Cooperative Behavior by Multi-agent Reinforcement Learning with Abstractive Communication

Jin Tanda
Department of Computer Science
Nagoya Institute of Technology
Nagoya, Japan
tanda.jin@itolab.nitech.ac.jp

Ahmed Moustafa
Department of Computer Science
Nagoya Institute of Technology
Nagoya, Japan
ahmed@nitech.ac.jp

Takayuki Ito
Department of Computer Science
Nagoya Institute of Technology
Nagoya, Japan
ito.takayuki@nitech.ac.jp

***Abstract*—Reinforcement learning (RL) is a major area of machine learning that aims to develop intelligent agents that are able to adapt in random environments appropriately. In this regard, RL has shown good results when applied to complex tasks such as playing video games. In addition, in multi-agent environments, RL has shown strong potential especially with the recent developments. However, there exist few studies that focus on developing cooperation among learning agents. In general, cooperative behavior among learning agents shows higher performance than independent agent behavior. Therefore, in this research, we focus on the cooperative behavior on Predator-Prey game in a continuous space, which is widely used as one of the typical simulation of Multi-agent environment. Especially we focus on predators that their goal is to catch a prey. We propose Leader-Follower model as the organization of predators, and investigate how they cooperate with each other to achieve their goal considering the prey's policy using a model of RL. The results of our work indicate that a communication between Leader and Followers affects high performance. In addition, we acquire an interesting result as a process of achieving their goal. We investigate the movement locus of them in three cases which is different reward settings, and in each case, they take different policy depending on the reward. We visualize the movement of locus, and discuss about their cooperation and effectiveness.**

***Index Terms*—Reinforcement learning, Multi-agent, Cooperative behavior**

I. Introduction

Recently, the machine learning field is rapidly developing several applications such as image recognition, natural language processing and robotic control. For example, face image recognition is an innovative technique, that has several applications in real world. Those machine learning algorithms are called supervised learning, and their goal is to generalize an optimal function using training data. On the other hand, reinforcement learning is also developing for several applications such as indoor robot navigation and video games [1] [2]. Recently, OpenAI published OpenAI Gym; environments in which we can experiment with renforcement learning algorithms easily. [3]. It can build a model without prepared training data beforehand because it makes the training data as an experience that is during the training. There are three reasons, that obtaining a behavior of agent by learning is necessary: 1) Designer cannot represent all state of an agent in an environment in advance [6] [9]. 2) Designer also cannot expect a change of an environment from hour to hour [6] [9]. 3) It is difficult to write a code that solves the problem directly [6]. In real world, there are not only single agent tasks, but also many multi-agent tasks which need to cooperate since agents interact with each other. For example, autonomous car needs to interact and consider other cars which have different policy in a road. In this paper, as a next step of reinforcement learning, we focus on a cooperative behavior in multi-agent environment through reinforcement learning. The goal of our work is to indicate the effectiveness of cooperation in reinforcement learning among agents that they have the same purpose like a team. We describe a model, which enables agents to cooperate by utilizing limited communication among agents.

II. Preliminaries

In this section, we introduce the background work of our research.

A. Partially Observable Markov Decision Process (POMDP)

In general, reinforcement learning environment is represented as a Markov Decision Process (MDP) to make the problem clear. MDP is defined by 4 elements: states, actions, a transition function, and a reward function.

$$S = \{s^1, s^2, ..., s^N\} \tag{1}$$

$$A = \{a^1, a^2, ..., a^K\} \tag{2}$$

$$T : S \times A \times S \to [0, 1] \tag{3}$$

$$R : S \times A \times S \to \mathbb{R} \tag{4}$$

The transition function $T(s, a, s')$ is a probability when the state is s and the action is a, and the next state is s', then,

$$T(s, a, s') = Pr(s_{t+1} = s'|s_t = s, a_t = a) \tag{5}$$

$R(s, a, s')$ is an immediate reward when the action a is executed when the state is s and the next state is s'.
On the other hand, when agents can only observe limited information, the environment is called POMDP [10]. Agents get $o \in \Omega$ as a set of observable information, which is not perfect information of the environment. Therefore, the agents need to learn using the incomplete information.

