A MINIMALIST ENSEMBLE METHOD FOR GENERALIZABLE OFFLINE DEEP REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep Reinforcement Learning (DRL) has achieved awesome performance in a variety of applications. However, most existing DRL methods require massive active interactions with the environments, which is not practical in real-world scenarios. Moreover, most current evaluation environments are exactly the same as the training environments, leading to the negligence of the generalization ability of the agent. To fulfill the potential of DRL, an ideal policy should have 1) the ability to learn from a previously collected dataset (i.e., offline DRL) and 2) the generalization ability for the unseen scenarios and objects in the testing environments. Given the expert demonstrations collected from the training environments, the goal is to enhance the performance of the model in both the training and testing environments without any more interaction. In this paper, we proposed a minimalist ensemble imitation learning-based method that trains a bundle of agents with simple modifications on network architecture and hyperparameter tuning and combines them as an ensemble model. To verify our method, we took part in the No Interaction Track of the SAPIEN Manipulation Skill (ManiSkill) Challenge and conducted extensive experiments on the ManiSkill Benchmark. The challenge rank and experimental results well demonstrated the effectiveness of our method.

1 INTRODUCTION

Deep Reinforcement Learning (DRL) has been widely studied and applied to a variety of applications including robotics (Akkaya et al., 2019) and autonomous driving (Filos et al., 2020; Zhao et al., 2022), and performed remarkably in ideal environments such as Atari (Bellemare et al., 2013) and MuJoCo (Todorov et al., 2012). While many DRL methods learn by online interacting with the environments, interactions with real-world environments are costly and sometimes unsafe. For example, in autonomous driving, deploying DRL methods on vehicles in road testing is expensive and may lead to traffic accidents. Offline DRL (Fujimoto et al., 2019; Kumar et al., 2020), in which no additional interactions with the environments are required after data collection, provides a promising way for making DRL methods applicable in practice. One way to solve the offline DRL problem is following the paradigm of Imitation Learning (Chen et al., 2020; Liu et al., 2021) to mimic the actions directly. However, offline DRL methods do not consider the generalization ability in the testing environments, especially for real-world complex, dynamic, and open-ended applications with unseen objects and new scenarios.

In this paper, we proposed a minimalist ensemble IL-based method and demonstrated its efficacy in the *No Interaction Track* of the ManiSkill Challenge (Mu et al., 2021). In this track, only a set of fixed expert demonstrations can be used for learning generalizable manipulation skills. Based on a base learner provided by the challenge, we modified the network architecture and tuned hyperparameters to make single base learner robust, and we further leveraged ensemble method with feature diversity improvement to enhance the final model performance. We participated in the No Interaction Track of the ManiSkill Challenge and conducted extensive local experiments on the ManiSkill Benchmark, and the results from the ManiSkill Challenge leaderboard and the additional experiments on the benchmark demonstrated the superiority of our method.

2 METHODOLOGY

2.1 OVERVIEW

We represent the environment using a Markov decision process (MDP) $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, reward function $\mathcal{R}$, state transition function $\mathcal{P}$ and discount factor γ. And we follow the setting of offline DRL that only trains the agent with a fixed dataset D without additional interaction. In ManiSkill Challenge tasks, the dataset $D = \{(s_i, a_i, r_i, s'_i), i = 1, \cdots, N\}$ is optimal (i.e., all data are expert demonstrations) and consists of N tuples where s is the current state, a is the continuous action vector, r is the reward and s' is the next state. Different from the conventional DRL problem, our goal is to maximize the success rate of a task instead of the excepted reward.

Given the expert demonstrations D, we efficiently train M independent base learners $\{\pi_j(\cdot)\}_{j=1}^{M}$ in parallel. Each single base learner takes the input s and predicts the action $\hat{a}_j$. To aggregate all base learners into an ensemble model, we use the Bagging algorithm (Breiman, 1996) and take the average action $\bar{a}$ as the final decision. In the remainder of this section, we describe the baseline method, how to make a robust base learner and how to aggregate multi-agents respectively in detail.

