Robotic Manipulation Datasets for Offline Compositional Reinforcement Learning

Marcel Hussing†,
University of Pennsylvania
mhussing@seas.upenn.edu

Jorge Mendez-Mendez†,
Massachusetts Institute of Technology
jmendezm@mit.edu

Anisha Singrodia
University of Pennsylvania
singroa@seas.upenn.edu

Cassandra Kent
University of Pennsylvania
dekent@seas.upenn.edu

Eric Eaton
University of Pennsylvania
eeaton@seas.upenn.edu

Abstract

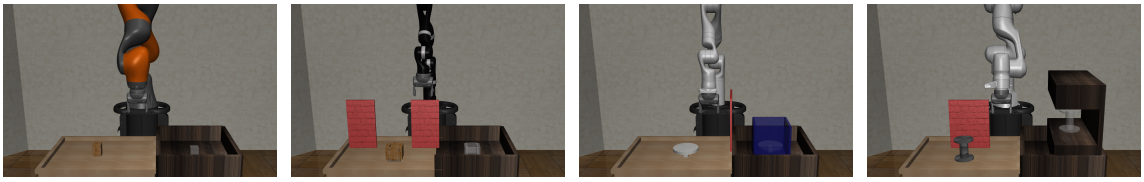
Offline reinforcement learning (RL) is a promising direction that allows RL agents to pre-train on large datasets, avoiding the recurrence of expensive data collection. To advance the field, it is crucial to generate large-scale datasets. Compositional RL is particularly appealing for generating such large datasets, since 1) it permits creating many tasks from few components, 2) the task structure may enable trained agents to solve new tasks by combining relevant learned components, and 3) the compositional dimensions provide a notion of task relatedness. This paper provides four offline RL datasets for simulated robotic manipulation created using the 256 tasks from CompoSuite (Mendez et al., 2022a). 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 show that current offline RL methods can learn the training tasks to some extent and that compositional methods outperform non-compositional methods. Yet current methods are unable to extract the compositional structure to generalize to unseen tasks, highlighting a need for future research in offline compositional RL.

1 Introduction

Large-scale data has generated much of the success of deep learning. We would expect robot learning techniques to similarly leverage vast amounts of data to solve multitudes of real-world problems. However, generating datasets for robotics is expensive and time consuming, even in simulation. Large-scale data collection is imperative to maximizing the utility of deep learning for robotics.

Much of the efforts in deep learning research for robotics have been devoted to reinforcement learning (RL). However, online RL methods require the agent to collect data over time, and therefore each new online RL experiment requires a new round of large-scale data collection. Offline RL approaches (Lange et al., 2011; Fujimoto et al., 2019) train on a fixed (previously collected) dataset, potentially permitting to learn high-quality policies without the need to obtain additional data. Once an agent has been pre-trained on offline data, its model can be fine-tuned to unseen tasks in the real world with little additional data (Chebotar et al., 2021). Despite these advantages, the offline setting comes with its own challenges. First, offline RL requires large datasets (Fu et al., 2020) labeled with reward functions. Image labels, the computer vision counterpart, can easily be crowdsourced, which has facilitated the creation of large vision datasets; crowdsourcing is not readily applicable to RL

† The two first authors contributed equally to this work.



⟨IIWA, box, no_obstacle, pick_and_place⟩ ⟨Jaco, hollow_box, object_door, push⟩ ⟨Gen3, plate, goal_wall, trash_can⟩ ⟨Panda, dumbbell, object_wall, shelf⟩

Figure 1: Examples of four CompoSuite tasks, showing each task's initial state. Each task is composed of one element from each of four compositional axes, involving a *robot* (`IIWA`, `Jaco`, `Gen3`, or `Panda`) manipulating an *object* (`box`, `hollow_box`, `plate`, or `dumbbell`) while avoiding an *obstacle* (`no_obstacle`, `object_door`, `goal_wall`, `object_wall`) to achieve a specific *objective* (`pick_and_place`, `push`, `trash_can`, or `shelf`). Images from Mendez et al. (2022a).

rewards. Second, offline RL agents are not allowed to explore new states during training, and must generalize to unseen states at evaluation time. Notably, unlike in supervised settings, RL agents do not make a single prediction on a new state, but instead the actions they choose lead them to traverse the state space, moving them increasingly far away from the original training distribution. This leads to a unique form of distribution shift. These issues are exacerbated by the fact that standard offline RL evaluations are limited to single-task problems, further restricting the scale of current datasets.

