Parallel Knowledge Transfer in Multi-Agent Reinforcement Learning

1st Yongyuan Liang
Robotics Institute
Carnegie Mellon University
Pittsburgh, United States
liangyy58@mail2.sysu.edu.cn

2nd Bangwei Li
School of Mathematics
Sun Yat-sen University
Guangzhou, China
libw5@mail2.sysu.edu.cn

***Abstract*—Multi-agent reinforcement learning is a standard framework for modeling multi-agent interactions applied in real-world scenarios. Inspired by experience sharing in human groups, learning knowledge parallel reusing between agents can potentially promote team learning performance, especially in multi-task environments. When all agents interact with the environment and learn simultaneously, how each independent agent selectively learns from other agents' behavior knowledge is a problem that we need to solve. This paper proposes a novel knowledge transfer framework in MARL, PAT (Parallel Attentional Transfer). We design two acting modes in PAT, student mode and self-learning mode. Each agent in our approach trains a decentralized student actor-critic to determine its acting mode at each time step. When agents are unfamiliar with the environment, the shared attention mechanism in student mode effectively selects learning knowledge from other agents to decide agents' actions. PAT outperforms state-of-the-art empirical evaluation results against the prior advising approaches. Our approach not only significantly improves team learning rate and global performance, but also is flexible and transferable to be applied in various multi-agent systems.**

***Index Terms*—Reinforcement Learning, Multi-Agent System, Attention, Transfer Learning**

I. Introduction

Knowledge transfer is a common method in the general learning process of a new task or in a new environment. The educational behavior in human society is an advanced form of knowledge transfer. In a multi-agent system, when an agent in an unfamiliar environment and learn to get more reward, the knowledge from other experienced agents is beneficial to the agent. Reinforcement Learning (RL) [1], as a popular framework, has been employed in sequential decision-making problems. And Transfer Learning (TL) [2] aims to improve learning through the learning experience from a related task, which also means knowledge reusing. In Reinforcement Learning domains, the source of informative knowledge varies from experienced agents(experts) to human guidance. In this paper, we work on applying knowledge transfer method in agents' behavior transfer for multi-agent team.

Cooperative Multi-agent Reinforcement Learning (MARL) has been applied in a series of meaningful problems such as multi-robot control [3] and team-game playing [4]. In Cooperative MARL, if an individual agent's learning is simply seen as independent RL with partial observations, the interactions between agents and the non-stationary environment will cause significant difficulties. To accelerate the team-wide learning efficiency and maximize the advantage of knowledge-transfer in the multi-agent domain, this paper targets the problem of optimizing knowledge-transfer between agents in Cooperative MARL under local constraints with a joint task or multiple tasks.

Our work is different from prior works that study inter-agent communication mechanisms in cooperative MARL [5]–[7]. These works require a centralized critic and decentralized execution framework. Considering the scale of an agent team in the real scenarios, centralized training is challenging considering training stability and high computation complexity with large computational costs. Jiang & Lu [8] and Das et al. [9] both proposed attention based communication protocol to exchange messages in MARL domains [8], [9], but these approaches did not consider agents' behavior knowledge. Communication methods in MARL domains are always designed for solve the problem of efficient sharing partial observation information. In a fully decentralized system, we try to find a method to reuse agents' experience information to guide the unfamiliar agents with less communication costs for the goal of improving team-wide learning. Our work concerns behavior knowledge reusing problem between cooperative agents with a teacher-student framework in order to improve team-wide performance in learning process, relevant to multi-agent teaching.

Regarding privacy constraints and communication cost in a multi-agent team, the main issues to be concerned are knowledge transfer decision, knowledge selection, and knowledge utilization. Invalid or confusing messages from other agents may cause a negative impact on agents' individual learning. Also, in an environment where information interaction is frequent, there is a danger of risk contagion, resulting in poor performance of the entire team. For instance, the phenomenon of "over-advising" mentioned in previous studies, has increased team-wide learning instability, especially in a team with more than two agents. In order to improve knowledge communication efficiency, we introduce an attention mechanism to dynamically distill knowledge from other agents' experience. Crucially, attention mechanism as a teacher selector is used to determine teacher agents' familiarity with the environment

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

and current policy effectiveness for the student agent. Hence, The student agent have the flexibility to accept the optimal advice from appropriate teachers for each time point.

In this paper, we propose a more robust and reliable parallel knowledge-transfer model with high efficiency in MARL framework. We here state the following based settings in our work:

- All agents simultaneously learn in an environment and make decisions with interactions with the environment and other agents.
- There is no optimal expert (good-enough agent) in the multi-agent team in the initial state.
- (In Parallel) For all agents, their local role of student or teacher is not fixed. An agent can use knowledge from other agents as a student and provide its own behavior knowledge as a teacher for students.
- All agents are learning for their local reward. But agents in the team are friendly to share knowledge. Our goal is to maximize the team-wide reward.

Our empirical results across a range of tasks and environments demonstrate the efficacy of our knowledge transfer architecture in multi-agent systems. We show that our attention selector is capable of teaching the student agent with the most confident advice from teacher agents. Compared with other multi-agent knowledge transfer frameworks, our approach successfully give rise to an obvious improvement in global performance and stability.

