Robotic Manipulation Datasets for Offline Compositional Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract: Offline reinforcement learning (RL) is a promising direction that would allow RL agents to be pre-trained from large datasets avoiding recurrence of expensive data collection. To advance the study of the field, it is crucial to generate large-scale datasets. Compositional RL is particularly appealing for generating such large data sets, since 1) it permits creating many tasks from few components, and 2) the task structure might enable trained agents to solve new tasks by combining relevant learned components. In this paper, we provide four offline RL datasets for simulated robotic manipulation created using the 256 tasks from CompoSuite [1]. Each dataset is collected from an agent with a different degree of performance, and consists of 256 million transitions. We provide training and evaluation settings for assessing an agent's ability to learn compositional task policies. Our benchmarking experiments on each of the provided training settings show that current offline RL methods can learn the training tasks to some extent, but are unable to extract their compositional structure to generalize to unseen tasks.

Keywords: Datasets for Robot Learning, Offline Reinforcement Learning, Compositional Generalization

1 Introduction

Much of deep learning's success at solving a wide variety of problems can be attributed to the comtemporary increase in freely available data [2, 3, 4]. Similar to other areas, we would expect robot learning techniques to leverage these vast amounts of data in order to solve multitudes of real-world control problems. However, the field of robotics has yet to fully take advantage of the capabilities that neural networks offer, as generating datasets for robotics is both expensive and time consuming, even in simulation. Large-scale data collection is imperative to maximizing the utility of deep learning.

Much of the research in deep learning for robotics is devoted to reinforcement learning (RL) [5, 6, 7, 8]. Classical RL methods require the agent to collect data over time, which is ostensibly problematic for neural networks that require an abundance of data. Offline RL approaches [9, 10]—those which train an agent solely on a fixed, previously collected dataset—may address this issue, as their goal is to learn policies comparable to those obtained by classical RL without requiring new data collection for each new training trial. Once an agent has been pre-trained on offline data, the learned model can be fine-tuned to new tasks in the real world with little additional data [11]. Despite these advantages, the offline setting comes with its own challenges. First, offline RL requires large datasets [12] labeled with reward functions, which cannot be as easily crowdsourced as image labels. Second, offline RL algorithms do not have the ability to explore new states at training time, and must generalize to specific states without having seen them during training [9]. This issue is enhanced by the fact that standard offline RL evaluations are limited to consider only single-task problems [10, 13, 14, 15, 16].

To address these issues, we consider compositional agents and environments. A compositional *agent* can decompose complex problems into sub-components, solve them, and re-use this acquired

Submitted to the Pre-training for Robot Learning Workshop at the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

knowledge throughout the state space, mitigating the state generalization issues of offline RL. Further, compositional RL agents move beyond single-task paradigms, showing sample efficiency improvements in multi-task and lifelong RL via generalizable components and behaviors that can be combined to solve new tasks [17, 18, 19, 20, 21, 22, 23, 24, 25]. On the other hand, compositional *environments* offer re-usability of reward functions to create a plethora of training behaviors [1].

To facilitate the combined study of offline RL and compositionality, we provide several datasets collected using CompoSuite [1]—a simulated robotic manipulation benchmark designed for studying online compositional RL—and experiment scenarios designed to answer questions related to the interplay of the two fields. Specifically, we contribute the following:

1. Four datasets of varying performance with trajectories from each of the 256 CompoSuite tasks,
2. Training-test split lists for evaluation to ensure comparability and reproducibility of results, and
3. Evaluation demonstrating both the utility of our datasets for offline compositional RL research, and the poor ability of current offline RL techniques to leverage compositional structures.

2 Preliminaries

Offline RL We formulate offline RL as a Markov decision process $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, R, \mathcal{P}, \gamma\}$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, R is the reward function, $\mathcal{P}$ are the transition probabilities, and γ is a discount factor. The goal is to find an optimal policy $\pi^*(a, s)$ that maximizes the expected return $\mathbb{E}_\pi \sum_{t=0}^{T} \gamma^t R(s_t, a_t)$. However, the agent does not have access to $\mathcal{M}$ directly for online interaction, but instead has access to a dataset $\mathcal{D} = \{(s_i, a_i, s'_i, r_i)\}_{i=1}^{N}$ of transition tuples containing a state, action, resulting state, and reward, collected from $\mathcal{M}$ using a behavioral policy π_β.

