amount of forgetting, as indicated by the lower triangle of the matrix, and has poor forward transfer ability, as reflected by the upper triangle of the matrix. While Concord performs significantly better than Vanilla. More results on other layouts can be found in Fig. 14-16 of App. C.1.

RQ2: How does the teammate representation mechanism work? As an important aspect of Concord, the learning of teammate representations helps AI agents prevent catastrophic forgetting and effectively adapt to different human teammates. The human recognizer encodes a representation embedding for each human team based on their behavior patterns. Using this embedding, an appropriate actor network is generated and utilized to effectively cooperate with that human team. We demonstrate the differences in behavior patterns among distinct human-like models and the corresponding relationship between embedding and behavior pattern in Fig. 8. Two main observations can be made: 1) The behavior patterns of human-like models are diverse and can be characterized through four key steps. Visualization results indicate that these models exhibit varying behavior patterns. The output representations from the recognizer network also differ significantly, forming distinct clusters. 2) Similar behavior patterns have similar embedding representations, demonstrating the effectiveness of our representation mechanism in identifying teammates' behavior patterns. For instance, the behavior patterns of models 3 and 4 are identical, and their embedding representations are similarly aligned. This pattern of similarity also holds for models 6 and 11, as well as 8 and 12. Additionally, models 5, 6, 8, 11, and 12, which all utilize the right pot for cooking, show embeddings that are clustered on the same side of a dividing line, in contrast to other models.

To further investigate the impact of teammate representations on AI agent behavior, we also present the behavior patterns displayed by AI actors generated by the hyper-network in response to different teammate representations. The findings indicate that the AI agent can adapt its behavior to different teammates,

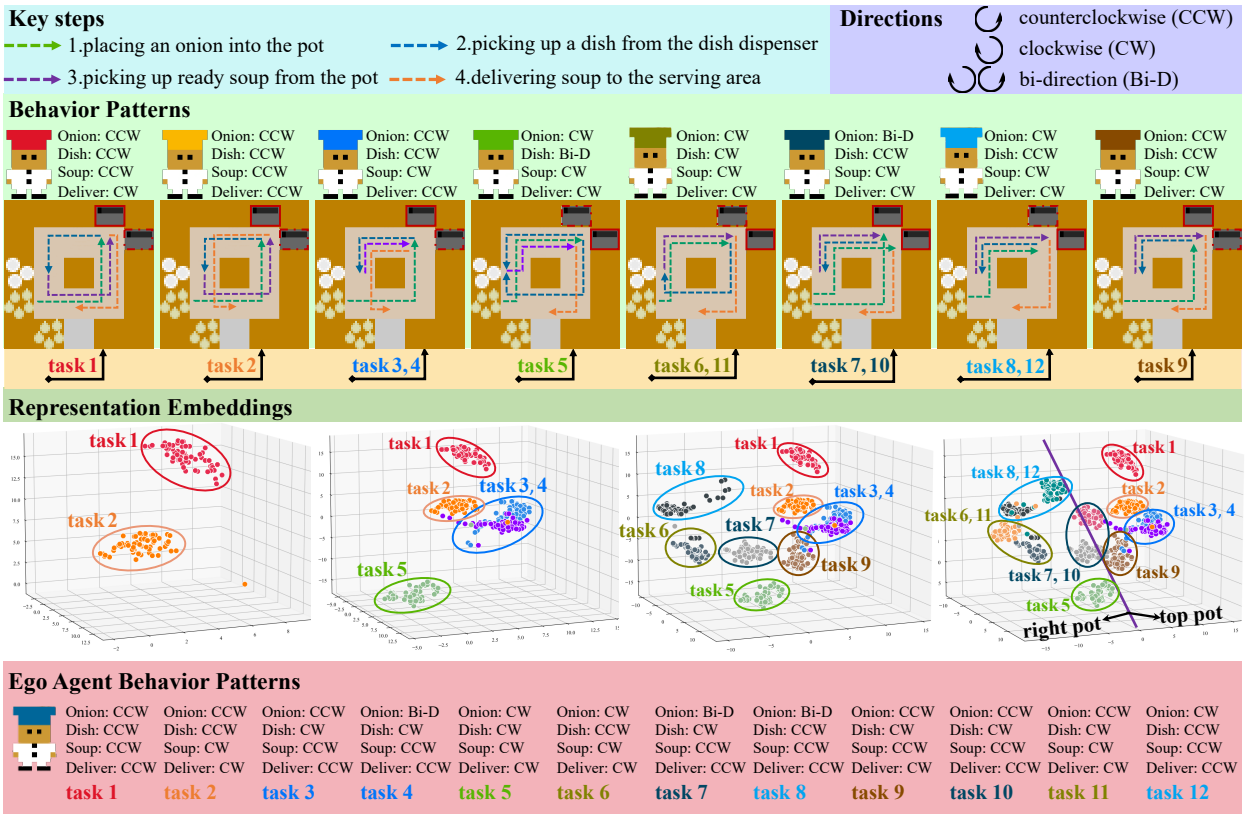
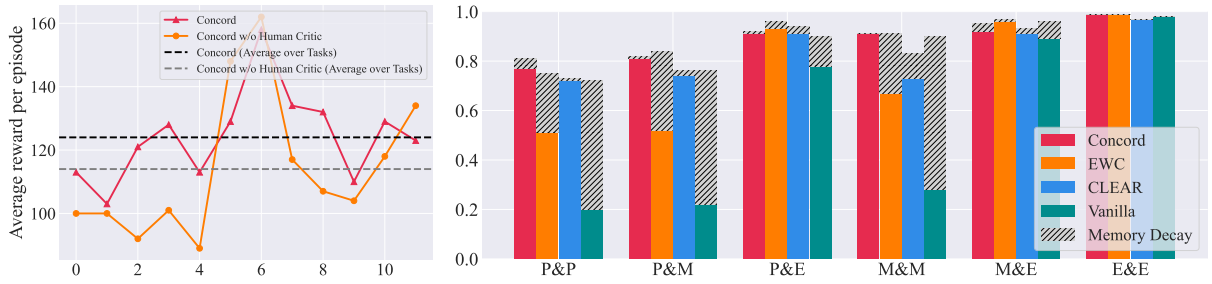


Figure 8: The t-SNE (Van der Maaten & Hinton, 2008) projection of different behavior patterns. Completing a delivery task requires several *key steps*, including placing an onion into the pot, taking a dish, taking a ready soup, and delivering the soup, and each step has three *directions* for human-like models: always clockwise (CW), always counterclockwise (CCW), and bi-direction (Bi-D). Then, the behavior pattern of a teammate can be defined as the directions used in the key steps.



(a) Study on value decomposition

(b) Experimental results on SMAC

Figure 9: (a) Performance comparison between Concord with and without value decomposition. (b) Experimental results on the map 4m. The gray shaded bar is used to denote the degree of forgetting, which equals to the highest test win rate during training minus the final test win rate. The x-axis represents the combination of two teammates, e.g., P&P means both teammates are poor (P) models.

demonstrating varied targeted actions accordingly. Notably, the agent exhibits similar behaviors when interacting with teammates who have comparable behavior patterns. For instance, when comparing the behaviors with the 3-rd and 4-th teammates, only minor differences are observed in how the AI agent places the onion into the pot.

RQ3: Is the value decomposition useful? One component of our approach design is the human-aware actor-critic architecture, which decomposes the global Q_{tot} into the individual Q values of teammates and AI agents. We here design an ablation that drops the value decomposition practice in Coord. Ring layout of Overcooked environment, which means treating the teammates as part of the environment and only learning the Q value of the AI agent. This reduces to applying single agent reinforcement learning algorithm like previous works (Heinrich et al., 2015; Strouse et al., 2021).

As shown in Fig. 9(a), where we report the testing performance on all 12 tasks after the whole continual training, the red curve consistently surpasses the orange one. This indicates that the complete Concord algorithm generally shows better anti-forgetting performance over the baseline w/o value decomposition, demonstrating the effectiveness of the human-aware actor-critic architecture to facilitate collaborating learning.

Other Ablation Studies and Further Analysis Due to space limitations, further analysis, such as the number of episodes for recognition, comparison between few-shot and zero-shot recognition, comparison with given teammate ID, ablation studies and sensitivity analysis of the recognition mechanism, as well as sensitivity analysis of the agent network can be found in Appendix C.2. Furthermore, we also include the discussions with ZSC methods, discussions with meta-RL methods, experiments with more continual teams, as well as experiments conducted on the SMAC environment in the rest of Appendix C.

4.3 Continual Coordination between Multiple Human-like teammates and Multiple AI Agents

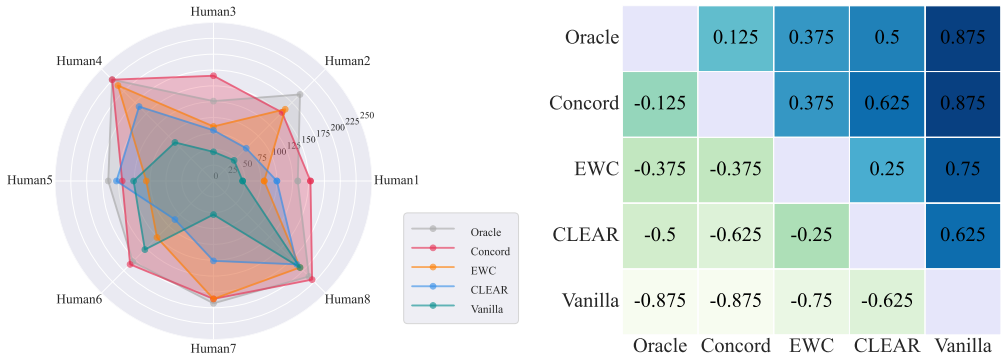
In the previous experiments, there was only one agent and one teammate in the environment. Multi-player human-AI coordination, i.e., the coordination between multiple agents and multiple teammates, is an understudied yet important setting in practical applications. In this section, to test the generalization ability of Concord, we use three maps in the SMAC environment (Samvelyan et al., 2019) to assess the coordination ability between multiple human-like teammates and multiple AI agents. We use DOP (Wang et al., 2021b) to train the human-like teammates. Three checkpoints with low, medium, high win rates during the training process were selected to simulate poor (P), medium (M) and expert (E) human players, respectively. Then, we train the agent via different algorithms to continually coordinate with single human-like teammate (maps 3m and 2s1z) or multiple human-like teammates (map 4m). Fig. 9(b) presents the results on map 4m, from which we can see that the gray shaded area of Concord is smallest, indicating that Concord generally has better anti-forgetting performance over other baselines. Results on maps 3m and 2s1z are reported in App. C.6.