II. Related Work

As a long-standing topic in the field of Reinforcement Learning, Multi-agent Reinforcement Learning (MARL) [10] track has a series of works in various ways to improve the performance and efficiency of team coordination. Deep Reinforcement Learning [11] uses deep neural networks to approximate the policy and value functions of agents in the environment to address the problem of large-scale action-value space in RL. And Knowledge transfer method has been studied in several related fields including imitation learning, learning from demonstration [4], and inverse reinforcement learning [12]. Several works [13] extended the source-target framework in transfer learning on reinforcement learning task. In the extensive teacher-student framework for transfer from expert policy to student policy, the student agent takes actions from the expert agent's advice. Student-initiated approaches mainly concern with the student's decision value, such as Ask Uncertain [14] and Ask Important [15]. In teacher-initiated approaches, teachers decide when to teach based on the comparison between student's and teacher's learning experience, such as Importance Advising [16], Early Correcting [15], and Correct Important [16]. Q-teaching [17] designs teaching rewards to help teachers determine when to advise. Moreover, episode-sharing mechanism [18] helps agents share individual successful episodes to accelerate learning.

However, all of the above works require an expert (all-knowing teacher) as the best agent to guide the learning of other agents. Zhan et al. [19] analyzed the case of negative transfer with the existence of a sub-optimal expert and present some theoretical results.

Recent works provide some solutions for multi-agent parallel advising problem:

AdHocVisit and AdHocTD [20] is an advisor-advisee framework without an expert in the multi-agent environment for agents learning simultaneously. The learning agents ask for advice and provide advising policy for other agents. Advisees use state visit counts to decide when to request advice and advisors evaluate their advice' reliability through confidence metrics to decide when and what to provide for advisees. For advice selection, AdHocVisit and AdHocTD follow majority vote [19].

LeCTR [21] is a new teacher-student framework, which targets peer-to-peer teaching in order to solve advising-level problem. Each agent in system learns when and what to advise. In LeCTR, teacher-student (advising-level) policies are trained using the multi-agent actor-critic approach (MADDPG) [5]. LeCTR sets the advising-level policies as decentralized actor and uses a centralized action-value function as critic with advising-level reward. It is worth mentioning that LeCTR considers the communication cost in information exchange. LeCTR only works in two-player games.

Motivated by attention [22] introduced for information extraction in deep learning, attention mechanism has recently emerged in reinforcement learning framework [23], [24]. In distributed MARL, Jiang & Lu [8] proposed an attentional communication method with independent actor-critic. Attention in this work encodes agents' individual observation before passing centralized communication channel. With centralized value estimate, TarMAC [9] is a targeted communication architecture to generate agents' internal state representations as input of centralized critic. Iqbal & Sha [25] introduced an attention-based critic to select agents in centralized training.

Although attention plays a core role in our idea, our motivation is different from aforementioned approaches. Our algorithm aims to reuse agents' accumulated knowledge in learning process but not limited to process individual observation with agents' interaction in multi-agent domains. Our approach is more effective and efficient to maximum team cooperation utility and flexible to extend in complex environments.

III. Preliminaries

In this work, we consider a decentralized multi-agent reinforcement learning scenario, multiple agents in cooperative team $\mathcal{G}$ simultaneously learn a joint task or multiple tasks. Our settings are formalized as a Decentralized POMDP (Dec-POMDP) in cooperative multi-agent system. All the agents in the environment receive local observation o_t^i at each time step, and interact with the environment by executing local action. Agents then update their policy parameters according to the feedbacks (reward) given by the environment.

The system is described as $(\mathcal{I}, S, A, T, R, \Omega, O, \gamma)$,

- S is a set of states,
- A is a set of joint actions, $\mathcal{A} = \times_i \mathcal{A}^i$,

- T is a set of conditional transition probabilities $T(s'|s,a)$ between states,
- R: $S \times A \to \mathbb{R}$ is the global reward function.
- Ω is a set of joint observations, $\boldsymbol{o} = \langle o^1, \ldots, o^n \rangle$
- O is a set of conditional joint observation probability, $P(\boldsymbol{o}|s', \boldsymbol{a}) = \mathcal{O}(\boldsymbol{o}, s', \boldsymbol{a})$,
- $\gamma \in [0, 1]$ is the discount factor.

At each time period, the environment is in some state $s \in S$. Agents take a joint action $\mathcal{A} \in A$. Then each agent receives a local reward $r_t^i = \mathcal{R}^i\left(s_t, \boldsymbol{a}_t^i\right)$. The process repeats.

The goal is for all agents to take actions at each time step that maximize the global expected future discounted reward: $E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$.

1) Reinforcement Learning: [1] is a standard framework to achieve the above goal of MDP (or POMDP). The process of value based reinforcement learning is to learn a policy which can maximize agent's final reward. Through collecting experience from environment, agent updates its value function$v^\pi(s) = \mathbb{E}_\pi\left[R_t|s_t = s\right]$and action value function$Q^\pi(s,a) = \mathbb{E}_\pi\left[R_t|s_t = s, a_t = a\right]$.