CompoSuite benchmark for compositional RL CompoSuite [1] is a recent simulated robotics benchmark for RL that builds on top of robosuite [26] and uses the MuJoCo simulator [27]. It is designed to study functional decomposition in RL algorithms. Every CompoSuite task is created by composing four different axes: a `robot` manipulator that moves an `object` to achieve an `objective` while avoiding an `obstacle`. Further, each axis consists of four elements yielding a total of 256 combinations of tasks. Note that, for a given `objective`, the reward function is constant across other variations, making it easy to scale the number of tasks without the need to label every single task individually. CompoSuite uses a combination of `robot`, `object`, `obstacle`, and `objective` information as well as a multi-hot task indicator as its observation space. The action space consists of target positions for each joint of the robotic manipulators as well as a gripper action. Each robot has seven degrees of freedom, and so the observation and action spaces are consistent across all tasks.

3 Datasets for Offline Compositional RL

We elaborate on the specific training setting we consider and structure of the datasets we provide, and detail several reproducible experiment configurations for analyzing offline compositional RL.

3.1 Data Collection

To collect our datasets, we trained several agents using Proximal Policy Optimization [6] on the CompoSuite benchmark and used them as our behavioral policies π_β. Similar to commonly used existing datasets [12], we collect one million transitions for every task in CompoSuite, totaling 256 million transitions per dataset. To achieve high success rate on all tasks, we use the compositional neural network architecture from Mendez et al. [25]. We provide the following four datasets.

Expert dataset: Transitions from an agent that was trained to achieve 90% success on every task.
Medium Dataset: Transitions from an agent that was trained to achieve 30% success on every task.
Random dataset: Transitions from an untrained agent (randomly initialized), achieving 0% success.
Medium-replay-subsampled dataset: Transitions that were stored during the training of an agent up to 30% success. For tasks that required more than one million steps to achieve 30% success, the one million transitions were uniformly sampled.

Note that the process of collecting all these datasets requires training a single compositional agent over all tasks, appropriately storing trained policies at various levels of performance for each task.

In the real world, expert data is rarely available. Instead, datasets have varying levels of performance, which the expert, medium, and random datasets represent. This allows for the construction of training sets containing data from trajectories of varying success rates (discussed more in Section 3.2), which both represents realistic data collection settings and lets researchers adapt the difficulty of offline RL tasks. In addition, the medium-replay-subsampled dataset contains data that an online RL agent would see during training, exhibiting various levels of proficiency at solving the task. Even though, intuitively, this should be sufficient to learn good policies via offline RL, existing work has shown that current approaches struggle in this setting [12].

3.2 Training Task Lists and Experimental Setup

We consider multiple training settings to analyze an agent's ability to functionally decompose a task and re-use its acquired knowledge. These settings are represented by different samplings of tasks across the various datasets. To facilitate comparability of results, we provide various lists splitting the tasks into training and zero-shot tasks, analogous to train-test splits in supervised learning problems[1]. Any of the sampling techniques can be used with any of the datasets from section 3.1. We differentiate between **uniform**, **compositional** and **restricted** task sampling as follows:

Uniform sampling This sampling corresponds to the standard multi-task setting which is used to evaluate training performance as well as zero-shot generalization to unseen tasks. We consider a train-test split of tasks similar to data splits in supervised learning with 224 training and 32 test tasks. The agent sees the training tasks but must perform zero-shot generalization to the test tasks.

Compositional sampling A slightly more realistic setting does not assume access to data of equal performance for every task. Instead, we split the data into 76 training tasks from the expert dataset, 148 additional training tasks from one of the other (non-expert) datasets, and 32 test tasks. This setting acts as a proxy for measuring compositionality in a learning approach; a model that can successfully decompose its knowledge about successful executions from the expert tasks should be able to combine this knowledge with the noisier information from remaining tasks to compositionally generalize to those other tasks. Note that if the training tasks for compositional sampling come from expert-random data, we obtain a dataset with similar average success as the medium dataset but with a significantly different success distribution across tasks.