To address these issues, we consider compositional agents and environments. A *compositional agent* decomposes complex problems into components, re-composes the components to solve the problems, and re-uses the acquired knowledge throughout the state space, improving state generalization. Further, compositional RL agents exhibit sample efficiency improvements in multi-task and lifelong RL via generalizable components and behaviors that can be combined to solve new tasks (Mendez et al., 2022b). On the other hand, *compositional environments* offer re-usability of reward functions to induce a plethora of training behaviors (Mendez et al., 2022a). They also enable the creation of numerous tasks with a clear notion of task relatedness along the different compositional dimensions, which is useful for selective transfer and for analyzing performance.

To facilitate the combined study of offline RL and compositionality, we provide multiple datasets collected using CompoSuite (Mendez et al., 2022a)—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]:

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. An evaluation demonstrating the utility of our datasets for offline compositional RL research, and the (relatively) poor ability of current offline RL techniques to leverage compositional structures. These results validate both the learnability and difficulty of the datasets using common learning techniques, and demonstrate the need for improved algorithms for offline compositional RL.

2 Preliminaries

Offline RL Standard *online* RL solves a Markov decision process $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, R, \mathcal{P}, \gamma, \mu\}$ via direct interaction with $\mathcal{M}$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, R is the reward function, $\mathcal{P}$ are the transition probabilities, γ is a discount factor, and μ is the distribution over starting states. In *offline* RL, the agent does not have access to $\mathcal{M}$ for training, but instead receives a dataset $\mathcal{D} = \{(s_i, a_i, s'_i, r_i)\}_{i=1}^N$ of transition tuples, where s'_i and r_i are the state and reward obtained by executing action a_i in state s_i using an unknown behavioral policy $a_i \sim \pi_\beta(s_i)$ in $\mathcal{M}$. Consequently, $\mathcal{D}$ is a sample from the distribution $d^{\pi_\beta}(s)\pi_\beta(s, a)$, where $d^{\pi_\beta}(s)$ is the marginal state distribution

[1]Datasets are available at datadryad.org/stash/dataset/doi:10.5061/dryad.9cnp5hqps; train-test split lists and code for the experiments can be found at github.com/lifelong-ml/offline-compositional-rl-datasets.

induced by π_β. The goal in offline RL is to find an optimal policy π^* which maximizes the expected cumulative return $J_\pi = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$ in $\mathcal{M}$ *without interacting with* $\mathcal{M}$.

Functional composition in RL Unlike traditional temporal sequencing of skills, functionally compositional RL considers the composition of functional transformations of the state by a chain of computations that results in a chosen action (Mendez et al., 2022b). These functional modules are akin to functions in programming, which consume as inputs the outputs of other functions and produce inputs to yet other functions. At each timestep, multiple functions are involved in computing the action to take from the current state; compare this to temporal composition (e.g., in the Options framework (Sutton et al., 1999b; Bacon et al., 2017)) in which only one module (an option) is active at each time. A set of RL tasks related via functional composition can be described formally as a compositional problem graph whose paths represent the transformations required to solve each task.

CompoSuite benchmark for compositional RL CompoSuite (Mendez et al., 2022a) is a recent simulated robotics benchmark for RL built on top of robosuite (Zhu et al., 2020), designed to study functional composition in RL. Every CompoSuite task is created by composing elements of four different axes: a `robot` manipulator that moves an `object` to achieve an `objective` while avoiding an `obstacle`. Each axis consists of four elements, for a total of 256 tasks (Figure 1). For a given `objective`, the reward function is constant across other axes, making it easy to scale the number of tasks without the need to craft labeling functions for each individual task.

3 Datasets and experimental setup 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. Figure 2 provides an overview of the dataset collection process and its use in training and evaluation.

3.1 Data shape and spaces

Following Fu et al. (2020), we collect one million transitions for each task, totaling 256 million transitions per dataset. Every transition contains: observation, action, reward, next observation, timeout indicator, and terminal indicator. Observations are vectors of size 93 containing proprioceptive robot information such as joint and finger positions and velocities, absolute and relative object, obstacle, and goal positions, and a multi-hot task indicator to identify the elements of the current task. The action space is eight-dimensional; the first seven dimensions correspond to target joint angles of the 7-DoF robots for joint position control, and the last dimension is a gripper action. Tasks use dense rewards to ensure that every transition has a non-zero reward. Rewards are specific to each objective and encourage the learning of a policy in stages (e.g., the rewards for pick-and-place encourage first reaching the object, then grasping the object, lifting the object, and finally approaching the target). The reward for being in a goal-satisfying state is 1 for all tasks. Episodes time out after 500 timesteps, and `push` tasks additionally terminate if the grasped object is lifted from the table.

