Prioritized Tasks Mining for Multi-Task Cooperative Multi-Agent Reinforcement Learning

Yang Yu
School of Artificial Intelligence, University of Chinese Academy of Sciences
Institute of Automation, Chinese Academy of Sciences
Beijing, China
yuyang2019@ia.ac.cn

Qiyue Yin
School of Artificial Intelligence, University of Chinese Academy of Sciences
Institute of Automation, Chinese Academy of Sciences
Beijing, China
qyyin@nlpr.ia.ac.cn

Junge Zhang
School of Artificial Intelligence, University of Chinese Academy of Sciences
Institute of Automation, Chinese Academy of Sciences
Beijing, China
jgzhang@nlpr.ia.ac.cn

Kaiqi Huang
School of Artificial Intelligence, University of Chinese Academy of Sciences
Institute of Automation, Chinese Academy of Sciences
CAS Center for Excellence in Brain Science and Intelligence Technology, Beijing, China
kqhuang@nlpr.ia.ac.cn

ABSTRACT

Multi-task learning improves data efficiency in cooperative multi-agent reinforcement learning, since agents can learn multiple related tasks simultaneously and the cooperation knowledge in a task can be utilized by others. However, existing methods mainly learn multiple cooperation tasks uniformly, regardless of their complexity and significance. In this paper, we propose a new framework called Prioritized Tasks Mining (PTM) for multi-task cooperation problems, which helps agents to identify and mine higher priority cooperation tasks, so as to learn more effective coordinated strategies for multiple cooperation tasks. Specially, agents will use the hindsight during training to identify the priority of different tasks, and make an exploration and exploitation in higher priority cooperative tasks to mine more sophisticated coordinated strategies. We evaluate PTM in challenging multi-task StarCraft micromanagement games with different scales, and results demonstrate that our method consistently outperforms all strong baselines.

KEYWORDS

Multi-Task Learning; Multi-Agent Cooperation; Reinforcement Learning

ACM Reference Format:
Yang Yu, Qiyue Yin, Junge Zhang, and Kaiqi Huang. 2023. Prioritized Tasks Mining for Multi-Task Cooperative Multi-Agent Reinforcement Learning. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

1 INTRODUCTION

Cooperative multi-agent reinforcement learning (MARL) has drawn increasing interest in recent years, since it can help address many challenging cooperative games, such as football games and DOTA2 [2, 12]. However, most cooperative MARL algorithms focus on the cooperative tasks with fixed team composition, and when cooperating with varying quantities or types of teammates, they need to retrain new coordinated strategies for agents from scratch. On the other hand, multi-task learning remains promising to improve the data efficiency in cooperative MARL, where a shared network learns to master multiple related cooperative tasks, and common cooperative knowledge can be transferred across these related tasks to improve data efficiency. Thus, the research of multi-task cooperative multi-agent reinforcement learning is important.

More recently, there are works focusing on the multi-task learning problem in cooperative MARL. Most approaches aim to design network architecture for agents to fit tasks with different observation and action spaces in multi-task setting, and utilize common cooperative knowledge in these related tasks by the shared network directly [8, 9]. Besides, there are works introducing auxiliary tasks to further help agents learn reusable cooperative pattern in related tasks, where common cooperative patterns are learned by learning value of the factorization of the team entities [10]. However, although the aforementioned approaches have made progress in multi-task cooperative reinforcement learning, they mainly learn different cooperative tasks uniformly and neglect the complexity and contribution that can be derived by different tasks. For example, in cooperative team-battle games 5 vs. 5 and 5 vs. 6, in the latter, agents need to learn more samples to win this harder asymmetric game. Moreover, more sophisticated coordinated strategies will also be discovered when agents try to conquer this game. In other words, the task 5 vs. 6 deserves more attention. Hence when learning in multiple cooperative MARL tasks with limited learning resources, an adjustment of the focus on different tasks is needed, and more focus on those noteworthy tasks is likely to help agents find more effective coordinated strategies that are suitable to the learning in multiple tasks.

However, due to the complexity in multi-agent system, it is challenging to identify the value of different cooperation tasks. Besides, how to effectively deal with those noteworthy cooperation tasks is also unknown. In this paper, to address above challenges existing

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

in multi-task cooperation problem, we propose a new framework called Prioritized Tasks Mining (PTM), which can automatically identify the priority of different tasks using the hindsight during training, and encourages agents to make exploration and exploitation in these preferential tasks to efficiently mine more effective and general coordinated strategies.

More specifically, in order to determine the priority of different cooperation tasks, we adopt the hindsight discovered during training. By training with different tasks, agents will obtain their performance corresponding to different tasks, which can be used to help identify the priority of different tasks, where the tasks with lower performance are more needed to be focused. This design is inspired by human beings. When humans want to master multiple tasks, they will also check the learning process of different tasks, and pay more attention to tasks they are not good at. And when agents try to overcome tasks they are not capable, they will discover more sophisticated skills [1, 5, 11, 15]. However in Cooperative MARL, though the priority of tasks can be determined, how to effectively utilize higher priority tasks to discover more sophisticated coordinated strategies is still a problem. To this end, we propose to do exploration and exploitation in higher priority tasks separately. The exploration wants to help agents find effective strategies. Considering that agents need to coordinate with different teammates in multi-task cooperation problem, we encourage agents to explore coordinated strategies that are robust to the variation of teammates by using the dropout method in the multi-agent team. On the other hand, to effectively transfer the coordinated knowledge learned in high priority tasks, we encourage agents to learn cooperation patterns emerging in such tasks by learning the randomized factorization of the team values.

Finally, we evaluate our method PTM with strong baselines in challenging multi-task StarCraft micromanagement games with different scales. Results demonstrate that PTM achieves a superior performance when learning in multiple cooperation tasks, and surpasses all strong multi-task learning baselines consistently.

We summarize our contributions as follow:

- We propose a novel method called Prioritized Tasks Mining to adjust the learning focus of different tasks in multi-task cooperation problem.
- We achieve an adaptive priority identification method with the help of the hindsight during training.
- We achieve an effective mining of higher priority tasks from the perspective of exploration and exploitation in such tasks, which help find more sophisticated and general coordinated strategies.
- Our method PTM consistently outperforms existing state-of-the-art methods in multi-task cooperative games with different scales.