2.2 THE BASE LEARNER

We borrow the baseline PointNet+Transformer from the ManiSkill Challenge (Mu et al., 2021). Given the expert action a and input $s = (c, o, m)$ including the agent state c, the point cloud observation o and segmentation mask m, the PointNet+Transformer $\pi(\cdot)$ firstly uses a PointNet (Qi et al., 2017) to extract the categorical feature for each segmentation category (i.e., C PointNets in total and C is the segmentation category number). All categorical features are then passed into a Transformer (Vaswani et al., 2017) and fused into a representative global feature $g \in \mathbb{R}^{b \times l}$ (b is the batch size and l is the dimension of the feature). And a final Multilayer Perceptron (MLP) generates the action $\hat{a}$ using the global feature. We train the PointNet+Transformer $\pi(\cdot)$ using the Behavior Cloning (BC) with the objective $\boldsymbol{L}_{\mathrm{bc}} = \|\hat{a} - a\|_2$.

2.3 TOWARDS A ROBUST SINGLE LEARNER

Larger Batch Size. To avoid making out-of-distribution (OOD) actions in both the training and testing environments, we hope to learn a conservative model fitting on the high-quality successful demonstrations and reduce the compounding error (Ross & Bagnell, 2010). As discussed in McCandlish et al. (2018), a larger batch size can approximate more accurately true gradients while a smaller batch size often generates gradients with higher variances. Thus we choose to use a large batch size to calculate accurate gradients. In our implementation, we used a large batch size $b = 1024$ rather than a small value $b = 128$ in the baseline PointNet + Transformer.

Dropout Regularization. To handle the overfitting problem (Codevilla et al., 2019) which is common in the BC method, we apply the dropout regularization technique (Srivastava et al., 2014) to our network. During the training stage, for the input features from the previous layer and the corresponding neurons in the dropout layer, the dropout technique randomly cuts off the connection with a probability p. As claimed in Srivastava et al. (2014), the dropout regularization can reduce generalization error by a large margin, which is exactly what we focus on. In our implementation, we added two dropout layers before the last two fully connected layers in the final MLP, where the probability p was simply set to a mild value of 0.15.

2.4 MINIMALIST ENSEMBLE MODELING

Besides making a single base learner more robust, we also observe that the ensemble methods are very useful to reduce the generalization error. Following the idea of Bagging (Breiman, 1996), we train M ($M = 20$ in our implementation) base learners $\pi_j(\cdot)$ independently and aggregate all base learners by taking the average of the predicted actions:

$$\bar{a} = \frac{1}{M} \sum_{j=1}^{M} \hat{a}_j = \frac{1}{M} \sum_{j=1}^{M} \pi_j(s). \tag{1}$$

As stated in the theory analysis in Breiman (1996), if the base learners are not very different from each other, the aggregation operation may not help a lot. Conversely, with truly independent base learners, the aggregation operation can reduce variance, prevent the overfitting problem and enhance generalization ability to new scenarios. Since base learners are generated by learning from the same training set D, it is difficult to obtain truly independent base learners. Here we use two methods to make each base learner more diverse and obtain good generalization performance by aggregation.

Bootstrapping Sampling Technique. To increase the data diversity, we use bootstrapping sampling (Tibshirani & Efron, 1993) to build different sub-datasets for each base learner (i.e., given a dataset D with N samples, we randomly resample N samples from D with replacement).

Feature Diversity Improvement. For each base learner $\pi_j(\cdot)$, we randomly generate a real symmetric matrix $A_j \in \mathbb{R}^{l\times l}$ by sampling from a uniform distribution $U(0,1)$. Then we produce an orthogonal basis $B_j \in \mathbb{R}^{l\times l}$ consisting of l eigenvectors of A_j. By multiplying the orthogonal basis B_j, the global feature $g_j \in \mathbb{R}^{b\times l}$ from the Transformer module (Vaswani et al., 2017) can be projected to a new feature space with a specified transformation

$$f_j = g_j B_j, \tag{2}$$

and then sent to the final MLP to generate final action $\hat{a}_j$. Note that since each base learner has a different orthogonal basis B_j, they can learn different patterns even if the differences on the dataset are not significant, allowing aggregation operation to reduce variance and improve generalization ability.