Restricted sampling This setting is similar to the restricted sampling from the original CompoSuite paper [1], which corresponds to a harder setting to evaluate an agent's ability to extract compositional information. This is achieved by restricting the training dataset to be smaller and to contain only a single task for a specific element. As an example, if the selected element is the IIWA arm, then the training set contains exactly one task which uses an IIWA arm. The training set contains a total of 56 tasks while the test set contains all the remaining 63 tasks that contain the IIWA arm.

4 Experimental Results

We experimentally evaluate several of the suggested settings from section 3.2. We run two baselines for each setting, Behavioral Cloning (BC) and Implicit Q-Learning (IQL), using the d3rlpy implementations [28]. BC imitates the behavioral policy π_β by learning to predict the correct action given a state from the dataset, and we expect it to perform well over data from high performing agents. However, for non-expert data one would hope to achieve better performance using an offline RL baseline, such as IQL, which can extrapolate information across states to better generalize beyond the training data distribution. After training each agent, we evaluate it online and report average cumulative return and success rate over one trajectory per task. We include both training and zero-shot test task performance.

[1]The set lists are available with the dataset release, and all experiments in Section 4 are carried out over the provided lists to facilitate reproducibility.

Table 1: Test and training return and success rates achieved by Behavioral Cloning (BC) and Implicit Q-Learning (IQL) on each dataset.

	Dataset: Expert Sampling: Unif.		Dataset: Medium Sampling: Unif.		Dataset: Replay Sampling: Unif.		Dataset: Random Sampling: Comp.	
	BC	IQL	BC	IQL	BC	IQL	BC	IQL
Train Return	339.54	276.66	185.05	178.73	104.28	150.89	119.31	94.48
Test Return	293.15	279.17	165.12	192.64	98.87	176.11	31.79	81.57
Train Success	0.87	0.69	0.23	0.33	0.00	0.15	0.30	0.22
Test Success	0.71	0.68	0.21	0.34	0.00	0.25	0.06	0.16

Table 2: Test and training return and success rates achieved by Behavioral Cloning (BC) and Implicit Q-Learning (IQL) on the expert dataset in the restricted sampling setting.

	Dataset: Expert Elem.: IIWA		Dataset: Expert Elem.: PickPlace		Dataset: Expert Elem.: Hollowbox		Dataset: Expert Elem.: ObjectWall	
	BC	IQL	BC	IQL	BC	IQL	BC	IQL
Train Return	368.78	285.52	360.60	268.71	354.60	297.82	357.19	265.08
Test Return	25.73	33.49	31.96	57.88	25.04	74.27	8.12	15.95
Train Success	0.96	0.73	0.93	0.66	0.93	0.79	0.89	0.66
Test Success	0.03	0.03	0.03	0.08	0.03	0.14	0.00	0.00

Training on Uniformly Sampled Datasets To evaluate learnability and characterize different levels of challenge among our scenarios, we train the BC and the IQL agents on the expert, medium, and medium-replay-subsampled datasets. We use uniform sampling of 224 training and 32 test tasks as discussed in section 3.2. The results are shown in the leftmost three columns of Table 1. The results in Table 1 (column 1) verify that both agents can achieve high performance on the expert datasets. However, IQL strictly outperforms BC in the settings where fewer successful trajectories are available, and generalizes better to test settings. BC is not able to gain any success when trained on replay data while IQL is still able to achieve a decent amount of success from this source.

Training on Expert-Random Composition Next, we demonstrate the importance of accounting for compositional structure by training agents using compositional sampling over an expert-random dataset combination. As shown in the right-most column of Table 1, both agents are able to extract some information from the expert datasets and achieve similar training performance. On the zero-shot tasks, IQL outperforms BC but achieves low overall success rates. For comparison, both agents perform significantly better on the medium dataset (column 2), indicating that they learn something closer to a mean behavior policy rather than extracting the compositional structure of the tasks.

Training on Restricted Sampling To further test current approaches' propensity for compositional learning, we compare the BC and IQL agents on four different restricted settings, and show the results in Table 2. In each experiment, we fix an element from a different axis to show the generality of the setting. These agents were trained on expert data that only contains a single task with the specified element present while the test tasks all contain this element. Across all settings, both agents achieve decent success on the training tasks but fail to generalize to the zero-shot settings. When compared to the expert dataset with uniform sampling setting in Table 1, it seems that having a large amount of data for every single task element is required to generalize to unseen tasks. This is further evidence that current agents are incapable of extracting and leveraging the compositional structure from the environment data.

