Parallel deep reinforcement learning with model-free and model-based methods

Eiji Uchibe. Dept. of Brain Robot Interface, Computational Neuroscience Labs., ATR International

Introduction

Background

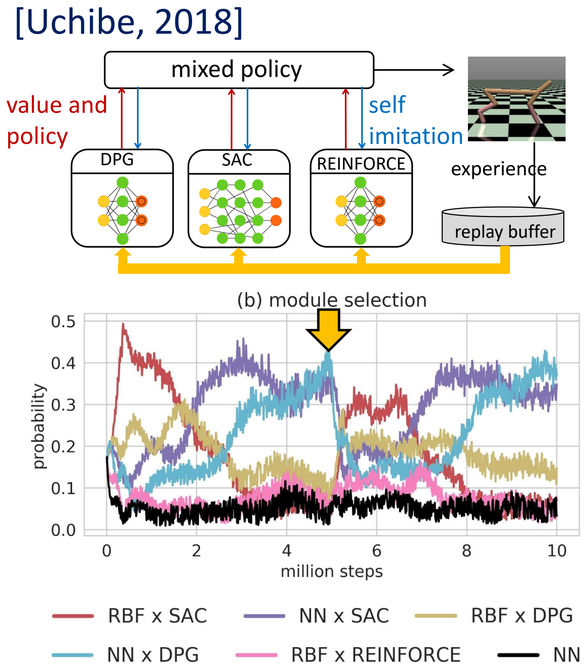
Reinforcement Learning (RL) can be categorized into model-based methods that exploit an (estimated) environmental model, and model-free methods that directly learn a policy through the interaction with the environment. To improve learning efficiency, we have proposed CRAIL (Uchibe, 2018), which dynamically selects a learning module from multiple heterogeneous model-free RL modules according to learning performance while multiple modules are trained simultaneously. However, CRAIL does not consider model-based methods.

Contribution of this study

This study extends CRAIL to deal with model-based and model-free methods and investigates whether dynamic switching between them contributes to the improvement of learning efficiency. The proposed method was evaluated by MuJoCo benchmark tasks. Experimental results show that a model-based method with a simple model was selected at the early stage of learning, and a model-based method with a complicated model was used at the later stage. Furthermore, model-free methods were selected when the network did not have sufficient capacity to represent the environmental dynamics.

Inverse RL using Density Ratio Estimation

Model-free Cooperative Reinforcement And Imitation Learning (CRAIL) [Uchibe, 2018]



- Multiple learning modules with different algorithms and networks
  - Mixed policy collect experiences
  - All the modules share the experiences
  - Trained with RL loss and self-imitation loss
- Mixing weights are determined by the value function
- The appropriate module is selected automatically
  - Early stage: RBF x SAC
  - Late stage: NN x DPG

Entropy-regularized reinforcement learning

Each module has a policy and a state value function

- Each module has a policy and a state value function
  - $\pi_i(u \mid x)$: prob. of the i-th module to select u at x, and $V_i(x)$: expected value of returns at x
  - γ: discount factor, and $r(x,u)$: immediate reward
- Behavior policy is given by the weighted sum of policies

$$\pi_b(u \mid x) = \sum_{i=1}^{M} \alpha(i \mid x)\pi_i(u \mid x)$$

- $\alpha(i \mid x)$: mixing weight is defined by

$$\alpha(i \mid x) \propto \exp\left(\beta V_i(x)\right)$$

the agent tends to select the module that has a large state value with high probability.

Prepared Model-free RL modules

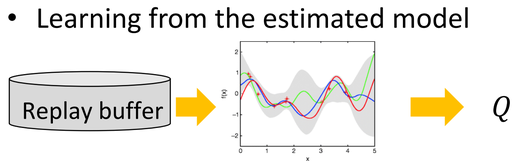
Soft Actor-Critic (SAC)

- Learning from the replay buffer

  - Uniform sampling from the replay buffer

Gaussian Process based SAC (GP-SAC)

- Learning from the estimated model



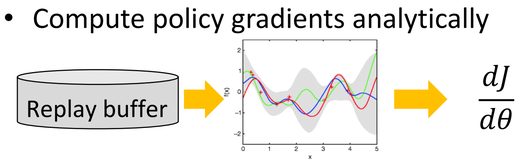
  - Dynamics and reward are estimated by Gaussian Process
  - DYNA-style (Sutton, 1990)

(Haarnoja et al., 2018)

Prepared Model-based RL modules

Probabilistic Inference for Learning Control (PILCO)

- Compute policy gradients analytically

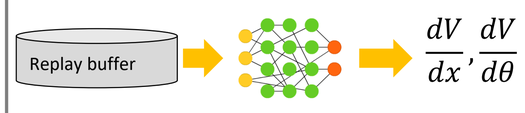


  - Multiple importance sampling for off-policy learning
  - Evaluate the long-term predictions by cascading 1-step prediction

(Deisenroth and Rasmussen, 2011; Deisenroth et al., 2015)

Stochastic Value Gradient (SVG)

