Abstract—Deep cooperative multi-agent reinforcement learning has demonstrated its remarkable success over a wide spectrum of complex control tasks. However, recent advances in multi-agent learning mainly focus on value decomposition while leaving entity interactions still intertwined, which easily leads to over-fitting on noisy interactions between entities. In this work, we introduce a novel interactOn Pattern disentanGling (OPT) method, to disentangle the entity interactions into interaction prototypes, each of which represents an underlying interaction pattern within a subgroup of the entities. OPT facilitates filtering the noisy interactions between irrelevant entities and thus significantly improves generalizability as well as interpretability. Specifically, OPT introduces a sparse disagreement mechanism to encourage sparsity and diversity among discovered interaction prototypes. Then the model selectively restructures these prototypes into a compact interaction pattern by an aggregator with learnable weights. To alleviate the training instability issue caused by partial observability, we propose to maximize the mutual information between the aggregation weights and the history behaviors of each agent. Experiments on single-task, multi-task and zero-shot benchmarks demonstrate that the proposed method yields results superior to the state-of-the-art counterparts.

Index Terms—Deep reinforcement learning, cooperative multi-Agent learning, interaction pattern disentangling.

I. INTRODUCTION

REINFORCEMENT Learning (RL) is instigated by the trial and error (TE) procedure in human behaviors and now becomes a well-established paradigm for building intelligent systems interacting with the environment, especially driven by the resurgence of deep learning [1], [2], [3], [4], [5], [6]. In the RL context, cooperative Multi-Agent Reinforcement Learning (MARL), where a group of agents works collaboratively for one common goal, is a long-standing research topic and a key tool used to address many real-world problems such as video games [7], [8], traffic light systems [9], [10], and smart grid control [11], [12], [13]. However, cooperative MARL is perceived as a significantly more challenging problem than the single-agent counterpart due to several peculiar characteristics [14], for example, credit assignment that the agents need to deduce their contributions in the presence of only global rewards, partial observability which necessitates the learning of decentralized policies, the curse of dimensionality that the joint action (and state) space grows exponentially with the number of agents, etc.

To alleviate these issues, value-based MARL has recently emerged as a powerful framework with the large capacity of deep neural networks and the Centralized Training and Decentralized Execution (CTDE) paradigm [15], [16], [17], [18]. Value-based MARL endows agents the ability to learn the individual optimal policies by decomposing the global value function according to their contributions. For example, Value Decomposition Network (VDN) [15] learns to additively factorize the team value function into agent-wise value functions, which enables the decentralized deployment of the centralized trained agents. QMIX [16] identifies that the full factorization of VDN is not necessary and employs a network that estimates joint action values as a complex non-linear combination of per-agent values that condition only on local observations. Furthermore, randomized entity-wise factorization for imagined learning (REFIL) [18] randomly divided entities into subgroups to construct an auxiliary regularization term, which facilitates discovering more generalized value functions. These works mainly focus on agent-wise value decomposition with centralized training for decentralized deployment, while ignoring the specific interconnections between the entities of interest. The decentralized policies, however, can generalize poorly at test time due to over-fitting on noisy interactions during the training time. Specifically, we observe that entity interactions can be disentangled into several interaction prototypes, as depicted in Fig. 1. Each interaction prototype corresponds to an underlying interaction pattern within a subgroup of the entities, while different interaction prototypes often simultaneously exist among entities in a complex environment. The interaction prototypes concealed in various-scale tasks are usually shareable, resulting in similar subgroup strategies of agents. Unfortunately, these patterns are

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
We propose OPT, a novel disentangling method to decompose entity interactions into interaction prototypes. OPT not only enjoys stronger generalizability, but also exhibits higher interpretability with the disentangled interaction prototypes.

- Experiments conducted on the StarCraft II micromanagement, Google Research Football and Predator-Prey benchmarks demonstrate that OPT yields significantly superior performance to state-of-the-art competitors.

The remainder of this manuscript is organized as follows. In Section II, we review some MARL topics highly related to this work. Then preliminaries of MARL are provided in Section III. The proposed interaction pattern disentangling is provided in Section IV. Experimental results, including the benchmark comparison and ablation study, are presented in Section V. Finally, we conclude this manuscript with some possible future works in Section VI.

II. RELATED WORK

We briefly review here some recent advances that are most related to the proposed work in the context of MARL.

A. Value-Based MARL

Centralized Training and Decentralized Execution (CTDE) has recently emerged as a powerful paradigm in MARL, where agents are granted access to additional global information during centralized training and deliver actions only based on their local observation in a decentralized way. The most common approach to the MARL problem heretofore is to learn value decomposition under the CTDE paradigm [21]. To enable effective CTDE, the Individual-Global-Max (IGM) principle guarantees the optimal consistency between the joint and individual actions [22]. Based on this IGM principle, VDN [15] factorizes the joint action-value function into a summation of individual agent terms. In contrast, QMIX [16] extends this additive value function factorization and imposes a monotonicity constraint. Furthermore, QTRAN [22], WQMIX [23], and QPLEX [17] progressively enlarged the family of functions that the mixing network can represent, while other advanced works exploit the value decomposition from the perspective of role-based learning [24], [25], large-scale transfer learning [26], [27], [28], and intrinsic motivation-driven exploration [29], [30], [31], [32], [33]. However, these works still treat all entities as a whole and ignore their internal interaction information.

B. Policy-Based MARL

Another line of MARL methods under the CTDE paradigm deploys policy gradient to learn local policies with a centralized critic [34], [35]. Prior works in this domain often directly employ the value decomposition mechanism to realize credit assignment in the centralized critic [36], [37], [38], [39]. Most of them follow the same off-policy fashion as value-based methods to avoid low sample efficiency. Recently, MAPPO [40] and HATRPO [41] impose the trust region constraint on multi-agent policy learning, and empirically show that on-policy policy-based methods can also provide competent efficiency and performance guarantees on many cooperative MARL benchmarks [42], [43]. To further improve MAPPO, some advanced works attempt to alleviate...
the high variance issue in multi-agent policy gradient estimation [44], [45], [46]. However, these works also encounter the difficulty of over-fitting on noisy interactions.

C. Model-Based MARL

Model-based MARL aims to build the environment model to promote policy searching [47]. The multi-agent environment model often consists of not only a dynamics model but also multiple opponent models [48]. Sessa et al. [49] propose to construct confidence intervals around the transition function and hallucinate optimistic value functions for agents to guide exploration. Zhang et al. [50] and Liu et al. [51] attempt to establish the theoretical analysis of sample complexity of model-based MARL in zero-sum Markov games. Moreover, Mahajan et al. [52] and Van Der Vaart et al. [53] factorize the action-value function based on tensor decomposition to model agent interactions, and further develop an effective model-based algorithm using low-rank transition and reward functions. This tensor decomposition mechanism, however, is not applicable to our setting where non-agent entities should also be considered.