2) Deep Q Learning (DQN): [11] is a value-based Reinforcement Learning approach combined with deep neural networks, which learns the action value function (Q-value) in continuous environment using value function approximation. Q-Network updates by minimizing the loss:$L(\theta) = \mathbb{E}[(r + \gamma \max_{a'} Q(s', a'; \theta) - Q(s, a; \theta)]$, and outputs the expected action value $Q(s, a; \theta)$.

3) Deterministic Policy Gradient (DPG): [26] is a policy-based Reinforcement Learning approach as an extension of policy gradient (PG) [27], which optimize policy by update policy parameters θ along the gradient direction,

$$\nabla_\theta J(\theta), \nabla_\theta J(\theta) = \mathbb{E}_{s\sim p^\pi, a\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s) Q^\pi(s, a; \theta)]$$

. Then, DPG is extended to **Deep Deterministic Policy Gradient (DDPG)** [28], with $\nabla_\theta J(\theta) = \mathbb{E}_{s\sim\mathcal{D}}[.\nabla_\theta \mu_\theta(a|s) \nabla_a Q(s, a; \mu)|_{a=\mu_\theta(s)}]$ using deterministic policy when the variance of action probability distribution approaches 0.

4) Attention Mechanism: is one of the most influential models which can be broadly interpreted as a vector of importance weights. It has recently been applied in reinforcement learning domains.

IV. Overview

Our work targets knowledge reusing between agents in cooperative MARL, where all the agents in the environment are not good enough. In this section, we provide a story-level overview of our main idea. The overview of our motivating scenario is presented in Fig. 1.

Considering our settings, all the agents act in the environment and update their self policy parameters with local rewards from the environment. The actions executed by the agents is dictated by their self policy parameters. Now, we explore a novel knowledge transfer framework, PAT. In our framework,

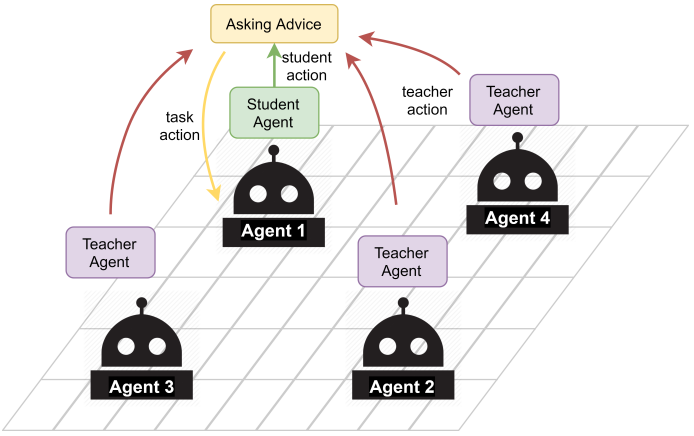

Fig. 1. Overview

each agent has two acting modes, student mode and self-learning mode. Before each agent takes action, agents' student actor-critic modules decide agents' acting modes with agents' hidden states. It is not an ad-hoc design for specific domain. Agents should learn to ideally learn from other agents.

In self-learning mode, agents take action based on their independent learned behavioral knowledge, which is represented as agents' behavior policy parameters. In this mode, agents' actions are independent of other agents' behavioral knowledge. All agents in self-learning mode are trained in an individual end-to-end manner using Deep Deterministic Policy Gradient [28] algorithm with an actor network and a critic network.

In student mode, if there are more than two agents in the system, a student agent receives multiple advice from other agents. We now refer to a new problem, teacher selecting, because not all teachers' knowledge is useful for the student agent. However, existing frameworks try to dodge this problem and have key limitation in the scenario where the number of agents is large, which also causes difficult in model transfer. We apply a soft attention mechanism in our work to select teachers' knowledge. The Attention Teacher Selector solves the problem by selecting contextual information in teachers' learning information and computing weights of teachers' knowledge. Considering from a different angle, our attentional module selectively transform the learning information from teachers with the target of solving student's problem. The attentional selecting approach is effective both in multi-task scenarios and joint task scenarios.

We make a few assumptions on agents' identities to support our framework. When an agent chooses student mode in t time step, other agents in the environment automatically become its teachers and provide their behavior policy and learning knowledge to the student agent.

Our parallel setting means that An agent in student mode can be a teacher of the other agents. When agent i is unfamiliar with its observation m_t^i, but at the same time, agent i may be familiar with agent j's observation m_t^i because of i's past trajectory, which means that agent i can be agent j's teacher. At this time step, agent i is in student mode but it also is

a teacher to transfer its knowledge to other agent. The core idea is agents' different learning experience. An student agent may have confidence in the other states and its behavior knowledge can help the other student agents. Our teacher selector module is designed to determine the appropriate teachers and transform teachers' local behavior knowledge into student's advising action. Moreover, our attention mechanism quantifies the reliability of teachers, so our scenarios do not need good-enough agents (experts).

