Expert Data Augmentation in Imitation Learning* (Student Abstract)

Fuguang Han, Zongzhang Zhang

National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
hanfg@lamda.nju.edu.cn, zzzhang@nju.edu.cn

Abstract

Behavioral Cloning (BC) is a simple and effective imitation learning algorithm, which suffers from compounding error due to covariate shift. One solution is to use enough data for training. However, the amount of expert demonstrations available is usually limited. So we propose an effective method to augment expert demonstrations to alleviate the problem of compounding error in BC. It operates by estimating the similarity of states and filtering out transitions that can go back to the states similar to ones in expert demonstrations during the process of sampling. The data filtered out along with original expert demonstrations are used for training. We evaluate the performance of our method on several Atari tasks and continuous MuJoCo control tasks. Empirically, BC trained with the augmented data significantly outperform BC trained with the original expert demonstrations.

Introduction

Reinforcement Learning (RL) can train excellent agents with well-designed reward functions in a wide variety of domains. But for some applications, it is difficult to design a reward function. Imitation learning offers an approach to solve the problem by training an agent to imitate expert policy given expert demonstrations. Behavioral Cloning (BC) is a simple imitation learning algorithm, which is essentially a supervised learning method.

RL solves the sequential decision problem, so later states are related to earlier state-action sequences. However, in supervised learning, the training data is assumed to be Independent and Identically Distributed (IID), which contradicts the RL setting. Therefore, for BC, the test data is not independently sampled. The setting of BC does not satisfy the IID assumption of supervised learning, and the error that occurs at a certain step will affect the next sampling state. So, in the case of few expert demonstrations, the deviation of the trajectory will further expand the error, eventually leading to the problem of compounding error.

A straightforward solution to the compounding error of BC is to expand the expert demonstrations. DAgger (Ross, Gordon, and Bagnell 2011) is a data augmentation algorithm that allows an agent trained by BC to query an expert for the correct actions corresponding to states in a trajectory. However, in many cases experts cannot be queried and the amount of available expert demonstrations is limited, making it infeasible to obtain more training data from experts.

In this paper, we propose a new data augmentation method. This method does not need to interact with experts, but only requires the agent to interact with the environment to collect data, thereby mitigating the compounding error of BC and improving the performance of BC.

Method

The motivation of our method is that when deviating from the expert trajectory, the policy should take actions that can return to the expert trajectory. We consider obtaining transitions that can reach the expert demonstration states to augment the original expert data, so that the trajectory generated by the policy will return to the expert trajectory when it deviates from the expert trajectory to avoid compounding error.

Specifically, we use Random Network Distillation (RND) (Burda et al. 2018) to estimate the similarity between states. RND was proposed to measure the novelty of states and encourage policy to explore novel states. RND involves a randomly initialized fixed target network $f_\theta : S \to \mathbb{R}^K$ and a predictor network $f_{\hat{\theta}} : S \to \mathbb{R}^K$ that is trained on data collected by the agent, where S represents a state space and K represents the dimensionality of the output of the network. The objective of the predictor network is to minimize the expected Mean Squared Error (MSE) $\|f_\theta(s) - f_{\hat{\theta}}(s)\|_2^2$ by gradient descent with respect to the parameter $\hat{\theta}$. After training, the prediction error for states similar to the training states is expected to be lower.

We use this property of RND to estimate the similarity between states and ones in the expert demonstrations. Unlike training with the collected data continuously during training in RND, we only pre-train RND with states in the expert demonstrations. After pre-training, we take the top quantile (98% in our implementation) of the prediction error for states in the expert demonstrations as the threshold δ.

During policy sampling, we augment the original expert demonstrations with data that can reach the expert trajectories. For the sampled trajectory $\tau = \{s_0, a_0, s_1, a_1, ..., s_T\}$ and the state prediction loss $l_i = \|f_\theta(s_{i+1}) - f_{\hat{\theta}}(s_{i+1})\|_2^2$,

*Corresponding author: Zongzhang Zhang. This work is supported by the NSFC (No. 62276126).