B. Reinforcement Learning

Reinforcement learning is a machine learning algorithm that obtains policy for an intelligent agent to achieve a goal in an environment. During the training, agent in an environment observe a state $s \in S$ and take an action $a \in A$, then it gets a reward r based on the state and action. Agent tries to get high reward in the trial and it connects to the goal. As deep reinforcement learning algorithms, Deep Q-Learning (DQN) [1] and Deep Deterministic Policy Gradient (DDPG) [11] are widely known.

DQN : DQN is expanded Q-Learning algorithm by deep learning. In Q-Learning algorithm, Q function is defined as an action value function, and optimize it. Agent choose an action from the Q value and the state.

$$\begin{aligned} Q(s,a) \leftarrow & Q(s,a)+ \\ & \alpha(R(s,a)+\gamma \max_{a'} \mathbb{E}[Q(s',a')]-Q(s,a)) \end{aligned} \tag{6}$$

This equation means that $Q(s,a)$ gets close to $R(s,a) + \gamma \max_{a'} E[Q(s',a')] - Q(s,a)$ during the training. DQN approximates the Q function using neural network. Conventional Q-Learning algorithm cannot expand to large-scale problem because it is a discrete method. In other words, Q-Learning needs to have and evaluate all values of states discretely. Therefore, if Q-Learning is used in a large-scale problem, memory shortage occurs, or the learning cannot converge. DQN overcomes the large-scale problem by the approximation. Since DQN approximates the Q function by neural network, memory shortage does not occur, and the learning becomes easy to converge more.

$$L = \frac{1}{2}\mathbb{E}[(R(s,a)+\gamma \max_{a'} Q_\theta(s',a') - Q_\theta(s,a))^2] \tag{7}$$

$R(s,a)+\gamma \max_{a'} Q_\theta(s',a')$ is treated as target, and the model optimizes $Q_\theta(s,a)$.

C. Multi-Agent Reinforcement Learning (MARL)

1) The issues of MARL: There are a lot of works of multi-agent reinforcement learning to realize intelligent agents for real world [7] [8]. On the other hand, reinforcement learning for single agent is also studied, and these made achievements until now [2] [11] [15] [16]. However, in the field of multi-agent reinforcement learning, it still have a lot of issues to solve problems. The problems are following points [9] : (1) when there are multiple agents which learns in the same environment, even if a state is good for an agent, it may be bad for other agent. Therefore, to define the goal of learning for agents is difficult. (2) During the training, since other agents also learn and change their behavior at the same time, each agent needs to consider the change. In other words, the training tends to be unstable. (3) Since the number of agents increases, the scale of the state also increases exponentially. For example recently, in response to the exponentially increased scale problem, a paper about effective exploration was published [5].

2) MARL for conventional environment: Before the development of deep reinforcement learning, many researches of MARL has worked on discrete problems such as grid world problem [8] [13]. These ideas and logics of the researches are surely useful for real world. However, almost of all environment in real world are formulated as continuous world problem. Therefore, a research of MARL which can apply to continuous environment, is required. It has to consider continuous problem beyond discrete problem, and these researches has attracted attentions using deep reinforcement learning. However, these still have issues for example, learning tend to be unstable, and it is hard to converge or to optimize policy. To solve these problems is also current topic of MARL [5].

3) Multi-Agent Deep Deterministic Policy Gradient: Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [4] is a deep reinforcement learning method for multi-agent domains. MADDPG is based on the Actor-Critic architecture, and it is supposed to follow some constraints: (1) At execution time, the learned policies can only use local information (i.e. their own observations); (2) they do not assume a differentiable model of the environment dynamics; (3) they do not assume any particular structure on the communication method between agents. The core point of the method is to use the centralized training with decentralized execution. The architecture is shown in Figure 1 [4]. The Q function is prepared for each agent as critic, and each Q function inputs all agent's observations and actions in training phase. The policy function is also prepared for each agent as actor. Each policy function is trained based on the Q function. In execution phase, each agent's information is not shared. Only policy functions are used for actions, so it is the kind of decentralized actor.