3.2 Data collection

To collect our datasets, we trained agents using standard online RL methods to obtain the behavioral policies π_β. For three of the datasets, we trained a single agent via proximal policy optimization (PPO; Schulman et al., 2017) across all tasks, storing trained policies at various levels of performance for each task. PPO can be parallelized and offers a fast algorithm to obtain these policies in terms of wall-clock time. To ensure that the agent would achieve high success rate on all tasks, we used the compositional neural network architecture of Mendez et al. (2022b). For the fourth dataset, we trained a separate agent on each task via soft actor critic (SAC; Haarnoja et al., 2018) to simulate a warmstart scenario in which some data from an RL run is available. Here, we use SAC due to its sample efficiency. Concretely, we provide the following four datasets.

- **Expert dataset:** Transitions from an agent trained to achieve 90% success on every task.

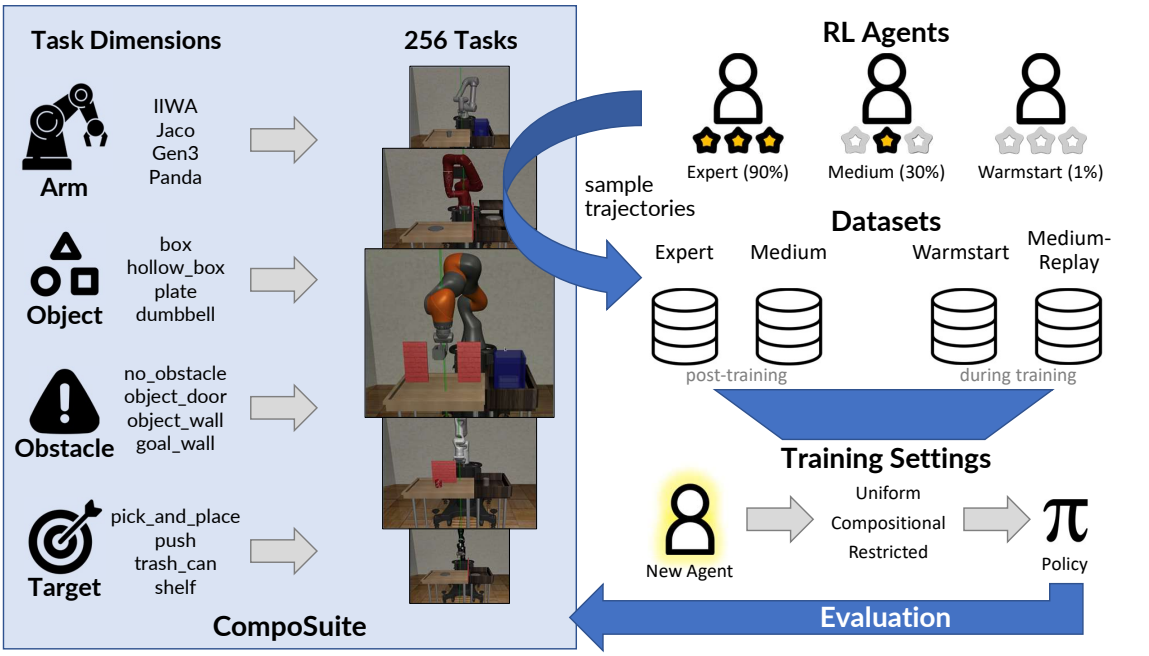


Figure 2: An overview of our dataset creation and training process. Manipulation tasks vary along four compositional dimensions, as taken from CompoSuite. Trajectories are sampled from pre-trained PPO agents, forming four different datasets of varying difficulties (Section 3.2). Three different training settings (Section 3.3) provide different views into these data for training, and evaluation of the learned policies is performed on the CompoSuite simulator using mujoco.

- **Medium dataset:** Transitions from an agent trained to achieve 30% success on every task.
- **Warmstart dataset:** Transitions that were stored during the training of one SAC agent per task for 1 million steps. The average success rate across all trajectories is in the order of 1%.
- **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 obtained by uniformly sampling trajectories.

The different datasets were collected to serve various research purposes. In the real world, expert data is rarely available. Instead, datasets have varying levels of performance, represented by the expert, medium, and warmstart datasets. This allows for the construction of training sets containing data from trajectories of varying success rates and task progress (as discussed in Section 3.3), which both offers realistic data collection settings and lets researchers experiment with diverse levels of difficulty of offline RL tasks. We chose to replace the random agents commonly used in the literature (e.g., Fu et al., 2020) with warmstart agents trained for a short period to ensure that the data contained diverse states covering the various stages of each task—because the tasks are long-horizon, a random policy would only cover a small portion of the state space, since it would never even grasp the object. Warmstart agents simulate a setting where a researcher can do *some* online-RL but is limited in time.