D. Interaction Pattern in MARL

Exploring complex interaction patterns has been widely studied in MARL. Several works [54], [55], [56] capture multi-agent interplay using the pre-defined graph. However, these prior relationships between entities are often unavailable. On the other hand, MAAC [19] and ATOC [20] introduce the attention mechanism to learn the interaction relationships, while UPDeT [57] and P[58] utilize attention-based semantic alignment between the input entities and the output actions to decompose the policy. Despite the promising results achieved, these works mainly rely on dense attention and have to focus on irrelevant entities. To remedy this issue, ALC-MADDPG [59] and CASEC [60] use the predefined threshold to prune the low correlation relationships between agents. Moreover, G2Anet [61] and S2RL [62] introduce the sparse attention mechanism to learn the dynamics among agents. Mahajan et al. [63] formalize the multi-agent generalization problem and provide theoretical bounds on generalization. Furthermore, REFIL [18] randomly partitions entities into sparse subgroups and forces each subgroup to learn the same objective function for all entities. This random grouping strategy serves as a regularization term to improve the generalization ability of value functions. Nevertheless, the interaction patterns in these methods still remain entangled. They only consider the sparsity of interaction patterns but ignore the diversity. Different interaction patterns often simultaneously exist among entities in a complex task. To this end, we introduce a novel method to realize the interaction pattern disentangling for both sparsity and diversity.

III. PRELIMINARY

In this section, we formally define the cooperative MARL problem under the decentralized partially observable Markov decision process (Dec-POMDP) with entities. Then we introduce the centralized training and decentralized execution (CTDE) paradigm for value decomposition.

A. Dec-POMDP With Entities

Cooperative MARL problem can be described as a Dec-POMDP with entities [18], [64] defined by a tuple \((\mathcal{E}, \mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{U}, P, r, \mathcal{O}, \Omega, \gamma)\), where \(\mathcal{E}\) is the set of entities in the environment. Each entity \(e\) has a state representation \(s^e\) and \(s = \{s^e | e \in \mathcal{E}\} \in \mathcal{S}\) is the global state of the environment. \(\mathcal{A} \subseteq \mathcal{E}\) is the finite set of agents, while the non-agent entities are part of the environment (e.g., uncontrollable enemies, obstacles and landmarks). \(\mathcal{T}\) is the set of tasks, where each task is composed of a different number of entities and agents. At each time step \(t\), each agent \(a \in \mathcal{A}\) receives an individual partial observation \(o^a_t \in \Omega\) according to the observation function \(O(s_t, a)\) and chooses an action \(a^a_t \in \mathcal{U}\), forming a joint action \(u_t\). This causes a transition to the next state \(s_{t+1}\) according to the state transition function \(P(s_{t+1} | s_t, u_t)\). All agents share the same reward function \(r(s_t, u_t)\) and \(\gamma \in [0, 1)\) is the discount factor. Each agent \(a\) has an action-observation history \(\tau^a \in \mathcal{T} \equiv (\Omega \times \mathcal{U})^*\) and learns its individual policy \(\pi^a(u^a_t | \tau^a)\) to jointly maximize the discounted return \(R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}\). \(\tau\) is used to denote joint action-observation history. The joint policy \(\pi\) induces a joint action-value function \(Q^\pi_{\text{tq}}(s_t, u_t) = \mathbb{E}_{s_{t+1} \sim \mathcal{E}, u_{t+1} \sim \mathcal{U}} [R_t | s_t, u_t]\).

B. CTDE Paradigm

The CTDE paradigm allows agents to learn their individual utility functions by optimizing the joint action-value function for credit assignment [65], [66], [67], [68]. During centralized training, the learning model has access to the global state and the action-observation histories of all agents, whereas each agent can only condition on its own action-observation history during decentralized execution. Due to the partial observability, \(Q_{\text{tot}}(\tau, u, s)\) is used in place of \(Q_{\text{tot}}(u, s)\). A mixing network is introduced to merge all individual action values into a joint one \(Q_{\text{tot}}(\tau, u, s; \theta_u) = f([Q_a(\tau^a, u^a; \theta_x)]_{a \in \mathcal{A}}, s; \theta_u)\), where \(Q_a\) is the utility network of each agent \(a\). The learnable parameter \(\theta = (\theta_e, \theta_u)\) can be updated by minimizing the following expected Temporal-Difference (TD) loss:

\[
\mathcal{L}_{TD}(\theta) = \mathbb{E}_{D} \left[ (y^t_{\text{tot}} - Q_{\text{tot}}(\tau, u, s; \theta_u))^2 \right].
\]

where \(D\) is the replay buffer of the transitions, \(y^t_{\text{tot}} = r + \gamma \max_{u^t} Q_{\text{tot}}(\tau^t, u^t, s^t; \theta_u)\) is the target value and \(\theta_{u}^{-}\) is the parameter of the target network [2].

IV. METHOD

In what follows, we first provide an overview of the framework based on the proposed interactOn Pattern disenTangling (OPT) module. Then we further detail the OPT module and summarize the complete optimization objective.
A. Overall Framework

Our framework adopts the CTDE paradigm, where each agent learns its individual utility network by optimizing the TD loss of the mixing network. As shown in Fig. 2, the OPT serves as a plug-in module in both the utility network and mixing network, jointly facilitating the learning of decision making and credit assignment. We use a Transformer architecture [69] to stack multiple layers of OPT modules, which can promote the generalization ability of the model. The outputs of the OPT modules are directly used for subsequent network computations. All the network modules, including OPT, MLP and GRU, perform entity-wise operations on their input.

Specially, the utility network of each agent \(a\) takes the current observation \(o^a_t\) and the history information \(h_{tot}^{a-1}\) as input, estimating the individual action value \(Q^a_t(s^a_t, \cdot)\) based on the OPT module. Moreover, each action value is calculated by its corresponding entity [18], [57]. Then the agent \(a\) selects an action \(u^a_t\) that maximizes the action value \(Q^a_t(s^a_t, u^a_t)\) to execute in the environment, while the action value \(Q^a_t(s^a_t, u^a_t)\) is used as the input to the mixing network. Here we adopt the annealed \(\varepsilon\)-greedy strategy for exploration. Our mixing network implementation uses QMIX [16] as a basic backbone for its robust performance, but it is readily applicable to the other mixing methods. The mixing network is a feed-forward neural network that merges all individual action values into a joint one \(Q_{tot}(s, s; \theta_v)\), where the weights of the mixing network are generated by a hyper-network that takes the global state \(s_i\) as input. The OPT module is used to handle the global state \(s_i\) in this hyper-network of the mixing network.

Intuitively, we can explain the intrinsic mechanism of OPT from the perspective of subgraph mining. The disentangled interaction prototypes represent several interaction subgraphs.

The method we adapt to generate these prototypes is based on the self-attention module in Transformer, which can capture the relationship weights between each pair of entities. As shown in Fig. 3, we start by introducing the explicit disentangling step to encourage sparsity and diversity among discovered prototypes. Then we selectively restructure these prototypes by a learnable aggregator, unweighting the irrelevant prototypes and emphasizing the important ones.

