Decomposed Deep Reinforcement Learning for Robotic Control

Extended Abstract

Yinzhao Dong
School of Computer Science & Technology, Dalian University of Technology, Dalian, China
1447866357@qq.com

Chao Yu*
School of Data & Computer Science, Sun Yat-Sen University, Guangzhou, China
yuchao3@mail.sysu.edu.cn

Paul Weng
UM-SJTU Joint Institute, Shanghai Jiao Tong University, Shanghai, China
paul.weng@sjtu.edu.cn

Ahmed Maustafa
Department of Computer science, Nagoya Institute of Technology, Nagoya, Japan
ahmed@nitech.ac.jp

Hui Cheng
School of Data & Computer Science, Sun Yat-Sen University, Guangzhou, China
chengh9@mail.sysu.edu.cn

Hongwei Ge
School of Computer Science & Technology, Dalian University of Technology, Dalian, China
hwge@dlut.edu.cn

ABSTRACT

We study how structural decomposition and interactive learning among multiple agents can be utilized by deep reinforcement learning in order to address high dimensional robotic control problems. We decompose the whole control space of a certain robot into multiple independent agents according to this robot's physical structure. We then introduce the concept of *Degree of Interaction* (DoI) to describe the level of dependencies (i.e., the necessity of coordination) among the learning agents. Three different methods are then proposed to compute the DoI dynamically during learning. The experimental evaluation demonstrates that the decomposed learning method is substantially more sample efficient than the state-of-the-art algorithms, and more explicit interpretations can be generated on the final learned policy as well as the underlying dependencies among the learning agents.

KEYWORDS

Robotic control; Deep reinforcement learning

ACM Reference Format:
Yinzhao Dong, Chao Yu*, Paul Weng, Ahmed Maustafa, Hui Cheng, and Hongwei Ge. 2020. Decomposed Deep Reinforcement Learning for Robotic Control. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION

Deep Reinforcement Learning (DRL) holds the promise of learning a wide range of robotic behaviors in challenging tasks such as locomotion and manipulation [1, 3, 5, 6, 9, 18]. However, directly applying DRL algorithms to real-world robotic control problems is difficult due to the high dimensional state/action space. In addition, the majority of the existing DRL algorithms [3, 15, 17] directly search in the entire state/action space and output the learned policy in an end-to-end manner. As a result, it becomes difficult to provide any meaningful interpretations on the performance of these learning algorithms.

In order to address these challenges, we propose a general framework, *Structure-Motivated Interactive LEarning* (SMILE), for efficient and interpretable DRL in robotic control. By considering the robot's physical structure, the whole robot structure is decomposed into multiple learning agents, each of which controls its own individual actions. This decomposition scheme avoids searching in the huge combinatorial and continuous state/action space, thus reducing the complexity required to solve the control problem [14, 21]. A coordination graph is used in order to synchronize the interactions among the agents. Each edge on this graph indicates that the two linked agents need to coordinate over their behaviors for better learning performance. In this context, the level of dependencies between any two agents is measured by a value that we call *Degree of Interaction* (DoI), which reflects the necessity of coordination. We then propose three different ways to compute the DoI during agents' learning process: the *ATTENTION* method to compute the DoI for a fully connected graph using an attention mechanism, the *Partially Observable Dynamics Topology* (PODT) method to compute DoI using the prediction errors in other agents' states and builds a graph that links those agents with the largest DoI, and *Attention-PODT* (A-PODT) that is able to take advantage of both previous two methods to obtain a trade-off between performance and complexity. The experimental evaluations in typical Mujoco environments [20] verify the effectiveness of the proposed methods.

There are a number of studies that focus on decomposed learning for a robotic control problem [2, 4, 7, 8, 10, 11, 13, 14, 21, 24]. Unlike all the existing approaches, where agents either learn independently without any explicit coordination, or coordinate with each other in a fixed procedure (e.g., by sharing a fixed amount of information), SMILE models the dynamic dependencies among the agents through computing the continuously changing DoI. In this way, the most

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.

important information can be naturally taken into account in order to increase the coordination efficiency during the learning process. Moreover, explicit interpretations can be derived regarding the underlying dependencies among the components of a robot in different motion postures.