In addition, the medium-replay dataset contains data that an online RL agent would see during training, exhibiting varied levels of proficiency at solving the task. Intuitively, this should be sufficient to learn good policies via offline RL, yet current approaches struggle in this setting (Fujimoto et al., 2019; Fu et al., 2020). Note that since our agents require substantially more than one million samples to converge in many of the tasks, the medium-replay dataset subsamples trajectories observed during training. Since some trajectories from `push` tasks might be truncated (due to the termination condition), we artificially truncate the last subsampled trajectory and place a time-out at the final position. This might, in rare cases, lead to one incorrect trajectory if the datasets are used for

explicitly finite horizon RL experimentation. However, the truncation ensures a consistent dataset size of one million across tasks and compatibility with other standard code implementations.

3.3 Training task lists and multi-task training

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 lists splitting the tasks into training and zero-shot tasks, analogous to train-test splits in supervised learning problems. Any of the following sampling techniques can be used with any of the datasets from section 3.2.

Uniform sampling This standard multi-task setting is used to evaluate training performance and zero-shot generalization. Akin to data splits in supervised learning, the agent trains offline on 224 tasks and is evaluated for generalization to 32 online test tasks without any data for those tasks.

Compositional sampling A more realistic setting should not assume access to data of equal performance for every task. To simulate this scenario, 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 zero-shot tasks. The 76 expert tasks contain all 16 CompoSuite components in equal proportions. 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 into the relevant components should be able to combine this knowledge with the noisier information from remaining tasks to compositionally generalize to those and the unseen tasks. Note that, if non-expert tasks are drawn from the warmstart dataset, the combined dataset has similar average success as the medium dataset but with a substantially different success distribution across tasks.

Restricted sampling Similar to the equivalent setting in online CompoSuite, restricted sampling constitutes 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. For example, if the selected element is the `IIWA` arm, then the training set contains exactly one task which uses an `IIWA` arm and 55 tasks that use other arms. The training set contains a total of 56 tasks while the zero-shot set contains the remaining 63 tasks that contain the `IIWA` arm.

4 Experiments

4.1 Implementation and experiment details

We evaluated various settings from Section 3.3 over three different random seeds controlling the choice of train-test split list, network parameter initialization, and data sampling within each algorithm—because our comparisons are over large numbers of tasks, the results have low variance and so three seeds are sufficient to show general trends. We consider four baselines[2]: Behavioral Cloning (BC), Compositional BC (CP-BC), Implicit Q-Learning (IQL; Kostrikov et al., 2022) and Compositional IQL (CP-IQL), using the d3rlpy implementations (Takuma Seno, 2021) (hyperparameters in Appendix B). 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 given high-performance data. IQL is an offline RL baseline designed to generalize beyond the training data distribution and is expected to achieve better performance given non-expert data. These two baselines use standard multilayer perceptrons (MLP) to encode policies and value functions. The CP versions of the algorithms instead employ a compositional neural network architecture as described by Mendez et al. (2022b;a) for all networks (see Appendix D for details). The compositional network architecture consists of hierarchically stacked modules that correspond to the various elements in CompoSuite. Each module operates in two stages: the pre-processing stage is an MLP that takes as input the module-specific state (e.g., object modules take only the object state as input); the post-processing stage is a second MLP that

[2]We additionally ran the evaluations using conservative Q-learning (Kumar et al., 2020), but found that it attained 0% success on all settings (including the training tasks with Expert data), so we omit it from our results.

Table 1: Test and training return and success rates achieved by Behavioral Cloning (BC), Implicit Q-Learning (IQL), Compositional BC (CP-BC), and Compositional IQL (CP-IQL) on the various datasets using 224 training tasks and 32 test tasks. All agents achieve decent success and generalize when given access to expert data (sub-table 1). IQL agents strictly outperform BC on the medium and replay datasets (sub-tables 2 and 3). When having to extract compositional information from expert data, the compositional policy yields some benefits over feed-forward networks but is still far from optimal (sub-table 4). Success rates are shaded from green (100%) to yellow (50%) to red (0%). All values represent mean ± standard deviation.