4.4 Continual Coordination with Humans

Finally, we conduct experiments on the Open Asy. Advantages layout to investigate the ability of coordinating with real humans. We first collect around 160 human-human trajectories (for a total of 64k environment timesteps) and partition them into two subsets, splitting each trajectory into two single-agent trajectories. Based on the collected human data, we construct 8 sets of human proxy models through behavior cloning. Then we train different methods by using these human proxy models as the teammates that appear sequentially. Note that the AI agents obtained by different methods will play with the real human participants in the test phase, not the human proxy models. More details about the construction of human proxy models and ethical statement concerning this experiment are provided in App. D.1 and D.2, respectively.

Main results. We show the single coordination performance after training in Fig. 10(a). Among these algorithms, Concord is the closest to oracle in terms of performance, and outperforms CLEAR and EWC baselines with most teammates, which is consistent with the results in Section 4.2. The Vanilla method performs the worst and shows catastrophic forgetting, which again illustrates the importance of the anti-forgetting in continual human-AI coordination.

Human preference. To check the preference of AI agents obtained by different algorithms of human players, we additionally introduce a popular subjective metric, i.e., human preference (Strouse et al., 2021; Yu et al., 2023), to evaluate different methods, with detailed explanation in App. D.3. As shown in Fig. 10(b), Concord is better than the three baselines and comparable with Oracle. For instance, the percentage of volunteers who



(a) Main results (b) Human preference

Figure 10: (a) Single coordination performance results with humans. (b) Human preference results for row method over column method. The value at i-th row and j-th column is the difference in the percentage of humans who prefer i-th method compared to those who prefer j-th method more.

prefer Concord method is 62.5% higher than those who prefer CLEAR. This shows that Concord is relatively more preferred by humans, demonstrating the value of Concord for practical human-AI coordination.

5 Related Work

Many real-world problems consist of multiple interactive agents, which can often be represented as a Multi-Agent Reinforcement Learning (MARL) problem (Busoniu et al., 2008; Zhang et al., 2021). Additionally, when these agents share a common objective, it falls under the category of cooperative MARL (Oroojlooy & Hajinezhad, 2022), which has exhibited significant advancements in diverse domains such as path finding (Sartoretti et al., 2019), active voltage control (Wang et al., 2021a), and dynamic algorithm configuration (Xue et al., 2022b). Numerous approaches have been proposed to facilitate agent coordination, including policy-based methods such as MADDPG (Lowe et al., 2017) and MAPPO (Yu et al., 2022), value-based techniques like VDN (Sunehag et al., 2018) and QMIX (Rashid et al., 2018), and other approaches like the transformer architecture (Wen et al., 2022). These methods have demonstrated remarkable coordination abilities across a wide range of tasks, including SMAC, Hanabi, and GRF (Yu et al., 2022). In addition to the mentioned approaches and their variants, numerous other methods have been proposed to investigate cooperative MARL, such as efficient communication strategies (Zhu et al., 2022), offline policy deployment (Zhang et al., 2023), model learning in MARL (Wang et al., 2022a), policy robustness in the presence of perturbations (Guo et al., 2022), and training paradigms like CTDE (centralized training with decentralized execution) (Lyu et al., 2021), and more.

Building AIs that can cooperate and coordinate with different teammates or human remains a fundamental requirement and challenge for AI (Dafoe et al., 2021). This ability is essential for numerous applications, including human-machine communication (Guzman & Lewis, 2020), cooperative autonomous vehicles (Toghi et al., 2021), and assistive robot control (Losey et al., 2022). A typical approach utilizes techniques like modelling (Albrecht & Stone, 2018) to capture others' intentions or behavior or (repeatedly) build an effective behavior model over human data and plan with the human model (Sheridan, 2016). These methods require an expensive and time-consuming data-collection process. Benefiting from the success of cooperative MARL (Wong et al., 2022), many related approaches like ad-hoc teamwork (AHT) (Mirsky et al., 2022), zero-shot coordination (ZSC) (Treutlein et al., 2021), few-shot teamwork (FST) (Fosong et al., 2022) have been developed recently. AHT is the research problem of designing agents that can coordinate with new teammates without prior coordination (Stone et al., 2010), including teammates type inference (Barrett & Stone, 2015; Chen et al., 2020), changing point detection (Ravula et al., 2019), partial observation solving (Gu et al., 2021), adversarial training (Fujimoto et al., 2022), robustness training (Rahman et al., 2022), etc. The goal of ZSC is to train the agent(s) that can coordinate effectively with a wide range of

unseen teammates (Hu et al., 2020; Treutlein et al., 2021). Many works emerging to improve the ability of multi-agent system from different aspects, including diversity measurement (Lupu et al., 2021; Zhao et al., 2023), training paradigm design (Hu et al., 2020; Strouse et al., 2021; Xue et al., 2022a), human bias based approaches (Yu et al., 2023), diverse teammates generation (Charakorn et al., 2023), etc. The agents in AHT aim to coordinate with teammates from a target population, while agents in ZSC need aim to coordinate with arbitrary unseen teammates without any prior coordination (Mirsky et al., 2022; Lauffer et al., 2023). Few-shot adaptation plays a crucial role in single agent reinforcement learning (Kumar et al., 2020; Osa et al., 2022; Gaya et al., 2022b), which samples K episodes by any policy to obtain a latent variable z for downstream tasks. In FST setting (Fosong et al., 2022), skilled agents trained in a team to complete one task are combined with skilled agents from different tasks and together must learn to adapt to an unseen but related task.

Another topic that is related to our work is Continual Learning, which involves sequential representations of tasks or samples (Parisi et al., 2019; Masana et al., 2020; Kudithipudi et al., 2022). Continual learning, or to say incremental learning or lifelong learning, is considered in many different fields, while in the field of continual reinforcement learning (Khetarpal et al., 2022), we expect the agents to sequentially learn decision-making across different tasks. Among the previous works, EWC (Kirkpatrick et al., 2017) conducts ℓ_2 distance-based weight regularization with the weights learned in the previous tasks. This algorithm requires additional supervision information to select a specific Q-function head and task-specific exploration schedule for different tasks. Different from EWC, CLEAR (Rolnick et al., 2018) is a task-agnostic method without need for task information during the continual learning process. It stores a large experience replay buffer and addresses the forgetting problem by sampling data of previous tasks from the buffer. Several other approaches, such as HyperCRL (Huang et al., 2021) and (Kessler et al., 2022a), utilize a learned world model to enhance the efficiency of continual learning. To address scalability issues in scenarios with a large number of tasks, CN-DPM (Lee et al., 2020) and LLIRL (Wang et al., 2022b) decompose the task space into subsets of data (tasks) and employ techniques like Dirichlet Process Mixture or Chinese Restaurant Process to expand the neural network for efficient continual supervised learning and reinforcement learning tasks, respectively. OWL (Kessler et al., 2022b) is a recently proposed approach that utilizes a multi-head architecture to achieve high learning efficiency. CSP (Gaya et al., 2022a) incrementally builds a subspace of policies for training a reinforcement learning agent on a sequence of tasks. Despite the various approaches that have been proposed to solve the continual learning problem, they mainly focus on the field of single-agent reinforcement learning. There is still limited research considering continual learning in multi-agent problem setting, let alone the problem of multi-agent continual coordination learning. MACPro (Yuan et al., 2023b) firstly formulates the multi-agent coordination problem, and develops an efficient algorithm for the setting. Macop (Yuan et al., 2023a) further endow the controllable agents in a multi-agent system with continual learning ability for a high generalization coordination policy.

Even the mentioned methods could endow the trained agents with the ability to coordinate more efficiently with different teammates. They need to train with all teammates at all times, which requires extensive computing and storage resources and extra design, such as prioritized sampling. In this work, we show these approaches are of low learning efficiency and take a further step for this problem by modeling it as a continual learning problem.

6 Conclusion and Future Work

Recognizing the importance and practicality of human-AI coordination, this study takes a significant stride towards addressing this challenge in a continual manner. We begin by formulating the problem as a MACMDP, where teammates are encountered sequentially. Subsequently, a mechanism based on hyper-teammate identification is proposed to avoid catastrophic forgetting while promote forward knowledge transfer for multi-agent coordination. To the best of our knowledge, our proposed method, Concord, is the first to address the human-AI coordination problem via continual training. Experiments on various multi-agent benchmarks validate the effectiveness of Concord, demonstrating its strong ability to continually coordinate with generated human-like models or real human participants across a range of evaluation metrics. Currently, Concord has the following two main limitations. Although Concord alleviates the issue of storage overhead compared to previous methods, it still necessitates a certain amount of storage to prevent forgetting. Be-

sides, addressing large-scale problems poses a challenge for current human-AI coordination methods. Further research is required to delve deeper into these two topics. Expanding Concord to decentralized cooperative decision-making settings beyond human-AI coordination is also an interesting future direction.

Acknowledgment

This work is supported by the Natural Science Foundation of Jiangsu (BK20243039, BK20241199).

References

Stefano V Albrecht and Peter Stone. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258:66–95, 2018.

Samuel Barrett and Peter Stone. Cooperating with unknown teammates in complex domains: A robot soccer case study of ad hoc teamwork. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2010–2016, 2015.

Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learnin. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38 (2):156–172, 2008.

Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-AI coordination. In *Advances in Neural Information Processing Systems 32*, pp. 5175–5186, 2019.

Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 233–248, 2018.

