SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

August 2024, Vol. 67, Iss. 8, 182101:1–182101:13
https://doi.org/10.1007/s11432-023-3862-1

Multi-agent policy transfer via task relationship modeling

Rongjun QIN[1,3], Feng CHEN[1], Tonghan WANG[2], Lei YUAN[1,3], Xiaoran WU[4], Yipeng KANG[5], Zongzhang ZHANG[1], Chongjie ZHANG[5] & Yang YU[1,3*]

[1]*National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing* 210000*, China;*
[2]*School of Engineering and Applied Sciences, Harvard University, Cambridge* MA 02138*, USA;*
[3]*Polixir Technologies, Nanjing* 211106*, China;*
[4]*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084*, China;*
[5]*Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing* 100084*, China*

Received 27 March 2023/Accepted 29 June 2023/Published online 22 July 2024

Abstract Team adaptation to new cooperative tasks is a hallmark of human intelligence, which has yet to be fully realized in learning agents. Previous studies on multi-agent transfer learning have accommodated teams of different sizes but heavily relied on the generalization ability of neural networks for adapting to unseen tasks. We posit that the relationship among tasks provides key information for policy adaptation. We utilize this relationship for efficient transfer by attempting to discover and exploit the knowledge among tasks from different teams, proposing to learn an effect-based task representation as a common latent space among tasks, and using it to build an alternatively fixed training scheme. Herein, we demonstrate that task representation can capture the relationship among teams and generalize to unseen tasks. Thus, the proposed method helps transfer the learned cooperation knowledge to new tasks after training on a few source tasks. Furthermore, the learned transferred policies help solve tasks that are difficult to learn from scratch.

Keywords multi-agent reinforcement learning, cooperative transfer learning, task relationship modeling, multi-agent policy reuse, multi-agent multi-task learning

1 Introduction

Human cooperation is characterized by resiliency to unforeseen changes and intentional adaptation to new tasks [1]. These adaptability and transferability of cooperative skills are a hallmark of human intelligence. Computationally, multi-agent reinforcement learning (MARL) [2] is a vital methodology for replicating human cooperation. Although recent MARL research has made prominent progress in various aspects of cooperation, such as policy decentralization [3–7], communication [8, 9], and organization [10–12], realizing the ability of group knowledge transfer has remained as an open question.

Compared with single-agent knowledge reuse [13], a unique challenge faced by multi-agent transfer learning is the varying size of agent groups. The number of agents and the length of observation inputs in unseen tasks may differ from those in source tasks. To solve this problem, existing multi-agent transfer learning approaches build population-invariant [14] and input-length-invariant [15] learning structures using graph neural networks [16] and attention mechanisms like transformers [17, 18]. Although these methods efficiently handle varying populations and input lengths, their ability to transfer knowledge to unseen tasks mainly depends on the inherent generalization ability of neural networks. The relationship among tasks in MARL has not been fully exploited for a more efficient transfer.

To make up for this shortage, we present herein the discovery and the utilization of common structures in multi-agent tasks and propose the multi-agent transfer reinforcement learning by modeling the task relationship (MATTAR). In this learning framework, we capture the common task structure by modeling the similarity among the transition and reward functions of different tasks. We specifically train a forward model for all source tasks to predict the observation, state, and reward at the next timestep

* Corresponding author (email: yuy@nju.edu.cn)

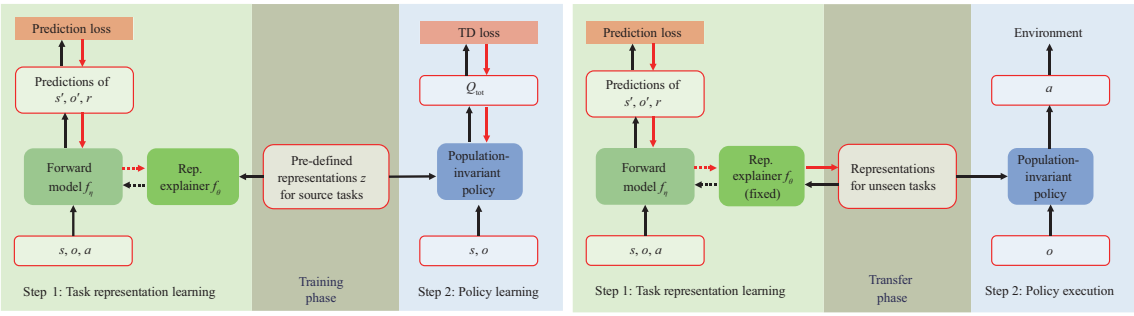

Figure 1 Transfer learning scheme of our method. The black arrows depict the data flow direction. The red ones show the gradient flow direction. The dashed arrows illustrate the flow between the hypernetwork and the generated network.

given the current observation, state, and actions. The challenge is how to embody the similarity and the difference among tasks in this forward model. We introduce the difference by giving each source task a unique representation. We model the similarity by generating the forward model parameters via a shared hypernetwork, which is herein referred to as the representation explainer.

Moreover, to learn a well-formed representation space that encodes the task relationship, we propose an alternative-fixed training method to learn the task representation and the representation explainer. During training, the source task representations are pre-defined and fixed as mutual orthogonal vectors. The representation explainer is learned by optimizing the forward model prediction loss on all source tasks. When facing an unseen task, we fix the representation explainer and backpropagate gradients through a fixed forward model to learn the new task representation by a few samples. We also design a population-invariant policy network conditioned on the learned task representation. During policy training, all source task representations are fixed, and the policy is updated to maximize the expected return over all source tasks. For an unseen task, we obtain the transferred policy by simply inserting a new task representation into the learned policy network.

On the SMAC [19] and MPE [3] benchmarks, we empirically show that the learned knowledge from the source tasks can be transferred to a series of unseen tasks with high success rates. We also pinpoint the other advantages of our method. First, when compared with learning from scratch, fine-tuning the transferred policy on unseen tasks results in a better performance. This indicates that the task representation and the pre-trained policy network provide a good initialization point. Second, training on multiple source tasks results in a better performance compared with training on them individually and other multi-task learning methods, showing that MATTAR also provides a method for multi-agent multi-task learning. Finally, although not designed for this goal, our structure yields comparable performance to single-task learning algorithms when trained on single tasks.

2 Method

In this study, we focused on the knowledge transfer among fully cooperative multi-agent tasks that can be modeled as a Dec-POMDP [20] comprising a tuple $G = \langle I, S, A, P, R, \Omega, O, n, \gamma \rangle$, where I is the finite set of n agents, $s \in S$ is the true environment state, and $\gamma \in [0, 1)$ is the discount factor. At each timestep, each agent i receives an observation $o_i \in \Omega$ drawn according to the observation function $O(s, i)$ and selects an action $a_i \in A$. Individual actions then form a joint action $\boldsymbol{a} \in A^n$ that leads to the next state s' according to the transition function $P(s'|s, \boldsymbol{a})$ and a reward $r = R(s, \boldsymbol{a})$ shared by all agents. Each agent has a local action-observation history $\tau_i \in T \equiv (\Omega \times A)^* \times \Omega$. Agents learn to collectively maximize the global action-value function $Q_{\text{tot}}(s, \boldsymbol{a}) = \mathbb{E}_{s_{0:\infty}, a_{0:\infty}}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \boldsymbol{a}_t) | s_0 = s, \boldsymbol{a}_0 = \boldsymbol{a}]$ (a little notation abuse: the subscript here for s and $\boldsymbol{a}$ indicates the timestep, while that for observation and action elsewhere in this paper indicates the agent index).