In a high-level summary, because of agents' different learning experience, agents in a cooperative team are good at different tasks or different parts of a joint task. Knowledge Transfer is a framework to help agents solve unfamiliar task with experienced agents' learning knowledge.

PAT's training and architecture details are presented in the next section.

V. Attention based Knowledge-Transfer Architecture

This section introduces our knowledge transfer approach with more design details of the whole structure and all training protocols in our framework. In our framework, each agent has two actor-critic model and an attention mechanism to support two acting modes.

A. Acting Mode

Different from original individual agent learning, after receiving observation from the environment, agents in our framework need to choose their action mode before taking action.

At t time step, agent i reprocesses the observation from environment with a hidden LSTM (or RNN) unit, which integrates information (observation) from i's observation history. The LSTM unit l^i outputs agent's observation encoding m_t^i, which represents agent's hidden state. Here, k is a scaling variable, which represents the time period covered in the hidden state. We will adjust k depending on different types of games.

$$l^i : (o_{t-k}^i, a_{t-k}^i, ..., o_t^i) \rightarrow m_t^i \tag{1}$$

Next, based on this step's memorized observation, m_t^i, agent i's student actor network takes this step's memorized observation, m_t^i, as input and output agent i's acting mode. Considering the efficiency of information exchange and communication cost, student actor is used to deciding agent i' confidence in t time step. If i has enough confidence with m_t^i, student actor chooses self-learning mode. Conversely, student actor chooses student mode and sends advice request to other agents.

Student actor and student critic is a deep deterministic policy gradient model. The student actor network outputs the probability of choosing student mode. When the probability exceeds a threshold value, agent will choose student mode as a deterministic action. The threshold value is a variable depending on different types of games.

Student actor and student critic represent the acting mode choosing model which determines whether agent i become a student and ask teacher agents for advice. We train student actor-critic using a student reward $\widetilde{r}_t^i$.

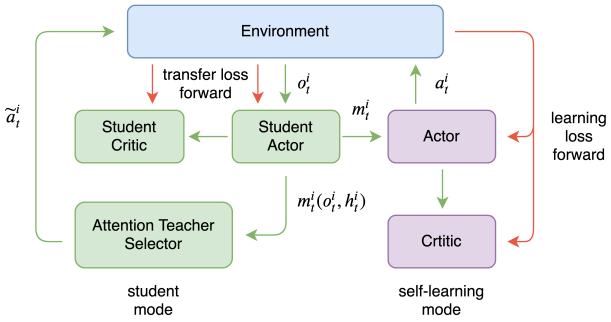

Fig. 2. PAT Architecture

$$\widetilde{r}_t^i = V\left(m_t^i; \theta_t'^i\right) - V\left(m_t^i; \theta_t^i\right) \tag{2}$$

$\theta_t'^i$ and θ_t^i, are agent i policy parameters in student mode and self-learning mode. The student reward measures gain in agent learning performance from student mode. The sharing of student actor-critic network parameters allows this module learning effectively in environment and easily extending to other settings.

In our experiments, student actor-critic is trained with the trained Attention Teacher Selector. Agent i's student critic is updated to minimize the student loss function:

$\widetilde{\mathcal{R}}$ is agent i's student policy transition set.

$$\mathcal{L}\left(\theta^{\widetilde{Q}}\right) = \mathbb{E}_{m_t, w_t, \widetilde{r}_t, m_{t+1} \sim \widetilde{\mathcal{R}}}\left[\left(\tilde{y}_t - \widetilde{Q}\left(m_t, w_t | \theta^{\widetilde{Q}}\right)\right)^2\right],$$
$$\tilde{y}_t = \tilde{r}_t + .\gamma \widetilde{Q}(m_{t+1}, w' | \theta^{\widetilde{Q'}})|_{w'=\widetilde{\mu}'\left(m_{t+1}|\theta^{w'}\right)} \tag{3}$$

Student policy network is updated by ascent with the following gradient:

$$\nabla_{\theta^{\widetilde{\mu}}} J = \mathbb{E}_{m_t, w_t \sim \widetilde{\mathcal{R}}}\left[\nabla_w \widetilde{Q}\left(m_t, w_t | \theta^{\widetilde{Q}}\right)\Big|_{w_t = \widetilde{\mu}(m_t)} \nabla_{\theta^{\widetilde{\mu}}} \widetilde{\mu}\left(m_t | \theta^{\widetilde{\mu}}\right)\right] \tag{4}$$

Here, $\widetilde{\mu}$ is agent i's student policy, which is parameterized by $\widetilde{\theta}$

B. Student Mode

1) Attention Teacher Selector: Inspired by the similarity between source task and target task in transfer learning, we use attention mechanism to evaluate the task similarity between student and teachers and teachers' confidence of student's state. Therefore, each agent's Attention Teacher Selector in student mode is used to select advice from teachers based on their similarity and confidence. The main idea behind our knowledge transfer approach is to learn the student mode by selectively paying attention to policy advice from other agents in the cooperative team. Fig. 3 illustrates the main components of our attention mechanism.