	Dataset: Expert; Sampling: Uniform			
	Train Return	Test Return	Train Success	Test Success
Behavioral Cloning	339.05 ±4.26	297.29 ±7.18	0.87 ±0.01	0.73 ±0.02
Implicit Q-Learning	264.97 ±2.16	279.67 ±33.92	0.65 ±0.01	0.68 ±0.07
CP Behavioral Cloning	380.42 ±2.44	354.61 ±11.23	0.97 ±0.01	0.88 ±0.05
CP Implicit Q-Learning	351.62 ±1.98	345.19 ±10.16	0.90 ±0.01	0.86 ±0.03

	Dataset: Medium; Sampling: Uniform			
	Train Return	Test Return	Train Success	Test Success
Behavioral Cloning	190.84 ±8.40	162.93 ±6.11	0.24 ±0.03	0.21 ±0.06
Implicit Q-Learning	176.84 ±11.46	150.66 ±22.31	0.30 ±0.02	0.24 ±0.02
CP Behavioral Cloning	211.10 ±3.27	190.04 ±21.61	0.28 ±0.02	0.22 ±0.08
CP Implicit Q-Learning	223.44 ±4.23	196.98 ±30.63	0.47 ±0.02	0.38 ±0.1

	Dataset: Medium-Replay; Sampling: Uniform			
	Train Return	Test Return	Train Success	Test Success
Behavioral Cloning	102.65 ±4.63	95.04 ±12.00	0.00 ±0.01	0.00 ±0.00
Implicit Q-Learning	138.37 ±1.68	142.35 ±23.51	0.10 ±0.02	0.09 ±0.05
CP Behavioral Cloning	95.31 ±1.04	91.60 ±5.02	0.00 ±0.00	0.00 ±0.00
CP Implicit Q-Learning	102.66 ±10.11	99.08 ±20.67	0.09 ±0.03	0.09 ±0.03

	Dataset: Warmstart (+ Expert); Sampling: Compositional			
	Train Return	Test Return	Train Success	Test Success
Behavioral Cloning	132.54 ±3.45	51.04 ±18.34	0.29 ±0.01	0.07 ±0.06
Implicit Q-Learning	98.64 ±3.13	57.82 ±11.90	0.18 ±0.01	0.07 ±0.03
CP Behavioral Cloning	153.36 ±7.94	89.86 ±10.51	0.35 ±0.01	0.17 ±0.01
CP Implicit Q-Learning	127.75 ±5.97	87.31 ±22.72	0.30 ±0.01	0.18 ±0.10

takes as input the concatenation of the output of the previous module (in the hierarchy) and the output of the pre-processing stage. Intuitively, encoding the tasks' inherent compositional structures into the policies should facilitate transfer to unseen tasks.

We trained each agent simultaneously on a subset of the 256 tasks, and evaluated it on held-out tasks per the task lists from Section 3.3. All BC and IQL agents were trained for 50,000 and 300,000 update steps respectively using a batch size of #training tasks $\times$ 256. Trained agents are evaluated online using CompoSuite (Mendez et al., 2022a), with the metrics from Appendix C. We report mean cumulative return and success rate over one evaluation trajectory per task for train and test tasks.

Table 2: Test and training return and success rates achieved by Behavioral Cloning (BC), Implicit Q-Learning (IQL) and Compositional BC (CP-BC) and Compositional IQL (CP-IQL) on the expert datasets in the restricted sampling setting. All agents achieve decent training success. However, transfer to unseen tasks remains a challenge, especially for non-compositional agents. All values represent mean ± standard deviation.

	Dataset: Expert; Fixed Element: `IIWA`			
	Train Return	Test Return	Train Success	Test Success
Behavioral Cloning	371.57 ±10.27	18.54 ±5.54	0.95 ±0.03	0.02 ±0.01
Implicit Q-Learning	273.71 ±17.64	34.85 ±5.25	0.70 ±0.05	0.03 ±0.02
CP Behavioral Cloning	386.30 ± 4.79	77.02 ±40.17	0.98 ±0.02	0.11 ±0.11
CP Implicit Q-Learning	361.49 ± 8.82	127.49 ±25.98	0.95 ±0.02	0.18 ±0.04

	Dataset: Expert; Fixed Element: `pick_and_place`			
	Train Return	Test Return	Train Success	Test Success
Behavioral Cloning	346.84 ±19.94	41.76 ± 9.27	0.88 ±0.06	0.06 ±0.03
Implicit Q-Learning	262.24 ±13.35	49.82 ±11.46	0.64 ±0.05	0.07 ±0.01
CP Behavioral Cloning	382.70 ±5.03	81.83 ±34.19	0.97 ±0.01	0.13 ±0.07
CP Implicit Q-Learning	368.92 ±7.38	75.18 ±21.34	0.93 ±0.02	0.16 ±0.08