Rujikorn Charakorn, Poramate Manoonpong, and Nat Dilokthanakul. Generating diverse cooperative agents by learning incompatible policies. In *The 11th International Conference on Learning Representations*, 2023.

Shuo Chen, Ewa Andrejczuk, Zhiguang Cao, and Jie Zhang. Aateam: Achieving the ad hoc teamwork by employing the attention mechanism. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 7095–7102, 2020.

Allan Dafoe, Yoram Bachrach, Gillian Hadfield, Eric Horvitz, Kate Larson, and Thore Graepel. Cooperative ai: machines must learn to find common ground. *Nature*, 593(7857):33–36, 2021.

Kenneth Derek and Phillip Isola. Adaptable agent populations via a generative model of policies. In *Advances in Neural Information Processing Systems 34*, pp. 3902–3913, 2021.

Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer vision–ECCV 2020: 16th European conference, Glasgow, UK, August 23–28, 2020, proceedings, part XX 16*, pp. 86–102. Springer, 2020.

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

Ted Fujimoto, Samrat Chatterjee, and Auroop Ganguly. Ad hoc teamwork in the presence of adversaries. *arXiv preprint arXiv:2208.05071*, 2022.

Jean-Baptiste Gaya, Thang Doan, Lucas Caccia, Laure Soulier, Ludovic Denoyer, and Roberta Raileanu. Building a subspace of policies for scalable continual learning. *arXiv preprint arXiv:2211.10445*, 2022a.

Jean-Baptiste Gaya, Laure Soulier, and Ludovic Denoyer. Learning a subspace of policies for online adaptation in reinforcement learning. In *The 10th International Conference on Learning Representations*, 2022b.

Pengjie Gu, Mengchen Zhao, Jianye Hao, and Bo An. Online ad hoc teamwork under partial observability. In *The 10th International Conference on Learning Representations*, 2021.

Jun Guo, Yonghong Chen, Yihang Hao, Zixin Yin, Yin Yu, and Simin Li. Towards comprehensive testing on the robustness of cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2204.07932*, 2022.

Andrea L Guzman and Seth C Lewis. Artificial intelligence and communication: A human–machine communication research agenda. *New Media & Society*, 22(1):70–86, 2020.

David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *The 5th International Conference on Learning Representations*, 2017.

Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020.

Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 805–813, 2015.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems 29*, pp. 4565–4573, 2016.

Hengyuan Hu, Adam Lerer, Alex Peysakhovich, and Jakob Foerster. "other-play" for zero-shot coordination. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 4399–4410, 2020.

Yizhou Huang, Kevin Xie, Homanga Bharadhwaj, and Florian Shkurti. Continual model-based reinforcement learning with hypernetworks. In *IEEE International Conference on Robotics and Automation*, pp. 799–805, 2021.

Samuel Kessler, Piotr Miłoś, Jack Parker-Holder, and Stephen J Roberts. The surprising effectiveness of latent world models for continual reinforcement learning. *arXiv preprint arXiv:2211.15944*, 2022a.

Samuel Kessler, Jack Parker-Holder, Philip J. Ball, Stefan Zohren, and Stephen J. Roberts. Same state, different task: Continual reinforcement learning without interference. In *Thirty-Sixth AAAI Conference on Artificial Intelligence*, pp. 7143–7151, 2022b.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.

Cory D Kidd and Cynthia Breazeal. Robots at home: Understanding long-term human-robot interaction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3230–3235, 2008.

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

Glen Klien, David D Woods, Jeffrey M Bradshaw, Robert R Hoffman, and Paul J Feltovich. Ten challenges for making automation a "team player" in joint human-agent activity. *IEEE Intelligent Systems*, 19(6): 91–95, 2004.

Raphael Koster, Jan Balaguer, Andrea Tacchetti, Ari Weinstein, Tina Zhu, Oliver Hauser, Duncan Williams, Lucy Campbell-Gillingham, Phoebe Thacker, Matthew Botvinick, et al. Human-centred mechanism design with democratic AI. *Nature Human Behaviour*, 6(10):1398–1407, 2022.

Dhireesha Kudithipudi, Mario Aguilar-Simon, Jonathan Babb, Maxim Bazhenov, Douglas Blackiston, Josh Bongard, Andrew P Brna, Suraj Chakravarthi Raja, Nick Cheney, Jeff Clune, et al. Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4(3):196–210, 2022.

Saurabh Kumar, Aviral Kumar, Sergey Levine, and Chelsea Finn. One solution is not all you need: Few-shot extrapolation via structured maxent RL. In *Advances in Neural Information Processing Systems 33*, pp. 8198–8210, 2020.

Niklas Lauffer, Ameesh Shah, Micah Carroll, Michael D. Dennis, and Stuart Russell. Who needs to know? Minimal knowledge for optimal coordination. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 18599–18613, 2023.

Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. In *The 8th International Conference on Learning Representations*, 2020.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems 30*, pp. 6467–6476, 2017.

Dylan P Losey, Hong Jun Jeon, Mengxi Li, Krishnan Srinivasan, Ajay Mandlekar, Animesh Garg, Jeannette Bohg, and Dorsa Sadigh. Learning latent actions to control assistive robots. *Autonomous robots*, 46(1): 115–147, 2022.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems 30*, pp. 6379–6390, 2017.

Andrei Lupu, Brandon Cui, Hengyuan Hu, and Jakob Foerster. Trajectory diversity for zero-shot coordination. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 7204–7213, 2021.

Xueguang Lyu, Yuchen Xiao, Brett Daley, and Christopher Amato. Contrasting centralized and decentralized critics in multi-agent reinforcement learning. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 844–852, 2021.

Anuj Mahajan, Mikayel Samvelyan, Tarun Gupta, Benjamin Ellis, Mingfei Sun, Tim Rocktäschel, and Shimon Whiteson. Generalization in cooperative multi-agent systems. *arXiv preprint arXiv:2202.00104*, 2022.

Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification. *arXiv preprint arXiv:2010.15277*, 2020.

Reuth Mirsky, Ignacio Carlucho, Arrasy Rahman, Elliot Fosong, William Macke, Mohan Sridharan, Peter Stone, and Stefano V Albrecht. A survey of ad hoc teamwork research. In *European Conference on Multi-Agent Systems*, pp. 275–293, 2022.

Bilge Mutlu, Allison Terrell, and Chien-Ming Huang. Coordination mechanisms in human-robot collaboration. In *Proceedings of the Workshop on Collaborative Manipulation*, pp. 1–6, 2013.

Afshin Oroojlooy and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, pp. 1–46, 2022.

Takayuki Osa, Voot Tangkaratt, and Masashi Sugiyama. Discovering diverse solutions in deep reinforcement learning by maximizing state–action-based mutual information. *Neural Networks*, 152:90–104, 2022.

Johannes Von Oswald, Christian Henning, João Sacramento, and Benjamin F. Grewe. Continual learning with hypernetworks. In *The 8th International Conference on Learning Representations*, 2020.

Xinlei Pan, Weiyao Wang, Xiaoshuai Zhang, Bo Li, Jinfeng Yi, and Dawn Song. How you act tells a lot: Privacy-leaking attack on deep reinforcement learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 368–376, 2019.

Georgios Papoudakis, Filippos Christianos, and Stefano Albrecht. Agent modelling under partial observability for deep reinforcement learning. In *Advances in Neural Information Processing Systems 34*, pp. 19210–19222, 2021.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Arrasy Rahman, Elliot Fosong, Ignacio Carlucho, and Stefano V Albrecht. Towards robust ad hoc teamwork agents by creating diverse training teammates. *arXiv preprint arXiv:2207.14138*, 2022.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 4295–4304, 2018.

Manish Ravula, Shani Alkoby, and Peter Stone. Ad hoc teamwork with behavior switching agents. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 550–556, 2019.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Greg Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems 32*, pp. 348–358, 2018.

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. The Starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2186–2188, 2019.

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

Thomas B Sheridan. Human–robot interaction: status and challenges. *Human factors*, 58(4):525–532, 2016.

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

Peter Stone, Gal Kaminka, Sarit Kraus, and Jeffrey Rosenschein. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pp. 1504–1509, 2010.

DJ Strouse, Kevin McKee, Matt Botvinick, Edward Hughes, and Richard Everett. Collaborating with humans without human data. In *Advances in Neural Information Processing Systems 34*, pp. 14502–14515, 2021.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinícius Flores Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2085–2087, 2018.

Gerald Tesauro. Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural computation*, 6(2):215–219, 1994.

Behrad Toghi, Rodolfo Valiente, Dorsa Sadigh, Ramtin Pedarsani, and Yaser P Fallah. Cooperative autonomous vehicles that sympathize with human drivers. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4517–4524, 2021.

Johannes Treutlein, Michael Dennis, Caspar Oesterheld, and Jakob Foerster. A new formalism, method and open issues for zero-shot coordination. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 10413–10423, 2021.

L. Van der Maaten and G. E. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9 (11):2579–2605, 2008.

Jianhong Wang, Wangkun Xu, Yunjie Gu, Wenbin Song, and Tim C. Green. Multi-agent reinforcement learning for active voltage control on power distribution networks. In *Advances in Neural Information Processing Systems 34*, pp. 3271–3284, 2021a.

Xihuai Wang, Zhicheng Zhang, and Weinan Zhang. Model-based multi-agent reinforcement learning: Recent progress and prospects. *arXiv preprint arXiv:2203.10603*, 2022a.

Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. DOP: Off-policy multi-agent decomposed policy gradients. In *The 9th International Conference on Learning Representations*, 2021b.

Zhi Wang, Chunlin Chen, and Daoyi Dong. Lifelong incremental reinforcement learning with online bayesian inference. *IEEE Transactions on Neural Networks and Learning Systems*, 33(8):4003–4016, 2022b.

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 *Advances in Neural Information Processing Systems 35*, pp. 16509–16521, 2022.

Maciej Wołczyk, Michał Zajkac, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Continual world: A robotic benchmark for continual reinforcement learning. In *Advances in Neural Information Processing Systems 34*, pp. 28496–28510, 2021.

