Robotic Manipulation with Reinforcement Learning, State Representation Learning, and Imitation Learning (Student Abstract)

Hanxiao Chen
Department of Automation, Harbin Institute of Technology
West DaZhi Street, Nangang District, Harbin
hanxiaochen@hit.edu.cn

Introduction

Humans possess the advanced ability to grab, hold, and manipulate objects with dexterous hands. What about robots? Can they interact with the surrounding world intelligently to achieve certain goals (e.g., grasping, object-relocation)? Actually, robotic manipulation is central to achieving the premise of robotics and represents immense potential to be widely applied in various scenarios like industries, hospitals, and homes. In this work, we aim to address multiple robotic manipulation tasks like grasping, button-pushing, and door-opening with reinforcement learning (RL), state representation learning (SRL), and imitation learning. For diverse missions, we self-built the PyBullet or MuJoCo simulated environments (**Fig. 1**) and independently explored three different learning-style methods to successfully solve such tasks: (1) Normal reinforcement learning methods; (2) Combined state representation learning (SRL) and RL approaches; (3) Imitation learning bootstrapped RL algorithms.

Kuka Grasping with Reinforcement Learning

As represented in **Fig. 1**, we established two Kuka-grasping PyBullet environments denoted as "KukaGymEnv-v0" and "KukaDiverseObjectEnv-v0". **Fig. 2** also provides basic information of two scenarios. Different from "KukaGymEnv-v0", "KukaDiverseObjectEnv-v0" grasps objects from multiple items instead of one single within the table slot and it considers high-level image data as input rather than relative x,y positions or Euler angles for robotic agents and targets. While training two grasping environments, we both utilized classical RL methods including Deep Q-Learning (DQN) (Mnih et al. 2013) and Proximal Policy Optimization (PPO) (Schulman et al. 2017). But separately, we applied the wrapped PPO & DQN algorithms in OpenAI baselines on the non-vision "KukaGymEnv-v0". For visual-based "KukaDiverseObjectEnv-v0", we self-implemented DQN & PPO RL algorithms with PyTorch and Tensorboard (**Fig. 3**), then recorded the **success percentage (SP)** with different evaluation episodes from 100 to 1000 and **average evaluation time** in each episode. Briefly speaking, our implemented DQN follows the training pipeline in **Fig. 4 (a)**. Its aim is training a policy that maximizes the discounted cumulative reward $R_{t_0} = \sum_{t=t_0}^{\infty} \gamma^{t-t_0} r_t$, where R_{t_0} denotes the *return*. The discount $\gamma \in [0, 1]$ ensures that the sum converges. In sum, pure RL algorithms (e.g., PPO & DQN) can effectively and efficiently solve basic grasping tasks.

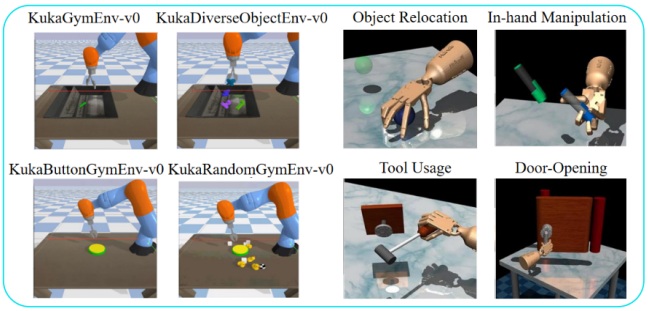


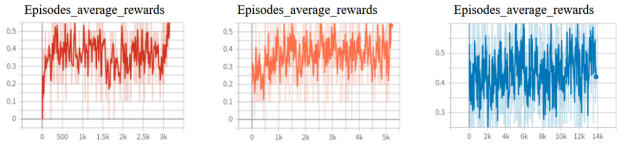
Figure 1: Left: Whole-arm-based 1-2 finger gripper manipulators. (Up: Grasping; Down: Button-pushing)
Right: Multi-fingered dexterous manipulators (four tasks).

Kuka Button-pushing with SRL & RL

Kuka Button-pushing PyBullet environments focus on goal-based robotic tasks and we built two simulated scenarios "KukaButtonGymEnv-v0" and "KukaRandomGymEnv-v0" with S-RL Toolbox (Raffin et al. 2018). To tackle the missions, we utilized a "self-supervised learning" style method which integrates state representation learning (SRL) with reinforcement learning (RL). SRL preliminarily extracts compact representations into the state space S (e.g., learn target and agent positions (x,y)) from raw observation data O), then we utilized the learned states $s_t \in S$ with RL methods to train a control policy π and output actions a_t to maximize rewards for specific tasks, thus addressing main challenges in RL: **sample inefficiency and instability**. To emphasize, efficiency is critical in robotics since experimenting an action is time-consuming, even defining interesting states for control tasks requires considerable manual engineering. Hence, we explored multiple effective SRL techniques (e.g., Auto-Encoder, VAE, Robotic Priors, etc.) to learn a compact, sufficient, disentangled, and generalizable state representation and benefit the following RL training significantly.