2 THE SMILE FRAMEWORK

In SMILE, a DoI value indicates the level of dependency between two agents. Each agent uses DoI to integrate the state information of other agents, and generates an enhanced state, which is defined as follows:

$$\hat{s}_i = Concat(w_{i,1} * s_1, \dots w_{i,j} * s_j, \dots w_{i,n} * s_n, s_g) \in \mathbb{R}^{\hat{M}} \quad (1)$$

where $\hat{s}_i$ denotes the $\hat{M}$-dimensional enhanced state of the *i-th* agent, and $w_{i,j} \in \mathbb{R}$ denotes the importance weight (i.e., DoI) of agent A_j as considered by agent A_i. Each agent then makes decisions based on its local actions and the enhanced states using various existing DRL algorithms (e.g., PPO [16]).

The ATTENTION Method. First, the local state of agent A_i is fed into a multi-layer perceptions (MLP) to obtain a feature vector with fixed-size b: $f_i = F_{in}(s_i)$, where F_{in} denotes an MLP, and $f_i \in \mathbb{R}^b$ is the feature vector of agent A_i. Then, the Scaled Dot-Product Attention algorithm [22] is used to calculate the attention value. The joint feature between two agents $\langle f_i, f_j \rangle$ is then fed into two MLPs: the first one is an attention MLP that outputs the similarity value $\mathbf{K}_{ij} \in \mathbb{R}^h$ between f_i and f_j, and the other one is a value MLP that outputs a mapping value $\mathbf{V}$ of the joint feature. A soft-max function is used to normalize the similarity value, and the attention of agent A_j for agent A_i is computed by:

$$Att(f_i, f_j) = \frac{e^{\mathbf{K}_{ii}^\top \cdot \mathbf{K}_{ij}}}{\sum_{k=1}^{n} e^{\mathbf{K}_{ii}^\top \cdot \mathbf{K}_{ik}}}. \quad (2)$$

Given the values of $Att(f_i, f_j)$ and $\mathbf{V}_{ij}$, the enhanced state $\hat{s}_i$ of agent A_i is computed as follows:

$$\hat{s}_i = Concat(Att(f_i, f_1) * \mathbf{V}_{i1}^\top, \dots Att(f_i, f_j) * \mathbf{V}_{ij}^\top, \dots s_g). \quad (3)$$

where $\mathbf{V}_{ij} \in \mathbb{R}^m$ is the mapping value of joint feature $\langle f_i, f_j \rangle$.

The PODT Method. The PODT method computes the DoI using the prediction errors of other agents' states. Each agent predicts the states of other agents in the next step based on its own state and action using a predictor (e.g., a neural network). Then, it utilizes the difference between the predicted states and the real states of the other agents to measure the DoI values for these agents. A coordination graph then can be built with a topology that links the agents with the top largest DoI values. By building a sparser graph, the PODT method can only achieve approximated solutions due to the loss of information of other agents.

The A-PODT Method. In this method, we first use the *ATTENTION* method to generate the coordination graph, and the gradient of the *ATTENTION* method can be further corrected to keep the graph closer to that using the PODT method. Denote adjacency matrices of the coordination graph using the *ATTENTION* method and the PODT method as $\mathcal{W}_a$ and $\mathcal{W}_p$, respectively. Denote $loss_{a \to p} = \|\mathcal{W}_a - \mathcal{W}_p\|_2$ as the distance between these two graphs. $loss_{a \to p}$ is only related to the parameters of the PODT method. Let Θ denote the parameters of the whole policy, Θ_a denote the parameters of the *ATTENTION* method, $\Theta - \Theta_a$ denote the parameters that are not in the *ATTENTION* method, and $loss_{policy}$ denote the loss of the PPO algorithm. The gradients of these parameters can be computed as follows:

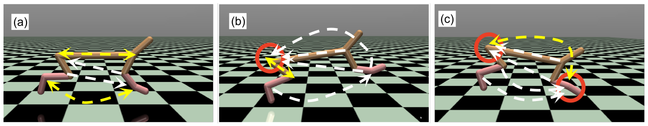

Figure 1: Interpretation of the learned policy while walking (a), jumping (b) and landing (c).

$$g_\theta = \frac{\partial loss_{policy}}{\partial \theta}, \text{ for } \theta \in \Theta - \Theta_a \quad (4)$$

$$g_\theta = (1 - \tau)\frac{\partial loss_{policy}}{\partial \theta} + \tau \frac{\partial loss_{a \to p}}{\partial \theta}, \text{ for } \theta \in \Theta_a \quad (5)$$

where τ denotes a trade-off hyper-parameter, which controls the proportion of correction.