Annie Wong, Thomas Bäck, Anna V Kononova, and Aske Plaat. Deep multiagent reinforcement learning: Challenges and directions. *Artificial Intelligence Review*, pp. 1–34, 2022.

Annie Xie, Dylan Losey, Ryan Tolsma, Chelsea Finn, and Dorsa Sadigh. Learning latent representations to influence multi-agent interaction. In *4th Conference on Robot Learning*, pp. 575–588, 2021.

Ke Xue, Yutong Wang, Cong Guan, Lei Yuan, Fu Haobo, Fu Qiang, Chao Qian, and Yang Yu. Heterogeneous multi-agent zero-shot coordination by coevolution. *arXiv preprint arXiv:2208.04957*, 2022a.

Ke Xue, Jiacheng Xu, Lei Yuan, Miqing Li, Chao Qian, Zongzhang Zhang, and Yang Yu. Multi-agent dynamic algorithm configuration. In *Advances in Neural Information Processing Systems 35*, pp. 20147–20161, 2022b.

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 *Advances in Neural Information Processing Systems 35*, pp. 24611–24624, 2022.

Chao Yu, Jiaxuan Gao, Weilin Liu, Botian Xu, Hao Tang, Jiaqi Yang, Yu Wang, and Yi Wu. Learning zero-shot cooperation with humans, assuming humans are biased. In *The 11th International Conference on Learning Representations*, 2023.

Lei Yuan, Lihe Li, Ziqian Zhang, Feng Chen, Tianyi Zhang, Cong Guan, Yang Yu, and Zhi-Hua Zhou. Learning to coordinate with anyone. *preprint arXiv:2309.12633*, 2023a.

Lei Yuan, Lihe Li, Ziqian Zhang, Fuxiang Zhang, Cong Guan, and Yang Yu. Multi-agent continual coordination via progressive task contextualization. *preprint arXiv:2305.13937*, 2023b.

Lei Yuan, Ziqian Zhang, Lihe Li, Cong Guan, and Yang Yu. A survey of progress on cooperative multi-agent reinforcement learning in open environment. *arXiv preprint arXiv:2312.01058*, 2023c.

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 11th International Conference on Learning Representations*, 2023.

Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.

R. Zhao, J. Song, H. Hu, Y. Gao, Y. Wu, Z. Sun, and Y. Wei. Maximum entropy population based training for zero-shot human-ai coordination. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI)*, pp. 6145–6153, 2023.

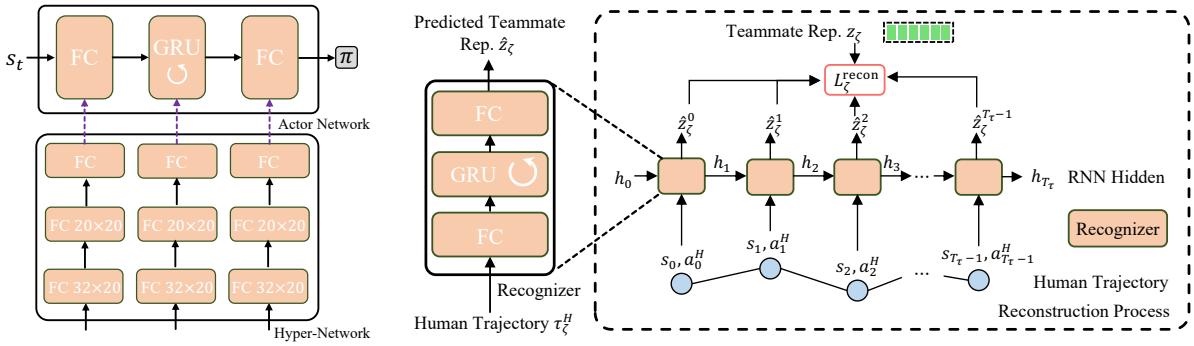
Zhi-Hua Zhou. Open-environment machine learning. *National Science Review*, 9(8), 2022.

Changxi Zhu, Mehdi Dastani, and Shihan Wang. A survey of multi-agent reinforcement learning with communication. *arXiv preprint arXiv:2203.08975*, 2022.

Appendix

A More Details about the Proposed Method

A.1 Network Architecture



(a) (b)

Figure 11: (a) Network architecture of the hyper-network and actor network. (b) Network architecture of the recognizer network. T_τ in the figure is short for $T_{\tau_\zeta^H}$, indicating the horizon of the human trajectory τ_ζ^H.

Hyper-Network The hyper-network in our method is used to generate the parameters of the AI actor by inputting the teammate representation to it. Thus, the network architecture determines the size of parameters generated by the hyper-network. We model the actor network with three layers as shown in Fig. 11(a), including a fully-connected layer, followed by a 64 bit GRU, and further followed by another fully-connected layer that outputs a probability distribution over local actions. In our design, the parameters of this actor network is completely generated by the hyper-network. We model the hyper-network as three multi-layer perceptions (MLPs), each generating the parameters of one layer of the actor network, which is also shown in Fig. 11(a). We set each MLP with two hidden layers of 20 neurons and ELU non-linearity function, and the input dimension of each MLP network equals to the dimension of teammate representation, which is set 32 throughout all our experiments.

Recognizer Network The recognizer network (see Fig. 11(b)) is utilized to reconstruct the teammate representation z_ζ from a few of human interaction trajectories. Specifically, the recognizer network is built as a RNN-based trajectory encoder, which takes the human trajectory as input. For each human trajectory τ_ζ^H, the recognizer will output a predicted teammate representation for each timestep t, which we denote as $\hat{z}_\zeta^t$, with the trajectory information before t. Then we update the recognizer network via minimizing the following reconstruction loss:

$$\mathcal{L}_\zeta^{\text{recon}} = \sum_{t=0}^{T_\tau-1} \lambda^{T_\tau-1-t} \|\hat{z}_\zeta^t - z_\zeta\|_2^2, \tag{2}$$

which is a weighted sum of the ℓ_2 distances between every $\hat{z}_\zeta^t$ and z_ζ, where we utilize one hyper-parameter λ to control the weights. On one hand, by minimizing the ℓ_2 distance between each $\hat{z}_\zeta^t$ and z_ζ, we ensure the recognizer's ability to reconstruct the representation z_ζ from variable-length trajectory information. On the other hand, we assign larger weights to the $\hat{z}_\zeta^t$ loss terms with larger t. The reason for this practice is that during the final test phase, only the predicted teammate representation at the last timestep $\hat{z}_\zeta^{T_\tau-1}$ is utilized as the input of the subsequent networks, which means that we hope that $\hat{z}_\zeta^t$ with larger t should be more accurate. From a different perspective, the reconstruction losses for $\hat{z}_\zeta^t$ where t is smaller than $T_\tau - 1$ are utilized as auxiliary losses to facilitate the recognizer network training. In practice, the hyper-parameter λ is set as 0.994 and 0.92 in Overcooked and SMAC, respectively. Above, we have explained how the recognizer

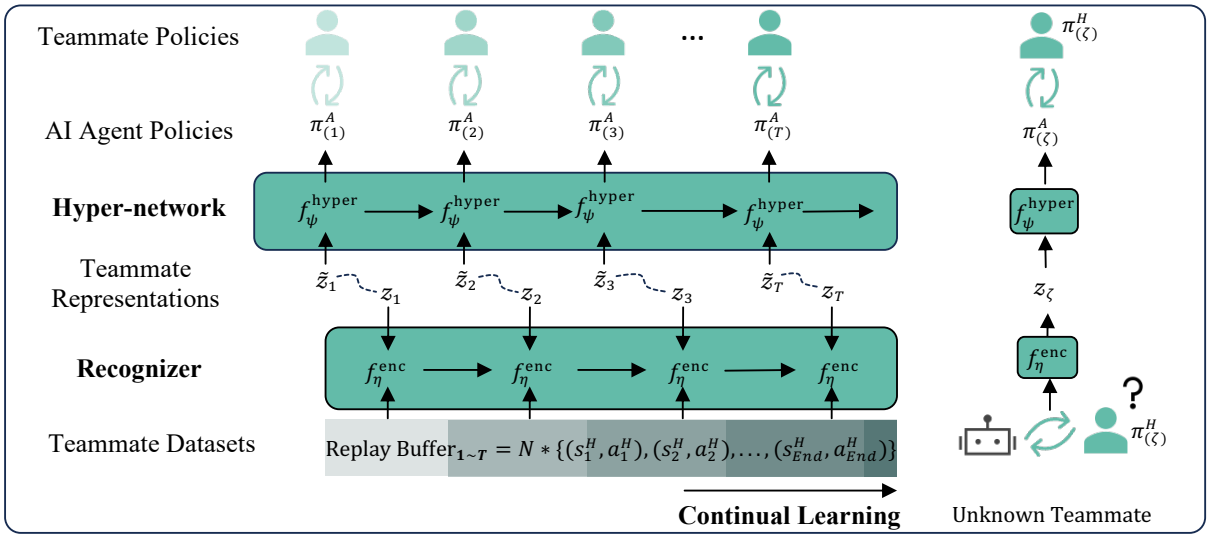


Figure 12: A high-level workflow showing the main idea of our approach. We use $\tilde{z}_\zeta$ to denote the initialized representation and z_ζ for the learned teammate representation. During the recognizer training, each task stores a small amount of data in the Replay Buffer, where N is the number of trajectories per teammate, and supervised learning is performed to align the behavioral trajectories and teammate representations.

network generates the reconstructed teammate representation with one human trajectory. However, we aim to utilize multiple trajectories to recognize teammates in our algorithm. In practice, we firstly generate representation with each human trajectory and then compute the mean vector of these representations.

A.2 Overall Workflow of Concord

Following the workflow of the multi-agent continual coordination learning in Fig. 2, the main idea of our approach is depicted in Fig. 12. In more details, Concord can be decomposed into two phases, the training phase and the test phase. During the training phase, we conduct the continual coordination learning, and during the test phase, we recognize the involved human teammates and let the AI agents to cooperate with them. The overall flow of these two phases are respectively shown in Alg. 1 and Alg. 2 in the maintext.