Our framework first trains on several source tasks $\{\mathcal{S}_i\}$ and then transfers the learned cooperative knowledge to unseen tasks $\{\mathcal{T}_j\}$. As shown in Figure 1, our learning framework achieves this by designing modules for both task representation and policy learning. The subsequent sections will introduce the design of the representation learning module and its learning scheme in different phases and describe policy learning in detail, including the population-invariant structure for dealing with inputs and outputs of varying sizes.

2.1 Task representation learning

We try to successfully achieve knowledge transfer among multi-agent tasks by capturing and exploiting both the common structure and the unique characteristics of tasks by learning task representation. A task distinguishes itself from other tasks by its transition and reward functions. Therefore, we incorporate our task representation learning component into the learning of a forward model that predicts the next state, local observations, and reward given the current state, local observations, and actions.

We associate each task i with a representation $z_i \in \mathbb{R}^m$ and expect it to reflect the relationship of the transition dynamics of tasks. In task similarity modeling, all source and unseen tasks share a representation explainer that inputs the task representations and outputs the forward model parameters. We then train the representation explainer on all source tasks. Concretely, for a source task $\mathcal{S}_i$, the forward model parameters are generated as $\eta_i = f_\theta(z_i)$, where θ denotes the representation explainer parameters. The forward model contains the three following predictors: $f^s_{\eta_i}$, $f^o_{\eta_i}$, and $f^r_{\eta_i}$. Given the current state s, agent j's observation o_j, and action a_j, these predictors estimate the next state s', next observation o'_j, and global reward r, respectively.

A possible method of training on source tasks is to backpropagate the forward model's prediction error to update both the representation explainer and the task representations. However, in practice, this scheme leads to representations with very small norms, which makes it difficult to obtain a meaningful representation space. To solve this problem, we propose the pre-determination of the task representation for each source task and the learning of the representation explainer by the prediction error backpropagation. This method helps form an informative task representation space and build a mapping from the task representation space to the forward model parameter space. In practice, source task representations are initialized as mutually orthogonal vectors. We first randomly generated vectors in $\mathbb{R}^m$ for the source tasks and then used Schmidt orthogonalization [21] on these vectors to obtain the source task representations. With the pre-defined task representations, we optimized the representation explainer to minimize the following loss function:

$$J(\theta) = \sum_{i=1}^{N_{\text{src}}} J_{\mathcal{S}_i}(\theta),$$

where N_{src} is the number of source tasks,

$$J_{\mathcal{S}_i}(\theta) = \mathbb{E}_{\mathcal{D}}\left[\sum_j \left[\|f^s_{\eta_i}(s, o_j, a_j) - s'\|_2^2 + \lambda_1 \|f^o_{\eta_i}(s, o_j, a_j) - o'_j\|_2^2 + \lambda_2 (f^r_{\eta_i}(s, o_j, a_j) - r)^2\right]\right]$$

is the per-task prediction loss, $\mathcal{D}$ is the replay buffer, and λ_1 and λ_2 are scaling factors.

We fixed the source task representations and learned the representation explainer during the training phase. In the current implementation, the training data were collected with the uniform random policy and were the same for the task representation learning in the transfer phase. Although more efficient approaches can be adopted for task representation learning, using a random policy was simple and effective in the experiment. In terms of the transfer phase, we aimed to find a good task representation that can reflect the similarity of the new task to the source tasks. To achieve this goal, we fixed the trained representation explainer and learned the task representation by minimizing the prediction loss of the forward model on the new task. We randomly initialized a task representation z, kept θ fixed, and had the forward model parameterized by $\eta = f_\theta(z)$. The task representation z was then updated by backpropagating the prediction loss for the transition and reward functions through the fixed f_η.

To keep the new task representation in the well-formed space of the source task representations, we learned the new task representation as a linear combination of the source task representations:

$$z = \sum_{i=1}^{N_{\text{src}}} \mu_i z_i \quad \text{s.t.} \quad \mu_i \geqslant 0, \ \sum_{i=1}^{N_{\text{src}}} \mu_i = 1.$$

We learned the weight vector μ in this manner. We additionally optimized an entropy regularization term $\mathcal{H}(\mu)$ to make the learning more stable. The final loss function for learning z is presented as follows:

$$J_{\mathcal{T}}(\mu) = \lambda \mathcal{H}(\mu) + \mathbb{E}_{\mathcal{D}}\left[\sum_j \left[\|f^s_\eta(s, o_j, a_j) - s'\|_2^2 + \lambda_1 \|f^o_\eta(s, o_j, a_j) - o'_j\|_2^2 + \lambda_2 (f^r_\eta(s, o_j, a_j) - r)^2\right]\right].$$

Appendix E describes the detailed architectures for task representation learning.

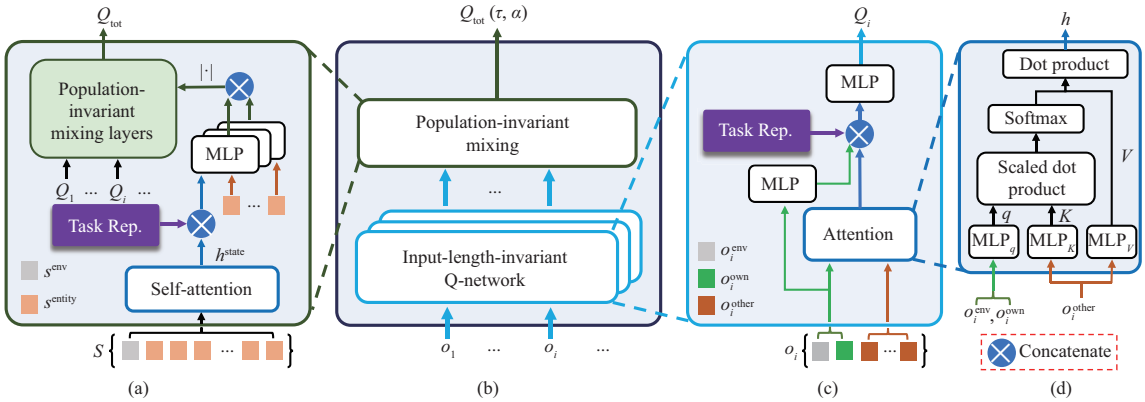

Figure 2 Population-invariant network structure for policy learning. (a) Population-invariant mixing; (b) overall framework; (c) input-length-invariant Q-network; (d) attention.

2.2 Task policy learning

After the task representation was learned by modeling the transition and reward functions, it was used to learn and transfer the policy on the source and unseen tasks.

A difficulty faced by multi-agent transfer learning is that the input and output dimensions vary across tasks. We used a population-invariant network (PIN, Figure 2) to solve this problem. This idea is not novel, and our contribution here is a lightweight structure that achieves a comparable, if not better, performance to other complex PIN modules [22]. The single-task experiments in Appendix G demonstrated its advantage, considering that the performance gains mainly came from our PIN design. Our PIN used the value decomposition framework and comprised two main components, namely, an individual Q-network shared by the agents and a monotonic mixing network [4] that learns the global Q-value as a combination of the local Q-values.