	MsPacman	Breakout	SpaceInvaders	Qbert
Expert	1870	412	570	4225
BC	207.6±4.8	0.9±0.8	175±58.5	0±0
Ours	1061.2±177.2	17.0±9.1	218.6±100.7	770.5±350.4
	Hopper	Ant	Walker2d	HalfCheetah
Expert	3608	4104	6540	4391
BC	94.9±28.9	521.8±157.1	165.9±43.3	-174.3±464.1
Ours	1722.4±278.7	931.8±477.6	422.3±319.6	43.7±102.9

Table 1: Multi-algorithm performance comparison on 4 Atari tasks and 4 MuJoCo tasks after training 2M timesteps with only one expert trajectory.

the dataset we obtain is

$$\{(s_i, a_i) \mid l_i \leq \delta\}.$$

After sampling, we train the policy by BC's augmented data.

The sampling policy can be any policy. However, we find that in the samples obtained by the random policy, few transitions can return to the expert trajectories, i.e., the sample efficiency is low. To improve sample efficiency, we use the policy trained by BC as the sampling policy. Furthermore, we propose a reward function

$$r(s, a, s') = \begin{cases} 1 & \text{if} \quad \|f_\theta(s') - f_{\hat{\theta}}(s')\|_2^2 \leq \delta \\ -1 & \text{otherwise} \end{cases}$$

and train the policy initialized by BC with the above reward function to guide the policy close to the expert demonstrations, further improving the sample efficiency.

Experiments

Experimental Setup

We evaluate the proposed method on 4 discrete control tasks in Atari (Bellemare et al. 2013) and 4 continuous control tasks in MuJoCo (Todorov, Erez, and Tassa 2012). To better demonstrate the compounding error of BC with few expert demonstrations and the effectiveness of our method, we use only one expert trajectory for training. We compare the performance (i.e., return, a.k.a., cumulative discount rewards with a discount factor of 0.9) of the policies trained by BC on data augmented by our method with policies trained on original expert data. More details are show in the appendix[1].

Results

In the Atari environments, we test our method on four different tasks. The results in Table 1 show that the policy trained by BC on the data augmented by our method outperforms policies trained on original expert demonstrations across most environments. Despite good expert demonstrations, the policies that BC trained on a handful of expert demonstrations fail on Breakout, Pong, and Qbert. After data augmentation with our method, the policy is able to learn something from new data in Breakout and Qbert.

In the MuJoCo environments, we also conduct experiments on four representative tasks. As shown in Table 1, although there is still a large performance gap compared to expert policies, our method achieves performance improvements over BC in all four tasks.

[1]https://www.lamda.nju.edu.cn/hanfg/file/aaai23stuapd.pdf

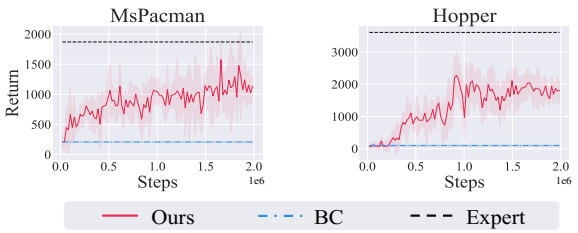

Figure 1: Learning curves on MsPacman and Hopper.

Due to space limitations, we only show the learning curves of the MsPacman and Hopper tasks in Figure 1. As training progresses and the amount of augmented data increases, the performance of policies trained with the augmented data continues to improve, demonstrating the effectiveness of the augmented data.

Discussion

We noticed that the preliminary results indicate that our method is likely unstable (i.e., high variance) on some tasks, e.g., SpaceInvaders and Qbert in Atari, Walker2d and HalfCheetah in MuJoCo. We think that this is due to the fact that the estimation of similarity by RND is not completely accurate. Inaccurate similarity estimation can introduce noisy data into the dataset, which will lead to the instability during the training of BC. A more accurate similarity estimation method can further improve the performance and stability of our method.

Conclusion

In this work, we address the problem of compounding error of BC by augmenting expert data. We propose a new data augmentation method that does not require interaction with experts, but only requires sampling data from the environment. Experiments in both Atari and MuJoCo environments demonstrate that augmented data can mitigate the compounding error of BC with few expert demonstrations.

References

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The Arcade learning environment: An evaluation platform for general agents. *JAIR*, 47: 253–279.

Burda, Y.; Edwards, H.; Storkey, A.; and Klimov, O. 2018. Exploration by random network distillation. In *ICLR*.

Ross, S.; Gordon, G.; and Bagnell, D. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 627–635.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. MuJoCo: A physics engine for model-based control. In *IROS*, 5026–5033.