A.3 Storage Requirement During Continual Learning

To address the potential forgetting problem of the hyper-network and recognizer network, Concord needs to store some variables when learning continually. Specifically, our algorithm requires storing the following two types of variables:

a. The parameters of the hyper-network at the last round, denoted as $\tilde{\psi}$, are used to compute $\mathcal{L}^{\text{reg}}$. This variable always occupies static storage space, which is not relevant to the continual learning length T. Therefore, the storage cost for $\tilde{\psi}$ is considered acceptable.

b. A small buffer $\mathcal{B}$ is utilized to store a few of human trajectories and teammate representations for previous teammates. For each encountered teammate policy π^H_ζ, we store 36 human trajectories and one representation vector with dimension of 32. The final storage cost for these variables grows linearly with the continual learning length T. In practice, we find that the storage cost for these variables are relatively acceptable because we only store a few of human trajectories and the storage cost for teammate representations is small.

B Experiment Setup

B.1 Hyper-parameters Selection

In Tab. 2, we list the selection of hyper-parameters that are critical to our algorithm. We adopted this set of hyper-parameters throughout all the experiments in this paper. However, we utilized different training periods per task on different environment scenarios, of which the details are listed in Tab. 3.

Table 2: Hyper-parameters selected for our proposed approach.

Hyper-parameter	Value
Discount factor γ	0.99
RMSprop learning rate for actor / critic	0.0005 / 0.0001
Adam learning rate for hyper and recognizer network	0.001
Batch size for actor learning	32
Buffer size for critic learning	5000
Max/Min ϵ value for ϵ-greedy exploration	0.5 / 0.05
Step number of exploration decay period	500k
Embedding dimension for teammate representation (D^{emb})	32
Regularization loss weight (β^{reg})	2.0
Episode number for teammate recognition	2
Episode number for training / validating recognizer network	32 / 4

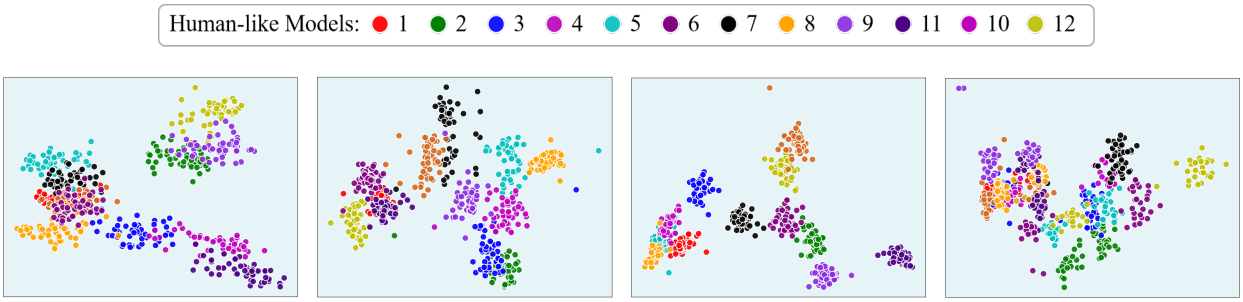
Table 3: Training period per task for different environment scenarios.

Environment	Scenario	Step number
Overcooked	Coord. Ring	5M
	Asy. Advantages	10M
	Many Orders	7M
	Open Asy. Advantages	7M
SMAC	3m	3M
	2z1s	1.5M
	4m	2M

B.2 Construction of Human-like Teammates

To evaluate the performance of coordination with human-like teammates, we should first construct a set of teammates. Building AIs that can cooperate with humans remains a fundamental challenge (Dafoe et al., 2021), as it requires human-like teammates under the property of high behavior diversity. We here use Maximum Entropy Population-based training (MEP) (Zhao et al., 2023) and Hidden-utility Self-Play (HSP) (Yu et al., 2023) algorithms to model a diversity of human-like behavioral policies, for their proven diverse agent ability in the Overcook environment. MEP trains partners with population entropy bonus to promote the pairwise diversity between partners and the individual diversity of partners themselves. HSP explicitly models human biases as hidden reward functions in the self-play objective thus can generate an augmented policy pool with biased policies. We use the code released by the HSP algorithm[1] to train MEP and HSP policies, and the number of our human-like models is defined as 12 (among them, 8 MEP policies and 4 HSP policies), to simulate real human behavior diversity and preference. As shown in Fig. 13, the diverse teammates allow the agent to learn more universal coordination behaviors, and therefore have stronger continual human-AI coordination performance.

[1]The code we used for HSP is at `https://github.com/samjia2000/HSP`



(a) *Coord. Ring* (b) *Asymm. Adv.* (c) *Many Orders* (d) *Open Asymm. Adv.*

Figure 13: Illustration of diversity of teammates. We plot the distribution of teammates' actions $\pi_p\{\cdot \mid s\}$ by t-SNE (Van der Maaten & Hinton, 2008) projection, where each point represents the t-SNE results of teammates' action distribution in each episode, and different teammates have different action distributions, thus the obtained teammates can be seen as diverse.

C Additional Experimental Results

C.1 Additional Results on Overcooked Environment

In Sec. 4.2 of the manuscript, we compare the detailed single coordination performance of Concord and Vanilla on the Asym. Adv. layout of Overcooked environment. To further analyze the situations on several other layouts, we present additional experimental results on the other three layouts here. Fig. 14-16 demonstrate that Concord continues to suffer less forgetting and achieves better forward transfer performance.

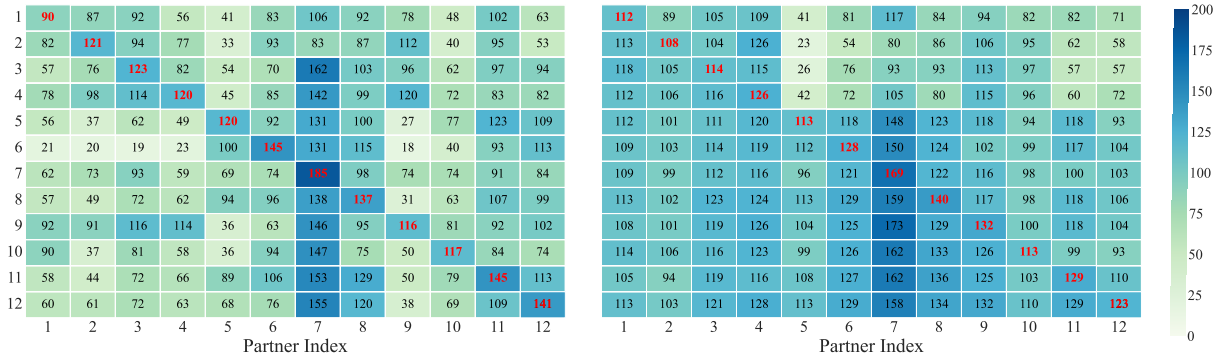


Figure 14: Single coordination performance of Vanilla (left) and Concord (right) on the Coord. Ring layout. The value at i-th row and j-th in the matrix is $P_i(j)$.

C.2 Additional Analysis

One important part of Concord is to utilize the recognizer network to recognize the teammates through a few of interaction trajectories during the test phase. To analyze whether such a few-shot mechanism is necessary and examine how the number of episodes used for teammate recognition influences the algorithm performance, we conduct ablation studies on the layout Asym. Adv. of Overcooked environment. We first provide the experimental results under different numbers of episodes for recognizing teammates, and then we further investigate the zero-shot coordination ability of different methods. Moreover, we also provide further analysis, including comparison with given teammate ID, ablation studies and sensitivity analysis of the recognition mechanism, as well as sensitivity analysis of the agent actor network architecture in this section.

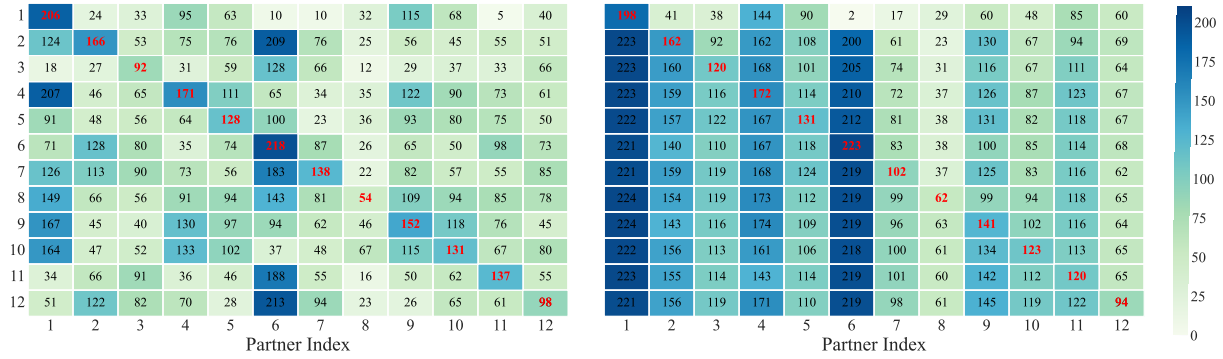


Figure 15: Single coordination performance of Vanilla (left) and Concord (right) on the Many Orders layout. The value at i-th row and j-th in the matrix is $P_i(j)$.

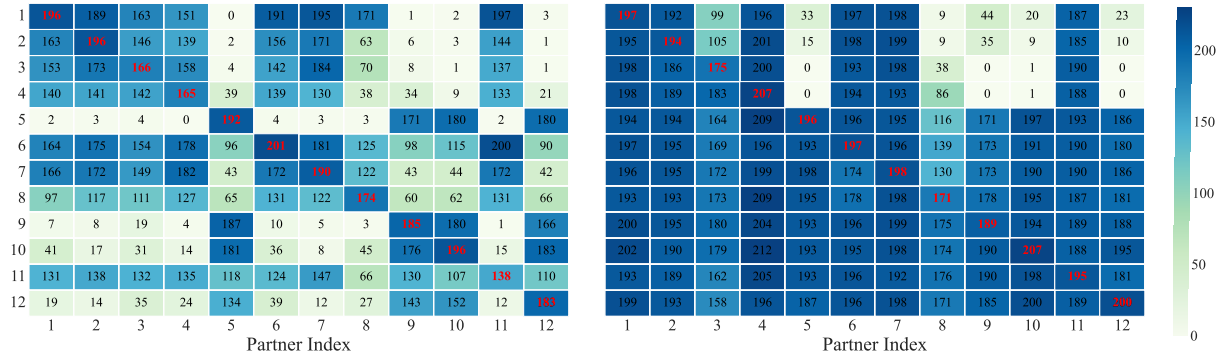


Figure 16: Single coordination performance of Vanilla (left) and Concord (right) on the Open Asy. Adv. layout. The value at i-th row and j-th in the matrix is $P_i(j)$.