Similar to a previous study [22], for the individual Q-network, we decomposed the observation o_i into parts related to the environment o_i^{env}, agent i itself o_i^{own}, and other entities $o_i^{\text{other}} = \{o_i^{\text{other}_{j\neq i}}\}$. We adopted the attention mechanism to obtain a fixed-dimensional embedding h:

$$q = \texttt{MLP}_q([o_i^{\text{env}}, o_i^{\text{own}}]), \quad K = \texttt{MLP}_K([o_i^{\text{other}_1}, \ldots, o_i^{\text{other}_j}, \ldots]),$$
$$V = \texttt{MLP}_V([o_i^{\text{other}_1}, \ldots, o_i^{\text{other}_j}, \ldots]), \quad h = \texttt{softmax}(qK^{\text{T}}/\sqrt{d_k})V,$$

where $[\cdot,\cdot]$ is the vector concatenation operation, d_k is the query vector dimension, and bold symbols are matrices. Embedding h was then fed into the subsequent network together with the task representation z for estimating the action values. For the mixing network, we decomposed the state s into parts related to different entities s^{entity} and the environment s^{env}. We applied a similar self-attention module to integrate the information from these state parts and obtain a fixed-dimensional embedding vector h^{state}. h^{state}, task representation z, and s^{entity} were used to generate the mixing network parameters. The number of actions in some multi-agent tasks varied in different tasks. We employed a mechanism similar to that in other popular population-invariant networks [17, 23] to deal with this issue and discussed it in detail in Appendix E.

We fixed the task representation z during the whole policy learning process. Compared with policy learning, which typically lasts for 2M timesteps, the training of task representation costs a few samples. We collected 50000 samples for learning the task representations and the representation explainer. When transferring to new tasks, we used the individual Q-network and the representation explainer trained on the source tasks. We learned the task representation for 50000 timesteps and inserted it into the individual Q-network. Agents performed an execution in a decentralized manner according to this Q-network.

3 Case study

For an intuitive grasp of task representation learning, we designed a case study using a two-player navigation game to illustrate how MATTAR works. In the experimental scenario, each task was associated with a goal position. Two agents were expected to simultaneously navigate from an initial point and

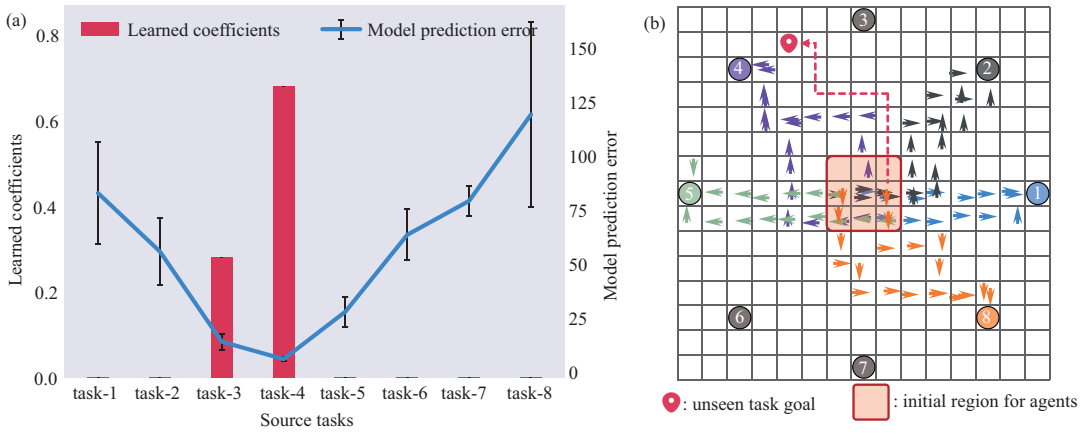

Figure 3 Case study on a two-player navigation game. (a) The task representation learned by MATTAR, and the prediction error on source tasks when the learned representation for the unseen task is used. (b) Agent policies on source tasks. Goals of tasks are represented by circles, and different tasks are distinguished by different colors. For simplicity, we only show trajectories on five source tasks. We test MATTAR on one unseen task and draw the trajectory of the transferred policy with a continuous red dashed arrow. More details can be found in Appendix A.

reach the specific goal. We designed eight source tasks with different goals and one unseen target task. Appendix A describes the experimental settings in detail.

We learned the representation explainer and the agent policy on eight source tasks during the training phase. We subsequently fine-tuned the task representation on the unseen task and transferred the learned policy. Figure 3(a) depicts the learned representation for the unseen target task. Source task-4 and task-3 corresponded to the two largest coefficients, which were approximately 0.7 and 0.3, respectively. Figure 3(b) illustrates that these two source tasks were more valuable because their goal positions were closer to that of the target unseen task when compared with the other source tasks. Furthermore, our mechanism for task representation learning captured the similarity between their reward functions. We confirmed our hypothesis by embedding the learned task representation into the representation explainer and testing the prediction error of the obtained forward model in each source task. Lower prediction errors were obtained in task-4 and task-3, implying that the learned representation can distinguish between tasks (e.g., rewards in these tasks).

We also visualized the trajectories of the successful policies in Figure 3. By encoding the similarity between the unseen target and source tasks in the task representation, the transferred policy reused the policy in similar source tasks. When embedding different source task representations, the agents exhibited a different behavior and reached their corresponding goals. In the unseen target task, the learned task representation guided the agent to move up first, similar to the behavior in task-4, and then move further to the target position when the goal is within sight. This demonstrated that MATTAR can extract the task similarity and encode it in the latent task representation, which helps solve unseen tasks.

4 Experiments

This section presents the experiments we designed to evaluate the following properties of the proposed method. (1) Generalizability to unseen tasks: our learning framework can extract knowledge from multiple source tasks and transfer the cooperation knowledge to unseen tasks, task representations play an indispensable role in transfer (Subsection 4.1); (2) good initialization for policy fine-tuning: fine-tuning the transferred policy can lead to success in super hard tasks, which cannot be solved when learning from scratch (Subsection 4.2); (3) benefits of multi-task training: our multi-task learning paradigm helps the model better leverage knowledge of different source tasks to boost the learning performance compared with individual training on source tasks (Subsection 4.3); (4) performance advantages on single tasks: although we did not design our framework for single-task learning, our network performs better against the underlying algorithm (Appendix G).

We evaluated MATTAR on the SMAC [19] and MPE [3] benchmarks. We better fitted the multi-task training setting by extending the original SMAC maps and sorting out three task series involving various numbers of Marines, Stalkers/Zealots, and Marines/Maneuvers/Medivacs. For the MPE tasks, we tested

Table 1 Transfer performance (mean win rates with variance) on the first series of SMAC maps[a)]

	Source tasks			Unseen tasks				
	2s3z	3s5z	3s5z_3s6z	1s8z	1s9z	2s8z	2s9z	7s3z
MATTAR	**1.00**±0.00	**0.99**±0.01	**0.48**±0.13	**0.79**±0.09	**0.60**±0.12	**0.93**±0.09	**0.84** ±0.04	**0.16**±0.12
w/o task rep.	0.99±0.01	0.96±0.02	0.20±0.08	0.12±0.13	0.07±0.12	0.47±0.18	0.25±0.20	0.15±0.19
0 task rep.	0.23±0.12	0.08±0.05	0.00±0.00	0.02±0.03	0.00±0.00	0.01±0.01	0.02±0.02	0.00±0.00
UPDeT-b	0.94±0.04	0.86±0.13	0.09±0.08	0.16±0.11	0.11±0.10	0.29±0.22	0.15±0.13	0.02±0.04
UPDeT-m	0.60±0.11	0.47±0.15	0.03±0.03	0.08±0.06	0.04±0.04	0.14±0.12	0.06±0.05	0.01±0.01
REFIL	0.75±0.09	0.43±0.13	0.01±0.01	0.08±0.04	0.03±0.01	0.08±0.05	0.05±0.04	0.06±0.04

a) The best results on each map are bolded. Besides, 3s5z_3s6z is short for 3s5z_vs_3s6z, and 3s5z_3s7z in the later text is short for 3s5z_vs_3s7z similarly.