3 Experiments

3.1 Experiments Setup

SAPIEN Manipulation Skill (ManiSkill) Challenge. To verify the effectiveness of our method, we conducted extensive experiments on the SAPIEN Manipulation Skill (ManiSkill) Benchmark and participated in the ManiSkill Challenge (Mu et al., 2021). Aiming to develop more generalizable manipulation skills, the ManiSkill Challenge is built on the ManiSkill Benchmark and consists of 4 object-centric manipulation tasks with 162 different objects in total. The 4 tasks are 1) *OpenCabinetDrawer* (Drawer), 2) *OpenCabinetDoor* (Door), 3) *PushChair* (Chair), and 4) *MoveBucket* (Bucket). For each task, the evaluation metric is the *success rate* rather than the accumulated rewards.

No Interaction Track. Specifically, we participated in the No Interaction Track of the ManiSkill Challenge that only allows to train the agent with the given demonstrations (i.e., unknown behavior policy and no more interaction with the environments). The demonstration dataset is collected from the training environments and consists of 7500, 12600, 7800, and 8700 successful trajectories for OpenCabinetDrawer, OpenCabinetDoor, PushChair and MoveBucket respectively, and about 1.5M frames in total. The testing environments contain 10 unseen objects for each task. The Challenge evaluates the model on both the training and the testing environments. The final score of each task is the mean success rate from both two environments.

Table 1: Success rates on the No Interaction Track of the ManiSkill Challenge. Final scores are the mean success rates of all four tasks in both the training and testing environments.

Team	**Final Score**	**Drawer**		**Door**		**Chair**		**Bucket**	
		Train	Test	Train	Test	Train	Test	Train	Test
Silver-Bullet-3D	0.5740	0.932	0.556	0.896	0.208	0.468	0.328	0.716	0.488
bigfish	0.3435	0.740	0.192	0.664	0.124	0.256	0.152	0.360	0.260
MI	0.3320	0.788	0.120	0.700	0.124	0.320	0.180	0.268	0.156
SieRra11799	0.1895	0.456	0.144	0.200	0.044	0.212	0.120	0.196	0.144
ic	0.1880	0.484	0.228	0.280	0.072	0.156	0.116	0.108	0.060
Zhihao	0.1835	0.416	0.120	0.272	0.068	0.216	0.096	0.176	0.104
Ours	0.4070	0.840	0.184	0.764	0.160	0.400	0.252	0.320	0.336

Table 2: Ablation studies on batch size "B" and dropout regularization "D" in terms of the success rate in the training environments.

#	B	D	Drawer	Door	Chair	Bucket
1	128	✗	0.370	0.300	0.180	0.150
2	128	✓	0.370	0.300	0.190	0.170
3	256	✗	0.410	0.310	0.180	0.165
4	256	✓	0.505	0.370	0.215	0.175
5	512	✗	0.545	0.415	0.250	0.180
6	512	✓	0.575	0.445	0.255	0.195
7	1024	✗	0.565	0.485	0.250	0.225
8	1024	✓	0.615	0.605	0.280	0.235

Table 3: Ablation studies on base learner number "M" and feature diversity improvement "F" in terms of the success rate in the training environments.