3 EXPERIMENTS AND RESULTS

We evaluate the proposed methods in Swimmer, Hopper, Walker and Half-Cheetah. The experimental evaluations show that the *ATTENTION* method is more suitable in dealing with lower-dimensional robot control problems, however, it has high computational complexity. Meanwhile, the PODT method can reduce this computational complexity by reducing the topology of the graph, but it is easy to fall into local optimum solutions. By combining the benefits of both methods, the A-PODT method is able to reduce the computational complexity to a certain extent, and at the same time, avoid falling into local optimum solutions. We also compare A-PODT to some state-of-the-art DRL algorithms including PPO [16], DDPG [9], AC [12], REINFORCE [23] and CEM [19]. In Swimmer, A-PODT performs slightly better than PPO, but much better than the other algorithms. The distinction of A-PODT becomes more apparent in higher dimensional environments, which fully demonstrates the effectiveness of our proposed method. Figure 1 shows an illustration of the coordination graph in Half-Cheetah using the A-PODT method, where the yellow arrow represents a bidirectional connection indicating that both agents are concerned with each other, the white arrow represents a one-way connection, and the red circle represents the most important joint with the highest attention by other agents. As we can see, reasonable interpretations can be derived regarding the underlying dependencies among the components of a robot in different motion postures.

ACKNOWLEDGMENTS

This work is supported by the Dalian Science and Technology Innovation Fund under Grant 2018J12GX046.

REFERENCES

[1] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. A brief survey of deep reinforcement learning. *arXiv preprint arXiv:1708.05866* (2017).

[2] Lucian Busoniu, Bart De Schutter, and Robert Babuska. 2006. Decentralized reinforcement learning control of a robotic manipulator. In *2006 9th ICARCV*. 1–6.

[3] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Abbeel. 2016. Benchmarking deep reinforcement learning for continuous control. In *ICML*. 1329–1338.

[4] Uladzimir Dziomin, Anton Kabysh, Vladimir Golovko, and Ralf Stetter. 2013. A multi-agent reinforcement learning approach for the efficient control of mobile robot. In *2013 IEEE 7th IDAACS*, Vol. 2. 867–873.

[5] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. 2017. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 ICRA*. 3389–3396.

[6] Tuomas Haarnoja, Vitchyr Pong, Aurick Zhou, Murtaza Dalal, Pieter Abbeel, and Sergey Levine. 2018. Composable deep reinforcement learning for robotic manipulation. In *2018 ICRA*. 6244–6251.

[7] David L Leottau, Javier Ruiz-del Solar, and Robert Babuška. 2018. Decentralized reinforcement learning of robot behaviors. *Artificial Intelligence* 256 (2018), 130–159.

[8] David L Leottau, Javier Ruiz-del Solar, Patrick MacAlpine, and Peter Stone. 2015. A study of layered learning strategies applied to individual behaviors in robot soccer. In *RoboCup*. 290–302.

[9] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[10] José Antonio Martin, H De Lope, et al. 2007. A distributed reinforcement learning architecture for multi-link robots. In *4th ICICAR*, Vol. 192. 197.

[11] Laëtitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. 2009. Design of semi-decentralized control laws for distributed-air-jet micromanipulators by reinforcement learning. In *2009 IROS*. 3277–3283.

[12] Jan Peters and Stefan Schaal. 2008. Natural actor-critic. *Neurocomputing* 71, 7-9 (2008), 1180–1190.

[13] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. 2018. Graph networks as learnable physics engines for inference and control. *arXiv preprint arXiv:1806.01242* (2018).

[14] Erik Schuitema. 2012. Reinforcement learning on autonomous humanoid robots. *Mechanical Maritime and Materials Engineering* (2012).

[15] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *ICML*. 1889–1897.

[16] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[17] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *ICML*. 387–395.

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

[19] István Szita and András Lörincz. 2006. Learning Tetris using the noisy cross-entropy method. *Neural computation* 18, 12 (2006), 2936–2941.

[20] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *2012 IROS*. 5026–5033.

[21] Sebastiaan Troost, Erik Schuitema, and Pieter Jonker. 2008. Using cooperative multi-agent Q-learning to achieve action space decomposition within single robots. In *1st ERLARS*. 23–32.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*. 5998–6008.

[23] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[24] Chao Yu, Dongxu Wang, Jiankang Ren, Hongwei Ge, and Liang Sun. 2018. Decentralized Multiagent Reinforcement Learning for Efficient Robotic Control by Coordination Graphs. In *PRICAI*. 191–203.