We now describe the Attention Teacher Selector mechanism in agent student mode. The Attention Teacher Selector (ATS) is a soft attention mechanism as a differentiable query-key-value model [23], [29]. After the student actor of student agent $i \in \mathcal{G}$ compute the memorized observation at t time step and choose student mode, ATS receives the encoding hidden state m_t^i. Then, from other agents in the team as teacher agents, ATS receives the teachers' encoding learning history $h_t^j = l^j(o_1^j, a_1^j, ..., o_t^j)$ and encoding policy parameter θ^j.

Now, ATS computes a query $Q_t^i = W_Q m_t^i$ as student query vector, a key $K_t^j = W_K h_t^j$ as teacher key vector, and a value $V_t^j = W_V \theta^j$ as teacher policy value vector, where W_K, W_Q and W_V are attentional learning parameters. After ATS receives all key-value (K^j, V^j) from all of teachers $j \in \mathcal{G}$, the attention weight α^{ij} is assigned by passing key vector from teacher and query vector from student into a softmax:

$$\alpha^{ij} = softmax\left(\frac{Q^i K^j}{\sqrt{D_K}}\right) \tag{5}$$

Here, D_K is the dimension of teacher j's key vector, which is used to resolve vanishing gradients (Vaswani et al. 2017). The final policy advice is a weight sum with a linear transformation:

$$v^i = W_T \sum_{j \neq i} \alpha^{ij} V^j \tag{6}$$

Here, W_T is a learning parameter for policy parameter decoding.

Behind the single attention head, we use a simple multi-attention head with a set of learning parameters (W_K, W_Q, W_V) to aggregate all advice from different representation subplaces. Besides, attention head dropout is applied to improve the effectivity of our attention mechanism.

Finally, student agent i obtains its action at this time with policy parameters from Attention Teacher Selector:

$$\widetilde{a}_t^i = v^i(m_t^i) \tag{7}$$

In our experiments, the attention parameters (W_K, W_Q, W_V) are shared across all agents, because knowledge transfer process is similar in all pairs of student-teacher, but different observations introduce different teacher weight vector. This setting encourages our approach to learn more efficient and make our model easy to be extended in different settings, such as larger number of agents or a different environment.

In this work, we consider scenarios where other agents' learning experience is useful to a student agent. Feeding student's observation information and teacher's learning experience into our attention mechanism helps to select action with other agents' behavioral policy for the student agent. This module is an end-to-end knowledge transfer method without any decentralized learning parameter sharing.

C. Self-learning Mode

If agent i's student actor chooses self-learning mode, the student actor sends i's encoding hidden state m_t^i to the actor network. In self-learning mode, agents learn as a common individual agent. Each agent's policy in self-learning mode is independently trained by DDPG [28] algorithm.

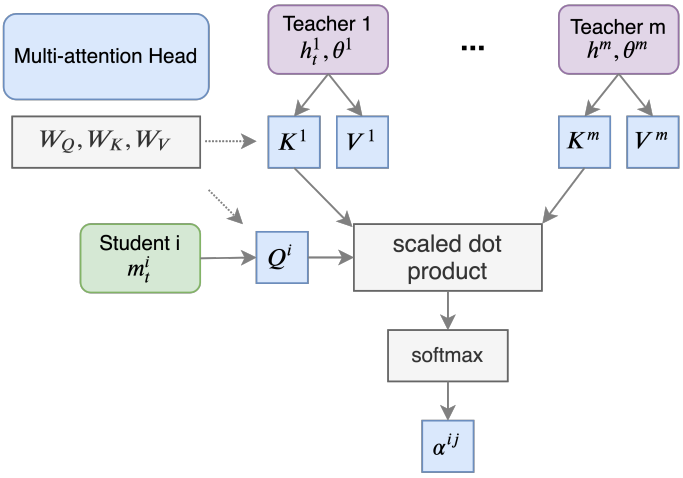

Fig. 3. Attention based Knowledge Selection

Agent i's critic network is updated by TD error, $\mathcal{R}$ is agent i's transition set:

$$\begin{aligned} \mathcal{L}\left(\theta^Q\right) &= \mathbb{E}_{m_t, a_t, r_t, m_{t+1} \sim \mathcal{R}}\left[\left(y_t - Q\left(m_t, a_t | \theta^Q\right)\right)^2\right], \\ y_t &= r_t + \gamma Q\left(m_{t+1}, a' | \theta^{Q'}\right)\Big|_{a'=\mu'\left(m_{t+1}|\theta^{\mu'}\right)} \end{aligned} \tag{8}$$

The policy gradient of agent i's actor network can be derived as:

$$\nabla_{\theta^\mu} J = \mathbb{E}_{m_t, a_t \sim \mathcal{R}}\left[\nabla_a Q\left(m_t, a_t | \theta^Q\right)\big|_{a_t = \mu(m_t)} \nabla_{\theta^\mu} \mu\left(m_t | \theta^\mu\right)\right] \tag{9}$$

In games with discrete action space, in self-learning mode, we refer to the modified discrete version of DDPG suggested by [5] in agents' actor-critic networks. Agents in self-learning update its actor network use,

$$\nabla_{\theta^\mu} J = \mathrm{E}_{m_t, a_t \sim \mathcal{R}}\left[\nabla_a Q(m_t, a_t) \nabla_{\theta^\mu} a_t\right] \tag{10}$$

.