5 Conclusion

In this paper we have introduced several novel datasets to study the intersection of offline and compositional RL. Our results indicate that current offline RL approaches do not capture the compositional structure of our tasks well, and that further research is required in this area. An interesting direction for future work is the explicit modeling of modularity in neural networks, or the discovery of modular structure, required to obtain networks that are capable of zero-shot generalization. We hope that, by releasing the datasets and the experimental settings described in this work, we can further research in offline and compositional RL for robotics applications.

References

[1] J. A. Mendez, M. Hussing, M. Gummadi, and E. Eaton. Composuite: A compositional reinforcement learning benchmark. In *1st Conference on Lifelong Learning Agents*, 2022.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[3] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[5] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations (ICLR-16)*, 2016.

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

[7] S. Fujimoto, H. van Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning (ICML-18)*, pages 1587–1596, 2018.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML-18)*, pages 1861–1870, 2018.

[9] S. Lange, T. Gabel, and M. Riedmiller. Batch Reinforcement Learning. In M. Wiering and M. van Otterlo, editors, *Reinforcement Learning: State of the Art*. Springer, in press, 2011.

[10] S. Fujimoto, D. Meger, and D. Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062, 2019.

[11] Y. Chebotar, K. Hausman, Y. Lu, T. Xiao, D. Kalashnikov, J. Varley, A. Irpan, B. Eysenbach, R. Julian, C. Finn, and S. Levine. Actionable models: Unsupervised offline reinforcement learning of robotic skills. *arXiv preprint arXiv:2104.07749*, 2021.

[12] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

[13] A. Kumar, J. Fu, M. Soh, G. Tucker, and S. Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[14] A. Nair, M. Dalal, A. Gupta, and S. Levine. Accelerating online reinforcement learning with offline datasets. *CoRR*, abs/2006.09359, 2020.

[15] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1179–1191. Curran Associates, Inc., 2020.

[16] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

[17] E. Brunskill and L. Li. PAC-inspired option discovery in lifelong reinforcement learning. In *Proceedings of the 31st International Conference on Machine Learning, ICML-14*, pages 316–324, 2014.

[18] C. Tessler, S. Givony, T. Zahavy, D. Mankowitz, and S. Mannor. A deep hierarchical approach to lifelong learning in Minecraft. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI-17*, 2017.

[19] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA-17)*, pages 2169–2176, 2017.

[20] A. Barreto, D. Borsa, J. Quan, T. Schaul, D. Silver, M. Hessel, D. Mankowitz, A. Zidek, and R. Munos. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *Proceedings of the 35th International Conference on Machine Learning, ICML-18*, pages 501–510, 2018.

[21] T. Haarnoja, V. Pong, A. Zhou, M. Dalal, P. Abbeel, and S. Levine. Composable deep reinforcement learning for robotic manipulation. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA-18)*, pages 6244–6251, 2018.

[22] B. Van Niekerk, S. James, A. Earle, and B. Rosman. Composing value functions in reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML-19)*, pages 6401–6409, 2019.

[23] G. Nangue Tasse, S. James, and B. Rosman. A Boolean task algebra for reinforcement learning. In *Advances in Neural Information Processing Systems 33, NeurIPS-20*, pages 9497–9507, 2020.

[24] A. Goyal, A. Lamb, J. Hoffmann, S. Sodhani, S. Levine, Y. Bengio, and B. Schölkopf. Recurrent independent mechanisms. In *9th International Conference on Learning Representations (ICLR-21)*, 2021.

[25] J. A. Mendez, H. van Seijen, and E. Eaton. Modular lifelong reinforcement learning via neural composition. In *10th International Conference on Learning Representations (ICLR-22)*, 2022.

[26] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

[27] E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-12)*, pages 5026–5033, 2012.

[28] M. I. Takuma Seno. d3rlpy: An offline deep reinforcement library. In *NeurIPS 2021 Offline Reinforcement Learning Workshop*, December 2021.