B. Disentangling

The goal of the disentangling step is to explore the diverse interaction prototypes from the input observation or state of entities. The disentangling steps of the utility network and mixing network are similar except for different inputs. Thus, we use the mixing network as an example to describe this process. At each time step \(t\), the input state of the mixing network is defined as \(s_t = [s^1_t, \ldots, s^M_t] \in \mathbb{R}^{M \times d_e}\), where \(M\) is the number of the entities and \(d_e\) is the state representation dimension of the entity. We first encode the state representation of each entity \(s^a_t\) via an entity-wise embedding function \(f(\cdot) : \mathbb{R}^{d_e} \rightarrow \mathbb{R}^{d_e}\) and output the embedding feature \(x^a_t = f(s^a_t)\) with the dimension of \(d_e\). Then the input of the disentangling step is denoted as \(x_t = [x^1_t, \ldots, x^M_t] \in \mathbb{R}^{M \times d_e}\). We assume that entities treat each other equally before disentangling, forming a fully connected interaction pattern. Therefore, our method focuses on disentangling this fully connected pattern into several interaction prototypes.

Each interaction prototype can be described as an attention map where the relationship weights between entities are computed by the query with the corresponding key. To simplify the following description, the embedding matrix \(X\) is used instead of \(x_t\). Specifically, we first transform the input embedding \(X\) into queries \(Q \in \mathbb{R}^{M \times d_e}\), keys \(K \in \mathbb{R}^{M \times d_e}\), and values \(V \in \mathbb{R}^{M \times d_e}\):

\[
\begin{bmatrix}
Q \\
K \\
V
\end{bmatrix} = \begin{bmatrix}
W_Q \\
W_K \\
W_V
\end{bmatrix} \cdot \begin{bmatrix}
X \\
X \\
X
\end{bmatrix},
\]

where \(\varphi = \{W_Q, W_K, W_V\}\) are parameter matrices and each matrix \(W \in \mathbb{R}^{d_e \times d_e}\) is trainable. Then we can calculate the dot product for queries and keys to obtain the relationship weights between entities, followed by an activation function to normalize the output weights to a probability distribution.

The conventional activation function used for normalizing is the softmax function, which never assigns a probability of zero to any weights. Due to the strictly positive probabilities, we can never rule out the irrelevant entities for the interaction prototypes. This density is wasteful and noisy, making the exploration space more complex. In a particular interaction prototype, an entity usually interacts with only several crucial entities, which implies a sparsity constraint over the probability distribution. Thus, a sparse activation function is desired to focus on the important entities and ignore the irrelevant entities. Inspired by the sparse mechanism used in [70], [71], we introduce sparsemax as an alternative to softmax, which tends to yield sparse probability

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
distribution:
\[
\text{sparsemax}(z) := \arg\min_{p \in \Delta^M} \|p - z\|_2,
\]

where \(z \in \mathbb{R}^M\) is the input vector and \(p \in \mathbb{R}^M\) is the output vector. \(\Delta^M := \{p \in \mathbb{R}^M \mid p \geq 0, \|p\|_1 = 1\}\) is the probability simplex. Sparsemax realizes a Euclidean projection of the input vector onto the probability simplex, which is likely to hit the boundary of the simplex and obtain the sparse solution. The solution of (3) is obtained as follows:
\[
p = \lfloor z - \sigma(z)\rfloor_+,
\]
where \([\cdot]_+ = \max(0, \cdot)\) is a clipping function and \(\mathbf{1} \in \mathbb{R}^M\) denotes the all-one vector. \(\sigma(\cdot): \mathbb{R}^M \rightarrow \mathbb{R}\) is a threshold function. We first sort the elements of vector \(z\) in descending order to derive a sorted vector \(\tilde{z}\). Then the number of non-zero elements in the sparse solution \(p\) is calculated as
\[
m(\tilde{z}) = \arg\max_{m \in \{1, 2, \ldots, M\}} \left\{ m\tilde{z}_m > \sum_{i \leq m} \tilde{z}_i - 1 \right\}.
\]
Thus, the threshold value is given by
\[
\sigma(z) = \frac{\sum_{i \leq m}(\tilde{z}_i - 1)}{m(\tilde{z})}.
\]

It is easy to prove that \(\sigma(z)\) satisfies \(\sum_i[z_i - \sigma(z)]_+ = 1\) for every \(z\). In this way, we can directly calculate the sparse probability distribution based on the threshold value. Sparsemax can retain most of the important properties of softmax, and meanwhile assign exactly zero probability to low-scoring choices.

This sparsity enables us to discover similar subgroup entities and promotes the generalization of the generated interaction prototypes. Consequently, an interaction prototype is constructed by
\[
P = \text{sparsemax}\left(\frac{QK^T}{\sqrt{d_x}}\right),
\]
where \(\sqrt{d_x}\) is the scaling coefficient. We construct \(N\) interaction prototypes. Each prototype can be represented by \(P_n \in \mathbb{R}^{M \times M}\) with its own trainable parameter matrices \(\varphi_n\). Moreover, the corresponding value matrix is defined as \(V_n\). Then these prototypes will be used for the next restructuring step. However, without any other constraint, some generated interaction prototypes may contain a similar structure, degrading the disentanglement performance and capacity of the model. We therefore design an additional loss function in the disentangle step, aiming to avoid the similarity of the generated interaction prototypes.

The motivation of the additional loss function is that, a well disentangled interaction prototype should be clearly distinguished from the others. However, obtaining the solution that all the disentangled interaction prototypes differ from each other without any constraint will severely damage the performance. The distance between different prototypes can be infinitely enlarged, resulting in divergence of the optimization process. We thus constrain the solution by using a contrastive disagreement (CD) loss:
\[
\mathcal{L}_{\text{CD}}(\theta, \varphi) = \mathbb{E}_{n,e} \left[ -\log \left( \frac{\exp\left( (P_n^e V_n)^T P_n^e V_n \right)}{\sum_{i=1}^N \exp\left( (P_n^e V_n)^T P_i^e V_i \right)} \right) \right],
\]
where \(P_n^e \in \mathbb{R}^{1 \times M}\) is the interaction prototype of the entity \(e\). In the contrastive disagreement loss, we use the prototype itself as the positive sample and the other prototypes as the negative sample. In this way, this loss only focuses on the negative prototypes and constrains them to uniformly distribute on a hypersphere without divergence [72]. With the contrastive disagreement loss, we directly encourage the diversity among

Fig. 3. An illustrative diagram of the proposed interaction pattern disentangling (OPT) method, which mainly consists of two steps: disentangling and restructuring.
interaction prototypes. This diversity enables the model to provide discriminative interaction prototypes for different tasks.

C. Restructuring

In the restructuring step, we aim to selectively restructure the interaction prototypes to form the final compact pattern. The input embedding \( x_t \in \mathbb{R}^{M \times d_x} \) of the restructuring step is the same as the previous disentangling step. Firstly, we use a mean pooling operation to extract the global information \( \bar{x}_t \in \mathbb{R}^{d_x} \) from \( x_t \) over all entities. Then the prototype aggregator \( p_\theta : \mathbb{R}^{d_x} \rightarrow \mathbb{R}^N \) takes \( \bar{x}_t \) as input to derive the aggregation weights \( \omega \in \mathbb{R}^N \). The prototype aggregator \( p_\theta \) parameterized by \( \theta \) is a one-layer MLP network with softmax function. Then we use the aggregation weights \( \omega \) to restructure the interaction prototypes and output the final compact pattern \( Y = \sum_{n=1}^{N} \omega_n \mathbf{P}_n \mathbf{V}_n, \in \mathbb{R}^{M \times d_y} \). The restructuring processing could facilitate the extraction of global information while adapting to different interaction prototypes with learnable weights. Then the output pattern is employed for the following decision making in the utility network or credit assignment in the mixing network.