	Dataset: Expert; Fixed Element: `hollow_box`			
	Train Return	Test Return	Train Success	Test Success
Behavioral Cloning	363.83 ±13.15	45.38 ±25.12	0.92 ±0.03	0.08 ±0.08
Implicit Q-Learning	278.81 ±41.53	63.48 ±13.99	0.69 ±0.12	0.11 ±0.04
CP Behavioral Cloning	383.11 ±0.62	103.27 ± 24.06	0.97 ±0.02	0.25 ±0.05
CP Implicit Q-Learning	377.45 ±0.69	60.50 ± 4.32	0.97 ±0.02	0.14 ±0.01

	Dataset: Expert; Fixed Element: `object_wall`			
	Train Return	Test Return	Train Success	Test Success
Behavioral Cloning	349.58 ±14.81	12.69 ± 3.31	0.88 ±0.04	0.02 ±0.03
Implicit Q-Learning	267.90 ±20.08	19.56 ± 2.39	0.64 ±0.07	0.02 ±0.02
CP Behavioral Cloning	393.10 ± 3.64	41.42 ±9.64	0.99 ±0.01	0.10 ±0.01
CP Implicit Q-Learning	377.43 ± 1.19	23.39 ±11.84	0.96 ±0.01	0.04 ±0.03

4.2 Experimental results

Training on uniformly sampled datasets To evaluate learnability and characterize different levels of challenge among our scenarios, we trained the BC, IQL, CP-BC and CP-IQL agents on the expert, medium, and medium-replay datasets. We used uniform sampling of 224 training and 32 zero-shot test tasks as discussed in section 3.3. The results in the first three sub-tables of Table 1 verify that the four baselines can achieve high performance on the expert datasets. IQL baselines strictly outperform BC baselines in the settings where fewer successful trajectories are available (medium and replay), and generalize better to unseen configurations. When trained on replay data, BC attains no success, while IQL achieves some success. Further, while the compositional architecture boosts performance on the medium dataset, its generalization capabilities remain far from optimal.

Training on Expert-Warmstart composition We demonstrate the importance of compositionality by evaluating agents using compositional sampling combining expert and warmstart datasets. As shown in the fourth sub-table of Table 1, all four agents are able to extract some information from the expert datasets. The compositional architecture leads to an increase in training performance,

which translates to better zero-shot performance. This indicates that the learned modules discover how to solve pieces of tasks on the training set, which are then re-used on the zero-shot tasks. BC and standard IQL agents perform substantially better on the medium dataset (sub-table 2) even though the medium dataset and the warmstart-expert dataset with compositional sampling contain a similar amount of successful trajectories. This suggests that they learn something akin to an "average" policy, instead of extracting the compositional structure and specializing it to each task.

Training on restricted sampling As one additional test of compositionality, we compared agents on four restricted settings, each restricting one element from a distinct axis (Table 2). Agents were trained on expert data that only contains one task with the restricted element, while all zero-shot tasks contain the restricted element. The four agents perform well on the training tasks, but fail to generalize to the zero-shot tasks. Together with the uniform sampling results in Table 1, these results demonstrate that the baselines require a large amount of data from varied task combinations for every single task element to generalize to unseen tasks. This is further evidence that current methods are incapable of extracting and leveraging the compositional structure of the environment. However, the compositional architecture shows signs of zero-shot generalization across all tasks, encouraging the study of compositional methods for robotic transfer learning problems.

5 Related work

Compositional RL Composition has been used in RL for decades in attempts to improve sampling efficiency (Mendez & Eaton, 2023). Intuitively, learning components of a problem may be easier than learning the full problem, and learned components can be combined with others to solve new tasks. The majority of such works in RL focused on learning temporally extended actions (skills or options) that can be sequenced to construct a compositional policy (Sutton et al., 1999a; Konidaris & Barto, 2009; Bacon et al., 2017; Tessler et al., 2017). Other common forms of composition include logical composition (Nangue Tasse et al., 2020; Barreto et al., 2018; Van Niekerk et al., 2019), state abstraction learning (Dayan & Hinton, 1993; Dietterich, 2000; Vezhnevets et al., 2017), and object-based RL (Li et al., 2020; Mu et al., 2020). We consider the *functional composition* perspective, described in Section 2, where components correspond to successive functional transformations of the state to generate actions (Devin et al., 2017; Goyal et al., 2021; Mendez et al., 2022b).