Environments	Observation	Action Mode	Reward Function	
			Initial Reward;	Details Description
KukaGymEnv-v0	Non-vision based (list)	Discrete (7)	-1000	Calculate closest points' number. Height of target object above the table.
KukaDiverseObjectEnv-v0	Vision Based	Discrete (7)	0	Getting close to the table and attempt to grasp any object.

Figure 2: Basic information for two Button-pushing tasks.



(a) DQN: 51.00 (b) PPO: 50.08 (c) PPO parallel: 50.13

Figure 3: Training Curves for "KukaDiverseObjectEnv-v0" (Algorithm [Left]: Average Reward Score [Right])

State Representation Learning

We explored 8 different SRL techniques belonging to 5 categories in S-RL Toolbox, even saved loss training curves and the learned state representation distribution visualizations[1].

- Reconstruction-based methods: Auto-Encoder (AE) & VAE (Variational AE) & DAE (Denoising AE).
- Forward & Inverse Model.
- Robotic Priors. (Raffin et al. 2018)
- Combining model with different loss functions.
- Splitting model that stacks representations learned with different objectives (e.g. Reward+Inverse+AE, etc).

SRL & RL Training and Evaluation

Following the SRL & RL pipeline **(Fig. 4 (b))** in S-RL Toolbox, after SRL training we combined the saved SRL model with various RL algorithms like DQN, PPO, and SAC (Haarnoja et al. 2018) to solve "KukaButtonGymEnv-v0" and "KukaRandomGymEnv-v0". While training, SRL model firstly transforms the input image observation into states, then chooses actions with RL methods. Also, we restricted training timesteps to be 50000, recorded the average reward scores for two button-pushing tasks and compared training performances of diverse SRL & RL approaches.

After training, we utilized two evaluation metrics for this self-supervised method: (1) Correlation between ground-truth states and learned representations, where we compute $GroundTruthCorrelation(GTC)_{mean}$ as follows,

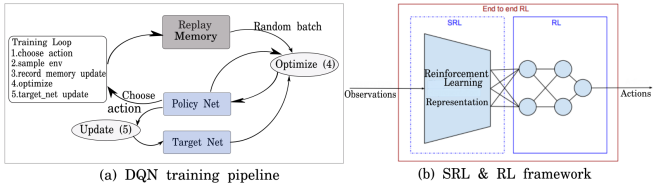
$$GTC_{mean} = E[GTC_{(i)}] = E[\max_j |\rho_{s,\tilde{s}}(i,j)|] \quad (1)$$

$$= E[\max_j |\frac{E[(s - \mu_s * (\tilde{s}) - \mu_{\tilde{s}})]}{\sigma_s * \sigma_{\tilde{s}}}|] \quad (2)$$

(2) Mean reward and evaluation time while evaluating two self-built new Button-pushing environments with different button scales for 30 epochs and recorded evaluation videos.

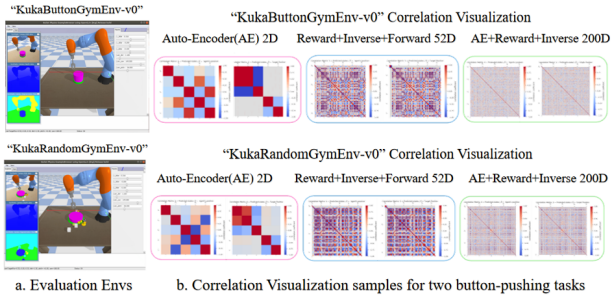
In a word, we find the SRL & RL pipeline performs well on goal-based manipulation tasks, even SRL can certainly extract crucial prior knowledge to benefit the RL phase. Also new SRL techniques can be explored on complex missions.

[1]Link: https://pan.baidu.com/s/1exP6yvJVzd1zpYOCco_g1Q (password: rxvc) for Cambridge AI Report & Experiments Code.



(a) DQN training pipeline (b) SRL & RL framework

Figure 4: Left: DQN train flow; Right: SRL & RL model.



a. Evaluation Envs b. Correlation Visualization samples for two button-pushing tasks

Figure 5: Left: Evaluation Envs. Right: Visualization.

Multi-fingered Dexterous Hand Manipulators

Towards the multi-fingered manipulation tasks, we successfully implemented the "Imitation Learning" style DAPG method (Rajeswaran et al. 2017) for **object relocation, pen rotation, tool usage, and door-opening**, even recorded task videos to observe directly. Using this approach, the agent becomes much flexible and efficient by imitating human hand demonstrations. For future work, we can apply advanced 3D computer vision techniques to facilitate the training process.

References

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290* .

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* .

Raffin, A.; Hill, A.; Traoré, R.; Lesort, T.; Díaz-Rodríguez, N.; and Filliat, D. 2018. S-RL toolbox: Environments, datasets and evaluation metrics for state representation learning. *arXiv preprint arXiv:1809.09369* .

Rajeswaran, A.; Kumar, V.; Gupta, A.; Schulman, J.; Todorov, E.; and Levine, S. 2017. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations. *CoRR* abs/1709.10087.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347.