Although the global state could be computed directly in the mixing network, the observation information in the utility network is incomplete due to the partial observability, resulting in the instability of the aggregation weights. To prevent this, we propose a conditional mutual information (CMI) objective:

\[
I(\omega^a_t, \tau^a_{t-1} | o^a_t) = H(\omega^a_t | o^a_t) - H(\omega^a_t | \tau^a_{t-1}, o^a_t) \tag{9}
\]

where \( H \) is the entropy and the aggregator of the agent \( a \) is regularized not only conditioned on its observation but also its history behaviors. Therefore, we can stabilize the generated aggregation weights over time via maximizing the CMI objective. However, directly optimizing this objective is quite difficult, so we design a variational approximator \( q_\psi \) parameterized by \( \psi \) to approximate the posterior over \( \omega^a_t \) given the observed trajectories \([24], [29]\). The variational inference provides a lower bound of the CMI objective to optimize, which can be formalized as a loss function:

\[
\mathcal{L}_{\text{CMI}}(\theta, \phi, \psi) = \mathbb{E}_D \left[ \text{KL} \left[ p_\theta(\omega^a_t | \bar{x}_t^a) \parallel q_\psi(\omega^a_t | h^a_{t-1}, \bar{x}_t^a) \right] \right], \tag{10}
\]

where \( \text{KL}(\cdot) \) is the KL divergence function, \( \bar{x}_t^a \) is the input embedding of the observation \( o^a_t \) after mean pooling operation, and a GRU network \([73]\) is used to encode the history trajectory \( \tau^a_{t-1} \) into \( h^a_{t-1} \). Compared to directly relying on the aggregator on history behaviors, the CMI objective derives a tractable lower bound, which provides a more solid objective to optimize. The proof is given in the appendix, available online.

D. Optimization Objective

To sum up, training our framework based on the OPT module contains three main parts. The first one is the original TD loss \( \mathcal{L}_{TD} \), which enables each agent to learn its individual policy by optimizing the joint action value of the mixing network. The second one is the CD loss \( \mathcal{L}_{CD} \) to facilitate the sparsity and diversity among interaction prototypes when disentangling. The last one is the CMI loss \( \mathcal{L}_{\text{CMI}} \) to stabilize the aggregation weights for restructuring. Thus, given the three corresponding loss items, the total loss of our framework is formulated as follows:

\[
\mathcal{L}(\theta, \phi, \psi; \tau) = \mathcal{L}_{TD}(\theta) + \alpha \mathcal{L}_{CD}(\theta, \phi) + \beta \mathcal{L}_{\text{CMI}}(\theta, \phi, \psi), \tag{11}
\]

where the \( \alpha \) and \( \beta \) are coefficients. The overall framework is trained in an end-to-end centralized manner. We summarize the full procedure in Algorithm 1.

**Algorithm 1: OPT for MARL.**

Initialize: Utility network \( \theta_u \), mixing network \( \theta_v \), target network \( \theta_v^\tau = \theta_v \), replay buffer \( D \)

1: repeat
2: \( \triangleright \) Collect the data
3: Sample a task from \( T \)
4: Obtain the initial global state \( s_0 \)
5: while not terminal do
6: for each agent \( a \) do
7: Obtain the observation \( o^a_t = O(s_t, a) \)
8: Calculate the input embedding \( \bar{x}_t^a \)
9: \( \triangleright \) Disentangling
10: Calculate \( \{Q_{t,n}^a, K_{t,n}^a, V_n^a\}_{n=1}^N \)
11: Extract the interaction prototypes \( \{\mathbf{P}_n^a\}_{n=1}^N \)
12: \( \triangleright \) Restructuring
13: Calculate the aggregation weights \( \omega^a_t \)
14: Construct the final compact patterns \( Y_{t,n}^a \)
15: \( \triangleright \) Policy
16: Calculate \( h_t^a = \text{GRU}(Y_{t,n}^a, h_{t-1}^a) \)
17: Calculate the action values \( Q_t^a(h_t^a, \cdot) \)
18: Sample \( u_t^a \) from \( Q_t^a(h_t^a, \cdot) \) based on \( \epsilon \)-greedy
19: end for
20: Execute the joint action \( u_t = [u_1^a, \ldots, u_{|A|}^a] \)
21: Receive the reward \( r_{t+1} \) and the next state \( s_{t+1} \)
22: end while
23: Store the episode to the replay buffer \( D \)
24: \( \triangleright \) Train the networks
25: Sample episodes from the replay buffer \( D \)
26: Calculate joint action values with mixing network
27: Compute \( \mathcal{L}_{TD}(\theta) + \mathcal{L}_{CD}(\theta, \phi) + \beta \mathcal{L}_{\text{CMI}}(\theta, \phi, \psi) \)
28: Update the network parameters \( \theta, \phi, \psi \)
29: Update the target network \( \theta_v^\tau = \theta_v \) every \( C \) episodes
30: until converge
V. EXPERIMENTS

To demonstrate the effectiveness of the proposed interaction pattern disentangling (OPT) method, we conduct experiments on the StarCraft Multi-Agent Challenge (SMAC)\(^1\) [74], Google Research Football (GRF) [75] and the Predator-Prey Game (PPG) [63]. Our methods are compared with various state-of-the-art baseline methods in three settings: (1) single-task SMAC and GRF. (2) multi-task SMAC.\(^2\) (3) Zero-shot PPG.

We adopt the Python MARL framework (PyMARL) [74] to run all value-based experiments and the On-policy framework [40] to run all policy-based experiments. The hyperparameters of the baselines are the same as those in their source codes. The implementation of our OPT method is also based on the above frameworks. The implementation of our OPT method is also based on the above frameworks. In the SMAC and PPG experiments, we use QMIX [16] as the backbone based on the PyMARL framework. In the GRF experiments, we use MAPPO [40] as the backbone based on the On-policy framework. The detailed hyperparameters are given as follows, where the common training parameters across different baselines are consistent to ensure comparability.

In the OPT module in all experiments, we set the layer number \(K\) to 2, the interaction prototype number \(N\) to 4, and the interaction prototype dimension \(d_x\) to 32. The CD loss coefficient \(\alpha\) is set to 0.5, and the CMI loss coefficient \(\beta\) is set to 0.1. In the value-based experiments, batches of 32 episodes are sampled from the replay buffer with the size of 5 K every training iteration. The target update interval is set to 200, and the discount factor is set to 0.99. We use the RMSprop Optimizer with a learning rate of \(5 \times 10^{-4}\), a smoothing constant of 0.99, and no momentum or weight decay. For exploration, \(\epsilon\)-greedy is used with \(\epsilon\) annealed linearly from 1.0 to 0.05 over 50 K training steps and kept constant for the rest of the training.