Our framework is adapted for both continuous action space and discrete action space.

VI. Empirical Evaluations

We construct three environments to test the team-wide performance of PAT and existing advising methods in multi-tasks and joint task scenarios. Also, we compare the scalability of all the approaches with the increase of number of agents. Additionally, the transferability of PAT is evaluated in different environments.

A. Setup

Empirical evaluations are performed on three cooperative multi-agent environments: Grid Treasure Collection, Moving Treasure Collection, Predator-Prey, and. We implement Grid Treasure Collection, a standard grid world environment. Moving Treasure Collection and Predator-Prey are implemented

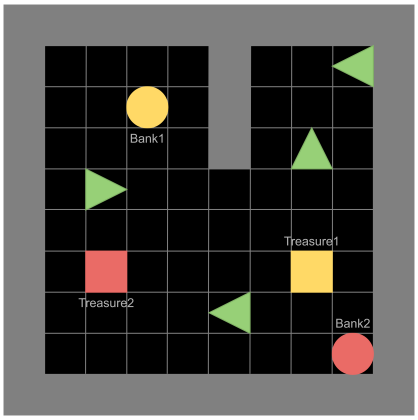

Fig. 4. Grid Treasure Collection, 4 agents, 2 treasure and 2 treasure banks

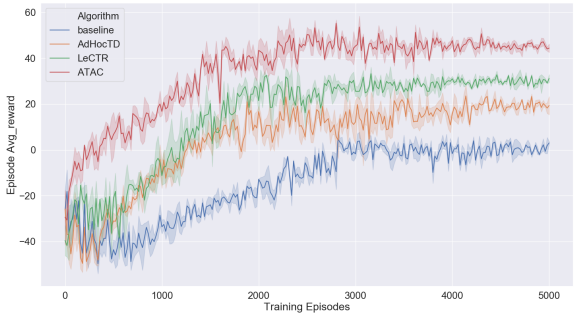

Fig. 5. Average reward per episode in Grid Treasure Collection

based on Multi-Agent Particle Environment [5], [30] where agents move around in a 2D space and involve interaction between agents. We briefly describe the three environments below:

1) Grid Treasure Collection: : There are M agents and M/2 treasure grids and M/2 treasure banks in the grid maze (shown in Fig. 4). Each treasure is corresponding to a treasure bank. When agents collect treasures in treasure grids, agents get small rewards, and then return treasures to its corresponding bank, agents receives big rewards. Each treasure grid has M treasures and agents can only obtain one treasure from each treasure grid. The ability of agents to carry treasures is not limited. But when agents return wrong treasures to treasure banks, agents receive big penalties.

2) Moving Treasure Collection: : The game rule in Moving Treasure Collection is similar to the above game, but in this environment, agents are green and treasures and treasure banks are moving randomly. And all objects are moving in a 2D open ground. Obstacles (large black circles) block the way in the environment.

3) Cooperative Navigation: : There are M agents (green) and M landmarks (purple) in this environment. Agents are rewarded based on how far any agent is from each landmark. Agents are required to position themselves covering all the landmarks. When an agent covers a landmark, it gets a local reward. Obstacles (large black circles) block the way.

To simplify our approach training, we set discrete action spaces in all the environments, allowing agents to move up, down, left, right, or stay. All agents receive partial observation at each step, and get local feedback from environments. We plan to evaluate our approach in both multi-task environment (Treasure Collection) and joint task environment

B. Baselines

We compare our approach, PAT with implementation of fully decentralized training approaches including Individual Deep Q-Learning (IQN) [11], AdHocTD [20] and LeCTR [21]. IQN, as a distributed baseline, is a reinforcement learning algorithm for single agent, which is trained independently for each agent with partial observation in our environments. AdHocTD, LeCTR and our method rely on action advising without any other information communication , which means each agent is unaware of the observations and rewards of other agents in the environment. We implement AdHocTD and LeCTR with original experimental parameters and public implementations. In LeCTR implementation, we use Majority Vote for multiple advice selecting. Hyperparameters are tuned based on our environments and performance. All implementation details will be released with code.

We evaluate all the models on two different agent teams , with M=4, M=8 and M=12 agents. When the number of agents increasing, the amount of calculation and difficulty increase rapidly.. Average step length per episode (Treasure Collection games' max episode length = 1000), success rate of covering all landmarks in Cooperative Navigation, and average sum of rewards per episode are three indicators used to evaluate performance of all models. All results are reported using 30 independent training setting different seeds. Table I shows the results in 4-agent and 12-agent environments for comparison. (The results in 8-agent environment will be presented in later released data)

C. Results and Analysis

Fig. 5 displays average team rewards per episode in Grid Treasure Collection. A thorough evaluation result is summarized in Table I.

In 4-agent Grid Treasure Collection (GTC) and Moving Treasure Collection (MTC) game, PAT outperforms all models with a higher average reward after convergence. In Comparison of all approachs in average episode length (Ave_step), PAT has no obvious advantage compared with LeCTR, but PAT greatly improves average team reward per episode (Avg_reward). Attention mechanism performs well and significantly fights out agents with useful experience (have successfully collected treasures), which can correctly guide unfamiliar agents to take more beneficial actions. AdHocTD selects teachers using the number of times the agent visited the current state. As a result, the phenomenon of over-advising appears in AdHocTD after trained over 2000 episodes, which means that students may take advice from teachers with more bad experience. It contributes to AdHocTD worse final

TABLE I
EVALUATION RESULTS

TASK	APPROACH	M=4 Avg_step	Success %	Avg_reward	M=12 Avg_step	Success %	Avg_reward
Grid Treasure Collection	Baseline	1000 ± 0	-	0.78 ± 1.43	2000 ± 0	-	-1.20 ± 0.04
Grid Treasure Collection	AdHocTD	876 ± 15	-	18.76 ± 1.14	1821 ± 29	-	12.56 ± 0.34
Grid Treasure Collection	LeCTR	812 ± 39	-	29.87 ± 1.23	1836 ± 23	-	11.68 ± 2.24
Grid Treasure Collection	PAT	762 ± 20	-	45.65 ± 2.32	1157 ± 37	-	20.34 ± 1.49
Moving Treasure Collection	Baseline	1000 ± 0	-	-1.84 ± 0.66	2000 ± 0	-	-3.59 ± 0.26
Moving Treasure Collection	AdHocTD	1000 ± 0	-	8.22 ± 1.79	1987 ± 67	-	-3.34 ± 0.21
Moving Treasure Collection	LeCTR	877 ± 46	-	20.76 ± 0.24	1889 ± 48	-	-2.79 ± 2.12
Moving Treasure Collection	PAT	854 ± 23	-	33.98 ± 2.13	1239 ± 32	-	-0.32 ± 0.43
Cooperative Navigation	Baseline	397 ± 25	52.2 ± 2.0	-1.78 ± 0.03	782 ± 4	32.7 ± 4.7	-4.68 ± 0.05
Cooperative Navigation	AdHocTD	320 ± 28	69.0 ± 3.7	-1.23 ± 0.27	547 ± 29	42.6 ± 2.0	-3.50 ± 0.08
Cooperative Navigation	LeCTR	278 ± 25	81.4 ± 3.2	-1.29 ± 0.39	672 ± 14	40.7 ± 1.9	-3.36 ± 0.70
Cooperative Navigation	PAT	289 ± 18	80.2 ± 3.2	-0.45 ± 0.07	498 ± 10	59.2 ± 1.3	-2.67 ± 0.02

reward performance than LeCTR and PAT. LeCTR learns how to teach by centralized training and decentralized executing, causing the learning instability in early episodes. Observing the results, PAT successfully avoids this problem by training a decentralized student actor-critic network for the decision of acting mode.

In 4-agent Cooperative Navigation, PAT surprisingly performs longer episode length and lower success rate than LeCTR. Attention mechanism in PAT tends to imitate successful action from other agents, which helps student agent gain more local rewards. But it might cause a bad impact on team coordination in a joint task. For example, agents tend to cover the same landmarks with successful experience, but the team needs to cover all the landmarks. PAT is hard to learn the team cooperative policy and gain more cooperative rewards. However, LeCTR uses a centralized critic network to calculate advising value concatenating all agents' observation, which exactly improves cooperation performance between agents. This result gives us a future direction that we should try to modify attention tendency in global reward for cooperative tasks.

1) Scalability: When the number of agents is 12 in all the environments, because of the increasing computation difficulty in selecting advice from a larger agent team, AdHocTD's advising probabilistic mechanism can not handle the problem in more information selecting, resulting in sub-optimal rewards. LeCTR, as an algorithm originally supporting two-player games, has low performance in a larger size of agent team. We suspect it due to LeCTR's centralized control of advising. With more agents in the environments, LeCTR's centralized advising-level critic performs poorly with limited training time. As expected, PAT's attention selector effectively filtering knowledge from more teachers and maintain student mode's accuracy, so our approach has a larger advantage with the increase in number of agents. Our experimental results confirm our inference.

In summary, PAT performs much better than all approaches and has a distinct advantage in average team-wide rewards in complex multi-task environments as expected. We also report PAT's disadvantage in the joint-task scenario. PAT scales better when agents are added in all the experiments, which shows that sharing attention mechanism is useful for information selecting in a large multi-agent team.

2) Transferablility: Analysing the above results, the behavior knowledge transfer becomes more difficult when number of agents increases. We design a new experiment to explore our model parameters transfer performance. We test our approach in two experiments with different numbers of agents. First, agents are trained in an environment with a small number of agents. Then, the trained parameters of agents' shared attention mechanisms are transferred/reused in an environment with more agents. We compared our transfer experimental performance with original training performance data of larger agent teams.