Table 2 Transfer performance (mean win rates with variance) on the second series of SMAC maps[a)]

	Source tasks			Unseen tasks				
	MMM	MMM2	MMM4	MMM0	MMM1	MMM3	MMM5	MMM6
MATTAR	**1.00**±0.00	**0.92**±0.20	**0.93**±0.12	**0.98**±0.02	**0.97**±0.04	**0.86**±0.10	**0.47**±0.15	**0.09**±0.02
w/o task rep.	0.94±0.05	0.23±0.39	0.33±0.25	0.81±0.15	0.37±0.36	0.07±0.05	0.22±0.30	**0.09**±0.17
0 task rep.	0.61±0.07	0.07±0.06	0.21±0.22	0.28±0.19	0.11±0.13	0.08±0.10	0.08±0.12	0.02±0.04
UPDeT-b	**1.00**±0.00	0.78±0.04	0.41±0.14	0.73±0.21	0.84±0.07	0.57±0.15	0.00±0.00	0.00±0.00
UPDeT-m	0.48±0.03	0.15±0.19	0.20±0.07	0.30±0.16	0.27±0.13	0.28±0.08	0.00±0.00	0.00±0.00
REFIL	0.97±0.01	0.04±0.02	0.06±0.03	0.93±0.02	0.38±0.06	0.12±0.04	0.00±0.00	0.00±0.00

a) The best results on each map are bolded.

Table 3 Transfer performance (mean win rates with variance) on the third series of SMAC maps[a)]

	Source tasks				Unseen tasks			
	5m	5m_6m	8m_9m	10m_11m	3m	4m	4m_5m	6m
MATTAR	**1.00**±0.00	0.72±0.05	**0.83**±0.05	0.81±0.09	**0.94**±0.27	**0.97**±0.02	0.04±0.05	**1.00**±0.00
w/o task rep.	0.97±0.01	0.01±0.02	0.01±0.01	0.01±0.03	0.86±0.03	0.88±0.04	0.00±0.00	0.95±0.03
0 task rep.	0.78±0.39	0.16±0.12	0.30±0.24	0.40±0.28	0.00±0.00	0.21±0.15	0.01±0.01	0.67±0.47
UPDeT-b	**1.00**±0.00	**0.93**±0.05	0.81±0.19	**0.94**±0.04	0.81±0.08	0.95±0.06	**0.29**±0.17	**1.00**±0.00
UPDeT-m	0.77±0.09	0.32±0.03	0.35±0.05	0.43±0.02	0.36±0.04	0.57±0.03	0.10±0.06	0.91±0.09
REFIL	0.73±0.03	0.00±0.00	0.01±0.01	0.03±0.02	0.68±0.06	0.74±0.02	0.00±0.00	0.71±0.02
	Unseen tasks							
	6m_7m	7m	7m_8m	8m	9m	9m_10m	10m	10m_12m
MATTAR	0.74±0.15	**1.00**±0.00	**0.83**±0.04	**1.00**±0.00	**1.00**±0.00	**0.84**±0.09	**1.00**±0.00	**0.07**±0.01
w/o task rep.	0.03±0.02	0.94±0.03	0.08±0.10	0.93±0.04	0.86±0.05	0.04±0.02	0.52±0.22	0.00±0.00
0 task rep.	0.31±0.22	0.67±0.47	0.49±0.35	0.67±0.47	0.66±0.46	0.32±0.24	0.65±0.46	0.00±0.00
UPDeT-b	**0.78**±0.05	0.99±0.01	0.73±0.11	0.99±0.02	0.99±0.01	0.80±0.16	0.99±0.01	**0.07**±0.04
UPDeT-m	0.35±0.10	0.92±0.03	0.38±0.05	0.83±0.05	0.66±0.11	0.33±0.09	0.17±0.08	0.03±0.02
REFIL	0.01±0.00	0.66±0.03	0.01±0.01	0.63±0.05	0.55±0.05	0.01±0.00	0.46±0.02	0.00±0.00

a) The best results on each map are bolded. Besides, xm_ym is short for xm_vs_ym (e.g., 5m_6m short for 5m_vs_6m).

on a discrete version of Spread and Gather, with different numbers of agents trying to cover one or all landmarks. Appendix B describes the detailed environmental settings. To ensure a fair evaluation, we performed all experiments with five random seeds. The results yielded a 95% confidence interval in Figures 4 and 5. For a more comprehensive description of the experimental details, please refer to Appendix D.

4.1 Generalizability to unseen tasks

As the major desiderata of our method, we expect MATTAR to have a better transfer performance on the unseen tasks. Note that only a few related studies are applicable to tasks of varying sizes. We compared our method against the state-of-the-art multi-agent transfer methods UPDeT [17] and REFIL [22]. UPDeT transfers knowledge from a single source task. For a fair comparison, we transferred from each source task to every unseen task and calculated the best (UPDeT-b) and mean (UPDeT-m) performance on each unseen task. For the test phase, we conducted transfer experiments on both the source and unseen tasks. Tables 1–3 show the SMAC benchmark results, while Tables 4 and 5 present

Table 4 Evaluation on MPE: transfer performance (mean success rates with variance) on `Spread` with different numbers of agents[a)]

	Source tasks				Unseen tasks			
	2	4	7	9	3	5	6	8
MATTAR	**1.00**±0.00	**0.97**±0.03	**0.17**±0.16	**0.12**±0.05	**0.98**±0.01	**0.75**±0.12	**0.19**±0.10	**0.09**±0.09
`w/o task rep.`	**1.00**±0.00	0.55±0.30	0.02±0.01	0.00±0.00	0.83±0.13	0.30±0.13	0.03±0.04	0.00±0.00
`0 task rep.`	0.98±0.01	0.96±0.03	0.13±0.18	0.09±0.04	0.94±0.07	0.70±0.17	0.09±0.09	**0.09**±0.07
UPDeT-b	0.73±0.23	0.09±0.00	0.02±0.02	0.00±0.00	0.41±0.16	0.05±0.02	0.00±0.00	0.02±0.02
UPDeT-m	0.39±0.12	0.06±0.00	0.00±0.00	0.00±0.00	0.18±0.08	0.02±0.01	0.00±0.00	0.01±0.01
REFIL	**1.00±0.00**	0.93±0.04	0.07±0.04	0.01±0.01	0.96±0.03	0.73±0.06	0.06±0.05	0.05±0.00

a) The best results on each task are bolded.

Table 5 Evaluation on MPE: transfer performance (mean success rates with variance) on `Gather` with different numbers of agents[a)]

	Source tasks				Unseen tasks			
	2	4	7	9	3	5	10	15
MATTAR	**1.00**±0.00	**1.00**±0.00	0.82±0.11	0.84±0.17	**1.00**±0.00	**1.00**±0.00	0.63±0.12	**0.55**±0.22
`w/o task rep.`	**1.00**±0.00	**1.00**±0.00	0.88±0.09	0.84±0.17	**1.00**±0.00	0.99±0.01	0.62±0.13	0.53±0.18
`0 task rep.`	**1.00**±0.00	**1.00**±0.00	0.73±0.26	0.75±0.18	**1.00**±0.00	0.99±0.01	0.54±0.26	0.50±0.29
UPDeT-b	0.66±0.34	0.38±0.38	0.25±0.12	0.09±0.09	0.61±0.36	0.36±0.27	0.03±0.03	0.06±0.06
UPDeT-m	0.32±0.16	0.15±0.15	0.07±0.02	0.04±0.04	0.20±0.10	0.11±0.07	0.01±0.01	0.02±0.02
REFIL	**1.00**±0.00	**1.00**±0.00	**0.99**±0.02	**0.99**±0.01	**1.00**±1.00	**1.00**±0.00	**0.73**±0.14	0.51±0.14

a) The best results on each task are bolded.

Table 6 Our method models task relationships and exploits it for knowledge transfer. Task representations of unseen tasks are learned as a linear combination of source tasks' representations, whose coefficients are shown in this table. These weights reveal the encoded task relationship[a)]

Source	Unseen tasks		Source	Unseen tasks		Source	Unseen tasks	
	`4m`	`10m_12m`		`3s4z`	`3s5z_3s7z`		MMM0	MMM6
`5m`	**0.61**	**0.43**	`2s3z`	0.21	0.18	MMM	**0.44**	0.30
`5m_6m`	0.13	0.07	`3s5z`	**0.59**	0.21	MMM2	0.15	0.14
`8m_9m`	0.14	0.08	`3s5z_3s6z`	0.21	**0.61**	MMM4	0.40	**0.56**
`10m_11m`	0.12	**0.43**						

a) For each target task, the largest source task coefficient is bolded.

the MPE results.

MATTAR showed a superior transfer performance on the unseen tasks, significantly outperforming UPDeT-b and REFIL, especially in complex settings requiring sophisticated coordination like the MMM series. MATTAR was effective, even for the unseen tasks that were very different from the source tasks. For example, the MATTAR policy learned on `2s3z`, `3s5z`, and `3s5z_vs_3s6z` can win 79% of the games on `1s8z`. On the MPE tasks, MATTAR also showed a good generalization performance on the unseen tasks with different numbers of agents, totally outperforming the baselines and the ablations. For example, on the `Spread` tasks, MATTAR exhibited a significant advantage over UPDeT on all the unseen tasks and outperformed REFIL with a 13% margin on the unseen task with six agents.

We will explain our method's performance by first checking the learned representations on the unseen tasks and then performing the ablation studies.

What representations are learned for the unseen tasks? When encountering an unseen task, we first learn its representation as a linear combination of the source task representations. We then directly update the coefficients of this linear combination by backpropagating the forward model's prediction error. For a deeper understanding of how our method transfers learned knowledge, we investigated herein the learned coefficients of the linear combination.

Table 6 presents the coefficients of two unseen tasks for each map series. The largest coefficient typically corresponded to the source task that was the most similar to the unseen task. For example, `5m` was the closest source task to the unseen task `4m`, and the `5m` coefficient was 61%. We, however, found some exceptions. For example, for the unseen task `10m_vs_12m`, the coefficients of the two source tasks, `5m` and `10m_vs_11m`, were equal. They together took up 86% of all coefficients. While `10m_vs_11m` was very similar to `10m_vs_12m` because the team composition was similar, the policy for solving `5m` differed from that for `10m_vs_12m`. Agents first formed groups to quickly set up an attack curve in `10m_vs_12m`. Therefore, on a local battlefield, approximately five allies are fighting against a similar number of enemies. In this case,

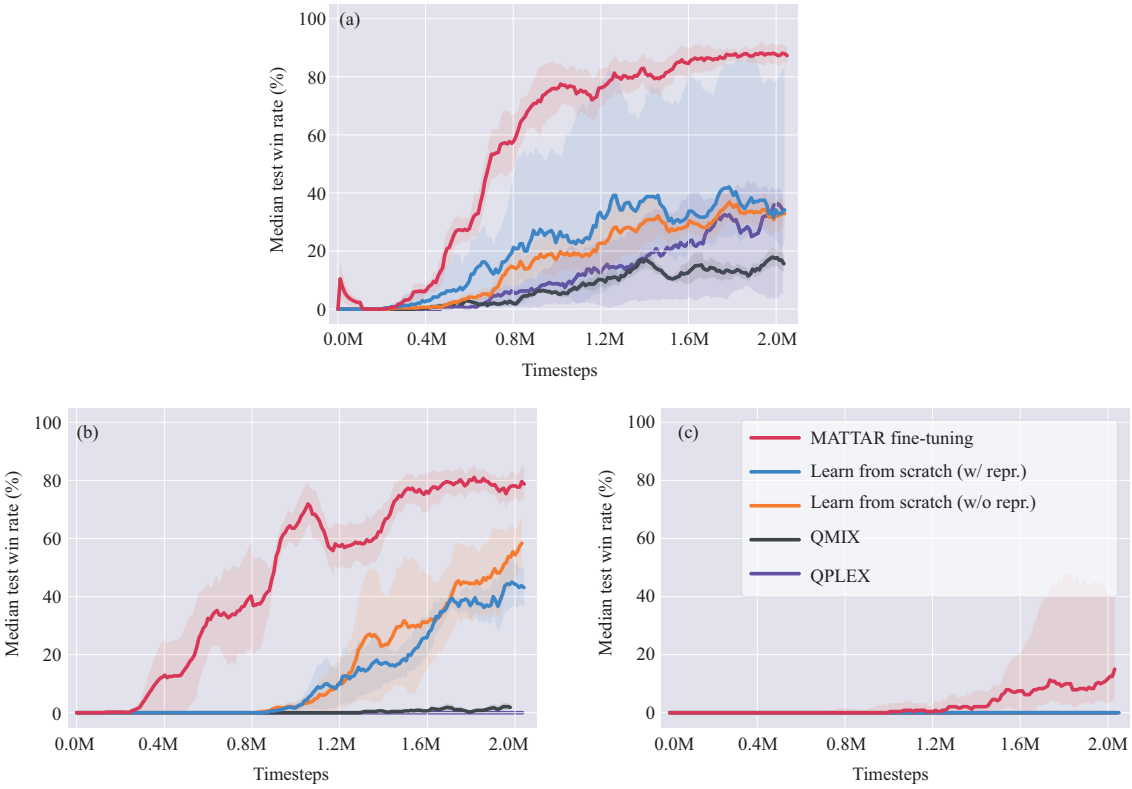

Figure 4 On unseen tasks: task representations provide a good initialization. Fine-tuning the policy can learn cooperation policies more effectively than learning from scratch. (a) `10m_vs_12m`; (b) MMM6; (c) `3s5z_vs_3s7z`.

the policy learned from `5m` can be locally used to promote coordination in `10m_vs_12m`.

In conclusion, in MATTAR, the task representation learning captures the task dynamics similarity and discovers policy reuse opportunities. These representations help unseen tasks to effectively leverage knowledge from the most similar source tasks or reuse knowledge from mixed source tasks, consistent with the case study results (Section 3).

Ablations. We further investigated the role of the task representations in our method by designing two ablations: (1) `0 task rep.` uses trained MATTAR models but feeds a zero-valued task representation into the policy network when transferring; (2) `w/o task rep.` denotes the complete removal of components related to the task representation, including the forward model, representation learning process, and task representation in the policy network. The population-invariant policy is then trained and transferred. This ablation can reveal the generalization ability of the policy network itself.

Tables 1–3 present the results. The ablations caused a large drop in the performance of MATTAR on nearly all the unseen tasks. For example, after being trained on `2s3z`, `3s5z`, and `3s5z_vs_3s6z`, MATTAR achieved a 0.79 win rate on `1s8z`, while `w/o task rep.` acquired a 0.12 win rate, and `0 task rep.` did not win any game. In summary, task representations play an indispensable role in policy transfer.

4.2 Good initialization for policy fine-tuning

When we evaluated the MATTAR performance on the unseen tasks, we only trained the task representations and reused the other parts of the policy. We investigated herein the performance of MATTAR after policy fine-tuning. On an unseen task, we first fixed the representation explainer and trained the task representation z for 50k timesteps. Next, we fixed the learned task representation and fine-tuned the policy for 2M timesteps. We then performed a comparison against learning from scratch in Figure 4. Learning from scratch meant the policy was randomly initialized and trained for 2M timesteps. We considered two versions for learning from scratch. The first version (`Learn from scratch w/o repr.`) totally removed the task representations from the policy, while the second version (`Learn from scratch w/ repr.`) inserted the same task representation z, as in the fine-tuning experiments.

The task representation and reused policy provided a good initialization. For example, on `10m_vs_12m`,

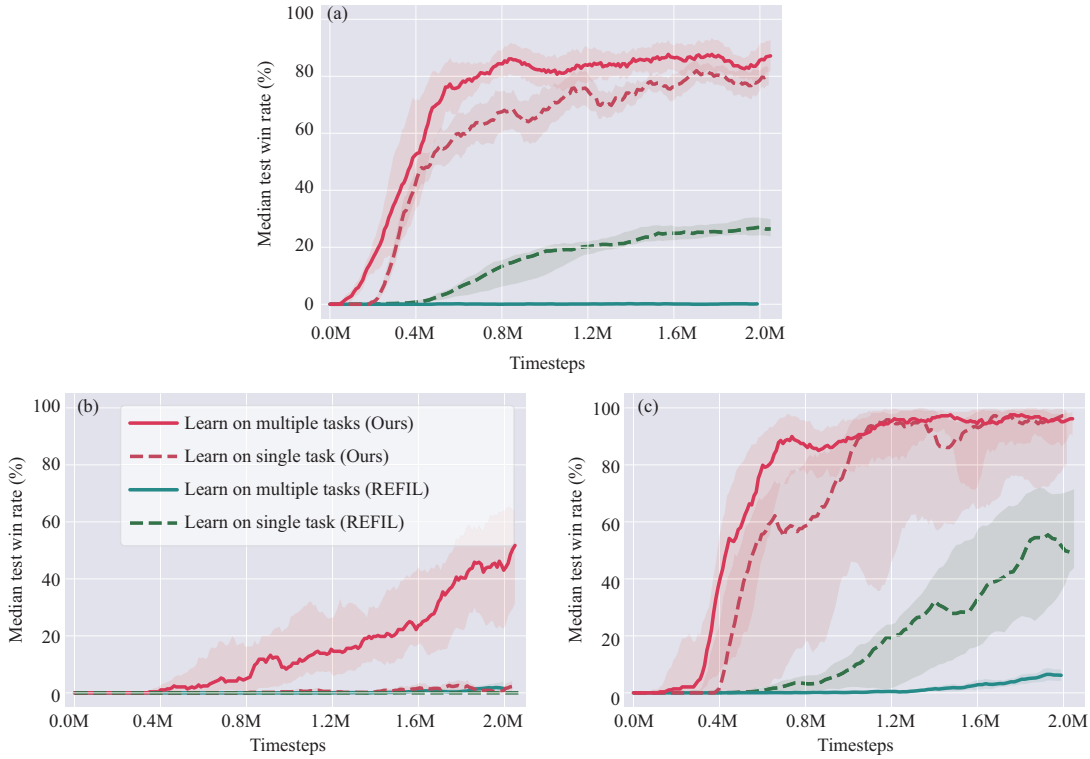

Figure 5 On source tasks: MATTAR provides a framework for multi-agent multi-task learning. Training on multiple tasks helps improve performance compared with learning on a single task. (a) `5m_vs_6m`; (b) `3s5z_vs_3s6z`; (c) MMM2.

after fine-tuning for 2M timesteps, `MATTAR fine-tuning` converged to a win rate of approximately 0.86, while both versions of learning from scratch only achieved a win rate of approximately 0.4. Empirically, only `MATTAR fine-tuning` achieved non-zero winning rates on `3s5z_vs_3s7z`, which was a task harder than the super hard map `3s5z_vs_3s6z`.

4.3 Benefits of multi-task learning

MATTAR adopted a scheme wherein multiple sources were simultaneously learned. Our aim was to leverage knowledge from more tasks and be able to generalize the learned knowledge to a larger set of unseen tasks. Empirically, this multi-task training setting helped not only the unseen tasks but also the source tasks.

Figure 5 presents the MATTAR performance on the source tasks when training with single and multiple source tasks. The experiments were performed on three tasks from three different series. Training on multiple tasks significantly boosted learning performance. For example, on `3s5z_vs_3s6z`, after 2M training samples, MATTAR with multiple tasks converged to the win rate of approximately 0.6, while training solely on this task can only achieve a win rate of approximately 0.05. MATTAR significantly outperformed the state-of-the-art deep multi-agent multi-task method (REFIL [22]). In other words, MATTAR provides a good learning framework for multi-agent multi-task learning and can leverage experience on other tasks to improve its performance on a similar task.

5 Related work

Transfer learning. Transfer learning promises to improve the sample efficiency of reinforcement learning [13], which learns knowledge from the source tasks, to accelerate the learning efficiency in unseen tasks. Previous studies used successor features, decoupled the transition dynamics and reward function, and learned faster in simulated navigation and robotic arm settings [24]. DSE [25] modeled the transfer process as a variational inference, further learning a latent space to transfer skills across different dynamics. Ref. [26] applied a model-based regularizer to learn the task-level transfer across various observation spaces. Related to our study, REPAINT [27] combined task representations with on-policy learning

and used an advantage-based experience selection approach to transfer useful samples. Our method differed from these studies in the following aspects: (1) its mechanism for alternately updating the task representation and the representation explainer; (2) its mechanism for handling different input/output sizes.

The basic idea behind multi-agent transfer learning [28] is knowledge reuse from other tasks or other learning agents corresponding to intra- and inter-agent transfers. Knowledge reuse is expected to accelerate coordination compared with learning from scratch. The inter-agent transfer paradigm aims to reuse knowledge from other agents with different sensors or (possibly) internal representations via communication. DVM [29] treats the multi-agent problem as a multi-task issue to combine knowledge from multiple tasks and distill this knowledge using a value-matching mechanism. Meanwhile, LeCTR [30] learns to teach in a multi-agent environment and advise others in a peer-to-peer manner. MAPTF [31] takes a further step by proposing an option-based policy transfer for multi-agent cooperation, significantly boosting the performance of existing methods in both discrete and continuous-state spaces. By contrast, intra-agent transfer refers to knowledge reuse from previous tasks, focusing on knowledge transfer across multi-agent tasks. The varying populations and input lengths impede transfer among agents, with which the graph neural networks and the transformer play promising roles. DyMA-CL [15] designs various transfer mechanisms across the curricula to accelerate the learning process based on a dynamic agent number network. EPC [14] proposes a curriculum learning paradigm via an evolutionary approach to scale up the agent population number. UPDeT [17] and PIT [18] use the generalization ability of the transformer to accomplish the multi-agent cooperation and transfer between tasks. Although the above mentioned methods can accelerate the learning efficiency of MARL algorithms, they do not exploit task similarity to achieve a better transfer performance. By contrast, our method explicitly models the task relationship by learning a hidden space in which similar-dynamics tasks have similar representations.

Multi-agent representation learning. Learning an effective representation in MARL is receiving significant attention for its effectiveness in solving many important problems. CQ-Learning [32] learns to adapt the state representation for multi-agent systems to coordinate with other agents. Ref. [33] learns useful policy representations to model an agent's behavior in a multi-agent system. LILI [34] learns latent representations to capture the relationship between an ego-agent's behavior and the other agent's future strategy. Meanwhile, RODE [12] uses an action encoder to learn action representations. It applies clustering methods to decompose the joint action space into restricted role action spaces and reduce the policy search space of multi-agent cooperation. MAR [35] learns meta-representation for multi-agent generalization. Our approach differs from the abovementioned methods because it learns representations for tasks and uses a representation explainer to achieve an efficient policy transfer.