2 RELATED WORK

2.1 Multi-Task Reinforcement Learning

Multi-task reinforcement learning aims to help agents utilize common knowledge contained in multiple related reinforcement learning tasks and performs better than learning each task independently [30]. There are two key factors in multi-task reinforcement learning. One is to identify the knowledge that can be shared or reused across multiple tasks, and the other is to make a balance between multiple tasks that are competing for limited resources of the learning system. In the first aspect, there are works proposing to learn the shared representations across multiple complex tasks considering the strong ability of neural network in state abstraction [3], or reusing partial neural network by designing progressive network or modular network [20, 32]. In the second aspect, there are works focusing on the balance when facing multiple reinforcement learning tasks. For example, in order to avoid the negative interference from other tasks, [27] proposed to learn a shared policy that distills the knowledge of each task policy instead of sharing parameters across different tasks directly. Besides, there are works finding that different density of magnitude of rewards in each task makes some tasks more salient than others, and proposing PopArt to automatically adapt the contribution of each task to the update of the network [7]. However, although great progress has been made in multi-task reinforcement learning, they all focus on the area of single-agent decision problems, while how to abstract common cooperative pattern and how to effectively balance multiple cooperative tasks are still unknown.

2.2 Multi-Task Cooperative Multi-Agent Reinforcement Learning

More recently, there are works beginning to study the problem of multi-task cooperative multi-agent reinforcement learning, *i.e.* learning coordinated policies in different but related cooperation tasks parallelly. Considering the fact that in different cooperative tasks, the quantity or the type of the teammates may be different, making the state or action space change, there are works studying the network design that is suitable to such varying state or action space. Then common knowledge will be transferred by this shared network [8, 9]. Besides, there are works studying how to leverage common cooperative patterns emerging among different cooperative tasks, by learning and transferring the value of randomly factorized team groups in different tasks[10]. Although these methods makes the learning of multi-task cooperative MARL more efficiently, they ignore the complexity and potential contribution of different tasks in multi-task cooperation problem. Due to the complexity of multi-agent system, the requirement of cooperation ability in different tasks is different. Besides, the potential of different tasks for agents to find more effective and general coordinated strategies is also different. However, existing methods make agents learn different tasks with no difference, and with limited learning resources, agents are likely to be dominated by tasks that are easy to learn and lose passion to find more effective strategies. Thus in this paper, we aim to take a step towards making a trade-off among multiple tasks to help agents learn more sophisticated strategies.

3 BACKGROUND

3.1 Problem Formulation

Cooperative MARL is the extension of reinforcement learning to help address cooperative multi-agent sequential decision problems, which are usually modeled as Dec-POMDP [34]. In this paper we consider the setting of Cooperative MARL with entities, described as Dec-POMDP with entities by $G = \langle S, U, P, r, O, A, \mathcal{E}, \mu, \gamma \rangle$ [22]. $\mathcal{E}$ is the space of entities, including the agent entity $a \in A \subseteq$

$\mathcal{E}$ and non-agent entities, such as the landmark or obstacles in the environment. For any $e \in \mathcal{E}$, it has a d_e dimensional state information $s^e \in \mathbb{R}^{d_e}$, and the global state is the set $S = \{s^e | e \in \mathcal{E}\}$. At each step, each agent a chooses an action $u^a \in U$ and the joint action is $\mathbf{u} = \{u^a | a \in A\}$. P is the state transition function which maps the current state s and the joint action $\mathbf{u}$ to the next state s', *i.e.* $P(s'|s, \mathbf{u})$. All agents receive a shared reward $r(s, \mathbf{u}) \in \mathbb{R}$. Note that the environment is partial observable, thus there is a observability function: $\mu(a, e) \in \{0, 1\}$, indicating whether agent a can observe entity e. The observation of the agent is $o^a = \{s^e | \mu(a, e) = 1, e \in \mathcal{E}\}$. The observation mask among agents and all entities is $M^\mu \in \mathbb{R}^{|A|\times|\mathcal{E}|}$, where $M^\mu[a, e] = \mu(a, e)$. $\gamma \in [0, 1)$ is the discount factor.

3.2 Cooperative MARL Methods

In order to address the non-stationary issue and the scalability problem in cooperative MARL [6, 14, 26, 34], existing cooperative MARL algorithms mainly adopt centralized training and decentralized execution (CTDE) paradigm. CTDE means during training, the learning algorithm has access to the full state and the action-observation histories of all agents, while during execution each agent makes decisions conditioning on its own local action-observation history. Based on CTDE paradigm, many value-based [17–19, 31] and policy-based [4, 13, 33, 35] cooperative MARL algorithms have been developed. Among these Cooperative MARL methods, value decomposition methods are the representative of value-based algorithms and become predominant in this area. In this paper, we also adopt the widely used paradigm CTDE, and base our method PTM on the representative value decomposition method QMIX [19] introduced later.

3.3 Value Decomposition Methods and QMIX

As the mainstream of value-based CTDE methods, value decomposition methods try to learn a joint state-action value function $Q^{tot}(s, \mathbf{u})$ to maximize the expected team rewards sum. $Q^{tot}(s, \mathbf{u})$ is usually represented by a neural network $Q^{tot}_\theta(s, \mathbf{u})$ or $Q^{tot}_\theta(\tau, \mathbf{u})$ in a partial observable environment, where τ is the trajectory consists of the observation and action of each agent. To learn this value function, Q-learning [25] is used and the loss function is:

$$\begin{aligned} L(\theta) &= E\left[\left(y^{tot}_t - Q^{tot}_\theta(\tau_t, \mathbf{u_t})\right)^2 | (\tau_t, \mathbf{u_t}, r_t, \tau_{t+1} \sim D)\right] \\ y^{tot}_t &= r_t + \gamma Q^{tot}_{\bar{\theta}}\left(\tau_{t+1}, \arg\max_{\mathbf{u_{t+1}}} Q^{tot}_\theta(\tau_{t+1}, \mathbf{u_{t+1}})\right) \end{aligned} \tag{1}$$

where D is the replay buffer and $\bar{\theta}$ is the target network periodically copied from θ for stable training [16]. The action input to the target network is chose by Q^{tot}_θ as advised by [28].