We modify several hyperparameters of the value-based experiments in some difficult scenarios and detail them below. In the policy-based experiments, MAPPO has performed large-scale hyperparameter searching for different scenarios [40]. Thus, we directly adopt the same hyperparameters as in MAPPO to conduct policy-based experiments. In the following, we will first introduce the environments and the baselines, and then the results of different methods and ablations.

A. Single-Task Environments

We conduct experiments on 6 single-task SMAC scenarios [74] and 3 single-task GRF scenarios [75]. The SMAC scenarios are classified into Easy (10m_vs_11m), Hard (5m_vs_6m), and Super Hard (MMM2, corridor, 6h_vs_8z, 3s5z_vs_3s6z). Two of these scenarios are homogeneous (10m_vs_11m, 5m_vs_6m), where the army is composed of only a single unit type, while the others are heterogeneous. The GRF scenarios consists of academy_3_vs_1_with_
Fig. 5. Learning curves of our method and policy-based baselines in 6 single-task SMAC scenarios. We also provide the additional results of value-based QMIX and QPLEX using 20 M training timesteps to ensure comparability.

Fig. 6. Learning curves of our method and baselines in 3 single-task GRF scenarios.

keeper, academy_counterattack_easy, and academy_counterattack_hard. The goal of the agents is to score a goal against the opponent team. In the academy_counterattack_easy and academy_counterattack_hard scenarios, we follow the dense-reward setting [40] to reward agents for moving the ball closer to the opponent goal. In the academy_3_vs_1_with_keeper scenarios, to increase the learning difficulty, we adopt the sparse-reward setting where only scoring leads to the rewards.

We compare the proposed OPT with various MARL baselines: (1) Value-based methods: IQL [74], VDN [15], QMIX [16], QTRAN [22], QPLEX [17], OWQMIX and CWQMIX [23]. (2) Policy-based methods: MAPPO [40], HAPPO and HATRPO [41]. For the value-based methods, the training timestep is set to 2 M or 4 M in different scenarios. For the policy-based methods, the training timestep is set to 20 M due to the low sample efficiency. We also provide the additional results of value-based QMIX and QPLEX using 20 M training timesteps for comparison. For the super hard SMAC scenarios that require more exploration, the test win rates of value-based methods often remain 0% with the default hyperparameters. In this way, we extend the epsilon anneal time to 500 K, and two of them (corridor, 6h_vs_8z) optimized with the Adam Optimizer for all the compared value-based methods.

The experimental results in SMAC compared with the state-of-the-art value-based methods are shown in Fig. 4 and Table I. In both easy and hard homogeneous scenarios (10m_vs_11 m,
5m_vs_6m), our proposed OPT successfully improves the learning efficiency and the final performance. In the easy homogeneous scenario (10m_vs_11m), the size of the armed forces between the agents and enemies is similar. Thus, several baselines, including VDN and QMIX, can also achieve the promising results with additive or monotonic value factorization, while the exploratory benefit brought by interaction pattern disentangling is not obvious. However, in the more difficult homogeneous scenario (5m_vs_6m), our method consistently outperforms baselines by a large margin during training. The results suggest that the disentangling can be utilized to explore the diverse cooperation mechanisms, which helps the agents to construct a more useful policy and achieve non-trivial performance.

To further test the potentiality of our proposed method, we compare our method in the super hard SMAC heterogeneous scenarios (MMM2, corridor, 6h_vs_8z, 3s5z_vs_3s6z). In these challenging scenarios, extracting interaction patterns becomes complex due to the different unit types. Furthermore, there is a great disparity in strength between the two teams, and it is impossible to beat the enemies in a reckless way. The agents have to learn proper tactics to win the battle, while the sophisticated tactics are laborious to explore. In the MMM2 and 6h_vs_8z scenarios, OWQMIX and CWQMIX use a weighted projection that allows more emphasis to be placed on better joint actions, giving their improved ability in cooperative exploration. Even so, they inevitably fall into suboptimal policies with low test win rates. In contrast, the proposed OPT provides an impressive improvement in the performance over the baselines, showing its robustness to an increased rate and quality of exploration. Especially in the corridor and 3s5z_vs_3s6z scenarios, almost all compared baselines cannot learn any effective policy and perform poorly. Our method still maintains a high learning efficiency, and in practice, leads to the superior performance compared to the baselines.

The experimental results in SMAC compared with the state-of-the-art policy-based methods are shown in Fig. 5 and Table I.

| Method     | 10m_vs_11m | 5m_vs_6m | MMM2 | corridor | 6h_vs_8z | 3s5z_vs_3s6z |
|------------|------------|----------|------|----------|----------|--------------|
| IQL [74]   | 0.09 ± 0.04| 0.19 ± 0.03| 0.00 ± 0.00| 0.01 ± 0.01| 0.00 ± 0.00| 0.00 ± 0.00  |
| VDN [15]   | 0.55 ± 0.02| 0.44 ± 0.04| 0.03 ± 0.03| 0.05 ± 0.06| 0.00 ± 0.00| 0.05 ± 0.04  |
| QMIX [16]  | 0.61 ± 0.12| 0.33 ± 0.10| 0.17 ± 0.12| 0.02 ± 0.02| 0.12 ± 0.10| 0.11 ± 0.19  |
| QTRAN [22] | 0.21 ± 0.08| 0.27 ± 0.08| 0.01 ± 0.01| 0.00 ± 0.00| 0.00 ± 0.00| 0.00 ± 0.00  |
| QPLEX [17] | 0.50 ± 0.06| 0.48 ± 0.09| 0.05 ± 0.02| 0.00 ± 0.00| 0.05 ± 0.09| 0.01 ± 0.02  |
| OWQMIX [23]| 0.47 ± 0.10| 0.41 ± 0.08| 0.28 ± 0.11| 0.00 ± 0.00| 0.29 ± 0.17| 0.09 ± 0.08  |
| CWQMIX [23]| 0.49 ± 0.08| 0.36 ± 0.09| 0.24 ± 0.11| 0.00 ± 0.00| 0.23 ± 0.17| 0.05 ± 0.04  |

Area Under Curve

| Ours       | 0.68 ± 0.04| 0.61 ± 0.05| 0.54 ± 0.08| 0.59 ± 0.05| 0.53 ± 0.12| 0.38 ± 0.13  |
|------------|------------|------------|------------|------------|------------|--------------|
| QMIX       | 0.93 ± 0.01| 0.67 ± 0.06| 0.83 ± 0.02| 0.59 ± 0.31| 0.15 ± 0.30| 0.62 ± 0.10  |
| QPLEX      | 0.82 ± 0.03| 0.75 ± 0.05| 0.83 ± 0.02| 0.60 ± 0.14| 0.15 ± 0.30| 0.69 ± 0.17  |
| MAPPO      | 0.90 ± 0.04| 0.77 ± 0.03| 0.82 ± 0.04| 0.88 ± 0.03| 0.67 ± 0.12| 0.46 ± 0.29  |
| HAPPO      | 0.88 ± 0.06| 0.02 ± 0.00| 0.42 ± 0.24| 0.02 ± 0.03| 0.00 ± 0.00| 0.20 ± 0.21  |
| HATPPO     | 0.91 ± 0.01| 0.70 ± 0.04| 0.60 ± 0.14| 0.05 ± 0.04| 0.01 ± 0.00| 0.23 ± 0.12  |