Number of episodes for recognition. The selection of the number of episodes for human recognition is a critical hyper-parameter in Concord. As depicted in Fig. 17 (a), utilizing 2 episodes yields a notable improvement compared to using only 1 episode. Although employing more episodes necessitates additional data, the impact is not substantial. Consequently, the number of episodes for human recognition has been set to 2 in our experiments.

Few-shot and zero-shot recognition. We ablate the few-shot mechanism for different methods, as shown in Fig. 17 (b). For the zero-shot version of our approach, we refrain from pre-sampling certain trajectories to recognize teammates. Instead, we utilize in-episode trajectory information to estimate the teammate representation, which is subsequently inputted into the hyper-network. Additionally, to better differentiate between the methods, we evaluate their generalization ability in this ablation study by conducting tests with an additionally generated teammate policy. As we can see from the results, Concord achieves better performance than other methods under zero-shot setting, while only Concord with few-shot mechanism achieves comparable performance to the Oracle baseline. This result demonstrates the necessity of the few-shot mechanism.

Comparison with given teammates ID. To investigate the influence of known teammates, we additionally test the performance of Concord (ID), which can be seen as Concord equipped with an Oracle identification mechanism. As show in Tab. 4, we can see that Concord (ID) shows a slight advantage over Concord, but has no significant differences. This means that the identification accuracy of our identification

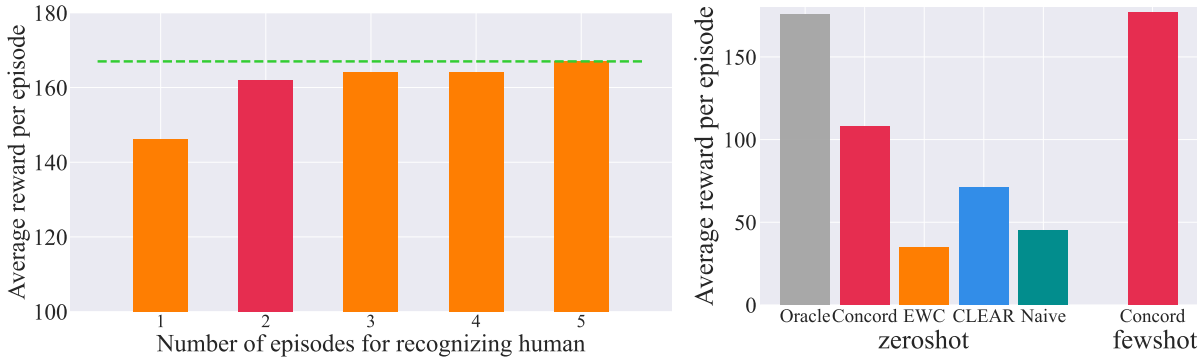


Figure 17: (a) The performance comparison with different episode numbers for the recognizer. (b) The performance of different algorithms in zero-shot testing and the performance of Concord algorithm in few-shot testing.

Table 4: Comparisons of distinct algorithms in all 4 layouts on average coordination performance with 5 random seeds. Concord (ID) is that the agents are given identities, which can be seen as Concord equipped with an Oracle identification mechanism. The best mean value on each problem except for Oracle is highlighted in bold. The symbols $+$, $-$, and $\approx$ indicate that the result is significantly superior to, inferior to, and almost equivalent to Concord, respectively, according to the Wilcoxon rank-sum test with a significance level of 0.05.

Environment	Oracle	EWC	CLEAR	Vanilla	Concord (ID)	Concord
Coord. Ring	127.05 ± 3.23	89.38 ± 11.87	102.54 ± 3.25	83.55 ± 12.83	$\mathbf{126.34 \pm 2.86}$	123.28 ± 3.01
Asy. Advantages	191.91 ± 2.42	51.48 ± 6.84	50.42 ± 3.28	47.28 ± 9.76	$\mathbf{169.27 \pm 6.39}$	160.08 ± 13.45
Many Orders	140.81 ± 4.22	98.03 ± 22.81	130.45 ± 6.01	79.85 ± 9.82	$\mathbf{137.22 \pm 5.43}$	134.76 ± 4.01
Open Asy. Advantages	197.83 ± 4.87	177.63 ± 3.58	106.95 ± 19.78	66.03 ± 3.24	$\mathbf{191.06 \pm 6.84}$	189.68 ± 5.71
$+/-/\approx$	2/0/2	0/4/0	0/3/1	0/4/0	0/0/4	/

mechanism is effective. Besides, the performance comparison between Concord (ID) and Oracle validates the effectiveness of other parts of Concord besides our identification module.

Ablation study for anti-forgetting mechanism. The anti-forgetting mechanism is also an important component of our method. To verify the effectiveness of this module, we design additional ablation experiments concerning this part. Specifically, we remove this anti-forgetting mechanism and conduct experiments on Overcooked environment, with the final results presented in Tab. 5. The results show that when the anti-forgetting mechanism is removed, our method suffers a significant performance drop, demonstrating the effectiveness of the anti-forgetting mechanism.

Sensitivity analysis of replay buffer size. To analyze the sensitivity of the replay buffer size in terms of addressing the forgetting issue, we add a sensitivity analysis in Tab. 6. It can be observed that Concord achieves relatively good performance across the ranges of buffer size for each teammate from 8 to 64. Note that Concord distinguishes itself from conventional replay-based methods to prevent any potential misunderstanding. In our approach, the replay buffer is solely employed to prevent the forgetting of teammate recognition rather than to help learn solving previous tasks. An example of conventional replay-based methods in our experiments is CLEAR.

Additionally, to further illustrate the role of teammate recognition, we conducted an experiment using random IDs. Unlike Concord, which identifies teammates by inputting trajectories into a recognizer network, Random ID randomly selects one representation from all stored teammate representations during the testing phase. We can observe that the performance of Concord given the random ID is worse than standard Concord. This demonstrates that our teammate recognition module can accurately identify teammates and leverage their information to achieve better collaboration.

Table 5: Experimental results of ablating the anti-forgetting mechanism. The "Ablation" in the table indicates Concord without anti-forgetting mechanism, and the better results are **bold**.

Method	Coord. Ring		Asy. Advantages		Many Orders		Open Asy. Adv.	
	Concord	Ablation	Concord	Ablation	Concord	Ablation	Concord	Ablation
Episode Reward	**123.28 ± 3.01**	85.25 ± 10.23	**160.08 ± 13.45**	50.12 ± 8.76	**134.76 ± 4.01**	81.13 ± 11.94	**189.68 ± 5.71**	90.99 ± 10.87
Forg.	**-1.31 ± 4.23**	-20.15 ± 2.52	**-16.11 ± 6.85**	-90.45 ± 18.96	**-1.27 ± 6.69**	-60.28 ± 2.71	**-6.83 ± 4.59**	-100.08 ± 20.31
F. Transfer	**103.09 ± 13.02**	78.84 ± 11.79	**119.59 ± 9.41**	59.89 ± 7.65	**100.85 ± 3.14**	68.87 ± 5.82	**162.29 ± 6.94**	122.24 ± 17.52

Table 6: The sensitivity analysis of replay Buffer. The best mean results for Episode Reward are in **bold**. Here, Forg. represents the metric of Forgetting, F. Transfer represents the metric of Forward transfer, Random Rep. represents the identification mechanism with randomly generated representations, Random ID represents the identification mechanism that randomly selects one teammate representation from all previously stored teammate representations, and Oracle ID represents Concord with an oracle identification mechanism.

Size	Coord. Ring			Asy. Advantages		
	Episode Reward	Forg.	F. Transfer	Episode Reward	Forg.	F. Transfer
4	109.97 ± 0.98	-3.67 ± 5.62	99.69 ± 1.31	81.97 ± 7.32	-64.62 ± 6.01	108.57 ± 14.17
8	117.72 ± 1.97	-7.78 ± 1.85	113.63 ± 2.72	128.61 ± 13.87	-51.54 ± 13.05	131.87 ± 8.31
16	122.16 ± 1.86	-2.01 ± 1.13	114.81 ± 1.72	136.19 ± 13.61	-28.36 ± 15.45	143.57 ± 8.46
24	**126.51 ± 1.94**	0.61 ± 4.21	117.09 ± 2.22	165.94 ± 3.64	-2.73 ± 8.24	136.42 ± 9.83
32	124.91 ± 1.96	-1.93 ± 4.92	113.57 ± 2.56	166.41 ± 4.47	-12.46 ± 9.71	131.01 ± 5.04
48	126.27 ± 1.45	-0.27 ± 0.65	113.94 ± 2.03	**167.75 ± 13.96**	-5.79 ± 16.45	148.96 ± 4.92
64	111.94 ± 0.48	1.84 ± 2.14	104.21 ± 2.08	166.86 ± 4.76	-13.12 ± 3.77	123.81 ± 16.62
Random Rep.	36.92 ± 6.94	-1.58 ± 21.68	19.45 ± 5.64	33.27 ± 8.15	-20.36 ± 14.83	39.36 ± 14.17
Random ID	87.63 ± 9.58	-1.27 ± 1.14	85.32 ± 11.92	110.61 ± 9.98	-5.42 ± 0.93	92.75 ± 7.34
Oracle ID	126.34 ± 2.86	1.76 ± 2.64	88.45 ± 0.25	169.27 ± 6.39	-3.36 ± 3.12	109.79 ± 2.02

Sensitivity analysis of the size of $\theta_i^{A_\zeta}$. The default configuration of the agent's network is a 3-layer Multilayer Perceptron (MLP) with a hidden size of 64. We aim to assess the agent's performance as a function of the size of $\theta_i^{A_\zeta}$. To do this, we modified the network structure to produce agents with varying $\theta_i^{A_\zeta}$ sizes and then compared their performances. "Concord (32)" denotes an agent with a hidden layer size of 32, whereas "Concord (4 layers)" refers to a 4-layer MLP with each hidden layer sized at 64. Our findings indicate that the parameters generated by our hyper-network are adequately expressive to accomplish the task. A slight reduction in the number of parameters does not significantly degrade performance. In contrast, as we continue to increase the size of the network parameters, performance slightly declines. This may be due to the larger parameter size requiring more data for training.