To enable decentralized execution, value decomposition methods factorize the $Q^{tot}(\tau_t, \mathbf{u_t})$ into the combination of per agent utility represented by $Q^a(\tau^a, u^a)$. For example, as a classical value decomposition method, QMIX factors the joint action-value Q^{tot} into a monotonic nonlinear combination of individual utilities Q^a. Therefore,

$$Q^{tot} = g\left(Q^1(\tau^1, u^1; \theta_Q), \cdots, Q^{|A|}(\tau^{|A|}, u^{|A|}; \theta_Q); \theta_g\right) \tag{2}$$

where θ_Q are the parameters of the agent utility network and θ_g are the nonnegative parameters for the mixing network. Note that the nonnegative parameters in θ_g are generated by a hyper-network with the input of the global state, which making the utilities of agents to be combined according to the current global state. The nonnegativity in the mixing network ensures that $\frac{\partial Q^{tot}((\tau,\mathbf{u})}{\partial Q^a(\tau^a,u^a)} \geq 0$, which in turn guarantees Individual-Global-Max (IGM) Condition [23], *i.e.* the optimal joint actions across agents are equivalent to the collection of individual optimal actions of each agent. QMIX is effective for decentralized execution, since during execution, each agent can independently make decisions by its own utility network without considering any global information. Because of the predominant performance in massive challenging multi-agent cooperation tasks, QMIX is widely used in MARL area. In this paper we build our work upon this method.

3.4 Attention QMIX

In particular, due to the varying state space in multi-task cooperative MARL, we base our method on Attention QMIX [9], which can deal with cooperative tasks with variable number of teammates. Specifically, a multi-head attention (MHA) layer [29] is augmented to the utility network of each agent. The input of the agent consists of two entries, (X, M^μ). $X \in \mathbb{R}^{|\mathcal{E}|\times d_h}$ represents the input features of each entity converted from the global state s, and d_h is the dimension of the feature of each entity. $M^\mu \in \mathbb{R}^{|A|\times|\mathcal{E}|}$ is the observation mask, where $M^\mu[a, e] = \mu(a, e)$ indicates whether agent a can observe the entity e. With these two entries, MHA layer can integrates the information of observable agent and non-agent entity with learnable attention weights. Then the integrated information of each agent is used to help calculate the agent utility Q^a. Besides, for the mixing network of QMIX, which requires the input of the global state to generate weights, a MHA layer is also used here to deal the global state with varying length. The input of the mixing network is (X, M^*), where the full observation is achieved by $M^*[a, e] = 1$. Then the integrated information of all agent and non-agent entities is used to generate weights to help mixing network combine the utility of each agent Q^a to calculate the joint state-action value Q^{tot}.

4 METHOD

As discussed above, PTM aims to adjust the focus on multiple tasks considering the potential contribution and complexity of different cooperation tasks, and focuses more on the mining of tasks with higher priority. In this section, we will introduce how to automatically identify the priority of different tasks, and how to handle and utilize these higher priority tasks effectively. The framework and pseudo-code are provided in Figure 1 and Alg. 1.

4.1 Higher Priority Tasks Identification

For multi-task cooperation problem, it is challenging to manually construct a set of cooperation tasks that are both related and have nearly same cooperation difficulty. Besides, due to the complexity of multi-agent system, it is also hard to determine the priority of different cooperation tasks just from the team composition in advance. In this paper, we propose a priority identification method inspired by humans. In general, when humans want to master multiple tasks, they will focus their studies on different tasks and pay more attention to those they learn not well [11, 15]. Similarly, in this paper,

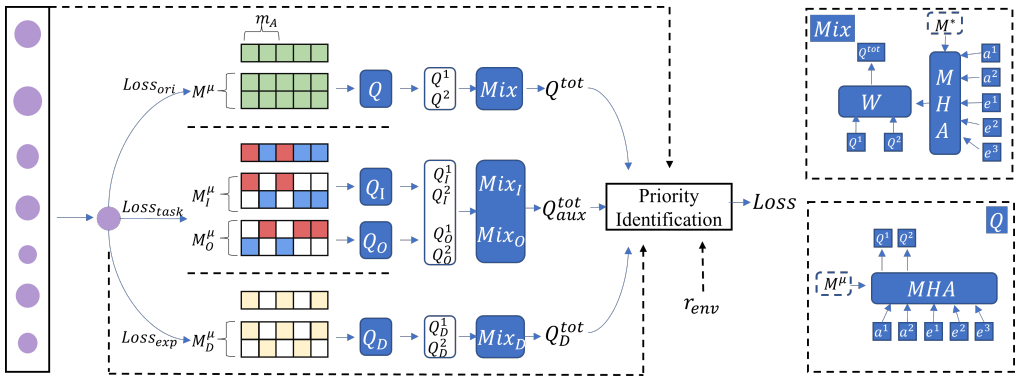


Figure 1: The framework of PTM. PTM aims to focus more on higher priority tasks. For the episode samples in the batch shown in purple, first it will be used to calculate the original loss in attention QMIX shown in the first row. In attention QMIX, agents in m_A will integrate all entities' information according to the observation mask M^μ in green, then output their utilities Q^a as shown in the utility network Q (bottom right). Here MHA is the multi-head attention layer. Then the mixing network in QMIX will combine all agent utilities to obtain the joint state-action value estimation Q^{tot} to approximate the team return, where the combination weights are generated by the attention to all entities with full observation mask M^* as shown in the mixing network Mix (top right). Then to encourage the mining in higher priority tasks, the sample will also be used to calculate the exploitation and exploration loss in the second and third branch. For exploration, agents need to learn with randomly dropped or remaining entities to find more robust coordinated strategies, with the input of the attention mask M_D^μ in yellow in the agent utility network Q and mixing network Mix. For exploitation, agents learn their utilities in two disjoint groups shown in red and blue, which are the factorization of the team value, by inputting the in-group attention mask M_I^μ and out-group attention mask M_O^μ to the network Q and Mix. Finally, PTM combines the original loss $Loss_{ori}$ and the loss $Loss_{exp}$ and $Loss_{task}$ used for higher priority tasks mining, according to the training performance of the task the sample belonging to.

instead of learning all tasks uniformly, we also encourage agents to focus their learning on different tasks, and give more priority to tasks they cooperate not well. Actually, compared to other learning tasks, the task with a lower cooperation rate often requires more sophisticated coordinated strategies, and thus the focus on them is likely to help agents discover more effective coordination [1, 5].