- NN model for computing two gradients



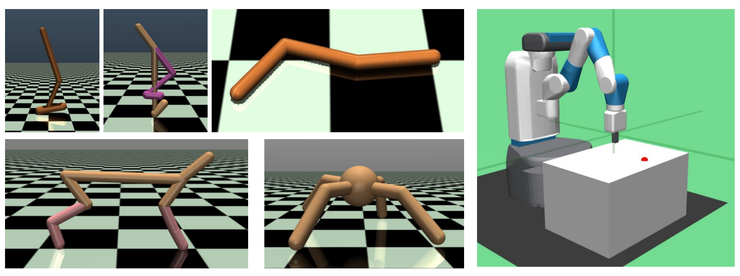
  - Multiple importance sampling for off-policy learning
  - Expand the Bellman recursion for 1-step

(Heess et al., 2015)

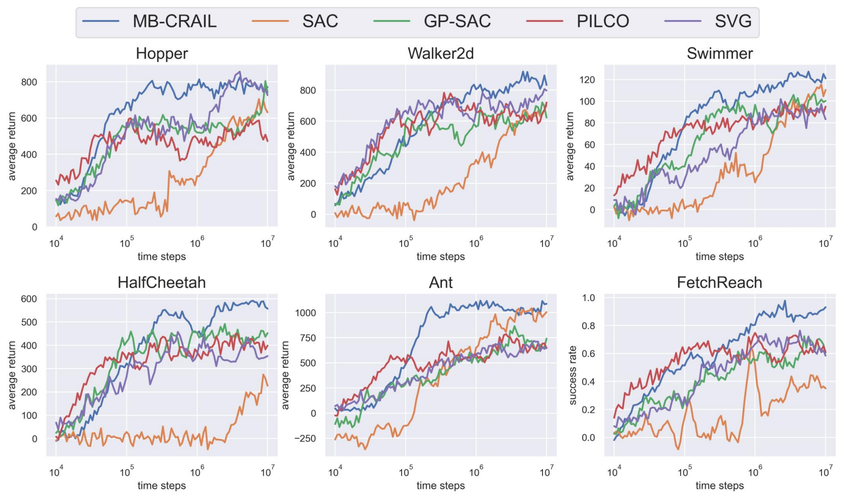
Experimental Results

Simulation: MuJoCo control tasks

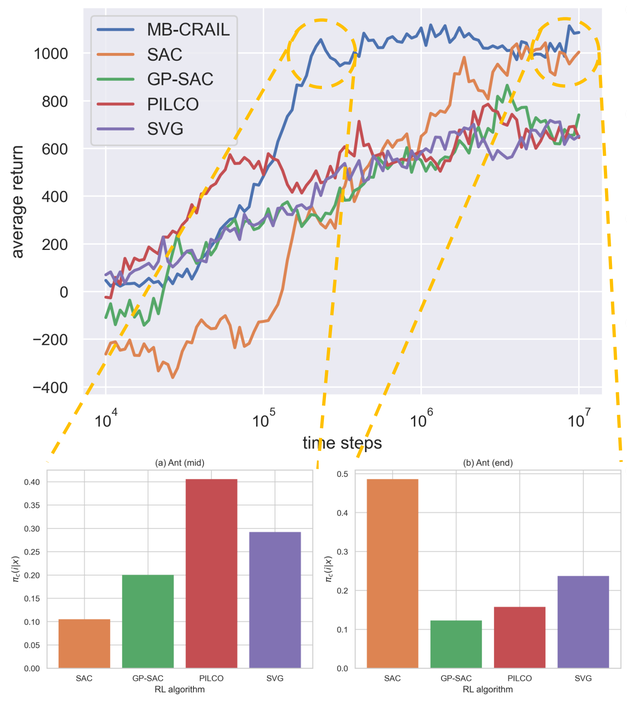
Benchmark tasks provided by OpenAI. For comparison, each module is trained independently



Training curves on all the control tasks



Comparison of learning curves



- The performance of MB-CRAIL increased after about 5×10^4 steps
- PILCO learned faster, but its performance saturated
- SAC learned more slowly but achieved the best performance
- The model-based methods (PILCO and SVG) have a large weight in the middle of learning
- SAC is often selected at the end of learning

Conclusion

Summary

We developed Model-based CRAIL, which integrates model-free and model-based RL modules and selects one module according to the learning progress. Our method outperformed the modules that were trained independently. We will consider the computing time in each step. For example, a simple module can make a decision faster than a complicated module. It is often appropriate in robot control.

Acknowledgments

This work was supported by Japan Society for the Promotion of Science KAKENHI Grant Numbers JP19H05001. This work was also supported by Innovative Science and Technology Initiative for Security Grant Number JPJ004596, ATLA, Japan and JST-Mirai Program Grant Number JPMJMI18B8, Japan.

References

1. Deisenroth, M., et al. (2015). Gaussian processes for data-efficient learning in robotics and control. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 408-423.
2. Haarnoja, T., et al. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proc. of the 35th International Conference on Machine Learning, pp. 1861-1870.
3. Heess, N. et al. (2015). Learning continuous control policies by stochastic value gradients. In Advances in Neural Information Processing Systems 28. 2015.
4. Uchibe, E. (2018). Cooperative and Competitive Reinforcement and Imitation Learning for a Mixture of Heterogeneous Learning Modules. Frontiers in Neurorobotics.
5. Sutton, R.S. (1990). Integrated Architecture for Learning, Planning, and Reacting Based on Approximating Dynamic Programming. In Proc. of the 7th International Conference on Machine Learning.