Multi-Agent/Robot Deep Reinforcement Learning with Macro-Actions (Student Abstract)

Yuchen Xiao, Joshua Hoffman, Tian Xia, Christopher Amato
Khoury College of Computer Science, Northeastern University
360 Huntington Ave, Boston, Massachusetts 02115
{xiao.yuch, hoffman.j, xia.tia}@husky.neu.edu, c.amato@northeastern.edu

Abstract

We consider the challenges of learning multi-agent/robot macro-action-based deep Q-nets including how to properly update each macro-action value and accurately maintain macro-action-observation trajectories. We address these challenges by first proposing two fundamental frameworks for learning macro-action-value function and joint macro-action-value function. Furthermore, we present two new approaches of learning decentralized macro-action-based policies, which involve a new double Q-update rule that facilitates the learning of decentralized Q-nets by using a centralized Q-net for action selection. Our approaches are evaluated both in simulation and on real robots.

Introduction

Performing high-quality collaborative behaviors under large, stochastic and uncertain environments requires robots to asynchronously execute high-level actions and simultaneously reason about cooperations between teammates. Formally, multi-agent asynchronous decision-making under uncertainty in fully cooperative tasks is modeled as *Macro-Action Decentralized Partially Observable Markov Decision Processes* (MacDec-POMDPs) (Amato, Konidaris, and Kaelbling 2014). Although several multi-agent deep reinforcement learning methods have recently achieved impressive performance under both cooperative and competitive domains (Omidshafiei et al. 2017, Lowe et al. 2017, Foerster et al. 2018, Rashid et al. 2018), they all assume synchronous primitive-action executions over agents. It is not clear how to incorporate asynchronous macro-actions into these methods.

In our work (Xiao, Hoffman, and Amato 2019), we bridge this gap by (a) proposing the first decentralized macro-action-based learning framework with a new buffer called *Macro-Action Concurrent Experience Replay Trajectories* (Mac-CERTs) that properly maintains the macro-action-based transitions for each agent; (b) introducing a novel centralized macro-action-based learning framework that generates *Macro-Action Joint Experience Replay Trajectories* (Mac-JERTs) to learn a joint macro-action-value function using a conditional target-value prediction method. In other recent work (Xiao et al. 2019), we improve macro-action-based decentralized policy learning by presenting *Macro-Action-Based Decentralized Multi-Agent Double Deep Recurrent Q-Net* (MacDec-MADDRQN) that enables each agent's local Q-net update to consider the effects of other agents' macro-actions. MacDec-MADDRQN introduces a hyper-selection by performing centralized exploration or decentralized exploration during training. The best choice is unknown without knowledge of the domain properties. Therefore, a generalized version of this method, called Parallel-MacDec-MADDRQN, is also proposed in (Xiao et al. 2019) that executes centralized and decentralized explorations in two separate environments and optimizes the centralized Q-net and decentralized Q-nets in parallel.

Approach

In this section, we present an overview of our contributions.

Learned Macro-Action-Based *Decentralized* Policy

In this framework, during execution, agents collect concurrent macro-action-observation experiences into the Mac-CERTs buffer, in which each transition is represented as a tuple $\langle z_i, m_i, z'_i, r^c_i \rangle$ for each agent i, where $r^c_i = \sum_{t=t_{m_i}}^{\tau} r_t$ is an accumulated reward from the beginning time-step t_{m_i} to the termination step τ of the macro-action m_i. During training, we combine Decentralized Hysteretic DRQN (Omidshafiei et al. 2017) with Double DQN (referred to Dec-HDDRQN) to update each agent's individual macro-action-value function $Q_{\theta_i}(h_i, m_i)$ using a concurrent mini-batch of sequential experiences sampled from Mac-CERTs, by minimizing the loss: $\mathcal{L}(\theta_i) = \mathbb{E}_{<z_i,m_i,z'_i,r^c_i>\sim\mathcal{D}}\Big[\big(y_i - Q_{\theta_i}(h_i,m_i)\big)^2\Big]$, where $y_i = r^c_i + \gamma Q_{\theta^-_i}\big(h'_i, \arg\max_{m'_i} Q_{\theta_i}(h'_i, m'_i)\big)$ and h_i denotes the macro-action-observation history of agent i.

Learned Macro-Action-Based *Centralized* Policy

This framework aims at learning a centralized macro-action-value function. At each time-step, the Mac-JERTs buffer collects a joint macro-action-observation experience, represented as a tuple $\langle \vec{z}, \vec{m}, \vec{z}\,', \vec{r}^{\,c} \rangle$, where $\vec{r}^{\,c} =$

$\sum_{t=t_{\vec{m}}}^{\vec{\tau}} r_t$ is a shared joint accumulated reward for the agents' joint macro-action $\vec{m}$ from its beginning time-step $t_{\vec{m}}$ to the ending time-step $\vec{\tau}$ when *any agent* terminates its macro-action. The centralized macro-action-value function $Q_\phi(\vec{h}, \vec{m})$ is then optimized by minimizing the loss: $\mathcal{L}(\phi) = \mathbb{E}_{<\vec{z}, \vec{m}, \vec{z}\,', \vec{r}^{\,c}> \sim \mathcal{D}}\Big[\big(y - Q_\phi(\vec{h}, \vec{m})\big)^2\Big]$, where $y = \vec{r}^{\,c} + \gamma Q_{\phi^-}\big(\vec{h}\,', \arg\max_{\vec{m}\,'} Q_\phi(\vec{h}\,', \vec{m}\,' \mid \vec{m}^{\text{undone}})\big)$. Here, $\vec{m}^{\text{undone}}$ is the joint-macro-action over the agents who have not finished their macro-actions. It considers agents' asynchronous macro-actions execution status. This method is referred to as Cen-DDRQN in the results section.