Concretely, the mastery of different tasks can be seen as the hindsight during human learning process. Similarly, when agents learn in multiple cooperation tasks, this kind of hindsight also emerges during training. When trained in multiple cooperation tasks, agents will obtain cooperation-related indicators of different tasks provided by the environment, such as the success rate of cooperation in full cooperation game or the win rate in team-battle game. We use c_i to represent this kind of cooperation indicator for task i, which is recorded and updated during training. Then for the episode sample x_i in the training batch, its priority ω_i is defined as:

$$\omega_i = \frac{exp(1 - c_{f(x_i)})}{\sum_i^B exp(1 - c_{f(x_i)})} \tag{3}$$

where f is the function mapping the sample x_i to its task index, and B is the number of samples in the batch.

Thus with the hindsight during training discussed above, we can obtain the priority of different tasks. Then in order to achieve a full use of these higher priority tasks, we encourage agents to mine these tasks from the perspective of exploration and exploitation separately, which are beneficial to the discovery and reuse of effective coordinated strategies in higher priority tasks.

Algorithm 1 PTM

Init: parameters for the agent network and mixing network θ_Q, θ_h
Init: buffer $D = \{\}$, ϵ, α, λ, task cooperation indicators $\{c_i\}_{i=1}^N = 0$
for each episode iteration **do**
  Agents interact with the environment based on their current policies θ_Q with ϵ-greedy strategy
  Update the cooperation indicators $\{c_i\}_{i=1}^N$ according to the performance of agents in each task
  Add tuples $\{\tau_t, \boldsymbol{u_t}, s_t, \tau_{t+1}, \boldsymbol{u_{t+1}}, r_t\}$ to D
  for each gradient step **do**
    Sample $\{\tau_t, \boldsymbol{u_t}, s_t, \tau_{t+1}, \boldsymbol{u_{t+1}}, r_t\}$ from replay buffer D
    Calculate the priority ω_i for samples by Eq. (3)
    Calculate the original QMIX loss $Loss_{ori}$ by Eq. (7)
    Calculate the exploration loss $Loss_{exp}$ by Eq. (10)
    Calculate the exploitation loss $Loss_{task}$ by Eq. (13)
    Update θ_Q, θ_h by the overall loss with ω_i, $Loss_{ori}$, $Loss_{exp}$, $Loss_{task}$ by Eq. (15)
  end for
end for

4.2 Exploration in Higher Priority Tasks

As discussed above, tasks with higher priority means that agents with current policies are not good at cooperating in such tasks, and thus they need to find other more effective coordinated policies. To this end, we encourage agents to do exploration in such tasks. However, it is worth noting that the problem is in a multi-task setting, where the state space of each task is different, and the joint

exploration in the whole state space is challenging. Thus, it is not tractable to apply traditional exploration methods which encourage agents to explore more novel or uncertain state. More importantly, whether novel states are helpful in such multi-task cooperation problem is unknown. To avoid the challenges above and achieve a more effective exploration that is suitable to multi-task cooperation setting, we propose a new exploration mechanism, which encourages agents to find more robust strategies to the variation of other entities.

Inspired by Dropout [24], which is used to mitigate the overfitting problem in neural network by randomly dropping units from the network during training, we encourage agents to learn their utilities in the team with randomly dropped entities. We use this dropout mechanism to prevent the co-adaption between entities, and aim to train more effective coordinated strategies that are robust to the variation of other entities, which is the typical problem existing in multi-task setting. Specifically, for an episode sample from higher priority tasks with $|\mathcal{E}|$ entities, the team with randomly dropped entities can be represented by a binary vector $m_D \in \{0,1\}^{|\mathcal{E}|}$, where m_e indicates whether entity e is dropped, and the subset of agent entities in all entities is denoted as $m_A : [m_a]_{a \in A}$. Recall that in attention QMIX, agents need to integrate the information of all entities in the team to make decisions, while in team with dropped entities, agents can only use information from remaining entities. We achieve this by calculate the attention mask in team with dropped entities, which is:

$$M_D = m_A m_D^T \vee \neg m_A \neg m_D^T \tag{4}$$

where $\neg m_A$ is the negation of m_A, and $M_D[a, e]$ indicates whether a can observe the information of the entity e in the dropout team. Then considering the problem is partial observable, the attention mask M_D need to be processed by the observation mask M^u as:

$$M_D^u = M_D \wedge M^u \tag{5}$$

which will be used to help agents integrate the information from remaining entities.

Note that in attention QMIX discussed above, agents integrate the information of entities they can observe by inputting the feature of entities and the observation mask M^μ to the MHA layer in the agent utility network, which will then output agents utilities $Q^1(s; \theta_Q, M^\mu), \cdots, Q^{|A|}(s; \theta_Q, M^\mu)$ separately. Then with the help of the mixing network in attention QMIX, the utilities of agents $Q^1, \cdots, Q^{|A|}$ are combined to calculate the joint state-action value Q^{tot}, *i.e.*

$$Q^{tot} = g(Q^1, \cdots, Q^{|A|}; h(s; \theta_h, M^*)) \tag{6}$$

where θ_h are the parameters of the hyper-network h to generate the weights of the mixing network g, and M^* is a full observable mask to help integrate the global state information in the hyper-network. Then the network parameters θ_Q, θ_h will be optimized by the original loss in QMIX:

$$Loss_{ori} = E\left[\left(y_t^{tot} - Q^{tot}(\tau_t, \mathbf{u_t})\right)^2 | (\tau_t, \mathbf{u_t}, r_t, \tau_{t+1} \sim D)\right] \tag{7}$$

$$y_t^{tot} = r_t + \gamma Q_{\bar{\theta}}^{tot}\left(\tau_{t+1}, \arg\max_{\mathbf{u_{t+1}}} Q_\theta^{tot}(\tau_{t+1}, \mathbf{u_{t+1}})\right) \tag{8}$$

where θ consists of θ_Q and θ_h.