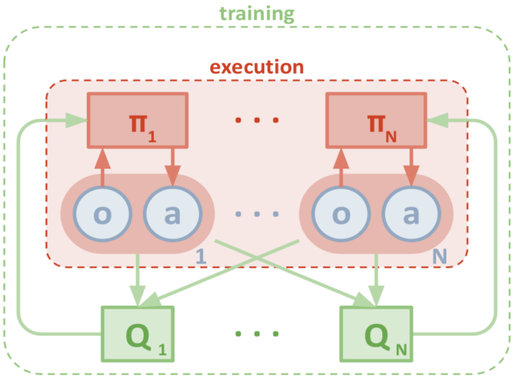

Fig. 1. Overview of the multi-agent decentralized actor, centralized critic approach [4].

III. Multi-Agent Cooperative Task Scenario

In this section, we introduce the scenario that agents need to cooperate with each other.

A. Predator-Prey Game

Predator-Prey game is often used as a representative game of multi-agent cooperative task [13]. The original version of the predator-prey game was introduced by Benda et al [12]. In the game, there are two types of agents which are predators and prey. They are allowed to move on an infinitely spread rectangular grid world. At each turn, each agent can move one cell. The goal of predators is to completely surround the prey by occupying the grid positions north, south, east and west of the red agent.

In our scenario, we use one of the Multi-agent Particle Environments (MPE), expanded Predator-Prey game. It is expanded from grid world (discrete world) to continuous world. There is no grid in the field, and agents are allowed to move smoothly according to the physical law. We show our environment of Predator-Prey game in Figure 2. In the case of conventional grid world, state and action is discrete. Therefore, all agents usually move the same distance which means one cell simultaneously, and it can be considered simply. However, once it expands continuous, the problem becomes more difficult because the speed is not the same, each agent has each speed. We define the speed that the maximum velocity of the prey is faster than predator one. Predators cannot catch or catch with the prey without cooperation because if one predator goes to the direction to the prey, it cannot catch up with the prey since the speed is slower. Even though it is more difficult, in general, problems of the real world are continuous world, so we consider that applying to continuous environment is more natural.

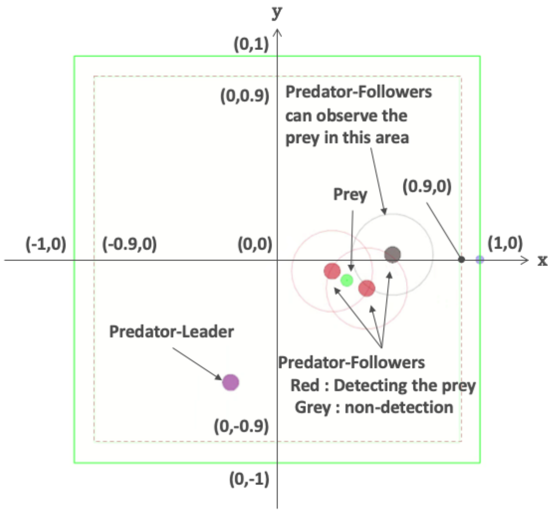

Fig. 2. Predator-Prey Game

To indicate the effectiveness of cooperation among agents, we focus on predators. The cooperative task for predators is to catch the prey, and we try to enable them to do it using our method.

The prey acts on the basis of following rules: 1) prey has a destination, and it always tries to go there. The destination is selected from nine candidate points which are the center of the field and eight around points (corners) of the center. Basically, the destination is decided by the sum of distance between predators and the point. In other words, the farthest point from predators is selected as the destination. For example, in the case of Figure 3, the bottom left point is selected.