Multi-task reinforcement learning. Multi-task reinforcement learning is another relevant research topic that enables an RL agent to leverage experience from multiple tasks to improve the sample efficiency and avoid learning from scratch on every single task. Various approaches were proposed to achieve multi-task learning, including distilling the knowledge of separate tasks into a shared policy [36–38], conditioning policies on tasks [39], mapping tasks to policy parameters [40–42], and solving the negative interference problem [43–45] indicating that gradients of different tasks may conflict.

However, most of these multi-task reinforcement learning methods utilize limited task-level context information, typically as simple as an ordinal task id. Similar to that in our study, multi-task learning with metadata [46, 47] approaches have been explored to exploit richer context information in the form of learned task embeddings, focusing on the task relation discovery. Sodhani et al. [48] used context information in high-level, under-specified, and unstructured natural language for multi-task learning. Compared with the settings presented in this study, multi-agent problems pose additional challenges. The transition dynamics and the reward functions in multi-agent tasks are influenced by the interaction of all agents; hence, the relation between tasks must be conditioned on the interaction pattern of multiple agents. Different from previous contextual multi-task RL methods, we included agent interaction modeling in the task representation learning module to deal with this problem. Nevertheless, we think that incorporating natural language as the prior high-level metadata of tasks can further improve our method's performance method in further study.

Context-based meta-RL. Related to our work, context-based meta-RL also reuses experience from multiple tasks and infers how to solve a new task from the collected small amounts of experience [49]. For example, PEARL [50] performs online task inference through probabilistic filtering of latent task variables. Other meta-RL methods exploit recurrent network dynamics for implicit task inference and fast adaptation [51,52]. These methods tend to form dynamic variables or embeddings, while our approach

utilizes context information to obtain fixed task representations. Although their task adaptation ability has been proven, context-based meta-RL has not been extended to multi-agent cases and cannot deal with varying numbers of actions and different observation input lengths. Future studies must discuss whether or not the inspiration of the context-based meta-RL can further improve the performance of our multi-agent multi-task learning framework.

Task or skill embedding learning. Learning task or skill embeddings for multi-task transfer reinforcement learning is also being extensively explored. When testing, Hausman et al. [53] learned a new skill embedding by interpolating the learned skill embeddings on the source tasks through approximate variational inference. Arnekvist et al. [54] focused on learning skill embeddings but conditioned on optimal Q-functions for different skills. Co-Reyes et al. [55] adopted a hierarchical framework and learned a high-level policy to control a low-level skill latent space. Similar to ours, Co-Reyes et al. [55] learned a low-level skill space by encoding the experience trajectories and decoding the states and actions. Another study similar to ours was that of Zhang et al. [56], which conditioned the policy on the latent representations learned by the dynamics and reward module. Lan et al. [57] learned task embeddings, fixed the policy, and updated the encoder during the test time. Our task representation learning method differed from these methods in the following aspects: (1) it is specially designed for multi-agent settings and considers varying input sizes and intra-agent interaction; (2) to the best of our knowledge, an alternatively fixed learning scheme is proposed for the first time.

Modular RL. Modular RL is another line of research on single-agent multi-task learning that can be effective in multi-agent settings. It decentralizes the control of multi-joint robots by learning the policies for each actuator, thereby holding the promise of dealing with input and output of varying lengths. Each joint has its controlling policy. They coordinate with each other via various message passing schemes. To do so, Wang et al. [58] and Pathak et al. [59] represented the robot's morphology as a graph and used graph neural networks (GNNs) as policy and message passing networks. Huang et al. [60] utilized both bottom-up and top-down message passing schemes through the links between joints for coordination. All GNN-like studies showed the benefits of modular policies over a monolithic policy in tasks tackling different morphologies. However, Kurin et al. [61] recently validated a hypothesis that any benefit the GNNs can extract from morphological structures is outweighed by the difficulty of message passing across multiple hops. They further proposed a transformer-based method called AMORPHEUS, which utilizes self-attention mechanisms as a message passing approach. Although modular RL can deal with varying action numbers of different tasks and implicitly model the interaction between agents through the GNN, transformer, or message passing, it still cannot cope with the varying observation lengths and does not incorporate the task-level context information compared with our method.

6 Conclusion

Herein, we investigated cooperative multi-agent transfer reinforcement learning by learning the task representations that model and exploit the task relationship. Previous studies on multi-agent transfer learning mainly focused on the varying population and input lengths, relying solely on the generalization ability of neural networks for cooperation knowledge transfer and ignoring the relationship among tasks. Our method improves the transfer performance by learning the task representations that capture the differences and similarities among tasks. When faced with a new task, our approach only must obtain a new representation before transferring the learned knowledge. Taking advantage of task relationship mining, our proposed method, MATTAR, achieves the best transfer performance and other bonuses compared with multiple baselines. An important direction for future research is the transfer of tasks from different task distributions. Whether or not a linear combination of the source task representations can fully represent unseen tasks is also a valuable topic.

Acknowledgements This work was supported in part by National Key Research and Development Program of China (Grant No. 2020AAA0107200), National Natural Science Foundation of China (Grant Nos. 61876119, 61921006), and Natural Science Foundation of Jiangsu (Grant No. BK20221442).