Similarly, the attention mask M_D^u is used to calculate the utilities of agents in dropout team, $Q_D^1(s; \theta_Q, M_D^\mu), \cdots, Q_D^{|A|}(s; \theta_Q, M_D^\mu)$. Then these agent utilities in dropout team will be combined directly to approximate the team value, *i.e.*

$$Q_D^{tot} = g(Q_D^1, \cdots, Q_D^{|A|}; h(s; \theta_h, M_D)) \tag{9}$$

and the exploration loss is defined as:

$$Loss_{exp} = E[(y_t^{tot} - Q_D^{tot}(\tau_t, \mathbf{u_t}))^2 | (\tau_t, \mathbf{u_t}, r, \tau_{t+1}) \sim D] \tag{10}$$

where y^{tot} is calculated using real episode samples as introduced in Equation (8).

By optimizing the loss above, agents learn their utilities in a dropout team, and will try to find strategies that are not overfitted to fixed team composition, thus the co-adaptation between agents and other entities is mitigated. This exploration mechanism which avoids co-adaptation is suitable to the multi-task cooperation setting, since the overfitting to one fixed team composition with strong cooperation performance is at the cost of other cooperation tasks.

4.3 Exploitation of Higher Priority Tasks

Once the higher priority tasks are identified and the exploration in such tasks are implemented, more sophisticated coordinated strategies will be discovered in such tasks. Now the problem is how to utilize the effective cooperation experience found in such higher priority tasks to help the learning in multiple tasks.

In the following, we will introduce how PTM exploits these higher priority tasks. Specifically, instead of straightly learning the cooperative knowledge in higher priority tasks by directly reweighting the loss from its samples, we encourage agents to explicitly learn cooperative patterns existing in such tasks. Note that learning a task by directly reweighting the loss of its samples only encourages agents to learn the value or contribution of each agent in the whole team, which is unlikely to be reused in other related but different tasks. Hence we encourage agents to learn value of the agent in a sub-group within the team, where such smaller cooperative patterns might also exist in related cooperative tasks and thus the learning of them is more likely to be useful in other tasks.

Here we learn cooperative patterns in hard tasks referring to [10], where the cooperative patterns can be learned by learning the value of the agent in a randomly selected sub-group within the whole team. In multi-agent cooperative games, agents need to make decisions based on the agent and non-agent entities in its observation, and reward feedback teaches them their value in such team and how to cooperate with these observed entities. Thus we can also learn the value of agents in a smaller sub-group, by "imaging" that agents only observe a subset of the entities they actually observe, which helps them learn the cooperation patterns in this sub-group and this knowledge can be utilized in other tasks.

Specifically, for an episode sample from higher priority tasks with $|\mathcal{E}|$ entities, its entities will be randomly partitioned into two disjoint groups, indicated by a binary vector $m \in \{0,1\}^{|\mathcal{E}|}$, where m_e indicates whether entity e is in the first group and $\neg m_e$ related the second group. The subset of agent entities in all entities is denoted as m_A as introduced above. Then the attention masks between the entities can be constructed as:

$$M_I = m_A m^T \vee \neg m_A \neg m^T, M_O = \neg M_I \tag{11}$$

where $M_I[a, e]$ indicates where agent a and entity e are in the same group and $M_O[a, e]$ indicates the opposite. Considering the environment is partial observable, then the attention masks also need to be combined with the observation mask as:

$$M_I^\mu = M^\mu \wedge M_I, M_O^\mu = M^\mu \wedge M_O \tag{12}$$

The size of these matrices are $|A| \times |\mathcal{E}|$. Then the matrices M_I^μ and M_O^μ will be used in the MHA layer of the agent network as the imaginary observation masks to help calculated the value in the sub-group, and obtain $Q_I^1(s; \theta_Q, M_I^\mu), \cdots, Q_I^{|A|}(s; \theta_Q, M_I^\mu)$, as well as the out-group values $Q_O^1(s; \theta_Q, M_O^\mu), \cdots, Q_O^{|A|}(s; \theta_Q, M_O^\mu)$, which are the potential interaction values affected by the out-group entities. Due to the lack of the returns of such in-group or out-group utilities, these estimation are grounded by the observed returns, and are optimized by the exploitation loss as:

$$\begin{aligned} Loss_{task} &= E[(y_t^{tot} - Q_{aux}^{tot}(\tau_t, \mathbf{u_t}))^2 | (\tau_t, \mathbf{u_t}, r, \tau_{t+1}) \sim D] \\ Q_{aux}^{tot} &= g(Q_I^1, \cdots, Q_I^{|A|}, Q_O^1, \cdots, Q_O^{|A|}; h(s; \theta_h, M_I), h(s; \theta_h, M_O)) \end{aligned} \tag{13}$$

By optimizing the loss above, agents will learn their values in each sub-group, thus learn the smaller cooperation patterns emerging in this small group. Note that there is an obvious difference between the exploration loss and the exploitation loss. In the exploration, the loss aims to use the utilities of agents with randomly dropped entities to directly approximate the team returns, which helps avoid the co-adaptation between agent entities and other entities. While in exploitation, the loss aims to help agents learn the value of the sub-group existing in the team, and the utilities of agents in one sub-group are used to approximate the portion of the team value instead of the whole team returns with the help of the out-group agent utilities.