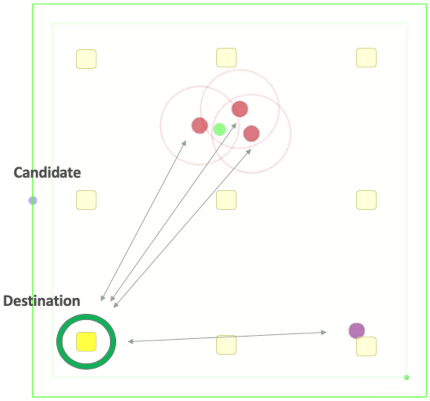

Fig. 3. Prey's behavior

IV. Leader-Follower Model

In real world, when team cooperation is needed for achieving a goal, leader is often provided to make the team effective. For example, soccer team definitely have a leader as a captain, and he often instructs and organizes the members to win.

According to the concept, we define the Leader-Follower model, and make the situation to apply it on Predator-Prey game as one of the example environment. In Predator-Prey game, we focus on Predator's team cooperation using Leader-Follower model. As shown in Figure 2, there are one predator-leader (Purple) and three predator-followers (Red and Grey) .

We indicate the performance difference between Leader and Follower in Table I.

TABLE I
Performance difference between Leader and Follower

	Leader	Follower
Observation	self position self velocity Follower's position Follower's velocity	
	(always) Prey's position (always) Prey's velocity	(limited) Prey's position (limited) Prey's velocity
	-	Leader's communication*
Action	x axis force y axis force	
	communication*	-

The area that agent is allowed to move is from(-1.0,-1.0) to (1.0,1.0). Both of Leader and Follower observe self position, self velocity, Follower's position, and Follower's velocity. The difference of observation ability between them is field of view which can find the prey around them. Leader have wide field of view, so it can find the prey anytime. On the other hand, Follower has a limited field of view, so only when the prey is within the Follower's view, Follower can find the prey as shown in Figure 2. The red agents (Followers) find the prey because the prey is within their field of view (Circle).

Leader can communicate with Follower as an action one-sidedly. However, we define that Leader cannot tell the direct complete information like the prey's position as a communication action but only tell abstractive communication as shown in Table II. Moreover, Leader only can tell a same instruction for all Followers. This is like a formation of team sports. For example, in soccer game, Leader usually tell abstractive communication such as "going forward" or "make formation A" because it is time-sensitive environment.

TABLE II
COMMUNICATION ACTIONS

ID	action command	force
0	free	0
1	left	$-x$
2	right	$+x$
3	down	$-y$
4	up	$+y$
5	come on	direction to Leader
6	far away	direction contrary to Leader

The communication action is described in Table II. There is 7 actions, and Leader chooses one action by neural network model, and tells the same ID to all Followers. Leader also adds small force to Followers depending on the action to make sense to the ID. In training, since Leader and Followers want to catch the prey to get a high reward, Leader comes to try to instruct exactly communication action, which means the neural network model optimize the choice of actions. As for Followers, the communication action is one input of their neural network model. We hypothesis that Followers learn the input by associating the small force.

V. EXPERIMENTS

In this section, we describe our experiments and the results.

A. Experiment Settings

We use MADDPG algorithm as MARL in Leader-Follower model. The model is trained with 50 steps per episode, and 45,000 episodes are executed in all of the cases. Moreover, the training is unstable, which means that the results tend to change every time as the beginning of the deep reinforcement learning is based on random trials. Then, for measurement, in each case the training is executed 10 times.