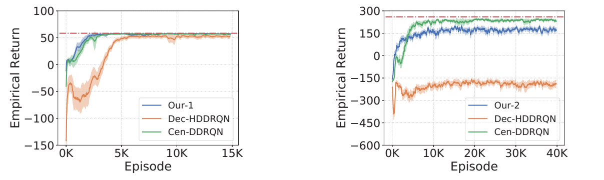
Macro-Action-Based *Decentralized* Multi-Agent Double Deep Recurrent Q-Net

This method adopts *centralized training decentralized execution* in deep Q-learning to learn the decentralized Q-net, Q_{θ_i}, for each agent i using the centralized Q-net, Q_ϕ. Here, the replay buffer is a merged version of Mac-CERTs and Mac-JERTs, containing the transition tuple $\langle \mathbf{z}, \mathbf{m}, \mathbf{z}', \mathbf{r}^{\mathbf{c}}, \vec{r}^{\,c} \rangle$, where $\mathbf{z} = \{z_0, ..., z_N\}$, $\mathbf{m} = \{m_0, ..., m_N\}$ and $\mathbf{r}^{\mathbf{c}} = \{r_0^c, ..., r_N^c\}$. During training, agents iteratively sample a mini-batch of sequential experiences to first optimize the centralized macro-action-value function Q_ϕ using Cen-DDRQN, and then train each decentralized macro-action-value function by performing gradient descent step on the loss: $\mathcal{L}(\theta_i) = \mathbb{E}_{<\mathbf{z}, \mathbf{m}, \mathbf{z}', \mathbf{r}^{\mathbf{c}}, \vec{r}^{\,c}> \sim \boldsymbol{\mathcal{D}}}\Big[\big(y_i - Q_{\theta_i}(h_i, m_i)\big)^2\Big]$, where $y_i = r_i^c + \gamma Q_{\theta_i^-}\Big[h_i', \big[\arg\max_{\mathbf{m}'} Q_\phi(\mathbf{h}', \mathbf{m}' \mid \mathbf{m}^{\mathbf{undone}})\big]_i\Big]$. This indicates the decentralized target Q-value is calculated in the double Q-learning manner, but using the centralized Q-net for the next macro-action selection of agent i under the conditional operation that considers teammates' behaviors and their asynchronous macro-action executions. Parallel-MacDec-MADDRQN also applies the above double Q-update rule for training each decentralized Q-net but purely based on decentralized experiences, while the centralized Q-net is trained only using the experiences generated by the centralized exploration in parallel in another environment.

Experiments and Results

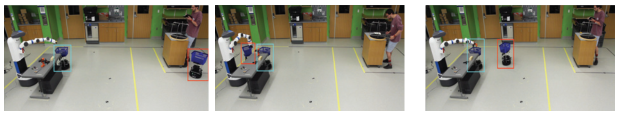
We evaluate our methods in both a benchmark problem (box pushing) and a warehouse domain. The warehouse task involves a human working on an assembly task in a workshop, which requires three particular tools for the future steps. A Fetch robot and two Turtlebots are respectively responsible for finding the correct tools on a table in the tool room and delivering the tools in the correct order to the human. The robots have no prior knowledge of the correct tools the human needs, so this has to be learned via training.

The simulation results shown in Fig. 1 first demonstrate that our macro-action-based policy learning frameworks enable the agents to perform near-optimal behaviors under both decentralized and centralized control in the box pushing domain, and under centralized control in the warehouse domain. Furthermore, the advantages of our new approaches MacDec-MADDRQN (Our-1) and Parallel-MacDec-MADDRQN (Our-2) are validated by achieving near-centralized performance and outperforming Dec-HDDRQN. The practical utility of our approach is proved by allowing real robots to perform reasonable collaborative behaviors to deliver the correct tools to the human in the proper order which is shown in Fig. 2 (Xiao et al. 2019).



(a) Box Pushing (20 × 20)   (b) Warehouse Tool Delivery

Figure 1: Averaged performance over 40 runs in two domains with optimal discounted returns as dash-dot lines.



(a) Pass tape measure   (b) Pass clamp   (c) Pass drill

Figure 2: Robots run the decentralized policies (learned via Parallel-MacDec-MADDRQN) in the warehouse domain

References

Amato, C.; Konidaris, G. D.; and Kaelbling, L. P. 2014. Planning with macro-actions in decentralized POMDPs. In *Proceedings of the Conference on Autonomous Agents and Multiagent Systems*.

Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018. Counterfactual multi-agent policy gradients. In *AAAI 2018: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; and Mordatch, I. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Neural Information Processing Systems (NIPS)*.

Omidshafiei, S.; Pazis, J.; Amato, C.; How, J. P.; and Vian, J. 2017. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2681–2690.

Rashid, T.; Samvelyan, M.; de Witt, C. S.; Farquhar, G.; Foerster, J.; and Whiteson, S. 2018. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML 2018: Proceedings of the Thirty-Fifth International Conference on Machine Learning*.

Xiao, Y.; Hoffman, J.; Xia, T.; and Amato, C. 2019. Multi-robot deep reinforcement learning with macro-actions.

Xiao, Y.; Hoffman, J.; and Amato, C. 2019. Macro-action-based deep multi-agent reinforcement learning. In *3nd Annual Conference on Robot Learning (CoRL)*.