Ours\(\dagger\)

\(\dagger\) corresponds to one standard deviation of the average evaluation over 5 trials. Bold indicates the best performance in each SMAC scenario. \(\dagger\) indicates the results using 20M training timesteps as shown in Fig.5.
can achieve gratifying performances on par with those obtained by the policy-based methods. Furthermore, OPT still provides competent efficiency and performance guarantees in all super hard scenarios except the corridor scenario where the final performance of OPT is only inferior to MAPPO. Besides, the additional results of QMIX and QPLEX show that more training timesteps enable value-based methods to converge to a higher performance, while OPT still maintains a better performance with a lower training cost.

Fig. 6 and Table II report the experimental results in GRF. In all scenarios, agents must learn diverse cooperative tactics to pass the ball and score the winning goal. Our proposed OPT successfully provides competent performance guarantees compared to the baselines. In the academy_3_vs_1_with_keeper scenario where agents are only rewarded for scoring, it is challenging for agents to find an effective policy due to the extreme sparsity of rewards. Thus, the compared baselines, except for MAPPO, all perform poorly. Despite the promising results achieved by MAPPO, MAPPO is easy to fall into local optimum. In contrast, OPT can benefit from interaction pattern disentangling to explore a more generalized cooperative policy. In the academy_counterattack_easy and academy_counterattack_hard scenarios, the difficulty of exploration increases as the number of entities increases. However, OPT still shows its robustness to cooperation promoting and offers impressive performance. Overall, these extensive results exhibit the advantage of boosting interaction pattern disentangling in single-task MARL, yielding significantly superior performance compared to the state-of-the-art counterparts.
TABLE III
THE AREA UNDER CURVE AND TEST WIN RATE OF OUR METHOD AND BASELINES IN 6 MULTI-TASK SMAC SCENARIOS

| Method       | 3-8csz | 3-8MMM | 3-8sz | 5-11csz | 5-11MMM | 5-11sz |
|--------------|--------|--------|-------|---------|---------|--------|
| Area Under Curve |       |        |       |         |         |        |
| Attn (VDN) [15] | 0.36 ± 0.03 | 0.18 ± 0.02 | 0.05 ± 0.01 | 0.29 ± 0.05 | 0.12 ± 0.01 | 0.01 ± 0.00 |
| Attn (QMIX) [16] | 0.22 ± 0.13 | 0.06 ± 0.08 | 0.05 ± 0.02 | 0.27 ± 0.06 | 0.06 ± 0.05 | 0.02 ± 0.00 |
| REFIL (VDN) [18] | 0.21 ± 0.03 | 0.12 ± 0.01 | 0.03 ± 0.01 | 0.14 ± 0.01 | 0.07 ± 0.01 | 0.00 ± 0.00 |
| REFIL (QMIX) [18] | 0.51 ± 0.05 | 0.35 ± 0.01 | 0.23 ± 0.02 | 0.48 ± 0.02 | 0.22 ± 0.01 | 0.10 ± 0.01 |
| Ours          | 0.65 ± 0.02 | 0.40 ± 0.03 | 0.32 ± 0.03 | 0.60 ± 0.03 | 0.33 ± 0.02 | 0.13 ± 0.02 |

| Test Win Rate |       |        |       |         |         |        |
|---------------|--------|--------|-------|---------|---------|--------|
| Attn (VDN) [15] | 0.52 ± 0.10 | 0.24 ± 0.04 | 0.06 ± 0.02 | 0.42 ± 0.10 | 0.15 ± 0.02 | 0.01 ± 0.01 |
| Attn (QMIX) [16] | 0.28 ± 0.27 | 0.08 ± 0.10 | 0.11 ± 0.07 | 0.56 ± 0.12 | 0.11 ± 0.09 | 0.01 ± 0.01 |
| REFIL (VDN) [18] | 0.39 ± 0.06 | 0.20 ± 0.04 | 0.06 ± 0.03 | 0.29 ± 0.08 | 0.12 ± 0.05 | 0.00 ± 0.00 |
| REFIL (QMIX) [18] | 0.66 ± 0.12 | 0.53 ± 0.07 | 0.44 ± 0.07 | 0.65 ± 0.08 | 0.27 ± 0.01 | 0.25 ± 0.05 |
| Ours          | 0.79 ± 0.06 | 0.53 ± 0.03 | 0.57 ± 0.06 | 0.75 ± 0.07 | 0.47 ± 0.10 | 0.27 ± 0.03 |

B. Multi-Task Environments

To further evaluate the proposed method in the multi-task setting, we conduct experiments on 6 multi-task SMAC scenarios, including 3 basic scenarios (3-8csz, 3-8MMM, 3-8sz) proposed by Iqbal and Sha [18] and 3 new harder scenarios (5-11csz, 5-11MMM, 5-11sz). The 3-8sz scenario contains 39 unique tasks of different scales, and the 3-8MMM and 3-8csz scenarios contain 66 tasks respectively. We follow the multi-task setting of Iqbal and Sha [18] and design 3 new harder scenarios: (1) 5-11csz contains two symmetrical teams, each of which is composed of 5 to 8 Stalkers/Zealots and 0 to 3 Colossi, resulting in 120 unique tasks. (2) 5-11sz contains two symmetrical teams, each of which is composed of 5 to 11 Stalkers/Zealots, resulting in 63 unique tasks. (3) 5-11MMM contains two symmetrical teams, each of which is composed of 5 to 8 Marines/Marauders and 0 to 3 Medics, resulting in 120 unique tasks. According to the difficulty of training, we can also divide these scenarios into three categories: Easy (3-8csz, 5-11csz), Hard (3-8MMM, 3-8sz) and Super Hard (5-11MMM, 5-11sz). We compare the proposed OPT with various multi-task baselines: VDN [15], QMIX [16], REFIL [18]. We realize the attention-based VDN and QMIX for the multi-task setting as in REFIL [18]. The models train simultaneously on multiple tasks sampled uniformly at each episode. For more exploration, we extend the epsilon anneal time to 500 K for all the compared methods.