The goal of predators is to catch with the prey. Reward functions are defined following equations. R_L is for Predator-Leader, and R_F is for Predator-Follower. The reward is used every step. The C function is a binary function. It affects the reward function when a predator catch with the prey. The $P(x, y)$ is a penalty function. When agents go out the field, it is applied.

$$R_L(s,a) = C(self) + \sum_{i=0}^{N} C(F_i) - P(x,y) \tag{8}$$

$$R_F(s,a) = C(self) + \sum_{F_i \neq self} C(F_i) + C(L) - P(x,y) \tag{9}$$

$$C(agent) = \begin{cases} r_{agent} & \text{when the agent catches with Prey} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

$$P(x,y) = \begin{cases} 0 \ (|x| < 0.9 \ and \ |y| < 0.9) \\ (|x| - 0.9) * w + (|y| - 0.9) * w \ \text{(otherwise)} \end{cases} \tag{11}$$

B. Preparation for experiments

1) Curriculum Learning [14]: At the beginning of the training, agents choose an action randomly because there is no experience yet. Then, it is difficult for predators to obtain success reward. Therefore, we apply Curriculum Learning, which is a method for successfully getting a reward deliberately. In this case, the prey behavior is changed after 5,000 episodes, the prey tries to catch with predator by itself. Then, success experience is obtained at the beginning of the training, and it can make good use of the experience for the future of the training.

C. Leader Communication

To indicate the utility of the Leader communication, we experiment learnings; using communication instruction and not using it. We call the learning model of using communication instruction "COM", and call the learning model of not using it "non-COM". We experiment three COM models and one non-COM model. The difference among three COM models and non-COM models is reward settings as shown in Table III. In each case, ten models are built, and 100,000 steps are executed using each model. The score of Figure 4 is the average of ten models evaluation.

TABLE III
REWARD SETTINGS

	Leader		Follower		
Capturer	Self	Follower	Self	Leader	Other
Case 1	+5	+5	+5	+5	+5
Case 2	+3	+4	+4	0	+2
Case 3	+6	+4			
non-COM	+5	+5	+5	+5	+5

Case 1 All predators (Leader and Follower) get the same reward when they catch with the prey.
Case 2 Leader can get higher reward when Follower catches with the prey than Leader catches. Moreover, Follower does not get a reward even when Leader catches with the prey. We make a hypothesis that it encourages the positiveness of Follower's behavior.
Case 3 Leader can get higher reward when Leader catches with the prey than Follower catches. The reward of Follower is the same as Case2.

The result is shown in Figure 4 and Figure 5. Figure 4 shows the number of captures among three cases and non-COM cases. All of scores of three cases with communication are higher than non-COM cases. It indicates that communication action affects good result from the point of achieving goal even though it is abstractive communication.

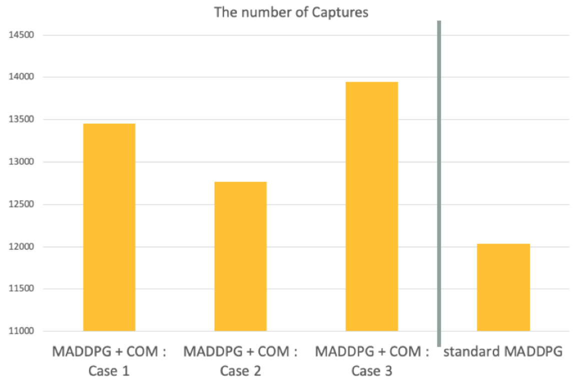

Fig. 4. Performance comparison between the case with communication action and non-communication action

Figure 5 shows the ratio of captures between Leader and Followers. In Case 1, all predators (Leader and Follower) get the same reward even if any other predators catch the prey. Then, as shown in Case 1 in Figure 5, since Leader can always observe the prey, almost all captures are due to Leader. Follower does not need to behave positively toward catching the prey because they can get the reward when Leader catches. In Case 2, almost all captures are due to Follower, and Leader almost does not catch. There are two mainly reasons of the result. First, Leader can get a higher reward when Follower catch the prey than itself catch. Second, Follower cannot get a reward when Leader catches the prey, and it can get a reward when itself catch or other followers catch. The result is in contrast to the result of Case 1. In Case 3, it is the best score of three cases. The difference compared to Case 2, is Leader's reward. Leader can get higher reward by catching the prey itself than followers catch. Then, Leader tries to catch by itself, and followers also try to catch positively. It is like an equilibrium point between Leader and Follower reward settings.