4.4 Learning in Multiple Cooperative Tasks

Here we introduce the overall optimization objective of PTM. In multi-task cooperation learning, we want to emphasize that all tasks are important while some tasks are more worthy of attention. Thus the learning loss in PTM consists of two parts, the uniform training loss $Loss_{ori}$ in all tasks as proposed by QMIX in Equation (1) or Equation (7), and the mining loss of higher priority tasks $Loss_{hpt}$. Besides, considering that both exploration and exploitation are important in the mining of higher priority tasks, here we encourage agents to learn a mixture of them with a weight α, which decreases linearly from 1 to 0, and the higher priority tasks training loss is:

$$Loss_{hpt} = \alpha Loss_{exp} + (1 - \alpha) Loss_{task} \tag{14}$$

Then, the total loss used for the learning in multiple cooperative tasks is:

$$\begin{aligned} Loss =& Loss_{ori} + \lambda \sum_{i=1}^{B} \omega_i Loss_{hpt}^i \\ =& Loss_{ori} + \lambda \sum_{i=1}^{B} \omega_i (\alpha Loss_{exp}^i + (1 - \alpha) Loss_{task}^i) \end{aligned} \tag{15}$$

where B is the number of episode samples in the batch, and λ weights the importance of the uniform training loss and higher priority tasks mining loss.

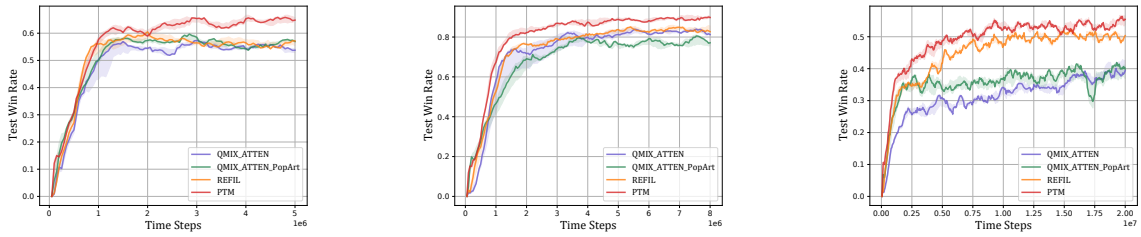
5 EXPERIMENTS

In this section, we design experiments to answer the following questions: 1) Is PTM more effective for learning in multi-task cooperation problems? 2) What is the influence of PTM to agents learning in multiple cooperation tasks? 3) What is the importance of different modules, such as the priority identification module, the exploitation module and exploration module in PTM? 4) Is the generalization ability of PTM also improved in related but unseen cooperative tasks compared to baseline methods? In the following, we will answer the questions above one by one in the challenging micromanagement tasks in StarCraft [21]. The StarCraft micromanagement task is a two-team battle game, where the algorithm needs to guide controlled alley agents to eliminate all enemy units in the other team. More details can be found in Appendix A.

In order to evaluate the effectiveness of PTM, we compare it to current state-of-the-art methods that are applicable to multi-task cooperative problems. We provide the performance of QMIX_ATTEN [9], which designs attention-based network to learn in multi-task setting, as well as REFIL [10], which helps agents utilize common cooperative patterns in multiple cooperative tasks. Besides, we also provide an extension of a strong multi-task single-agent reinforcement learning method PopArt [7] in cooperative setting *i.e.* QMIX_ATTEN_PopArt, which aims to normalize rewards in different tasks to adjust their contribution and balance the competence of different tasks. Each algorithm is evaluated using 3 independent training runs with different random seeds, and the resulting plots include the median performance shown in dark color as well as the 25%-75% percentiles shown in the shaded area. Besides, in PTM, the probability of being dropped in the exploration module or in the first group in exploitation module is obtained by that, we first draw $p \in (0, 1)$ uniformly, then the probability for each entity is sampled from a Bernoulli(p) distribution. This design makes a uniform distribution over all possible sub-groups within the team as discussed in [10]. Besides, λ is 10 fixed for all tested tasks. The exploration annealing period for α is 2 million time steps. We use the winrate (the number of successful episodes diveded by the total interaction episodes) of agents in each StarCraft task as their related cooperation indicator. More details can be found in Appendix B.

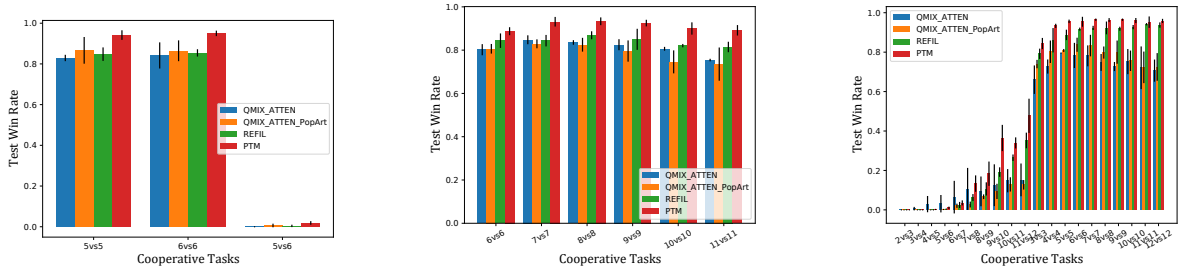
5.1 Performance in Multi-Task Cooperation

To answer the first question, we evaluate PTM and baseline methods in a customized StarCraft environment [10]. Concretely, we consider a multi-task setting, where the method is trained in multiple related tasks simultaneously, and demonstrate their average performance in multi-task tasks with different scales as shown in Figure 2. In Figure 2, the small_mt environment consists of 3 multi-agent cooperative tasks: $5m\_vs\_5m$, $6m\_vs\_6m$ and $5m\_vs\_6m$ (m is the abbreviation of marine units in StarCraft), where the first two are symmetric tasks while the last is asymmetric and is more challenging for agents to learn coordinated strategies. The medium_mt environment includes 6 cooperative tasks, where in each task the composition of two teams is symmetric, but the number of units in each task is varying from $6m\_vs\_6m$ to $11m\_vs\_11m$. In medium_mt environment, the variation in the number of teammates affects the cooperation difficulty of the cooperative task and requires different cooperation ability. The large_mt environment contains a richer



(a) small_mt (b) medium_mt (c) large_mt

Figure 2: Test win rate during training of different algorithms in multi-task StarCraft environment, where the curve represents the median performance and the shaded areas show 25%-75% percentiles.



(a) small_mt (b) medium_mt (c) large_mt

Figure 3: Performance of different algorithms in each cooperative task, where the error bar represents the standard deviation.

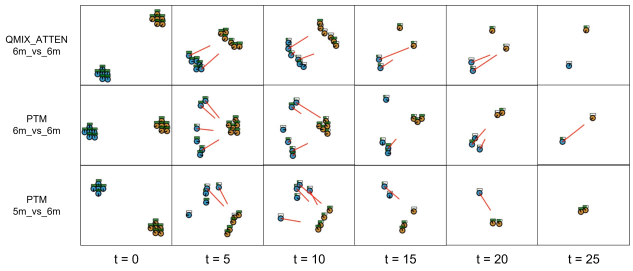


Figure 4: Snapshots of episode samples in different tasks trained by different methods. The ally agents are shown in blue while the enemies shown in orange. The red lines are attacks launched by ally agents and the green bar shows the health of the related agent. In this figure, agents trained by PTM learn to use the loose cooperation pattern found in $5m\_vs\_6m$ to solve $6m\_vs\_6m$ tasks and obtain an advantage in health in the last few snapshots.

variety of cooperative tasks, where tasks are different in symmetry or the number of teammates or both, such as $5m\_vs\_6m$ and $10m\_vs\_10m$. In such multi-task environments, though all tasks are related, the cooperation ability they require are not the same, thus learning difficulty and the potential contribution to the emerging of effective coordinated strategies is also different.

The results in Figure 2 show that PTM consistently outperforms all baselines in multi-task cooperation environment with different scales, demonstrating its effectiveness for learning in multi-task setting. Besides, we note that though compared to QMIX_ATTEN, REFIL improves the performance by reuse the cooperation patterns existing in multiple tasks, it still treats all tasks equally and thus can not share more effective cooperation patterns in higher priority tasks, which makes it weaker than PTM. Moreover, although PopArt-related method also focuses on the balance among tasks, it is mainly used to mitigate the unbalance caused by the magnitude of the in-task rewards, which is not permanent in our tested environments, thus the effectiveness of QMIX_ATTEN_PopArt is limited. Specially, we also demonstrate the detailed performance of these methods in each task in Figure 3. The results in Figure 3 of different methods verify that, due to the complexity in multi-agent system, though the team composition is nearly the same, the training difficulty of them varies greatly in different tasks. More importantly, we note that the performance of PTM in each concrete task is higher than baselines, which justifies that, when training in multiple related tasks, the adjustment of the focus on different tasks and paying more attention to unlearned tasks are helpful to find more effective and general strategies.

5.2 Visualization of Strategies Trained by PTM

Further, we want to investigate the influence of PTM to agents learning in different tasks. To this end, we visualize the episode samples in small_mt environment. Concretely, we render the samples trained by QMIX_ATTEN in $6m\_vs\_6m$ and PTM in $6m\_vs\_6m$ and $5m\_vs\_6m$ in Figure 4. In each plot, the blue units are alley

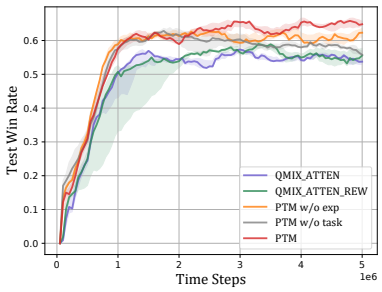


Figure 5: Ablation study of PTM, where QMIX_ATTEN_REW means directly reweighting the loss of samples from prior tasks, and PTM w/o exp (task) means only exploiting (exploring) the prior tasks without exploration (exploitation).

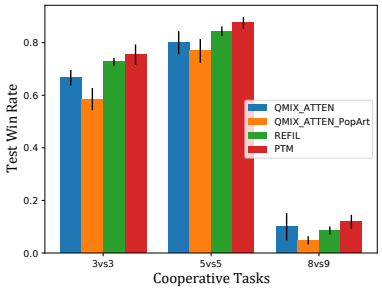


Figure 6: Generation performance of different multi-task cooperation methods in related but unseen tasks.

agents controlled by the evaluated method and the orange units are enemy agents controlled with build-in algorithms in StarCraft. The red lines are attacks launched by our alley agents and the green bar represents the health of the related agent. We show the snapshots in each episode every 5 time steps.

Note that QMIX_ATTEN is a method which learns different tasks uniformly, and achieves the knowledge sharing by the shared neural network in multiple tasks. In the first row of Figure 4, we can found when learning these tasks uniformly, agents are likely to be dominated by symmetry tasks and stuck in naive coordinated patterns, *i.e.* getting together to attack enemies, since such strategies can always obtain rewards in partial tasks. And with limited learning resources, agents lose passion to find other more effective strategies. Different from QMIX_ATTEN, agents trained by PTM will adjust the focus on different tasks, and in small_mt environment, they will focus more on the task 5*m_vs_6m*. This is a asymmetric game, and to win the game, agents need to form a loose cooperation pattern to create a larger attack range, and avoid being surrounded by enemies due to the lack of quantity advantage, as shown in the third row in Figure 4. From Figure 3(a) and the medium row in Figure 4, we find that although the discovered cooperation pattern in 5*m_vs_6m* can not help them win the game by a landslide, these strategies are still more advanced and the utilization of them in other tasks like 6*m_vs_6m* improves the task performance. In summary, the visualization of training samples show that the focus adjusted by PTM encourages agents to focus more on tasks they are not capable, where they can find more effective coordinated strategies to guide the cooperation in multiple tasks.

5.3 Ablation Study

We make an ablation study to verify the importance of different modules used in PTM. The contribution of PTM comes from its priority identification module and higher priority tasks mining loss. Thus we mainly make ablation studies about these two factors and show results in Figure 5. We find that directly reweighting the loss of samples from higher priority tasks with the priority ω_i (QMIX_ATTN_REW) hardly improves the performance. As discussed above, this reweighting only makes agents learning more of the agent utilities facing the whole team, which is less useful than learning the emerging cooperation patterns. Thus the higher priority tasks mining is relatively more important to the performance improvement. Besides, we notice that only mining the higher priority tasks with exploitation (PTM w/o exp) indeed improves the performance, while exploration is also necessary to find more effective strategies. More details can be found in Appendix C.

5.4 The Generalization Ability of PTM

In this section we investigate the generalization ability of PTM since it is more applicable if the model trained on multiple tasks can generalize well to other related but unseen tasks. Specially, we design 3 tasks to test the generalization ability of PTM and baseline models that trained in symmetric medium_mt environment, which are 3*m_vs_3m*, 5*m_vs_5m* and 8*m_vs_9m*, and show the results in Figure 6. The results show that in more related symmetric tasks like 3*m_vs_3m* and 5*m_vs_5m*, PTM demonstrates a higher generalization ability compared to other multi-task methods, which also justifies the effectiveness of the agents strategies mined by PTM in such symmetric tasks. Besides, we also found that when applied to the asymmetric task 8*m_vs_9m*, both PTM and other baselines are not that useful. As discussed above, asymmetric tasks always need more sophisticated cooperation that can not be discovered by training in symmetric tasks. The results in Figure 6 shows this phenomenon, and illustrates that a reasonable adjustment of focus on different tasks is necessary, where focusing on the tasks with more potential to generate sophisticated coordinated strategies is beneficial, but the opposite is not.

6 CONCLUSION

In this paper, we propose PTM for multi-task cooperative multi-agent reinforcement learning, which adjusts the focus on different tasks based on their complexity and potential in emerging effective coordinated strategies. By encouraging agents to make exploration and exploitation of higher priority tasks to mine effective strategies, PTM outperforms all strong multi-task cooperation methods in challenging multi-task cooperative games. Note that in this paper we assume that in multi-task setting, multiple tasks are related and the effective strategies found in higher priority tasks is beneficial to multiple tasks. However, there may be multiple tasks where the relationship among them is like a zero-sum game, and the performance improvement of some tasks is at the cost of the others. How to learn in such multi-task cooperation setting efficiently and effectively is worthy of future study.

ACKNOWLEDGMENTS

This work is supported in part by Basic Cultivation Fund project, CAS (JCPYJJ-22017), the National Natural Science Foundation of China (No.61876181), and the Youth Innovation Promotion Association CAS.

REFERENCES

[1] Trapit Bansal, Jakub Pachocki, Szymon Sidor, Ilya Sutskever, and Igor Mordatch. 2017. Emergent complexity via multi-agent competition. *arXiv preprint arXiv:1710.03748* (2017).

[2] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. 2019. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680* (2019).

[3] Diana Borsa, Thore Graepel, and John Shawe-Taylor. 2016. Learning shared representations in multi-task reinforcement learning. *arXiv preprint arXiv:1603.02041* (2016).

[4] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[5] Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, SM Eslami, et al. 2017. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286* (2017).

[6] Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. 2017. A survey of learning in multiagent environments: Dealing with non-stationarity. *arXiv preprint arXiv:1707.09183* (2017).

[7] Matteo Hessel, Hubert Soyer, Lasse Espeholt, Wojciech Czarnecki, Simon Schmitt, and Hado van Hasselt. 2019. Multi-task deep reinforcement learning with popart. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3796–3803.

[8] Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. 2020. UPDeT: Universal Multi-agent RL via Policy Decoupling with Transformers. In *International Conference on Learning Representations*.

[9] Shariq Iqbal, Christian A Schroeder de Witt, Bei Peng, Wendelin Böhmer, Shimon Whiteson, and Fei Sha. 2020. Ai-qmix: Attention and imagination for dynamic multi-agent reinforcement learning. *arXiv preprint arXiv:2006.04222* (2020).

[10] Shariq Iqbal, Christian A Schroeder De Witt, Bei Peng, Wendelin Böhmer, Shimon Whiteson, and Fei Sha. 2021. Randomized Entity-wise Factorization for Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. PMLR, 4596–4606.

[11] Daniel Kahneman. 1973. *Attention and effort*. Vol. 1063. Citeseer.

[12] Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, et al. 2020. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 4501–4510.

[13] Ryan Lowe, YI WU, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *Advances in Neural Information Processing Systems* 30 (2017), 6379–6390.

[14] Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. 2012. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review* 27, 1 (2012), 1–31.

[15] Edward C Merrill and Michael Peacock. 1994. Allocation of Attention and Task Difficulty. *American Journal on Mental Retardation* 98, 5 (1994), 588–593.

[16] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[17] Chunyi Peng, Minkyong Kim, Zhe Zhang, and Hui Lei. 2012. VDN: Virtual machine image distribution network for cloud data centers. In *2012 Proceedings IEEE INFOCOM*. IEEE, 181–189.

[18] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. 2020. Weighted QMIX: Expanding Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. *Advances in Neural Information Processing Systems* 33 (2020).

[19] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*. PMLR, 4295–4304.

[20] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671* (2016).

[21] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. 2019. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043* (2019).

[22] Christian Schroeder de Witt, Jakob Foerster, Gregory Farquhar, Philip Torr, Wendelin Boehmer, and Shimon Whiteson. 2019. Multi-agent common knowledge reinforcement learning. *Advances in Neural Information Processing Systems* 32 (2019).

[23] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning*. PMLR, 5887–5896.

[24] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[25] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[26] Ming Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*. 330–337.

[27] Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. 2017. Distral: Robust multitask reinforcement learning. *Advances in neural information processing systems* 30 (2017).

[28] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[30] Nelson Vithayathil Varghese and Qusay H Mahmoud. 2020. A survey of multi-task deep reinforcement learning. *Electronics* 9, 9 (2020), 1363.

[31] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. 2020. Qplex: Duplex dueling multi-agent q-learning. *arXiv preprint arXiv:2008.01062* (2020).

[32] Ruihan Yang, Huazhe Xu, Yi Wu, and Xiaolong Wang. 2020. Multi-task reinforcement learning with soft modularization. *Advances in Neural Information Processing Systems* 33 (2020), 4767–4777.

[33] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. 2021. The surprising effectiveness of mappo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955* (2021).

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

[35] Meng Zhou, Ziyu Liu, Pengwei Sui, Yixuan Li, and Yuk Ying Chung. 2020. Learning implicit credit assignment for multi-agent actor-critic. *arXiv preprint arXiv:2007.02529* (2020).