Table 7: The sensitivity analysis of $\theta_i^{A_\zeta}$. Concord (32) represents a hidden layer size of 32, while Concord (4 layers) represents a 4-layer MLP with hidden layer size of 64.

Coord. Ring	Average coordination performance.	Forgetting	Forward Transfer	# Parameters
Concord (64)	123.28 ± 3.01	-1.31 ± 4.23	103.09 ± 13.02	631398
Concord (32)	119.83 ± 10.92	-4.18 ± 1.82	90.765 ± 9.95	188550
Concord (48)	120.83 ± 9.05	-4.05 ± 2.87	79.27 ± 16.07	377718
Concord (96)	126.55 ± 3.77	-2.15 ± 5.21	87.69 ± 10.03	1332294
Concord (128)	118.66 ± 8.24	-3.93 ± 3.68	84.96 ± 3.05	2291238
Concord (256)	108.83 ± 7.99	-7.21 ± 4.28	76.99 ± 8.01	8707494
Concord (4 layers)	121.83 ± 2.21	-3.53 ± 1.37	79.02 ± 0.61	719838

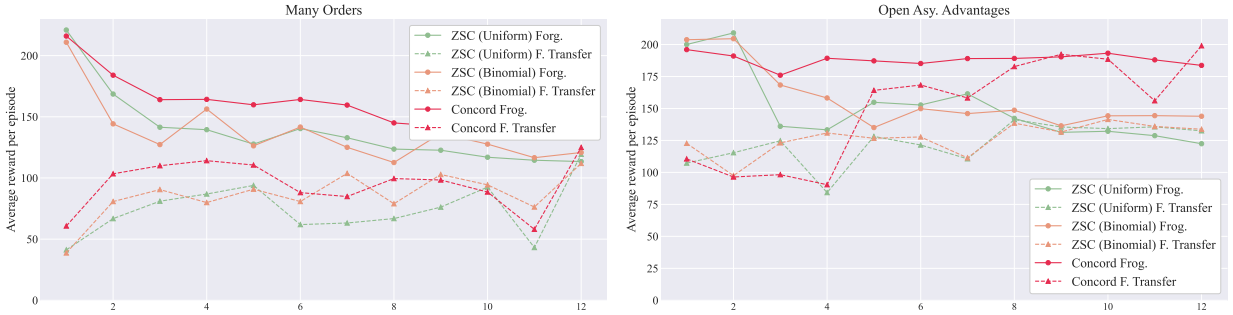


Figure 18: The performance comparison between Concord and ZSC (i.e., multi-task learning) on Many Orders and Open Asy. Advantages.

C.3 Discussions with Zero-Shot Coordination (ZSC) Methods

In human-AI collaboration, the classic approach (i.e., Zero-Shot Coordination, ZSC (Strouse et al., 2021; Zhao et al., 2023; Yu et al., 2023)) is to pre-generate all the diverse teammates and allow the AI agent to simultaneously train with them. In many real-world applications, humans do not collaborate with robots in this way. We do not want an AI to stop updating after collaborating with humans. Instead, as more and more people collaborate with the agent, the robot should continue to learn and improve its collaboration skills. This is the desired capability for an agent in human-AI collaboration scenarios. Thus, the training approach of ZSC has several drawbacks, such as necessitating access to all of them during training is unfeasible and enumerating all potential teammates in advance is very hard, as we discussed in our paper.

In the aforementioned realistic scenarios, we hope that after the agent collaborates with a human to complete a task, it will retain the ability to collaborate with them and improve its ability to cooperate with other teammates in the future. These two requirements correspond to the two important indicators in continual learning: anti-forgetting and forward transfer. We frame this challenge within the context of continual learning and seek solutions accordingly.

The essence of ZSC is multi-task learning, which allows the AI agent to train with multiple teammates simultaneously. This differs from our continual scenario where we believe that teammates arrive in sequence and the agent can only train with the current team, not past ones. Besides, the focus of ZSC methods is to provide a set of good partners for teammate training, while our work proposes a new problem formulation. We can also use ZSC techniques, as we did when constructing human-like teammates in our experiments.

Here, we conduct additional experiments to enhance the comparison between our setup and ZSC, which is akin to multi-task learning. In our experimental design, teammates are introduced sequentially, and both Concord and ZSC are allocated the same training budget. However, unlike Concord, ZSC is permitted to train with past teammates. When a new teammate (here we consider 2-player scenarios for simplicity) arrives, there should be a strategy to allocate the training timesteps between the new teammate and old ones. We propose two strategies, respectively "Uniform" and "Binomial" strategies. The "Uniform" strategy implies that all teammates are allocated an equal number of training timesteps during the current training session. The "Binomial" strategy indicates that the new teammate and all old teammates each receive half of the timesteps. It is worth noting that if the new teammate uses all the timesteps without training with the old teammates, the ZSC method is indeed the "Vanilla" method we discussed in our paper. This comparison more clearly differentiates our setup from ZSC. Note that this setup is not fair to us, as we can not retrain with past teammates.

The experimental results can be found in Fig. 18. Concord significantly outperforms ZSC using different strategies, demonstrating the significant advantages of our algorithm in this practical scenario.

Table 8: Comparison with two Meta RL baselines. The best results are **bold**.

Method	Coord. Ring			Asy. Advantages		
	Episode Reward	Forg.	F. Transfer	Episode Reward	Forg.	F. Transfer
Concord	$\mathbf{123.28 \pm 3.01}$	$\mathbf{-1.31 \pm 4.23}$	$\mathbf{103.09 \pm 13.02}$	$\mathbf{160.08 \pm 13.45}$	$\mathbf{-16.11 \pm 6.85}$	$\mathbf{119.59 \pm 9.41}$
PEARL	87.19 ± 1.31	-43.43 ± 1.28	87.28 ± 6.14	122.25 ± 5.44	-108.25 ± 5.2493	68.15 ± 3.30
PEARL w/ Anti-Forg	88.50 ± 1.66	-32.15 ± 3.09	81.85 ± 1.70	133.50 ± 8.88	-90.50 ± 4.08	79.75 ± 4.55

C.4 Discussions with Meta RL Methods

Our paper proposes a new formulation MACMDP, where the agents learn to cooperate with different teammates continually. Another domain investigating the agent's adaptability is meta RL. However, although continual RL and meta RL involves the changes of task distribution, but they are two completely different problem formulations. Continual RL emphasizes maintaining performance on both old and new tasks in a changing environment, while meta RL learns how to learn and focuses on improving the learning efficiency on one new task. We stress that Concord follows continual RL setting, and our evaluation phase involves only testing with teammates that have already been encountered, rather than involving coordination with unseen teammates like in meta RL.

Though our Concord follows continual RL, a setting different from meta RL, it shares certain similarities in practice with context-based meta RL method, as both involve learning task representations. However, EWC and CLEAR do not have such recognization mechanism. To strengthen baseline, we introduce PEARL, a classic context-based meta RL method, to our problem setting.

We design two variants to adapt to our setting, PEARL and PEARL w/ Anti-Forg. PEARL, compared to Vanilla, adds an encoder that processes trajectory data to model teammate representations. This encoder is optimized using reinforcement learning losses and a standard normal distribution regularization, as described in the original PEARL study. PEARL w/ Anti-Forg mitigates the challenge of forgetting in continual learning by utilizing a small buffer to store trajectories from previous training stages like us. When training with the current teammate, it regularize the outputs of the current AI policy and encoder networks to be close to those of the last-round networks on the old data. This is a typical practice in regularization-based methods (Castro et al., 2018; Douillard et al., 2020).

From the results, we can find that PEARL performs poor in our experiments, which reveals the significant differences between meta RL and our problem setting, where meta RL trains the policy on a set of tasks during the meta-train phase, while PEARL needs to face the tasks one-by-one in our problem setting. Besides, though PEARL w/ Anti-Forg performs slightly better than PEARL, our Concord still exhibits significant superiority over it. The reason why PEARL w/ Anti-Forg does not achieve ideal results might be that to constrain the output of the policy network to be close to the old network often requires storing more old data. Thus, when both storing 64 trajectories, our method shows better performance, reflecting the advantage of employing the hyper-network.

C.5 Experiments with More Continual Teams

An interesting questions is to investigate the performance when we have more teams to be continually coordinate. In this section, we double the teams to 24 to show that. For better presentation, we compared Vanilla and Concord and plotted the entire single coordination performance, which can be used to obtain all other metrics. The results can be found in Fig. 19. We found that, even though having more teammates brings significant challenges to continual learning, our Concord method still performs well and significantly outperforms Vanilla.