D. *Investigate cooperative behavior*

We discussed about these experiments from the point of score which is the number of captures in Section V-C. In this section, we discuss about cooperative behavior. Figure 6 shows the movement locus of agents. the purple locus is Leader movement, and the green is the prey movement. The black is Follower movement who do not observe the prey at the time, and the red is also Follower movement but who observe the prey by itself at the time.

In Case 1, since Followers get the same reward even when Leader catch the prey, Followers do not try to catch aggressively. In addition, we discovered interesting fact of their movements. Followers keep being around a specific position as you can see in Figure 6. This is cooperative behaviors of predators. Leader instruct to keep being around there to Followers, and Followers also try to keep doing it. Then, the area of the prey's movement is restricted because of the rule which is how to select the target point where the prey goes. It can be said that Leader and Followers could learn the prey's movement policy, and take measures using communication interaction. On the other hand, In Case 2, since Followers cannot get the reward when Leader catch the prey, and can only get it when Followers catch, Followers try to catch by themselves. In addition, Leader gets higher reward when Followers catch the prey than itself catch. Therefore, Leader has the policy that it keeps being around specific position to restrict the prey's area of movement and to let Followers catch the prey. Followers try to catch the prey aggressively. In Case 3, both of Leader and Followers try to catch the prey because of the rewards. They move around evenly. In conclusion of this section, Leader and Followers cooperate with each other, and it depends on the reward. We indicated it as action patterns which is movement locus of agents. However, we still do not indicate it quantitatively and it is one of the issue to evaluate what cooperative behavior is.

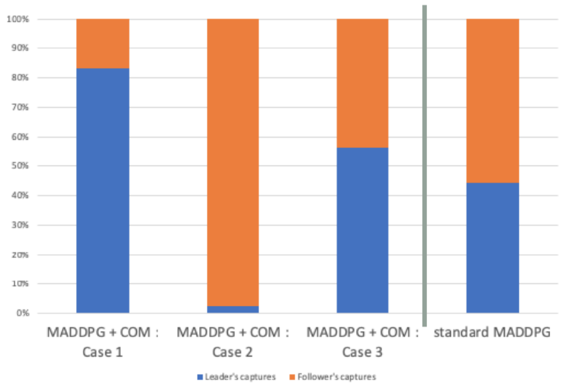

Fig. 5. The ratio of captures between Leader and Followers

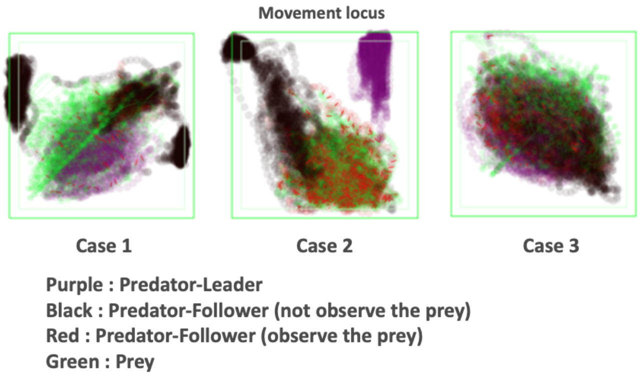

Fig. 6. Movement locus of agents

VI. Discussion

We defined the Leader-Follower model, and experimented and investigated cooperations between Leader and Follower. As we mentioned in Section V-C, even the leader's communication action is abstractive, it supports to achieve their goal. The interesting point is that Follower learns the meaning of Leader's abstractive communication through the training by theirselves implicitly. Leader also learns how to instruct as a communication action in training.

In these cases of the experiments, we found an interesting cooperative behavior among predators. For example, in Case2, Leader does not try to catch with the prey. It tries to stay at appropriate position. This strategy is better for predators because the prey decide the destination by the sum of distance between predators and the point, and this Leader's behavior prevents the prey from escaping far point like the situation of Figure 2. Predators learn also the rule of Prey's moving in training. It exactly can be said that this strategy is invented by the communication. Leader stays at appropriate position to let the range of the prey's behavior narrow, and it also instruct Followers to catch up easily.