Offline RL Offline RL research has grown steeply in recent years (Lange et al., 2011; Levine et al., 2020). Most methods operate in the single-task setting (Fujimoto et al., 2019; Kumar et al., 2019b; Nair et al., 2020; Kumar et al., 2020; Ma et al., 2021; Kostrikov et al., 2022), failing to leverage related datasets to train more powerful and general policies. Works on multi-task offline RL have been successful, but their scale remains limited (Siegel et al., 2020; Yu et al., 2021). Recently, large-scale datasets have enabled generalization of multi-task offline Q-learning (Kumar et al., 2023). Offline meta-RL, which pre-trains models on varied tasks to rapidly adapt to new tasks, shares motivation with our work. While most such methods consider offline fine-tuning (Mitchell et al., 2021; Dorfman et al., 2021; Li et al., 2021), others instead adapt the policy to new tasks online via exploration (Pong et al., 2022; Zhao et al., 2022). These prior works share information across tasks in an unstructured way, without considering common elements. Our explicitly compositional datasets promote the study of algorithms that reason about compositional relations across tasks.

Datasets and benchmarks Large image datasets have driven many advancements of deep learning (Deng et al., 2009; Krizhevsky & Hinton, 2009). Conversely, (online) RL has been restricted to the use of simulation benchmarks for assessing new methods' performance, primarily focused on single-task training (Bellemare et al., 2013; Brockman et al., 2016; Vinyals et al., 2017). More recent work has developed online benchmarks for multi-task, lifelong, and meta-RL (James et al., 2020; Cobbe et al., 2020; Chevalier-Boisvert et al., 2019; Henderson et al., 2017; Ahmed et al., 2021; Yu et al., 2020; Tomilin et al., 2023). With the advent of offline RL, it has become desirable to leverage large datasets for standardized RL training. Several such datasets have been proposed to benchmark offline RL (Mandlekar et al., 2018; Dasari et al., 2020; Fu et al., 2020; Gulcehre et al., 2020; Qin et al., 2022; Zhou et al., 2022; Liu et al., 2023) and offline multi-agent RL (Qu et al.,

2023) approaches. However, none of these study the interplay of deep RL and compositionality. Our dataset construction follows D4RL (Fu et al., 2020) to collect data from existing online benchmarks (in particular, CompoSuite; Mendez et al., 2022a). Offline RL datasets are important for moving towards benchmarks on robot hardware (Collins et al., 2020; Gürtler et al., 2023), and so our work relates to the development of low-cost (Yang et al., 2019; Ahn et al., 2019) or remote-access (Pickem et al., 2017; Paull et al., 2017; Kumar et al., 2019a) robotic platforms for data collection.

6 Limitations

While our datasets are collected from simulation of commercially available robotic manipulators, it is well known that most learning algorithms suffer from a simulation-to-real (sim2real) performance gap. In consequence, work that seeks to apply learned policies or modules directly to physical robots would need to develop mechanisms to bridge this gap. Notably, compositionality might enable one such technique: training a new module for the physical robot in combination with pre-trained modules for the remaining task components. In addition, our focus is on releasing large-scale data sets to enable pre-training of powerful, generalizable offline RL models. As such, the computing requirements for running experiments on the full data sets would be prohibitive for organizations without access to powerful computers (see Appendix A for details of the computational setup used in our experiments). That being said, several of the proposed experimental settings require substantially less computation, making them more accessible for such organizations. Beside these limitations, our datasets inherit some of the limitations of the original CompoSuite benchmark. Namely, our datasets use symbolic (and not image) observations, the observation space reveals the compositional structure of the tasks explicitly, and the tasks contain a fixed number of four compositional axes (Mendez et al., 2022a).

7 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. Other directions include the study of offline to online transfer in a multi-task setting as well as a continual learning setting. An interesting open scientific question would be whether increasing the variety of compositional tasks has significant benefits over training on single tasks. We hope that, by releasing the datasets and the experimental settings described in this work, we can further research efforts in offline and compositional RL for robotics applications.

Acknowledgments

JMM's research was funded by an MIT-IBM Distinguished Postdoctoral Fellowship. MH, AS, CK, and EE's research was partially supported by DARPA Lifelong Learning Machines grant FA8750-18-2-0117, DARPA SAIL-ON contract HR001120C0040, DARPA ShELL agreement HR00112190133, Army Research Office MURI grant W911NF20-1-0080, and DARPA Triage Challenge award HR001123S0011. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, the Army, or the US government.

References

Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Manuel Wuthrich, Yoshua Bengio, Bernhard Schölkopf, and Stefan Bauer. CausalWorld: A robotic manipulation benchmark for causal structure and transfer learning. In *9th International Conference on Learning Representations, ICLR-21*, 2021.

Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. ROBEL: RObotics BEnchmarks for Learning with low-cost robots. In *Conference on Robot Learning (CoRL)*, 2019.