C.6 Complete Value Decomposition Study Results

In Sec. 4.1 of the manuscript, we study the influence of value decomposition in the Coord. Ring layout of Overcooked environment with line charts, of which the concrete statistics are presented in Tab. 9. To

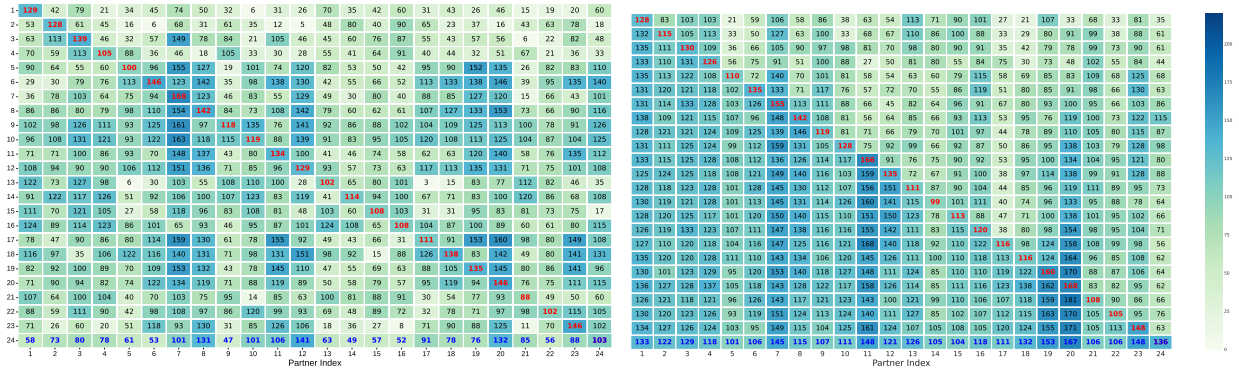


Figure 19: Single coordination performance of Vanilla (left) and Concord (right) on the Asym. Adv. layout with more partners. The value at i-th row and j-th in the matrix is $P_i(j)$.

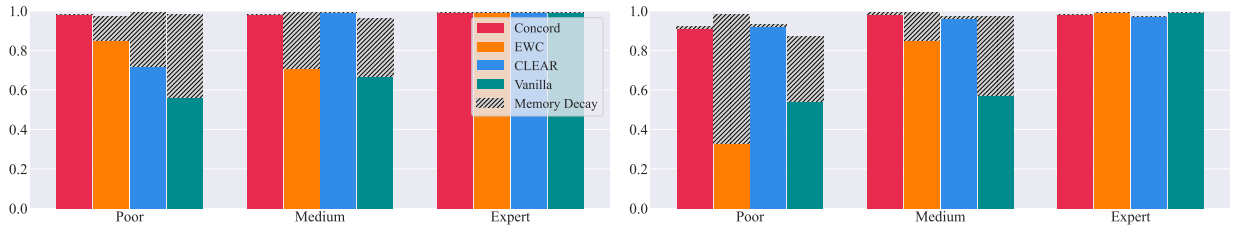


Figure 20: Experimental results on SMAC maps 3m and 2z1s. The gray shaded bar is used to denote the degree of forgetting, which equals to the highest test win rate during training minus the final test win rate.

further validate the effectiveness of value decomposition when there exist multiple AI agents cooperating with multiple human-like teammates, we further conduct value decomposition study on different SMAC maps, of which the results are reported in Tab. 10-12. Similar to the results in Overcooked, Concord with value decomposition also achieves better performance on different SMAC maps, which further significantly demonstrates the effectiveness of our value decomposition.

Table 9: Performance comparisons on eleven tasks between Concord with and without value decomposition in the Coord. Ring layout of Overcooked environment.

Algorithm	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8	Task9	Task10	Task11	Task11	Average
Concord	113	103	121	128	113	129	158	134	132	110	129	123	124
Concord w/o Human Critic	100	100	92	101	89	148	162	117	107	104	118	134	114

C.7 Additional Results on SMAC Environment

In Sec. 4.2 of the manuscript, we compare Concord with other methods on the SMAC map 4m. Here, we further provide the experimental results on the other two SMAC maps in Fig. 20 to demonstrate the consistency of the conclusion across different maps. From the results, we can observe that the Concord method still exhibits the smallest gray shaded region in these two maps, indicating that generality of the anti-forgetting ability of our approach.

D Detailed Information for Human Participation

D.1 Experimental Settings for Continual Coordination with Humans

To assess the performance of Concord in coordination with humans, we recruited 8 volunteers (25% female, 75% male, aged between 18 and 35). The volunteers were divided into four groups and given a comprehensive

Table 10: Performance comparisons on three tasks between Concord with and without value decomposition on the SMAC map 3m.

Algorithm	Task1	Task2	Task3	Average
Concord	0.98	0.98	0.99	0.9833
Concord w/o Human Critic	0.97	0.97	0.99	0.9766

Table 11: Performance comparisons on three tasks between Concord with and without value decomposition on the SMAC map 2z1s.

Algorithm	Task1	Task2	Task3	Average
Concord	0.91	0.99	0.98	0.96
Concord w/o Human Critic	0.92	0.93	0.97	0.94

Table 12: Performance comparisons on three tasks between Concord with and without value decomposition on the SMAC map 4m.

Algorithm	Task1	Task2	Task3	Task4	Task5	Task6	Average
Concord	0.81	0.82	0.91	0.91	0.92	0.99	0.8933
Concord w/o Human Critic	0.75	0.75	0.89	0.89	0.95	0.98	0.8683

introduction to the basic gameplay and experimental procedures. All volunteers were fully informed of their rights, and the experiments were approved by the relevant department. The Overcooked game was remotely deployed on a server accessible through the volunteers' browsers. Using this setup, we were able to collect trajectories of human players interacting with each other, as shown in Fig. 21. We collect around 160 human-human trajectories (for a total of 64k environment timesteps) and partition the these into two subsets, splitting each trajectory into two single-agent trajectories. Based on the collected human data, we construct 8 sets of human proxy models through behavior cloning. Then we train different methods by using these human proxy models as the teammates that appear sequentially. Note that the AI agents obtained by different methods will play with the real human participants, not the human proxy models. In our preliminary experiments, we observed that human models trained through behavior cloning outperformed those trained with Generative Adversarial Imitation Learning (GAIL) (Ho & Ermon, 2016). Hence, we decided to employ the former throughout our experiments.

D.2 Ethical Statement

In terms of the experiments that has human participation, we claim that we have employed effective practice to avoid possible ethical issues.

Test introduction Before the experiments, we informed the volunteers of the specific details of the entire experiments, including that they would be arranged to play the game with AI agents, and that their playing data during test and the final test results would be utilized for the academic research of this work. All the collected data is only for experiment purposes and we only make the testing results public in this paper. The volunteers voluntarily choose whether to participate in the experiment, and all participating volunteers were informed of and agreed to our practice.

Possible Risk for Humans and Broader Impact The main risks for the volunteers are twofold: the potential leakage of their personal information and the time cost. Regarding the former, we only invited the volunteers to participate in our experiments without the requirement for any unnecessary personal information. Additionally, we will keep the volunteers' identity information completely confidential. As for the latter, we first adjusted the frequency of the AI decision-making in the experiment to make the

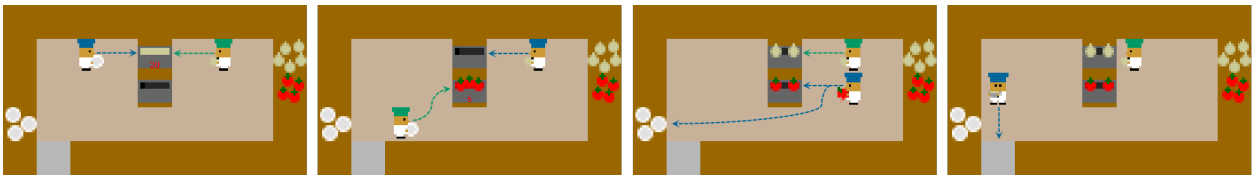


Figure 21: Open Asymmetric Advantages. The layout of the two rooms, where all players are initialized in the right room, and players need to coordinate to finish cooking, delivery or cooking and delivery at the same time, as well as ingredients (tomatoes or onions).The example trajectories represent different Human-AI coordination policies.

coordination process between humans and AI more comfortable. Besides, we provided appropriate material compensation for the volunteer's participation in the experiments.

Furthermore, the experimental results are encouraging in the sense that we demonstrate Concord is a promising method for dealing with Human-AI coordination in multi-agent systems via continual learning. It is not yet at the application stage and does not have a broader impact. However, this work learns one by one coordination via continual learning, making Concord more practical in real-world applications.

D.3 Definition of the Human Preference Metric

In Sec. 4.3 of the main paper, we validate our approach to conduct continual coordination with real humans. To provide a more comprehensive comparison of our approach with other baselines, we present the experimental results in two different dimensions. Among them, the main results are in the form of the single coordination performance, which is an objective metric. To further consider the humans' subjective feelings, we additionally include one metric we refer to as "Human preference" in the main paper (Strouse et al., 2021; Yu et al., 2023). To calculate the human preference, some human participants have tasks as judges. Note the real human participants in our experiments we recruited have two tasks:

- Players. We first collect some trajectories of them, then construct human proxy models based on these trajectories. Then we train different methods by using these human proxy models as the partners that appear sequentially. When testing, the obtained agents by different methods will play with the real human participants, not the human proxy modes. The final single coordination performance is shown in Fig. 10(a).
- Judges. We assign a different participant to each player as a judge, which will judge the preferences in the game between trained agents and the corresponding human player.

In terms of the formal definition of this metric, the human preference for method A over method B can be calculated as follows. Let N be the total number of human players participating in the experiment, N_A be the number of human players who rank A over B, and N_B be the number of those who rank B over A. Then, Human preference for method A over method B is computed as $\frac{N_A}{N} - \frac{N_B}{N}$. The human players will first play with the trained agents by different algorithms. The game replays are recorded for the following judging process, where each algorithm is evaluated against with a different algorithm. There is 5 different algorithms (i.e., Oracle, Concord, EWC, CLEAR, and Vanilla), thus there is total 10 rounds of evaluations. In each evaluation, human judges are asked to watch the replays of the two algorithms and vote for the algorithm that is preferred. For example, let $N = 8$ be the number of human judges, A and B are the two methods compared in a round, $N_A = 3$ be the number of human players who rank A over B, and $N_B = 5$ be the number of those who rank B over A. Then, Human preference for method A over method B is computed as $\frac{N_A}{N} - \frac{N_B}{N} = \frac{3}{8} - \frac{5}{8} = -0.25$. Similarly, human preference for method B over method A is 0.25. The final human preference is shown in Fig. 10(b).