The proposed method is particularly beneficial in this multi-task setting, which enables us to explore the diverse common interaction prototypes and generalize across various-scale tasks. Fig. 7 and Table III report the experimental results of all comparison methods on 6 multi-task SMAC scenarios. The results show that our proposed method consistently outperforms the baselines, which demonstrates the effectiveness of interaction pattern disentangling. In the easy scenarios (3-8csz, 5-11csz), REFIL can also learn effective cooperation strategies based on its random grouping mechanism. However, the latent interaction patterns still remain entangled in REFIL, which may severely downgrade the convergence rate. Conversely, our proposed OPT allows for the explicit shareable pattern disentangling across tasks, improving the learning efficiency and performance of the agents. In the hard 3-8MMM scenario, OPT achieves the performances on par with REFIL and still maintains a higher learning efficiency. Moreover, in the more difficult 3-8sz and 5-11MMM scenarios, OPT demonstrates its advantages in the
transferrability of the common prototypes and achieves gratifying results. In the super hard 5-11sz scenario, the performance gap between OPT and REFIL is not obvious, but most baselines including Attn (VDN), Attn (QMIX), REFIL (VDN) fail to learn any effective policy. To this end, OPT also successfully discovers several effective common interaction prototypes.

C. Zero-Shot Generalization Analysis

We further design a multi-task Predator-Prey Game [63] to test the zero-shot generalization ability of OPT. Specifically, \( N_a \) agents (predators), \( N_p \) prey and \( N_o \) fixed obstacles are initialized in a grid world, where \( \{N_a, N_p, N_o\} \) describes the scale of each task. Besides, each agent is equipped with an attack capability \( C_a \), and each prey is equipped with a defense capability \( C_p \). At each time step, agents with limited sight range can move, stop or capture, while prey randomly move and obstacles block movement. The goal of agents is to capture all prey, where one prey can be captured only when the total attack capability of agents is greater than or equal to the defense capability of the prey. The initialized scale and the capability are randomly sampled for each task during training. We conduct experiments on 3 multi-task Predator-Prey scenarios, where the agents are tested in the tasks with: (1) Unseen Capability. (2) Unseen Scale. (3) Unseen Capability & Scale (C & S).

Fig. 8 and Table IV report the zero-shot testing results of all comparison methods on 3 multi-task Predator-Prey scenarios, where our OPT consistently outperforms baselines by a large margin. All multi-task baselines, even the state-of-the-art REFIL, perform poorly in the unseen tasks. The results empirically suggest that the random grouping strategy of REFIL is limited to adopting policies in the trained tasks. In contrast, OPT disentangles the intertwined entity interactions in an explicit way, which encourages agents to extract the shareable interaction prototypes that can be generalized to the unseen tasks.

D. Ablations and Visualization

To understand the superior performance of OPT, we carry out ablation studies to test the contribution of its three main components: sparsemax, contrastive disagreement (CD) loss and conditional mutual information (CMI) loss. Following methods are included in the evaluation: (i) OPT without sparsemax (denoted by \( \text{w/o Sparse} \)); (ii) OPT without CD loss (denoted by \( \text{w/o CD Loss} \)); (iii) OPT without CMI loss (denoted by \( \text{w/o CMI Loss} \)). The results on both the single-task and multi-task scenarios are shown in Fig. 9(a). By comparing OPT without sparsemax and without CD loss, we can conclude that sparsity can help facilitate learning but the significant performance of OPT is mostly due to the restriction of CD loss. CD loss constrains the interaction prototypes to uniformly distribute on a hypersphere without divergence, which makes a major contribution to interaction pattern disentangling. Sparsemax mainly serves as an auxiliary means that constructs the more compact interaction prototypes. Moreover, CMI loss aims to stabilize the training process. When without CMI loss, the optimization results often derive a high standard deviation, especially in the multi-task scenario. This instability also leads to a significant drop in the performance.

Moreover, to study the impact of the interaction prototype number on the performance of OPT, we conduct an ablation study on the interaction prototype number.

---

**Table IV**

The area under curve and test win rate of our method and baselines in 3 multi-task predator-prey scenarios, where the results are provided by testing the trained agents on the unseen tasks.

| Method       | Unseen Capability | Unseen Scale | Unseen C & S |
|--------------|-------------------|--------------|--------------|
| **Area Under Curve** |                   |              |              |
| Attn (VDN) [15]  | 0.03 ± 0.00        | 0.10 ± 0.02  | 0.04 ± 0.01  |
| Attn (QMIX) [16] | 0.13 ± 0.06        | 0.20 ± 0.03  | 0.08 ± 0.02  |
| REFIL (VDN) [18] | 0.01 ± 0.00        | 0.08 ± 0.04  | 0.01 ± 0.01  |
| REFIL (QMIX) [18]| 0.04 ± 0.01        | 0.08 ± 0.01  | 0.04 ± 0.00  |
| **Ours**       | 0.55 ± 0.07        | 0.55 ± 0.08  | 0.37 ± 0.09  |

| Method       | Unseen Capability | Unseen Scale | Unseen C & S |
|--------------|-------------------|--------------|--------------|
| **Test Win Rate** |                   |              |              |
| Attn (VDN) [15]  | 0.05 ± 0.04        | 0.14 ± 0.03  | 0.05 ± 0.04  |
| Attn (QMIX) [16] | 0.37 ± 0.13        | 0.33 ± 0.17  | 0.07 ± 0.02  |
| REFIL (VDN) [18] | 0.01 ± 0.02        | 0.12 ± 0.07  | 0.01 ± 0.01  |
| REFIL (QMIX) [18]| 0.08 ± 0.08        | 0.20 ± 0.12  | 0.08 ± 0.04  |
| **Ours**       | 0.89 ± 0.05        | 0.78 ± 0.07  | 0.75 ± 0.06  |
Fig. 10. A visualization example of the common interaction prototypes that emerge across tasks in the 5-11csz scenario. Please zoom in for a better view.

(a) Each combat team of 2c3s5z is composed of 2 Colossi, 3 Stalkers and 5 Zealots. (b) Each combat team of 1c4s4z is composed of 1 Colossus, 4 Stalkers and 4 Zealots. Green and red shadows indicate the agent and enemy formations, respectively. Green and red arrows indicate the moving direction of the agent and enemy formations, respectively. White arrows indicate the attack action of the agents. (c) The interaction prototypes of 2c3s5z in the mixing network at phase 4. (d) The interaction prototypes of 1c4s4z in the mixing network at phase 4. The entities on the axes are arrayed by agents first and enemies later. Diverse common tactics based on the common prototypes emerge in combat tasks with different scales: (1) Hit and Run based on Prototype 1&4, i.e., drawing the fire, moving the agents away and fighting back again. (2) Coordinated Move and Force Fire based on Prototype 2, i.e., moving together and focusing fire on an enemy. (3) Sneak Attack based on Prototype 3, i.e., sneaking around behind the enemy and attacking the alone enemy.

study as shown in Fig. 9(b). When the number of interaction prototypes increases from 2 to 4, the performance gain of OPT becomes larger. This result verifies the effectiveness of the interaction pattern disentangling. Different interaction prototypes still remain entangled when the number of interaction prototypes is small, resulting in a worse performance. However, as the number of interaction prototypes becomes too large, disentanglement will be more challenging, yielding lower performance gains. After reaching the peak at 4, the performance slightly drops, but in general, our method is not very sensitive when the number of interaction prototypes is large. More interestingly, we find that the standard deviation of the results becomes smaller as the number of interaction prototypes increases, which indicates that providing more interaction prototypes for restructuring enables the agents to stabilize the final policy.