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, pp. 1726–1734, 2017.

Andre Barreto, Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Zidek, and Remi 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*, pp. 501–510, 2018.

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, JAIR*, 47:253–279, 2013.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016.

Yevgen Chebotar, Karol Hausman, Yao Lu, Ted Xiao, Dmitry Kalashnikov, Jake Varley, Alex Irpan, Benjamin Eysenbach, Ryan Julian, Chelsea Finn, and Sergey Levine. Actionable models: Unsupervised offline reinforcement learning of robotic skills. *arXiv preprint arXiv:2104.07749*, 2021.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. BabyAI: First steps towards grounded language learning with a human in the loop. In *7th International Conference on Learning Representations, ICLR-19*, 2019.

Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML-20*, pp. 2048–2056, 2020.

Jack Collins, Jessie McVicar, David Wedlock, Ross Brown, David Howard, and Jürgen Leitner. Benchmarking simulated robotic manipulation through a real world dataset. *IEEE Robotics and Automation Letters*, 5(1):250–257, 2020. doi: 10.1109/LRA.2019.2953663.

Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura (eds.), *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 885–897. PMLR, 30 Oct–01 Nov 2020.

Peter Dayan and Geoffrey E Hinton. Feudal reinforcement learning. In *Advances in Neural Information Processing Systems 6, NIPS-93*, pp. 271–278, 1993.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey 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)*, pp. 2169–2176, 2017.

Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research, JAIR*, 13:227–303, 2000.

Ron Dorfman, Idan Shenfeld, and Aviv Tamar. Offline meta reinforcement learning – identifiability challenges and effective data collection strategies. In *Advances in Neural Information Processing Systems 34 (NeurIPS-21)*, pp. 4607–4618, 2021.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062, 2019.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. In *9th International Conference on Learning Representations (ICLR-21)*, 2021.

Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Tom Le Paine, Sergio Gómez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel Mankowitz, Cosmin Paduraru, Gabriel Dulac-Arnold, Jerry Li, Mohammad Norouzi, Matt Hoffman, Ofir Nachum, George Tucker, Nicolas Heess, and Nando deFreitas. Rl unplugged: Benchmarks for offline reinforcement learning, 2020.

Nico Gürtler, Sebastian Blaes, Pavel Kolev, Felix Widmaier, Manuel Wuthrich, Stefan Bauer, Bernhard Schölkopf, and Georg Martius. Benchmarking offline reinforcement learning on real-robot hardware. In *The Eleventh International Conference on Learning Representations (ICLR-23)*, 2023.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey 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)*, pp. 1861–1870, 2018.

P. Henderson, W.-D. Chang, F. Shkurti, J. Hansen, D. Meger, and G. Dudek. Benchmark environments for multitask learning in continuous domains. *ICML Lifelong Learning: A Reinforcement Learning Approach Workshop*, 2017.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. RLBench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

George Konidaris and Andrew Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. In *Advances in Neural Information Processing Systems 22 (NeurIPS-09)*, 2009.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Ashish Kumar, Toby Buckley, Qiaozhi Wang, Alicia Kavelaars, and Ilya Kuzovkin. Offworld gym: open-access physical robotics environment for real-world reinforcement learning benchmark and research. *CoRR*, abs/1910.08639, 2019a. URL http://arxiv.org/abs/1910.08639.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey 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 (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019b.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191. Curran Associates, Inc., 2020.

Aviral Kumar, Rishabh Agarwal, Xinyang Geng, George Tucker, and Sergey Levine. Offline Q-learning on diverse multi-task data both scales and generalizes. In *The Eleventh International Conference on Learning Representations (ICLR-23)*, 2023.

S. Lange, T. Gabel, and M. Riedmiller. Batch Reinforcement Learning. In M. Wiering and M. van Otterlo (eds.), *Reinforcement Learning: State of the Art*. Springer, in press, 2011.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Lanqing Li, Rui Yang, and Dijun Luo. FOCAL: Efficient fully-offline meta-reinforcement learning via distance metric learning and behavior regularization. In *International Conference on Learning Representations (ICLR-21)*, 2021.

Yunzhu Li, Hao He, Jiajun Wu, Dina Katabi, and Antonio Torralba. Learning compositional Koopman operators for model-based control. In *8th International Conference on Learning Representations, ICLR-20*, 2020.

Zuxin Liu, Zijian Guo, Haohong Lin, Yihang Yao, Jiacheng Zhu, Zhepeng Cen, Hanjiang Hu, Wenhao Yu, Tingnan Zhang, