

# Variational Offline Multi-agent Skill Discovery

Jiayu Chen<sup>1</sup>, Tian Lan<sup>2</sup>, Vaneet Aggarwal<sup>3</sup>

<sup>1</sup>Carnegie Mellon University

<sup>2</sup>The George Washington University

<sup>3</sup>Purdue University

jiayuc2@andrew.cmu.edu, tlan@gwu.edu, vaneet@purdue.edu

## Abstract

Skills are effective temporal abstractions established for sequential decision making, which enable efficient hierarchical learning for long-horizon tasks and facilitate multi-task learning through their transferability. Despite extensive research, research gaps remain in multi-agent scenarios, particularly for automatically extracting subgroup coordination patterns in a multi-agent task. In this case, we propose two novel auto-encoder schemes: VO-MASD-3D and VO-MASD-Hier, to simultaneously capture subgroup- and temporal-level abstractions and form multi-agent skills, which firstly solves the aforementioned challenge. An essential algorithm component of these schemes is a dynamic grouping function that can automatically detect latent subgroups based on agent interactions in a task. Further, our method can be applied to offline multi-task data, and the discovered subgroup skills can be transferred across relevant tasks without retraining. Empirical evaluations on StarCraft tasks indicate that our approach significantly outperforms existing hierarchical multi-agent reinforcement learning (MARL) methods. Moreover, skills discovered using our method can effectively reduce the learning difficulty in MARL scenarios with delayed and sparse reward signals. The codebase is available at: <https://github.com/LucasCJYSDL/VOMASD>.

## 1 Introduction

Skill discovery aims at extracting useful temporal abstractions from decision-making sequences. The downstream policy learning can be much more efficient by simply composing the discovered skills temporally into complex maneuvers. Also, skills can potentially be transferred among tasks to facilitate multi-task learning. Despite considerable research on single-agent skill discovery [Eysenbach *et al.*, 2019; Chen *et al.*, 2023a], skill discovery in multi-agent reinforcement learning (MARL) remains under-explored. A straightforward approach is to discover single-agent skills for each agent independently and then learning a multi-agent meta policy to coordinate their use, as in [Yang *et al.*, 2020;

Sachdeva *et al.*, 2021]. However, multi-agent coordination can not be abstracted in such individual skills. On the other hand, there are a limited number of works [Chen *et al.*, 2022; Yang *et al.*, 2023] on discovering skills for the entire team of agents. However, in multi-agent tasks, coordination patterns can emerge within subgroups of varying scales (from 1 to  $n$ ), and team skills (i.e.,  $n$ -agent skills) can be inflexible to use.

Notably, complex multi-agent tasks can often be decomposed as a series of subtasks, each of which requires participation of a subgroup of agents for a certain duration. The policies for these subgroups can be abstracted as multi-agent skills. While agents can explore various forms of collaboration in an online setting (by interacting with the environment), offline multi-agent skill discovery in contrast must infer latent coordination patterns from agent interactions in the offline data, with the subgroup size arbitrarily varying from 1 to  $n$ . This gives rise to a combinatorial problem of dynamic subgroup division and forming temporal abstractions within each subgroup for skill discovery, which is a significant new challenge. **To the best of our knowledge, this is the first work to fully automate the extraction of collaborative patterns among agents as subgroup skills from offline data.** We also note that the problem is different from (online) role-based MARL [Xu *et al.*, 2023; Zhou *et al.*, 2024]. They instead focus on partitioning agents into subdivisions that consist of agents with similar responsibilities (i.e., roles), sharing the same policy and thus homogeneous behaviors. Our goal is to learn multi-agent skills – a collective set of single-agent skills taken by a subgroup where agents could have distinct yet coordinated behaviors.

To be specific, we provide effective auto-encoder frameworks for extracting embeddings of subgroup coordination patterns from offline data as a codebook, where each code corresponds to a multi-agent skill and provides abstractions in both subgroup- and temporal-level. We propose two scheme designs for this purpose: VO-MASD-3D and VO-MASD-Hier. In VO-MASD-3D, three-dimensional codebooks are adopted, where each multi-agent skill code consists of several single-agent skill codes such that it can be used to represent subgroup behaviors. In contrast, VO-MASD-Hier employs a two-level codebook: the bottom codes encode individual behaviors, while each top code is aggregated from a set of bottom codes to encode subgroup behaviors. Further, to enable automatic grouping while forming temporal abstractions, we

co-train a grouping function with the proposed auto-encoder schemes. Using this function, agents can be dynamically grouped based on the environment state, and each subgroup can then be assigned a multi-agent skill of the corresponding size. Importantly, our algorithm is designed to work with multi-task data, such that the discovered skills can be utilized in multiple relevant tasks (without retraining). Empirical results on challenging StarCraft tasks [Samvelyan *et al.*, 2019; Ellis *et al.*, 2024] demonstrate the superiority of the discovered multi-agent skills using our algorithm even in previously unseen tasks, and show the great advantages brought by the use of skills in long-horizon multi-agent tasks characterized by sparse reward signals.

## 2 Background

**Dec-POMDP.** This work focuses on a fully cooperative multi-agent setting with only partial observation for each agent, which can be modeled as a decentralized partially observable markov decision process (Dec-POMDP) [Oliehoek *et al.*, 2016] and described with a tuple  $G = \langle n, I, S, O, F, A, \mu, P, R, \gamma \rangle$ . At a time step, each agent  $i \in I = \{1, \dots, n\}$  would obtain a local observation  $o^i \in O$  from the observation function  $F(s, i) : S \times I \rightarrow O$ , where  $s$  is the real state of the environment, and determine its action  $a^i \in A$ . This would lead to a state transition in the environment according to the function  $P(s' | s, \vec{a}) : S \times A^n \times S \rightarrow [0, 1]$  and all agents would receive a shared team reward  $r = R(s, \vec{a}) : S \times A^n \rightarrow \mathbb{R}$ . To mitigate the issue of partial observability, each agent  $i$  holds an action-observation history  $\tau_{t-1}^i = (o_1, a_1, \dots, o_{t-1}, a_{t-1})$  and decides on its action  $a_t^i$  based on a policy  $\pi^i(a_t^i | o_t^i, \tau_{t-1}^i)$ . The goal of MARL in a Dec-POMDP can be formally defined as  $\max_{\vec{\pi}} \mathbb{E}_{\mu, \vec{\pi}, P, R} [\sum_{t=0}^{\infty} \gamma^t r_t]$ , where  $\vec{\pi} = (\pi^1, \dots, \pi^n)$  and  $\mu(s_0) : S \rightarrow [0, 1]$  denotes the distribution of the initial state.

**CTDE.** The paradigm of centralized training with decentralized execution (CTDE) [Oliehoek *et al.*, 2008] is proposed for solving Dec-POMDP and has gained substantial attention. In this paradigm, agents learn their policies with access to global information (e.g., the state  $s$ ) during centralized training and only rely on their local action-observation histories for decentralized execution. Notably, the discovered skills with our algorithm can be easily integrated into the CTDE paradigm. We select MAPPO [Yu *et al.*, 2022b] as the base CTDE MARL algorithm throughout this work, as it has shown superior performance across various MARL benchmarks. It learns a decentralized actor  $\pi(a_t^i | o_t^i, \tau_{t-1}^i)$ , which is used for each agent  $i \in I$  to determine its action  $a_t^i$  based on its individual action-observation history  $(o_t^i, \tau_{t-1}^i)$ , and a centralized critic  $V(s_t)$ . Viewing the  $n$  agents as a whole, the critic function is trained as in a single-agent RL algorithm – PPO [Schulman *et al.*, 2017], while the actor (of each agent) is trained to maximize the advantage function defined with the team reward and centralized critic.

**Skill & Task Decomposition.** In single-agent scenarios, skills are used as temporal abstractions of an agent’s behaviors. This is inspired by the fact that complex tasks can usually be decomposed as a sequence of subtasks and each subtask can be handled with a corresponding subpolicy, i.e., a

skill. With skills, an agent learns a hierarchical policy, where the low-level part  $\pi_l(a | s, z)$  is the skill policy and the high-level part  $\pi_h(z | s)$  determines the skill selection. Each skill  $z \in \Omega^z$ , after being selected, will be executed for  $H$  time steps – a predefined subtask duration. However, in multi-agent scenarios, task decomposition occurs not just at the temporal level but also at the agent level, since the overall multi-agent task can be viewed as several subgroup tasks executed in parallel. Here, we define such a task decomposition:

**Definition 1.** *Given a cooperative multi-agent task  $\langle n, I, S, O, F, A, \mu, P, R, \gamma \rangle$ , at a time step, it can be decomposed into a set of  $m$  subtasks, each of which is solved by a subgroup of agents for  $H$  time steps and can be represented as a tuple  $\langle n_j, I_j, S, O, F, A, \mu, P, R_j, \gamma \rangle$ . Here,  $\sum_{j=1}^m n_j = n$ ,  $\cup_j I_j = I$ , and  $I_j \cap I_k = \emptyset$  ( $\forall j \neq k$ ).*

Certain subtasks may frequently occur, such as passing and cutting cooperation among two or three players in a football match, and their subpolicies, showing coordination patterns, can be extracted as multi-agent skills and transferred across similar tasks for reuse. In this work, we propose an algorithm for discovering such multi-agent skills  $Z \in \Omega^Z$  from multi-agent interaction data.

**Related Works.** In Appendix A of an extended version of this paper<sup>1</sup>, we provide a thorough review of research on applying skills in MARL. There are three main categories: MARL with single-agent skills [Lee *et al.*, 2020; Yang *et al.*, 2020; Chen *et al.*, 2023a; Chen *et al.*, 2023c], role-based MARL [Yang *et al.*, 2022; Xu *et al.*, 2023], and team skill discovery [Chen *et al.*, 2022; Chen *et al.*, 2023b]. As a summary, research on multi-agent skill discovery is still at an early stage of development, especially for the offline setting. Even without prelearned skills, when dealing with a complex multi-agent task, the agents would implicitly learn to decompose the overall task into several subtasks, assign a subgroup for each subtask, and develop a joint policy (i.e., multi-agent skill) within the subgroup to handle the corresponding subtask. Replacing primitive actions with single-agent skills or role policies could make such a learning process more efficient, as agents can assemble these higher-level abstractions to obtain the required subgroup policies more easily. As the first offline multi-agent skill discovery algorithm, our work takes one step further by directly identifying subgroups, which could change throughout a decision horizon, and extracting their coordination patterns as multi-agent skills. With these joint skills, the MARL process could be greatly simplified, since agents only need to select correct multi-agent skills without considering grouping with others or forming subgroup policies. Thus, multi-agent skills represent a more efficient form of knowledge discovery.

## 3 Proposed Approach

Variational Offline Multi-agent Skill Discovery (VO-MASD) aims to extract a finite set of multi-agent skills from given offline trajectories. Proposed for Computer Vision, VQ-VAE [van den Oord *et al.*, 2017; Chen *et al.*, 2024] provides a fundamental manner to learn discrete representations for com-

<sup>1</sup><https://arxiv.org/abs/2405.16386>

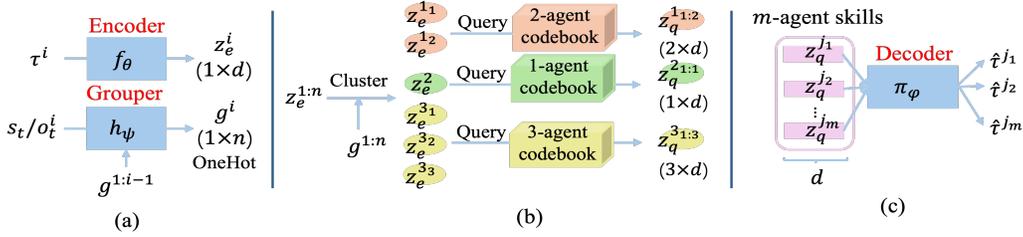


Figure 1: Multi-agent skill discovery based on a VQ-VAE with 3D codebooks.

plex, high-dimensional data. Besides the encoder and decoder as used in VAEs [Kingma and Welling, 2014], a codebook containing a finite set of codes, each of which is a latent representation of the data, is learned. In this case, VQ-VAE is a natural choice for skill discovery, with each code working as a skill embedding  $Z$ . Each  $Z$  would correspond to a skill policy  $\pi_i(\vec{a} \mid \vec{\tau}, Z)$  that leads to continuous multi-agent behaviors. In this section, we present two schemes of VO-MASD based on VQ-VAE by adopting novel codebook designs and involving an automatic grouping module. The challenge is to extract temporal-level abstractions (i.e., useful control sequences) and agent-level abstractions (i.e., multi-agent coordination) at the same time, **without using domain knowledge or task-specific reward signals**. In this way, VO-MASD can be applied to a mixture of multi-task data and the learned skills are generalizable to relevant tasks.

### 3.1 VO-MASD Based on 3D Codebooks

VQ-VAE typically adopts a 2D codebook  $[e_1, \dots, e_k] \in \mathbb{R}^{k \times d}$ , where  $e_i \in \mathbb{R}^{1 \times d}$  is a latent representation. However, in our case, there are three levels of abstractions: primitive actions  $\rightarrow$  single-agent skills  $\rightarrow$  multi-agent skills. As part of our novelty, we propose to use 3D codebooks within  $\mathbb{R}^{k \times m \times d}$  to represent a set of (i.e.,  $k$ )  $m$ -agent skills. Each code  $e_i = [e_{i,1}, \dots, e_{i,m}] \in \mathbb{R}^{m \times d}$  represents a multi-agent skill composed of  $m$  single-agent skills. Note that  $m$  ranges from 1 to  $n$  (i.e., the size of the team).

A straightforward approach to utilize such codebook design for skill discovery is repeatedly applying a VQ-VAE with an  $m$ -agent codebook to  $m$ -agent skill discovery, for  $m = 1, \dots, n$ . As some variational methods for single-agent skill discovery [Campos *et al.*, 2020; Ajay *et al.*, 2021], the objective for learning  $m$ -agent skills could be minimizing the reconstruction error of  $m$ -agent trajectory segments. Ideally, after training, each code can represent a coordination pattern among  $m$  agents and the code-conditioned decoder can be used as an  $m$ -agent skill policy. However, if there are no coordination involving  $m$  agents in the offline data, the effort to discover  $m$ -agent skills would be wasted. Also, in this way, the learning processes for skills involving different numbers of agents are independent and cannot benefit from each other.

In this case, we introduce a grouping function  $h_\psi$  that dynamically groups agents throughout an episode to identify existing coordination patterns in the offline data and unify the training of skills with different numbers of agents. The skill discovery process is illustrated as Figure 1. As shown in (a), at time step  $t$ , for each agent  $i$ , we encode its follow-

ing  $H$  time steps, i.e.,  $\tau^i = [o_t^i, a_t^i, \dots, o_{t+H-1}^i, a_{t+H-1}^i]$ , into a skill embeddings  $z_e^i$  using the encoder  $f_\theta$ . Also, each agent  $i$  selects its group based on the global state  $s_t$  and group choices of previous agents  $g^{1:i-1}$  using a grouping function  $h_\psi$ . There can be at most  $n$  groups, when all agents choose to use individual skills. Notably, both  $h_\psi$  and  $f_\theta$  are shared by all agents. Subsequently, in (b), the skill embeddings  $z_e^{1:n}$  from the encoder are first clustered based on the grouping result  $g^{1:n}$ : if  $m$  agents choose the same group (indicated by the one-hot output  $g^i$ ), they aim to form an  $m$ -agent coordination skill and their respective embeddings will be concatenated in the sequence of their agent indices, resulting in an  $m \times d$  joint embedding  $z_e^{j:1:m}$ . Then, as in VQ-VAE, the code that is the closest to  $z_e^{j:1:m}$  in the  $m$ -agent codebook (i.e.,  $z_q^{j:1:m}$ ) is queried to work as the skill code. Finally, in (c), a decoder  $\pi_\phi$  maps the skill code back to an  $m$ -agent trajectory segment, i.e.,  $\hat{\tau}^{j:1:m}$ . The training objective for subgroup  $j_{1:m}$  is:

$$L^{3D}(\tau^{j_{1:m}}) = - \sum_{l=0}^{H-1} \sum_{i=1}^m \log \pi_\phi(a_{t+l}^{j_i} \mid o_{t+l}^{j_i}, z_q^{j_i}) + \sum_{i=1}^m \left[ \|\text{sg}(z_e^{j_i}) - e^{j_i}\|_2^2 + \beta \|z_e^{j_i} - \text{sg}(e^{j_i})\|_2^2 \right] \quad (1)$$

As shown in Figure 1,  $z_e^{j_i} = f_\theta(\tau^{j_i})$ ,  $e^{j_{1:m}} = \arg \min_{e \in E_m} \|z_e^{j_{1:m}} - e\|_2$  ( $E_m$  denotes the  $m$ -agent codebook), and  $z_q^{j_i} = e^{j_i}$ .  $L^{3D}(\tau^{j_{1:m}})$  is an objective with respect to (w.r.t.)  $\theta, \phi, E_m$ . As in VQ-VAE, the first term in Eq (1) is a reconstruction loss of trajectory segments, and the last two terms move the skill codes (e.g.,  $e^{j_i}$ ) and encoder embeddings (e.g.,  $z_e^{j_i}$ ) towards each other, where  $\text{sg}$  represents the stop gradient operator. Through reconstructing  $m$ -agent ( $m \in \{1, \dots, n\}$ ) trajectory segments in an auto-encoder framework, representations of  $m$ -agent skills can be extracted as codes in the codebook. The overall objective for VO-MASD-3D is as below:

$$\min_{\theta, \phi, E_{1:n}} L^{3D} = \min_{\theta, \phi, E_{1:n}} \mathbb{E}_{\tau^{1:n} \sim \mathcal{D}_H} \sum_j L^{3D}(\tau^{j_{1:m}}) \quad (2)$$

Here,  $\mathcal{D}_H$  is a (multi-task) offline dataset with trajectories segmented every  $H$  time steps; each  $n$ -agent trajectory segment is partitioned into subgroups (e.g.,  $j_{1:m}$ ) based on the grouping function  $h_\psi$ . Note that, unlike  $f_\theta$ ,  $E_{1:n}$ , and  $\pi_\phi$ ,  $h_\psi$  cannot be trained in an end-to-end manner by minimizing Eq (2), since its output  $g^{1:n}$  are used for clustering which is not an differentiable operation. Thus, we choose to optimize  $h_\psi$  with MAPPO, where each agent  $i$  takes an action  $g^i$  to maximize the global return  $-L^{3D}$ . In this way, all modules in Figure 1 are effectively updated with a common objective.

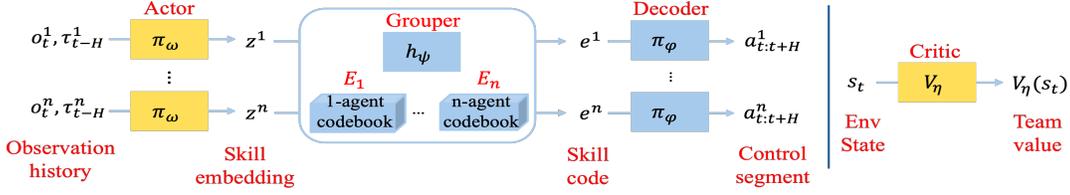


Figure 2: Utilizing discovered skills for downstream CTDE MARL. Compared to standard CTDE MARL, individual actions  $a^{1:n}$  are replaced with skill embeddings  $z^{1:n}$  as the actor’s output. These embeddings are then translated into skill codes and control segments using the pretrained VO-MASD components, as shown in Figure 1. Thus, only the individual actor  $\pi_\omega$  and centralized critic  $V_\eta$  need to be trained.

---

**Algorithm 1** MAPPO with learned skills
 

---

Input:  $\pi_\omega, V_\eta, \pi_\phi, h_\psi, E_{1:n}, \text{Env}$   
 Initialize  $\pi_\omega, V_\eta$   
**while** not converged **do**  
   Buffer  $\leftarrow \emptyset$   
   **for**  $b = 1 \dots B$  **do**  
     Initialize  $\tau_{-H}^{1:n}, \text{Traj} \leftarrow \emptyset, \tilde{r} \leftarrow 0$   
     **for**  $t = 0 \dots T$  **do**  
       **if**  $t \% H == 0$  **then**  
          $z_t^i, \tau_t^i \leftarrow \pi_\omega(o_t^i, \tau_{t-H}^i), i = 1 \dots n$   
         Get  $e^{1:n}$  based on  $z_t^{1:n}$  using  $h_\psi$  and  $E_{1:n}$ , following Fig 1 (b)  
         Add  $(\tilde{r}, s_t, o_t^{1:n}, \tau_{t-H}^{1:n}, z_t^{1:n})$  to Traj  
          $\tilde{r} \leftarrow 0$   
       **end if**  
        $a_t^i \leftarrow \pi_\phi(o_t^i | e^i), i = 1 \dots n$   
        $r_t, s_{t+1}, o_{t+1}^{1:n} \leftarrow \text{Env}(a_t^{1:n}), \tilde{r} += r_t$   
     **end for**  
     Buffer  $\leftarrow \text{Buffer} \cup \text{Traj}$   
   **end for**  
   Train  $\pi_\omega, V_\eta$  based on Buffer using MAPPO  
**end while**

---

This framework is advantageous: (1) the training of skills with different number of agents can facilitate each other, as they share all modules but the codebook; (2) the modeling of temporal- and agent-level abstractions within multi-agent skills are decoupled as training the decoder to reconstruct single-agent trajectories and training the grouper for automatic grouping; (3) the grouper is trained to form subgroups only when it enhances the overall pattern extraction objective (i.e., Eq (2)), ensuring that each subgroup, along with its policy, corresponds to a genuine coordination pattern.

We illustrate how to utilize the discovered skills in CTDE MARL in Figure 2. Also, Alg 1 outlines the detailed training process for a decentralized actor  $\pi_\omega$  and centralized critic  $V_\eta$  using MAPPO in a multi-agent task Env, leveraging the pretrained components  $h_\psi, E_{1:n}$ , and  $\pi_\phi$ . In particular, every  $H$  time steps, the actor produces a skill embedding  $z^i \in \mathbb{R}^{1 \times d}$  for each agent  $i$ .  $z^{1:n}$  are mapped to the closest multi-agent skill codes  $e^{1:n}$  using the grouper  $h_\psi$  and codebook  $E_{1:n}$ , following Figure 1 (b). Then, for the next  $H$  time steps, each agent  $i$  interacts with Env using corresponding  $\pi_\phi(a^i | s^i, e^i)$ , i.e., the decoder working as the skill policy. Based on the interaction transitions, i.e.,  $\{(s_t, o_t^{1:n}, \tau_{t-H}^{1:n}, z^{1:n}, \tilde{r}_t, s_{t+H})\}$ ,

$\pi_\omega$  and  $V_\eta$  can be trained with MAPPO, where  $\tau_{t-H}^{1:n}$  are the skill – observation (i.e.,  $z - o$ ) history,  $z^{1:n}$  can be viewed as (high-level) actions, and  $\tilde{r}_t = \sum_{l=t}^{t+H-1} r_l$  is the skill reward.

Besides the approach illustrated in Figure 2, we propose two alternative methods for mapping  $z^{1:n}$  to  $e^{1:n}$  in Appendix B. All the three methods assign each multi-agent ( $m \times d$ ) subgroup, such that the collaboration pattern encoded in the multi-agent code can be utilized. Alternatively, each ( $m \times d$ ) code can be decomposed into a set of ( $m$ ) single-agent codes. Each agent  $i$  could then independently select its skill from the set of single-agent codes. In particular, the single-agent skill code closest to the agent’s actor output  $z^i$  would be selected. We denote this algorithm as ‘VO-MASD-Mixed’. Later, we empirically compare these four skill assignment manners. Importantly, no matter which manner we choose, we only need to train a decentralized actor and a centralized critic for (online) MARL, without any additional learning effort beyond standard CTDE MARL approaches.

### 3.2 VO-MASD Based on a Hierarchical Codebook

Here, we present VO-MASD-Hier, which is an alternative to VO-MASD-3D and adopts a hierarchical codebook as in [Razavi *et al.*, 2019]. Although [Razavi *et al.*, 2019] is originally proposed for image generation, its top and bottom codebooks perfectly echo the two-level structure of multi-agent and single-agent skill embeddings. Thus, we propose to learn top and bottom codebooks as agent- and temporal-level abstractions, respectively, for multi-agent skill discovery. The overall framework of VO-MASD-Hier is shown as Figure 3. It contains a two-level codebook, i.e.,  $E_{\text{top}}, E_{\text{btm}}$ . VO-MASD-Hier does not need to learn  $n$  codebooks (i.e.,  $E_{1:n}$ ) as in VO-MASD-3D, while VO-MASD-3D can potentially make better use of domain knowledge. For example, if the subgroup scale (e.g.,  $m$ ) is known in advance, VO-MASD-3D only needs to learn  $E_1$  and  $E_m$ , while VO-MASD-Hier cannot specify the number of agents within a multi-agent skill.

The skill discovery process is detailed as follows. In Figure 3 (a), the embedding process of each individual trajectory segment  $\tau^i$  ( $i = 1, \dots, n$ ) is the same as the one of VO-MASD-3D (i.e., Figure 1 (a)). Subsequently, in (b), the skill embeddings  $z_{\text{btm}}^{1:n}$  are clustered based on the output from the grouping function, i.e.,  $g^{1:n}$ , and then embeddings within the same subgroup (e.g.,  $z_{\text{btm}}^{1:2}$ ) are aggregated to a higher-level representation (e.g.,  $z_{\text{top}}^1$ ) which is then used to query a top code (e.g.,  $q_{\text{top}}^1$ ). Note that the aggregator  $f_{\theta_{\text{top}}}$  uses a

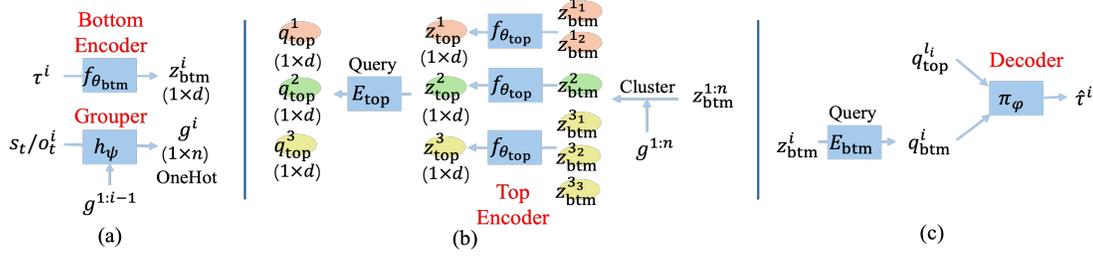


Figure 3: Multi-agent skill discovery based on a VQ-VAE with a hierarchical codebook.

multi-head attention module [Vaswani *et al.*, 2017] to process varied-length inputs and so can be shared by all subgroups (of varied sizes). In Figure 3 (c), for each agent  $i$ , a bottom code  $q_{\text{btm}}^i$  is assigned based on its individual skill embedding  $z_{\text{btm}}^i$ . Finally,  $q_{\text{btm}}^i$  and  $q_{\text{top}}^i$ , involving temporal- and agent-level abstractions respectively, are used to decode/reconstruct  $\tau^i$ . The overall objective is  $\min_{\theta_{\text{top, btm}}, E_{\text{top, btm}}, \pi_{\phi}} \mathbb{E}_{\tau^{1:n} \sim \mathcal{D}_H} L^{\text{Hier}}(\tau^{1:n})$ :

$$\begin{aligned}
 L^{\text{Hier}}(\tau^{1:n}) = & - \sum_{j=0}^{H-1} \sum_{i=1}^n \log \pi_{\phi}(a_{t+j}^i \mid o_{t+j}^i, q_{\text{btm}}^i, q_{\text{top}}^i) \\
 & + \sum_{i=1}^n \left[ \|\text{sg}(z_{\text{btm}}^i) - q_{\text{btm}}^i\|_2^2 + \beta \|z_{\text{btm}}^i - \text{sg}(q_{\text{btm}}^i)\|_2^2 \right] \\
 & + \sum_{i=1}^n \left[ \|\text{sg}(z_{\text{top}}^i) - q_{\text{top}}^i\|_2^2 + \beta \|z_{\text{top}}^i - \text{sg}(q_{\text{top}}^i)\|_2^2 \right]
 \end{aligned} \quad (3)$$

This loss function is similar with Eq (1), i.e., to reconstruct the input multi-agent trajectory segment, and move the codes and corresponding skill embeddings towards each other.

As for the design intuition, considering the first term in Eq (3), the gradient w.r.t. the bottom code  $q_{\text{btm}}^i$  only comes from reconstructing agent  $i$ 's individual skill trajectory  $\tau^i$ . However, the gradient w.r.t. the top code  $q_{\text{top}}^i$  is derived from reconstructing the multi-agent skill trajectories of the subgroup  $l_i$  that  $i$  belongs to, since each agent  $j$  in  $l_i$  would adopt  $q_{\text{top}}^i$  as the decoder condition to reconstruct corresponding  $\tau^j$ . This reflects that the top and bottom codebooks are trained to embed subgroup- and temporal-level abstractions, respectively. Notably, both VO-MASD-3D and VO-MASD-Hier follow the inductive bias: primitive actions  $\rightarrow$  single-agent skills  $\rightarrow$  multi-agent skills. That is, each single-agent skill code is trained to embed an individual trajectory and each multi-agent skill code is a composition of single-agent ones. In VO-MASD-3D, each  $(m \times d)$  multi-agent code contains a set  $(m)$  of  $(1 \times d)$  single-agent codes; while for VO-MASD-Hier, each multi-agent embedding  $z_{\text{top}}$  is obtained through aggregating individual skill embeddings  $z_{\text{btm}}$  from the same subgroup using an attention mechanism, as in Figure 3 (b).

To utilize the discovered skills in downstream online MARL, Alg 1 can be applied to VO-MASD-Hier by replacing the process in Figure 1 (b)(c) with corresponding ones in Figure 3 (b)(c). Specifically, a decentralized actor  $\pi_{\omega}$  gives out skill embeddings  $z_{\text{btm}}^{1:n}$  every  $H$  time steps.  $h_{\psi}$ ,  $E_{\text{top, btm}}$ ,  $f_{\theta_{\text{top}}}$ , and  $\pi_{\phi}$  are fixed during online MARL, transforming  $z_{\text{btm}}^{1:n}$  to multi-agent and single-agent skill codes, i.e.,  $q_{\text{top}}^{1:n}$  and  $q_{\text{btm}}^{1:n}$ .

The decoder is then used to produce skill trajectories of length  $H$ , according to  $\pi_{\phi}(a_t^i \mid o_t^i, q_{\text{top}}^i, q_{\text{btm}}^i)$ .

## 4 Evaluation and Main Results

Experiments are conducted on the StarCraft multi-agent challenge (SMAC) [Samvelyan *et al.*, 2019] – a commonly-used benchmark for cooperative MARL. Following ODIS [Zhang *et al.*, 2023], we adopt two SMAC task sets to test the discovered multi-task multi-agent skills. In each task set, agents control some units like marines, medivacs, and marauders, but the number of controllable agents or enemies varies across tasks in a task set. We refer to the two task sets as ‘marine’ and ‘MMMs’, which evaluate algorithm performance in scenarios with homogeneous and heterogeneous agents, respectively, and are detailed in Appendix C. For each task set, we discover skills from offline trajectories of source tasks, and then apply these skills to each task in the task set (including source and unseen tasks) for online MARL. The offline trajectories are collected with well-trained MAPPO agents, which can be viewed as expert demonstrations **for the source tasks**, and are included in our released code folder.

Next, we show evaluation results on several aspects. (1) In Section 4.1, we compare skills discovered using different algorithms on two SMAC task sets, based on their utility for downstream online MARL, to demonstrate the superiority of the multi-agent skills discovered by our methods. (2) In Section 4.2, we test the algorithms on a task set from another benchmark – SMACv2 [Ellis *et al.*, 2024], which features stochastic environments. (3) In Section 4.3, we show that, for MARL tasks with sparse reward signals, hierarchical learning with skills discovered using our methods can significantly outperform usual MARL algorithms. Notably, the skills are from relevant but different tasks. (4) In Appendix F of the extended version, we provide visualizations of the skills, evaluate our algorithms on offline datasets of varying quality, and present an ablation study to support our design choices.

### 4.1 Utility of Discovered Skills for Online MARL

The first group of results is shown as Figure 4, where ‘3d’, ‘hier’, ‘mixed’, ‘single’, and ‘odis’ refer to VO-MASD-3D, VO-MASD-Hier, VO-MASD-Mixed, VO-MASD-Single, and ODIS, respectively. As mentioned in Appendix A, ODIS is the only existing algorithm for discovering multi-agent temporal abstractions from offline multi-task data, and is a representative of role-based MARL. Notably, ODIS has

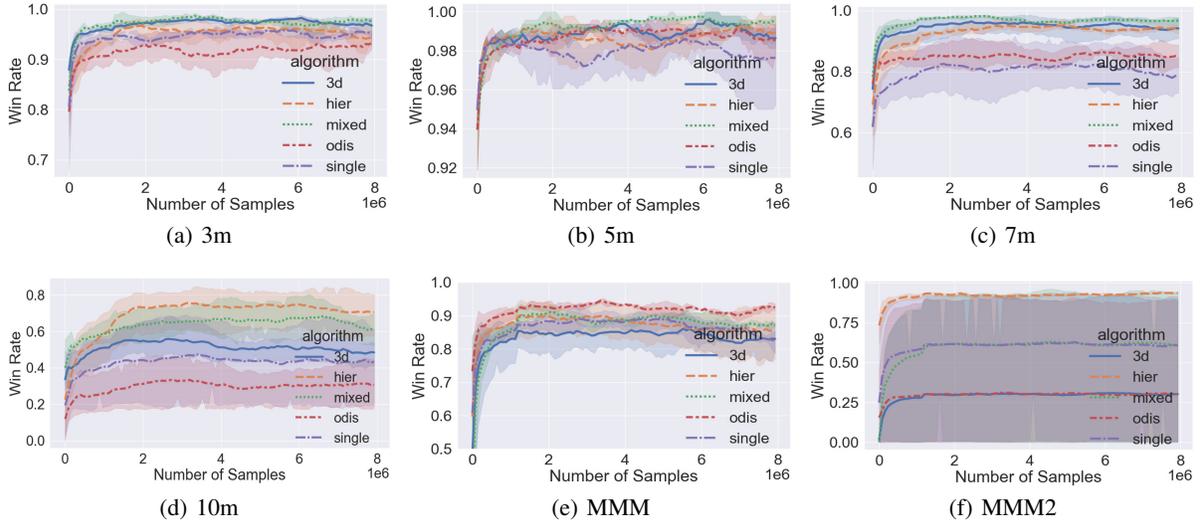


Figure 4: Evaluation of effectiveness of the discovered skills using different algorithms for online MARL.

demonstrated superior performance compared to direct imitation learning from the offline dataset, MADT [Meng *et al.*, 2021] (an offline MARL algorithm using pretraining), and UPDeT [Hu *et al.*, 2021] (a SOTA multi-task MARL method), making it a strong baseline for comparison<sup>2</sup>. VO-MASD-Single represents the other main branch of hierarchical MARL – learning a set of single-agent skills and collaboratively utilizing them for MARL, which is realized through removing  $E_{\text{top}}$  and  $f_{\theta_{\text{top}}}$  in VO-MASD-Hier (i.e., Figure 3). VO-MASD-Single discovers and utilizes single-agent skills, while VO-MASD-Mixed discovers multi-agent skills as in VO-MASD-3D but employs the learned skills as single-agent ones, which is detailed in the last paragraph of Section 3.1. Thus, the baselines include SOTA algorithms in this field and two variations of our algorithms to respectively show the effect of **discovering** and **utilizing** skills as multi-agent units.

Skills (of length 5) discovered from source tasks are applied to both source and unseen tasks for online MARL using Alg 1. In marine, 3m and 5m are source tasks; while in MMMs, MMM is the source task. We believe that the learning performance on unseen tasks with higher-complexity is the best way to testify the utility and generality of skills discovered with different algorithms. In particular, we track the change of win rates as the number of training samples increases, presenting the mean and 95% confidence intervals as solid lines and shaded areas, respectively. Several conclusions can be drawn from Figure 4. **(1)** ODIS and VO-MASD-Single, which represent two main existing approaches of applying skills in MARL, exhibit inferior performance compared to the others, especially in unseen tasks. This underscores the importance of **discovering** coordination patterns as multi-agent skills, which can significantly enhance performance and generality in new multi-agent tasks. **(2)** In

marine tasks, the performances of VO-MASD-3D and VO-MASD-Hier are comparable, with VO-MASD-Hier performing better in 10m and VO-MASD-3D excelling in the others. However, VO-MASD-3D’s performance deteriorates in MMMs, suggesting that its design may not be well-suited for heterogeneous-agent tasks like MMM and MMM2 and indicating a potential future research direction for improvement.

**(3)** VO-MASD-Mixed follows the same skill discovery process as VO-MASD-3D but adopts the skills as single-agent ones. Surprisingly, VO-MASD-Mixed consistently outperforms VO-MASD-3D. While VO-MASD-3D utilizes fixed combinations of single-agent ( $1 \times d$ ) codes from the discovery stage, VO-MASD-Mixed explores all possible combinations of these ( $1 \times d$ ) codes to achieve a higher return, which explains its better performance. However, in the most challenging settings (i.e., 10m and MMM2), VO-MASD-Hier demonstrates better results, showing the potential benefit of **utilizing** discovered multi-agent skills as complete units. **(4)** The evaluation on MMM2 – a super-hard task setting [Samvelyan *et al.*, 2019], demonstrates the superiority of VO-MASD-Hier over other algorithms. All algorithms, except for VO-MASD-Hier, exhibit large variance across different runs.

## 4.2 Evaluation Results on SMACv2

SMACv2 [Ellis *et al.*, 2024] uses procedural content generation [Risi and Togelius, 2020] to address SMAC’s lack of stochasticity. In SMACv2, for each episode, team compositions and agent start positions would be generated randomly. Thus, it is no longer sufficient for agents to repeat a fixed action sequence, but they must learn to coordinate across a diverse range of scenarios. Specifically, we select the Terran task, which involves teams composed of three types of units: marines, marauders, and medivacs. As in SMAC tasks, we train an MAPPO policy as the offline data collector. However, the learned policy can only achieve a win rate around 55% on the source tasks upon convergence, highlighting the difficulty of SMACv2. For different tasks in this task set, we

<sup>2</sup>In ODIS, the discovered skills are used for offline MARL. For fair comparisons, we instead integrate skills from ODIS with online MARL, as in VO-MASD-3D and VO-MASD-Hier.

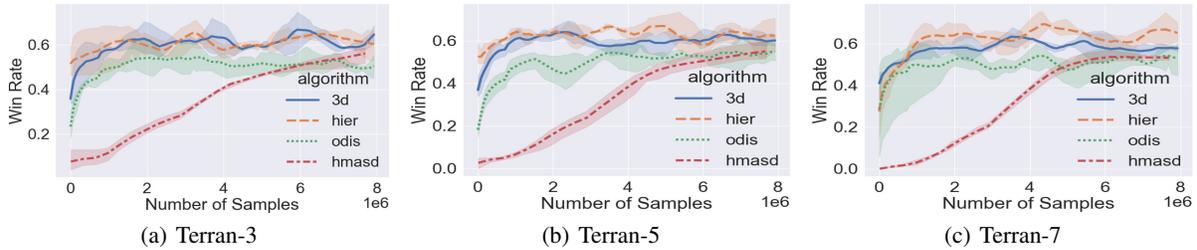


Figure 5: Evaluation results on SMACv2. We compare the performance of online MARL using skills discovered with our methods and ODIS. As an additional baseline, we also include HMASD, an online hierarchical MARL method that discovers task-specific skills during training.

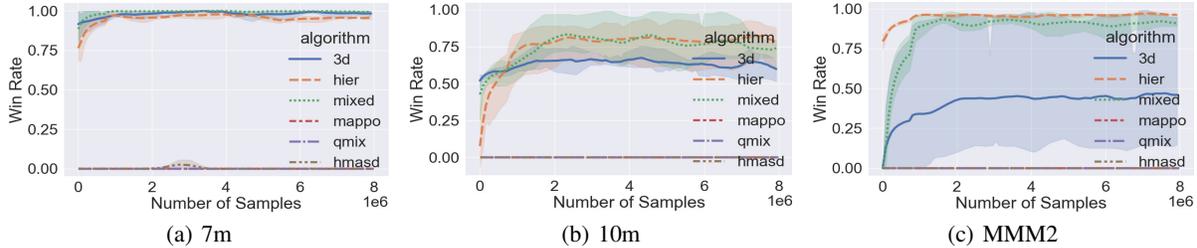


Figure 6: The effectiveness of discovered skills in online MARL with sparse reward signals.

vary the team size, selecting the less challenging Terran-3 and Terran-5 as source tasks and Terran-7 as the target task.

Our methods consistently outperform ODIS, discovering more effective skills for downstream MARL. We also adopt HMASD [Yang *et al.*, 2023] as a baseline. HMASD is a SOTA **online** hierarchical MARL algorithm that discovers skills while forming a hierarchical policy for a specific task. Notably, for Terran-7, the skills used by our algorithms are discovered from Terran-3 and Terran-5 and remain fixed during hierarchical policy learning, whereas HMASD develops specific skills for Terran-7. Despite this, our algorithm still achieves superior performance. As discussed in Appendix A, HMASD discovers only single-agent and team skills, rather than multi-agent skills for subgroups of varying sizes. Team skills can be less flexible to use, particularly when the team composition randomly changes (as in SMACv2).

### 4.3 Performance in MARL with Sparse Rewards

With pretrained skills, only a high-level policy  $\pi_\omega$  for skill selection needs to be trained for downstream task learning, as detailed in Alg 1, and the decision horizon of  $\pi_\omega$  is reduced to the original one divided by the skill length  $H$ . Thus, learning with skills (i.e., hierarchical learning) is particularly advantageous for long-horizon tasks with sparse and delayed reward signals. To testify this, we modify the reward setups of the unseen tasks: 7m, 10m, MMM2, to be sparse, where agents receive a reward of 20 only upon eliminating all enemies; otherwise, they receive a reward 0. These three tasks, with maximum episode horizons of 110, 120, and 180 respectively, are particularly challenging.

We apply two online MARL algorithms: MAPPO [Yu *et al.*, 2022a] and QMIX [Rashid *et al.*, 2018], to these tasks, and they consistently fail with all-zero win rates. Al-

though they have been proposed for years, MAPPO and QMIX remain the most robust algorithms in online MARL, as verified by extensive empirical studies [Yu *et al.*, 2022a; Hu *et al.*, 2023]. In contrast, with skills discovered using our algorithms: VO-MASD-3D, VO-MASD-Mixed, VO-MASD-Hier, the performance can be greatly improved, as shown in Figure 6. Note that (1) skills are discovered from source tasks (rather than 7m, 10m, or MMM2) and (2) only sparse rewards are adopted for downstream online MARL. This highlights the effectiveness of hierarchical MARL when employing the multi-agent, multi-task skills discovered by our algorithms. As in Figure 4, VO-MASD-Hier achieves the best overall performance, followed by VO-MASD-Mixed. Further, we compare our methods with HMASD, which discovers skills through interaction with the environment and has proven effective in sparse reward settings. However, HMASD fails in all three tasks, highlighting the superiority of the skills learned with our methods, even though they are discovered from offline data of different tasks.

## 5 Conclusion

In this work, we propose novel algorithms for discovering coordination patterns among agents as multi-agent skills from offline multi-task data. The key challenge lies in abstracting agents’ behaviors at both the temporal and agent levels in a fully automatic manner. We address this challenge by developing novel encoder-decoder architectures and co-training the encoder-decoder with a grouping function that dynamically groups agents. Empirical results demonstrate that multi-agent skills discovered using our methods significantly enhance learning in downstream MARL tasks, particularly in scenarios with sparse reward signals.

## Ethical Statement

There are no ethical issues.

## References

- [Ajay *et al.*, 2021] Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. OPAL: offline primitive discovery for accelerating offline reinforcement learning. In *ICLR*. OpenReview.net, 2021.
- [Campos *et al.*, 2020] Victor Campos, Alexander Trott, Caiming Xiong, Richard Socher, Xavier Giró-i-Nieto, and Jordi Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *ICML*, volume 119, pages 1317–1327. PMLR, 2020.
- [Chen *et al.*, 2022] Jiayu Chen, Jingdi Chen, Tian Lan, and Vaneet Aggarwal. Scalable multi-agent covering option discovery based on kronecker graphs. In *NeurIPS*, 2022.
- [Chen *et al.*, 2023a] Jiayu Chen, Vaneet Aggarwal, and Tian Lan. A unified algorithm framework for unsupervised discovery of skills based on determinantal point process. In *NeurIPS*, 2023.
- [Chen *et al.*, 2023b] Jiayu Chen, Marina Haliem, Tian Lan, and Vaneet Aggarwal. Multi-agent deep covering option discovery, 2023.
- [Chen *et al.*, 2023c] Jiayu Chen, Tian Lan, and Vaneet Aggarwal. Hierarchical deep counterfactual regret minimization. *CoRR*, abs/2305.17327, 2023.
- [Chen *et al.*, 2024] Jiayu Chen, Bhargav Ganguly, Yang Xu, Yongsheng Mei, Tian Lan, and Vaneet Aggarwal. Deep generative models for offline policy learning: Tutorial, survey, and perspectives on future directions. *Transactions on Machine Learning Research*, 2024. Survey Certification.
- [Ellis *et al.*, 2024] Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob Foerster, and Shimon Whiteson. Smacv2: An improved benchmark for cooperative multi-agent reinforcement learning. *NeurIPS*, 36, 2024.
- [Eysenbach *et al.*, 2019] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *ICLR*. OpenReview.net, 2019.
- [Hu *et al.*, 2021] Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. Updet: Universal multi-agent reinforcement learning via policy decoupling with transformers. *CoRR*, abs/2101.08001, 2021.
- [Hu *et al.*, 2023] Jian Hu, Siying Wang, Siyang Jiang, and Musk Wang. Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning. In *The Second Blogpost Track at ICLR 2023*, 2023.
- [Kingma and Welling, 2014] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.
- [Lee *et al.*, 2020] Youngwoon Lee, Jingyun Yang, and Joseph J. Lim. Learning to coordinate manipulation skills via skill behavior diversification. In *ICLR*. OpenReview.net, 2020.
- [Meng *et al.*, 2021] Linghui Meng, Muning Wen, Yaodong Yang, Chenyang Le, Xiyun Li, Weinan Zhang, Ying Wen, Haifeng Zhang, Jun Wang, and Bo Xu. Offline pre-trained multi-agent decision transformer: One big sequence model tackles all SMAC tasks. *CoRR*, abs/2112.02845, 2021.
- [Oliehoek *et al.*, 2008] Frans A. Oliehoek, Matthijs T. J. Spaan, and Nikos Vlassis. Optimal and approximate q-value functions for decentralized pomdps. *JAIR*, 32:289–353, 2008.
- [Oliehoek *et al.*, 2016] Frans A Oliehoek, Christopher Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.
- [Rashid *et al.*, 2018] Tabish Rashid, Mikayel Samvelyan, Christian Schröder de Witt, Gregory Farquhar, Jakob N. Foerster, and Shimon Whiteson. QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*, volume 80, pages 4292–4301. PMLR, 2018.
- [Razavi *et al.*, 2019] Ali Razavi, Aäron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. In *NeurIPS*, pages 14837–14847, 2019.
- [Risi and Togelius, 2020] Sebastian Risi and Julian Togelius. Increasing generality in machine learning through procedural content generation. *Nature Machine Intelligence*, 2(8):428–436, 2020.
- [Sachdeva *et al.*, 2021] Enna Sachdeva, Shauharda Khadka, Somdeb Majumdar, and Kagan Tumer. Maedys: multi-agent evolution via dynamic skill selection. In *Genetic and Evolutionary Computation Conference*, pages 163–171. ACM, 2021.
- [Samvelyan *et al.*, 2019] 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 *AAMAS*, pages 2186–2188. IFAAMAS/ACM, 2019.
- [Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.
- [van den Oord *et al.*, 2017] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *NeurIPS*, pages 6306–6315, 2017.
- [Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.
- [Xu *et al.*, 2023] Zhiwei Xu, Yunpeng Bai, Bin Zhang, Dapeng Li, and Guoliang Fan. HAVEN: hierarchical cooperative multi-agent reinforcement learning with dual coordination mechanism. In *AAAI Conference on Artificial Intelligence*, pages 11735–11743. AAAI Press, 2023.

- [Yang *et al.*, 2020] Jiachen Yang, Igor Borovikov, and Hongyuan Zha. Hierarchical cooperative multi-agent reinforcement learning with skill discovery. In *AAMAS*, pages 1566–1574. IFAAMAS/ACM, 2020.
- [Yang *et al.*, 2022] Mingyu Yang, Jian Zhao, Xunhan Hu, Wengang Zhou, Jiangcheng Zhu, and Houqiang Li. LDSA: learning dynamic subtask assignment in cooperative multi-agent reinforcement learning. In *NeurIPS*, 2022.
- [Yang *et al.*, 2023] Mingyu Yang, Yaodong Yang, Zhenbo Lu, Wengang Zhou, and Houqiang Li. Hierarchical multi-agent skill discovery. In *NeurIPS*, 2023.
- [Yu *et al.*, 2022a] Chao Yu, Akash Velu, Eugene Vinytsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *NeurIPS*, 35:24611–24624, 2022.
- [Yu *et al.*, 2022b] Chao Yu, Akash Velu, Eugene Vinytsky, Jiaxuan Gao, Yu Wang, Alexandre M. Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *NeurIPS*, 2022.
- [Zhang *et al.*, 2023] 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 *ICLR*. OpenReview.net, 2023.
- [Zhou *et al.*, 2024] Guangchong Zhou, Zhiwei Xu, Bin Zhang, Dapeng Li, Zeren Zhang, and Guoliang Fan. Constructing informative subtask representations for multi-agent coordination, 2024.