To further explain the learned interaction prototypes from OPT, we conduct a qualitative analysis. As shown in Fig. 10(a) and (b), we find that OPT can effectively grasp diverse tactics and utilize these common tactics across various-scale tasks. At the beginning of the battle, two combat teams first get closer to each other. Then our combat team assembles the agents into formations based on their unit types. Since Colossus is a heavy support unit with high defense and a large shooting range, Colossi learn to draw the fire by kiting the enemies. Colossi make the enemies give chase while maintaining enough distance, so that little or no damage is incurred. Moreover, Stalkers often tend to move together and focus fire on an enemy due to their high attack damage. Most interestingly, Zealots master the ability of sneak attack, which bypasses the disadvantages of low defense and low attack damage. The visualization of the interaction prototypes in Fig. 10(c) and (d) shows that our proposed method successfully disentangles the interaction pattern into several interaction prototypes that are sparse and distinguishable. Moreover, the interaction prototypes between different tasks are highly similar, which verifies the generalization of these prototypes. The shareable interaction prototypes enable our agents to construct and reuse the effective policy, and finally improve the performance of the model.
VI. CONCLUSION

In this work, we propose a novel interaction pattern disentangling approach, termed as OPT, that enables us to take advantage of the factorized interaction prototypes for multi-agent reinforcement learning. OPT follows a restructurings-by-disentangling scheme, where several distinguishable sparse prototypes are generated and then selectively reassembled together for the final focused policy. We validate OPT over the StarCraft II microgamemanagement, Google Research Football and Predator-Prey benchmarks, and showcase that it yields results significantly superior to the state-of-the-art techniques in both single-task and multi-task settings. To our best knowledge, this paper is the first attempt towards interaction pattern disentangling in MARL. In our future work, we will extend our method to design a more lightweight disentanglement framework for the large-scale problem, which is more challenging in terms of the exponential complexity as the number of agents increases.

REFERENCES

[1] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. Cambridge, MA, USA: MIT Press, 2018.
[2] V. Mnih et al., “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529–533, 2015.
[3] D. Silver et al., “Mastering the game of Go with deep neural networks and tree search,” Nature, vol. 529, no. 7587, pp. 484–489, 2016.
[4] W. Li, X. Wang, B. Jin, D. Luo, and H. Zha, “Structured cooperative reinforcement learning with time-varying composite action space,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 11, pp. 8618–8634, Nov. 2022.
[5] R. Akour, D. Tateo, and J. Peters, “Continuous action reinforcement learning from a mixture of interpretable experts,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 10, pp. 6795–6806, Oct. 2022.
[6] Q. Li, Z. Peng, L. Feng, Q. Zhang, Z. Xue, and B. Zhou, “MetaDrive: Composing diverse driving scenarios for generalizable reinforcement learning,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 3, pp. 3461–3475, Mar. 2023.
[7] T. Vinyals et al., “Grandmaster level in StarCraft II using multi-agent reinforcement learning,” Nature, vol. 575, no. 7782, pp. 350–354, 2019.
[8] M. Jaderberg et al., “Human-level performance in 3D multiplayer games with population-based reinforcement learning,” Science, vol. 364, no. 6443, pp. 859–863, 2019.
[9] J. Wu, J. Wang, L. Codecá, and Z. Li, “Multi-agent deep reinforcement learning for large-scale traffic signal control,” IEEE Trans. Intell. Transp. Syst., vol. 21, no. 3, pp. 1086–1095, Mar. 2020.
[10] T. Wu et al., “Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks,” IEEE Trans. Veh. Technol., vol. 69, no. 8, pp. 8243–8256, Aug. 2020.
[11] Z. Yan and Y. Xu, “A multi-agent deep reinforcement learning method for cooperative load frequency control of a multi-area power system,” IEEE Trans. Power Syst., vol. 35, no. 6, pp. 4599–4608, Nov. 2020.
[12] X. Xu, Y. Jia, Y. Xu, Z. Xu, S. Chai, and C. S. Lai, “A multi-agent reinforcement learning-based data-driven method for home energy management,” IEEE Trans. Smart Grid, vol. 11, no. 4, pp. 3201–3211, Jul. 2020.
[13] J. Wang, W. Xu, Y. Gu, W. Song, and T. Green, “Multi-agent reinforcement learning for active voltage control on power distribution networks,” in Proc. Int. Conf. Neural Inf. Process. Syst., 2021, pp. 3271–3284.
[14] K. Zhang, Z. Yang, and T. Bašar, “Multi-agent reinforcement learning: A selective overview of theories and algorithms,” 2019, arXiv: 1911.10635.
[15] T. Sunehag et al., “Value-decomposition networks for cooperative multi-agent learning based on team reward,” in Proc. 17th Int. Conf. Auton. Agents MultiAgent Syst., 2018, pp. 2085–2087.
[16] T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. N. Foerster, and S. Whiteson, “QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 4295–4304.
A Concise Introduction to Decentralized Learning with Branching and Sparsification

Shunyu Liu received the BEng degree in software engineering from the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China, in 2019. He is currently working toward the PhD degree with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. His current research interests include reinforcement learning, multi-agent learning, and power system control.

Jie Song (Member, IEEE) received the BEng degree in computer science and technology from Sichuan University, Chengdu, China, in 2015, and the PhD degree in computer science and technology from the College of Computer Science and Technology, Zhejiang University, Hangzhou, China, in 2020. He is currently an associate professor with the College of Software Technology, Zhejiang University. His research interests include transfer learning, model compression, and interpretable machine learning.

Yihe Zhou received the BEng degree in computer science and technology from the College of Computer Science and Technology, Zhejiang University, Hangzhou, China, in 2022. He is currently working toward the MS degree with the College of Computer Science and Technology, Zhejiang University. His current research interests include deep learning and reinforcement learning.

Na Yu received the BSc degree in information and computing science from the College of Science, Northeast Agricultural University, Harbin, China, in 2018. She is currently working toward the PhD degree with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. Her current research interests include deep learning, automatic feature engineering, and feature disentanglement.
Kaixuan Chen received the BEng degree in software engineering from the School of Computer Science and Engineering, Jiangsu University of Science and Technology, Zhenjiang, China, in 2016, and the MS degree in computer technology from the School of Internet of Things, Jiangnan University, Wuxi, China, in 2019. He is currently working toward the PhD degree with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. His current research interests include deep learning, graph representation learning, and Riemannian manifold learning.

Zunlei Feng received the BEng degree in software engineering from Soochow University, Suzhou, China, in 2014, and the PhD degree in computer science and technology from the College of Computer Science and Technology, Zhejiang University, Hangzhou, China, in 2019. He is currently an associate professor with the College of Software Technology, Zhejiang University. His research interests include computer vision, medical image analysis, and model optimization.

Mingli Song (Senior Member, IEEE) received the PhD degree in computer science from Zhejiang University, Hangzhou, China, in 2006. He is currently a professor with the College of Computer Science and Technology, Zhejiang University. His research interests include image information processing, machine vision and pattern recognition, deep model knowledge amalgamation, and graph machine learning. He was awarded the Microsoft Research Fellowship in 2004.