#	M	F	Drawer	Door	Chair	Bucket
1	–	✗	0.615	0.605	0.280	0.235
2	3	✗	0.715	0.685	0.325	0.255
3	3	✓	0.750	0.715	0.345	0.275
4	5	✗	0.785	0.740	0.345	0.255
5	5	✓	0.805	0.760	0.355	0.285
6	10	✓	0.825	0.755	0.340	0.300
7	15	✓	0.825	0.780	0.345	0.300
8	20	✓	**0.830**	**0.790**	**0.375**	**0.310**

3.2 Results on ManiSkill Challenge

Table 1 shows the detailed results on the No Interaction Track of the ManiSkill Challenge in terms of mean success rates. Except for the team Silver-Bullet-3D, our results outperformed other teams by a large margin in most environments. For instance, our method achieved a success rate of 0.4000 on the training environments for task PushChair, which is up to 14.4% ahead of the third team bigfish. It is worth noting that all teams suffered dramatic performance drops on all 4 tasks when transferring from the training environments to the testing environments, especially for tasks OpenCabinetDrawer and OpenCabinetDoor. We therefore inferred that the variations of unseen objects can lead to dramatic changes in decision making. For tasks PushChair and MoveBucket, we noticed that the performance gap between the training and testing environments are relatively small, which indicates the variations of the objects in these two tasks are not large enough to cause more failures.

3.3 Ablation Study

To better understand the effectiveness of each component and technique used in our method, we conducted detailed ablation studies on ManiSkill Benchmark. Due to fact that we can not access the testing environments of each task on the local benchmark, we evaluated our models on the training environments of 4 tasks over 200 times and took the final average success rates as the results. Note that because we evaluated locally and used different seeds, the results are different from the challenge results on the training environments in Table 1.

Table 2 shows the results for different batch sizes and dropout regularization used or not. For the second and third columns, "B" is the abbreviation for batch size and "D" is the abbreviation for dropout regularization. From the settings 1, 3, 5 and 7 in Table 2, we can observe that there are consistent improvements as the batch size increases. For example, the mean success rate on the task OpenCabinetDrawer increased from 0.370 to 0.565 as the batch size increased from 128 to 1024. The dropout regularization can also enhance the performance according to the comparisons between settings 2, 4, 6, 8 and settings 1, 3, 5, 7 respectively. Table 3 shows the results for different numbers "M" of the base learners and the feature diversity improvement "F" in the ensemble model. The ensemble models consistently achieved higher success rates as the number of the base learners increased on all 4 tasks. For instance, a single base learner can only obtain a success rate of 0.615 on the task OpenCabinetDrawer, while the ensemble model of 20 base learners achieved 0.830 leading to a huge improvement of 0.215.

4 Conclusion

In this paper, based on the baseline PointNet+Transformer from the ManiSkill Challenge, we propose a minimalist ensemble method to enhance the generalization ability of the model. Towards a robust single agent, we simply add dropout regularization and increase the batch size to handle the overfitting problem and reduce generalization error. To further enhance the generalization ability, we use bagging algorithm to aggregate a bundle of models. The experimental results on the ManiSkill Challenge and Benchmark demonstrated the superiority of our method.

REFERENCES

Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 2013.

Leo Breiman. Bagging predictors. *Machine learning*, 1996.

Xinyue Chen, Zijian Zhou, Zheng Wang, Che Wang, Yanqiu Wu, and Keith Ross. Bail: Best-action imitation learning for batch deep reinforcement learning. *Advances in Neural Information Processing Systems*, 2020.

Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

Angelos Filos, Panagiotis Tigkas, Rowan McAllister, Nicholas Rhinehart, Sergey Levine, and Yarin Gal. Can autonomous vehicles identify, recover from, and adapt to distribution shifts? In *International Conference on Machine Learning*, 2020.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2020.

Minghuan Liu, Hanye Zhao, Zhengyu Yang, Jian Shen, Weinan Zhang, Li Zhao, and Tie-Yan Liu. Curriculum offline imitating learning. *Advances in Neural Information Processing Systems*, 2021.

Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Cathera Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 2014.

Robert J Tibshirani and Bradley Efron. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 1993.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, 2012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 2017.

Yinuo Zhao, Kun Wu, Zhiyuan Xu, Zhengping Che, Qi Lu, Jian Tang, and Chi Harold Liu. Cadre: A cascade deep reinforcement learning framework for vision-based autonomous urban driving. *arXiv preprint arXiv:2202.08557*, 2022.