However, evaluating of the cooperative behavior like teamwork quantitatively is difficult, and finding how to evaluate quantitatively is future task.

While we define the constraints of communication action which means Leader can only tell abstractive information; for example, it cannot tell Prey's position, it is indicated that even the communication is abstractive, it performs effectively. We consider like the situation which can use only abstractive or limited information as a communication in the real world.

Through the experiments, we found that Leader-Follower model with communication ability is effective for multi-agent environment, and it can integrate to multi-agent reinforcement learning. While we experiment the simple cooperative task using MADDPG, we are sure that the architecture has a possibility for achieving more complex cooperative task.

VII. Conclusion

In this paper, we proposed Leader-Follower model as the organization of predators in Predator-Prey game in a continuous space, and investigated how they cooperate with each other to achieve their goal considering the opponent policy using a model of RL. We indicated that a communication between Leader and Followers in the model affects high performance. Moreover, from the view of process that predators try to achieve their goal, we investigated the movement locus of them, and indicated the policies. These policies make sense, and we can understand why they take these policies when compared to reward settings. We evaluated cooperative behavior in Leader-Follower model from the points of how many times they achieve their goal, and movement locus of predators visually. However, we think it is not robust to evaluate cooperative behavior, it is not direct evaluation. Therefore, we think we need to find the evaluation of cooperation or cooperative behavior more quantitatively. As future works, we will propose a decentralized MARL algorithm instead of MADDPG considering real world applications. Decentralized MARL algorithms are more realistic approach to apply real world applications actually.

References

[1] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).

[2] Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." Nature 518.7540 (2015): 529.

[3] Brockman, Greg, et al. "Openai gym." arXiv preprint arXiv:1606.01540 (2016).

[4] Lowe, Ryan, et al. "Multi-agent actor-critic for mixed cooperative-competitive environments." Advances in Neural Information Processing Systems. 2017.

[5] Burda, Y., Edwards, H., Storkey, A., and Klimov, O. Exploration by Random Network Distillation. arXiv:1810.12894 [cs, stat], October 2018b.

[6] Russell, S. and Norvig, P.: Artificial Intelligence: A Modern Approach, Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition (2009).

[7] Tan, Ming. "Multi-agent reinforcement learning: Independent vs. cooperative agents." Proceedings of the tenth international conference on machine learning. 1993.

[8] Yu, Chao, Minjie Zhang, and Fenghui Ren. "Coordinated learning by exploiting sparse interaction in multiagent systems." Concurrency and Computation: Practice and Experience 26.1 (2014): 51-70.

[9] Buşoniu, L. et al.: A comprehensive survey of multi- agent reinforcement learning, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol. 38, No. 2, pp. 156-172 (online), DOI: 10.1109/TSMCC.2007.913919 (2008).

[10] Littman, Michael L. "Markov games as a framework for multi-agent reinforcement learning." Machine learning proceedings 1994. Morgan Kaufmann, 1994. 157-163.

[11] Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971 (2015).

[12] Benda, M.; Jagannathan, V.; and Dodhiawalla, R. 1985. On Optimal Cooperation of Knowledge Sources. Technical Report BCS-G2010-28, Boeing AI Center.

[13] Osawa, Eiichi. "A Metalevel Coordination Strategy for Reactive Cooperative Planning." ICMAS. Vol. 95. 1995.

[14] Bengio, Yoshua, et al. "Curriculum learning." Proceedings of the 26th annual international conference on machine learning. ACM, 2009.

[15] Hessel, Matteo, et al. "Rainbow: Combining improvements in deep reinforcement learning." Thirty-Second AAAI Conference on Artificial Intelligence. 2018.

[16] Salimans, Tim, and Richard Chen. "Learning Montezuma's Revenge from a Single Demonstration." arXiv preprint arXiv:1812.03381 (2018).