Table II shows the transfer performance of 4-agent attention mechanism in 8-agent environment, and Table III presents the comparison between original learning performance and 6-agent model transfer data in 12-agent environment. In transfer experiments, the team of 8 or 12 agents trains their new individual student actor-critic network and self-learning network based on the transferred attention mechanism trained by a smaller team.

According to the compared results summarized in Table II and III, our approach can be efficiently transferred. The transferred model successfully achieve nearly 90% of the original training performance, which also better than all other approaches with fully training. It can save large computational costs for large-size team of agents. . Our shared attention selector can effectively solve new tasks based on related experience.

VII. CONCLUSIONS AND FUTURE WORK

We introduce a parallel knowledge-transfer framework, PAT for decentralized multi-agent reinforcement learning. Our key idea is designing two acting mode for agents and using a shared attention mechanism to select behavior knowledge

TABLE II
TRANSFER EVALUATIONS IN M=8

Task M=8	Original Avg_step	Avg_reward	M=4 Transfer Avg_step	Avg_reward
Grid	914 ± 65	27.80 ± 3.49	1021 ± 18	24.86 ± 2.45
Moving	1174 ± 42	10.79 ± 3.58	997 ± 10	9.37 ± 3.02

TABLE III
TRANSFER EVALUATIONS IN M=12

Task M=12	Original Avg_step	Avg_reward	M=8 Transfer Avg_step	Avg_reward
Grid	1157 ± 37	20.34 ± 1.49	1230 ± 9	18.85 ± 0.98
Moving	1239 ± 32	-0.32 ± 0.43	1389 ± 24	3.06 ± 0.09

from other agents to accelerate student agent learning. We empirically evaluate our proposed approach against all state-of-the-art advising or teaching methods in multi-agent environments. Results in experiments of scaling the number of agents and model transfer are also shown. Extending knowledge transfer in joint task learning and more complicated multi-agent systems is our future research direction.

REFERENCES

[1] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[2] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1633–1685, 2009.

[3] L. Matignon, L. Jeanpierre, and A.-I. Mouaddib, "Coordinated multi-robot exploration under communication constraints using decentralized markov decision processes," in *Twenty-sixth AAAI conference on artificial intelligence*, 2012.

[4] H. M. Le, Y. Yue, P. Carr, and P. Lucey, "Coordinated multi-agent imitation learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1995–2003.

[5] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, 2017, pp. 6379–6390.

[6] S. Sukhbaatar, R. Fergus *et al.*, "Learning multiagent communication with backpropagation," in *Advances in Neural Information Processing Systems*, 2016, pp. 2244–2252.

[7] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 2137–2145.

[8] J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Advances in Neural Information Processing Systems*, 2018, pp. 7254–7264.

[9] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, and J. Pineau, "Tarmac: Targeted multi-agent communication," *arXiv preprint arXiv:1810.11187*, 2018.

[10] L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," in *Innovations in multi-agent systems and applications-1*. Springer, 2010, pp. 183–221.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[12] D. Hadfield-Menell, S. J. Russell, P. Abbeel, and A. Dragan, "Cooperative inverse reinforcement learning," in *Advances in neural information processing systems*, 2016, pp. 3909–3917.

[13] F. L. Da Silva and A. H. R. Costa, "A survey on transfer learning for multiagent reinforcement learning systems," *Journal of Artificial Intelligence Research*, vol. 64, pp. 645–703, 2019.

[14] J. A. Clouse, "On integrating apprentice learning and reinforcement learning." 1997.

[15] O. Amir, E. Kamar, A. Kolobov, and B. Grosz, "Interactive teaching strategies for agent training," 2016.

[16] L. Torrey and M. Taylor, "Teaching on a budget: Agents advising agents in reinforcement learning," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 1053–1060.

[17] A. Fachantidis, M. Taylor, and I. Vlahavas, "Learning to teach reinforcement learning agents," *Machine Learning and Knowledge Extraction*, vol. 1, no. 1, pp. 21–42, 2018.

[18] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 330–337.

[19] Y. Zhan, H. B. Ammar *et al.*, "Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer," *arXiv preprint arXiv:1604.03986*, 2016.

[20] F. L. Da Silva, R. Glatt, and A. H. R. Costa, "Simultaneously learning and advising in multiagent reinforcement learning," in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2017, pp. 1100–1108.

[21] S. Omidshafiei, D.-K. Kim, M. Liu, G. Tesauro, M. Riemer, C. Amato, M. Campbell, and J. P. How, "Learning to teach in cooperative multiagent reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6128–6136.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[23] J. Oh, V. Chockalingam, S. Singh, and H. Lee, "Control of memory, active perception, and action in minecraft," *arXiv preprint arXiv:1605.09128*, 2016.

[24] J. Choi, B.-J. Lee, and B.-T. Zhang, "Multi-focus attention network for efficient deep reinforcement learning," in *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[25] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," *arXiv preprint arXiv:1810.02912*, 2018.

[26] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," 2014.

[27] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in neural information processing systems*, 2000, pp. 1057–1063.

[28] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[29] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," *arXiv preprint arXiv:1410.5401*, 2014.

[30] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.