Continuous control in deep reinforcement learning with direct policy derivation from Q network

Aydar Akhmetzyanov, Alexandr Klimchik, Rauf Yagfarov, Salimzhan Gafurov, Mikhail Ostanin

1 Center for Technologies in Robotics and Mechatronics Components, Innopolis University, Innopolis, Russia

{a.akhmetzyanov, r.yagfarov, s.gafurov, m.ostanin, a.klimchik[}@](mailto:%7d@)innopolis.ru

**Abstract.** Reinforcement learning approach allows to learn desired control policy in different environments without explicitly providing system dynamics. Model-free deep Q-learning algorithm is proven to be efficient on a large set of discrete-action tasks. Extension of this method to the continuous control task usually solved with actor-critic methods which approximate a policy function with additional actor network and uses Q function to speed up policy network training. Another approach is to discretize action space which will not give a smooth policy and is not applicable for large action spaces. A direct continuous policy derivation from Q network leads to optimization of action on each inference and training step which is not efficient but provides the optimal and continuous action. Time-efficient Q function input optimization is required in order to apply this method in practice. In this work we implement efficient action derivation method which allows using Q-learning in real time continuous control tasks. In addition, we test our algorithm on robotics control tasks from robotics gym environments and compare this method with modern continuous RL methods. The results have shown that in some cases proposed approach learns smooth continuous policy keeping the implementation simplicity of the original discreet action space Q-learning algorithm.

**Keywords:** Reinforcement learning, Q-learning, Continuous control, Robotics, Deep learning

Introduction

What is RL

Reinforcement learning as one of the three general machine learning paradigm allows to learn goal-based policy without explicit task dynamics knowledge by means of trials and errors. Reinforcement learning paradigm includes the following basic concepts: environments, agent, observation, reward, and action. Agent acts in the environment, choosing the actions based on observations, maximizing discounted cumulative reward formalized by bellman optimality principle [1]. The goal of the learning is to obtain a mapping from observation to action, known as a policy function.

RL applications

Reinforcement learning paradigm is proven to be efficient for a large set of tasks. For instance, RL as a method for optimizing non-differentiable loss function is capable of improving neural machine translation algorithms, review by Lijun Wu , Fei Tian [2] Besides NLP task, RL is also capable to provide end to end solution for financial portfolio optimization problem, classically solved by model based or analytical methods [3]. In robotics, RL successfully applied on number of continuous control tasks. For example, the work called Learning Dexterous In-Hand Manipulation demonstrates pre-training in simulation and knowledge transfer to reality for efficient goal-oriented robotics hand control. [4] Tasks mentioned above imply continuous control signal, thus classified as continuous control tasks which are common in the real scenarios.

Short about Q learning (limitation of pure rl from atari)

Q-learning [5] is one of possible Reinforcement Learning paradigm implementation. Q-learning is model-free algorithm; thus, it does not require predefined model of the environment in order to learn efficient behavior. The goal of Q-learning is to learn action-value function or Q function. Q function outputs expected discounted reward for specific state and set of possible actions which allows to derive a policy be selecting the action with the maximum Q value.

Continuous and discrete Action/state spaces

Actions and observation are classified into discrete and continuous. If both action space and observation space are discrete, the policy can be described as a table which maps states to finite actions. However, if the action space is continuous or state space is large, appropriate approximation of policy function needs to be used. In the deep Q-learning algorithm [6], deep reinforcement learning successfully applied on Atari game task with deep neural network with the raw pixels input, by means of optimization techniques combination such as fixed target network and experience replay.

Existing approaches to solve continuous action space

Q-learning could be extended to continuous control domain by various ways. The simpler option is to discretize continuous space, which is infeasible and inefficient in some cases and lead to expansion of action space. The most common approach, for instance used in DDPG algorithm [7], imply utilization of Q-function to train separate policy network.

Our approach description

In this work we consider pure deep Q-learning method with state-action input and single action value output shown in Fig. 1. This Q network architecture allows to work with continuous state-action domain. However, we can’t apply argmax function to select the most appropriate action in each state, instead in order to select optimal action on each iteration we need to apply optimization to find network input which maximize output of Q network.

Fig 1

Our work contribution experiments

Background

* 1. **Model-free RL**
  2. **Q learning**
  3. **Continuous action space Q learning methods**
  4. **Baselines**

Method

* 1. **Method description**
  2. **Policy derivation**
  3. **Method hyperparameters**

Experimental results

* 1. **Sample efficiency comparison**
  2. **Training process evaluation**
  3. **Policy execution complexity**

Conclusion

Short conclusion

Further work

Acknowledgments. This research was supported by the Russian Education Ministry under the Agreement No. 075-10-2018-017 on granting a subsidy from 11.26.2018 “Development of a commercial urban transport with the intellectual driver assistance system “City Pilot”. The customer is a federal target program. The unique identifier of the project RFMEFI60918X0005. Industrial partner - KAMAZ

References

1. Bellman, R.E. Dynamic Programming. (1957)
2. Wu, Lijun, et al. "A study of reinforcement learning for neural machine translation." arXiv preprint arXiv:1808.08866 (2018).
3. Jiang, Zhengyao, Dixing Xu, and Jinjun Liang. "A deep reinforcement learning framework for the financial portfolio management problem." arXiv preprint arXiv:1706.10059 (2017).
4. Andrychowicz, Marcin, et al. "Learning dexterous in-hand manipulation." arXiv preprint arXiv:1808.00177 (2018).
5. Watkins, Christopher JCH, and Peter Dayan. "Q-learning." Machine learning 8.3-4 (1992): 279-292.
6. Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).
7. Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971 (2015).