Supporting information Appendixes A–I. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Tjosvold D. Cooperation theory and organizations. Hum Relat, 1984, 37: 743–767
2 Zhang K Q, Yang Z R, Başar T. Multi-agent reinforcement learning: a selective overview of theories and algorithms. In: Handbook of Reinforcement Learning and Control. Berlin: Springer, 2021. 321–384
3 Lowe R, Wu Y, Tamar A, et al. Multi-agent actor-critic for mixed cooperative-competitive environments. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017. 6382–6393
4 Rashid T, Samvelyan M, Schroeder C, et al. QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. In: Proceedings of the 35th International Conference on Machine Learning, 2018. 4292–4301
5 Wang J H, Ren Z Z, Liu T, et al. QPLEX: duplex dueling multi-agent Q-learning. In: Proceedings of the 9th International Conference on Learning Representations, 2021
6 Wang Y H, Han B N, Wang T H, et al. DOP: off-policy multi-agent decomposed policy gradients. In: Proceedings of the 9th International Conference on Learning Representations, 2021
7 Cao J H, Yuan L, Wang J H, et al. LINDA: multi-agent local information decomposition for awareness of teammates. 2021. ArXiv:2109.12508
8 Foerster J, Assael Y M, de Freitas N, et al. Learning to communicate with deep multi-agent reinforcement learning. In: Proceedings of the 30th International Conference on Neural Information Processing Systems, 2016. 2145–2153
9 Jiang J C, Lu Z Q. Learning attentional communication for multi-agent cooperation. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, 2018. 7265–7275
10 Jiang J C, Dun C, Huang T J, et al. Graph convolutional reinforcement learning. In: Proceedings of the 8th International Conference on Learning Representations, 2020
11 Wang T H, Dong H, Lesser V, et al. ROMA: multi-agent reinforcement learning with emergent roles. In: Proceedings of the 37th International Conference on Machine Learning, 2020. 9876–9886
12 Wang T H, Gupta T, Mahajan A, et al. RODE: learning roles to decompose multi-agent tasks. In: Proceedings of the 9th International Conference on Learning Representations, 2021
13 Zhu Z D, Lin K X, Zhou J Y, et al. Transfer learning in deep reinforcement learning: a survey. 2020. ArXiv:2009.07888
14 Long Q, Zhou Z H, Gupta A, et al. Evolutionary population curriculum for scaling multi-agent reinforcement learning. In: Proceedings of International Conference on Learning Representations, 2019
15 Wang W X, Yang T P, Liu Y, et al. From few to more: large-scale dynamic multiagent curriculum learning. In: Proceedings of AAAI Conference on Artificial Intelligence, 2020. 7293–7300
16 Agarwal A, Kumar S, Sycara K, et al. Learning transferable cooperative behavior in multi-agent teams. In: Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, 2020. 1741–1743
17 Hu S Y, Zhu F D, Chang X J, et al. UPDeT: universal multi-agent reinforcement learning via policy decoupling with transformers. In: Proceedings of the 9th International Conference on Learning Representations, 2021
18 Zhou T Z, Zhang F B, Shao K, et al. Cooperative multi-agent transfer learning with level-adaptive credit assignment. 2021. ArXiv:2106.00517
19 Samvelyan M, Rashid T, de Witt C S, et al. The StarCraft multi-agent challenge. In: Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems, 2019. 2186–2188
20 Oliehoek F A, Amato C. A Concise Introduction to Decentralized POMDPs. Berlin: Springer, 2016
21 Björck Å. Numerics of Gram-Schmidt orthogonalization. Linear Algebra Appl, 1994, 197: 297–316
22 Iqbal S, de Witt C S, Peng B, et al. Randomized entity-wise factorization for multi-agent reinforcement learning. In: Proceedings of the 38th International Conference on Machine Learning, 2021. 4596–4606
23 Wang W X, Yang T P, Liu Y, et al. Action semantics network: considering the effects of actions in multiagent systems. In: Proceedings of the 8th International Conference on Learning Representations, 2020
24 Barreto A, Borsa D, Quan J, et al. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In: Proceedings of the 35th International Conference on Machine Learning, 2018. 501–510
25 Petangoda J C, Adam V, Vrancx P, et al. Disentangled skill embeddings for reinforcement learning. 2019. ArXiv:1906.09223
26 Sun Y C, Zheng R J, Wang X Y, et al. Transfer RL across observation feature spaces via model-based regularization. In: Proceedings of the 10th International Conference on Learning Representations, 2022
27 Tao Y Z, Genc S, Chung J, et al. REPAINT: knowledge transfer in deep reinforcement learning. In: Proceedings of the 38th International Conference on Machine Learning, 2021. 10141–10152
28 Silva F L, Costa A H R. Transfer learning for multiagent reinforcement learning systems. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, 2016. 3982–3983
29 Wadhwania S, Kim D K, Omidshafiei S, et al. Policy distillation and value matching in multiagent reinforcement learning. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2019. 8193–8200
30 Omidshafiei S, Kim D K, Liu M, et al. Learning to teach in cooperative multiagent reinforcement learning. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, 2019. 6128–6136
31 Yang T P, Wang W X, Tang H Y, et al. An efficient transfer learning framework for multiagent reinforcement learning. In: Proceedings of the 35th Conference on Neural Information Processing Systems, 2021. 17037–17048
32 de Hauwere Y M, Vrancx P, Nowé A. Learning multi-agent state space representations. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, 2010. 715–722
33 Grover A, Gupta J, Burda Y, et al. Learning policy representations in multiagent systems. In: Proceedings of the 35th International Conference on Machine Learning, 2018. 1802–1811
34 Xie A, Losey D, Tolsma R, et al. Learning latent representations to influence multi-agent interaction. In: Proceedings of Conference on Robot Learning, 2021. 575–588
35 Zhang S N, Shen L, Han L. Learning meta representations for agents in multi-agent reinforcement learning. 2021. ArXiv:2108.12988
36 Levine S, Finn C, Darrell T, et al. End-to-end training of deep visuomotor policies. J Mach Learn Res, 2016, 17: 1334–1773
37 Ghosh D, Singh A, Rajeswaran A, et al. Divide-and-conquer reinforcement learning. In: Proceedings of the 6th International Conference on Learning Representations, 2018
38 Xu Z Y, Wu K, Che Z P, et al. Knowledge transfer in multi-task deep reinforcement learning for continuous control. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020. 15146–15155
39 Deisenroth M P, Englert P, Peters J, et al. Multi-task policy search for robotics. In: Proceedings of IEEE International

Conference on Robotics and Automation, 2014. 3876–3881
40 da Silva B C, Konidaris G, Barto A G. Learning parameterized skills. In: Proceedings of the 29th International Conference on Machine Learning, 2012. 1443–1450
41 Kober J, Wilhelm A, Öztop E, et al. Reinforcement learning to adjust parametrized motor primitives to new situations. Auton Robot, 2012, 33: 361–379
42 Yang R H, Xu H Z, Wu Y, et al. Multi-task reinforcement learning with soft modularization. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020. 4767–4777
43 Du Y S, Czarnecki W M, Jayakumar S M, et al. Adapting auxiliary losses using gradient similarity. 2018. ArXiv:1812.02224
44 Suteu M, Guo Y K. Regularizing deep multi-task networks using orthogonal gradients. 2019. ArXiv:1912.06844
45 Yu T H, Kumar S, Gupta A, et al. Gradient surgery for multi-task learning. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020. 5824–5836
46 You Q, Wu O, Luo G, et al. Metadata-based clustered multi-task learning for thread mining in web communities. In: Proceedings of International Conference on Machine Learning and Data Mining in Pattern Recognition, 2016. 421–434
47 Zheng Z M, Wang Y Q, Dai Q Y, et al. Metadata-driven task relation discovery for multi-task learning. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019. 4426–4432
48 Sodhani S, Zhang A, Pineau J. Multi-task reinforcement learning with context-based representations. In: Proceedings of the 38th International Conference on Machine Learning, 2021. 9767–9779
49 Zintgraf L, Schulze S, Lu C, et al. VariBAD: variational Bayes-adaptive deep RL via meta-learning. J Mach Learn Res, 2021, 22: 13198–13236
50 Rakelly K, Zhou A, Finn C, et al. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In: Proceedings of the 36th International Conference on Machine Learning, 2019. 5331–5340
51 Wang J X, Kurth-Nelson Z, Soyer H, et al. Learning to reinforcement learn. In: Proceedings of Annual Meeting on Cognitive Science Society, 2017
52 Duan Y, Schulman J, Chen X, et al. RL^2: fast reinforcement learning via slow reinforcement learning. 2016. ArXiv:1611.02779
53 Hausman K, Springenberg J T, Wang Z Y, et al. Learning an embedding space for transferable robot skills. In: Proceedings of the 6th International Conference on Learning Representations, 2018
54 Arnekvist I, Kragic D, Stork J A. VPE: variational policy embedding for transfer reinforcement learning. In: Proceedings of International Conference on Robotics and Automation, 2019. 36–42
55 Co-Reyes J D, Liu Y X, Gupta A, et al. Self-consistent trajectory autoencoder: hierarchical reinforcement learning with trajectory embeddings. In: Proceedings of the 35th International Conference on Machine Learning, 2018. 1637–1647
56 Zhang A, Satija H, Pineau J. Decoupling dynamics and reward for transfer learning. In: Proceedings of the 6th International Conference on Learning Representations, 2018
57 Lan L, Li Z G, Guan X H, et al. Meta reinforcement learning with task embedding and shared policy. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019. 2794–2800
58 Wang T W, Liao R J, Ba J, et al. NerveNet: learning structured policy with graph neural networks. In: Proceedings of the 6th International Conference on Learning Representations, 2018
59 Pathak D, Lu C, Darrell T, et al. Learning to control self-assembling morphologies: a study of generalization via modularity. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems, 2019. 2295–2305
60 Huang W L, Mordatch I, Pathak D. One policy to control them all: shared modular policies for agent-agnostic control. In: Proceedings of the 37th International Conference on Machine Learning, 2020. 4455–4464
61 Kurin V, Igl M, Rocktäschel T, et al. My body is a cage: the role of morphology in graph-based incompatible control. In: Proceedings of the 9th International Conference on Learning Representations, 2021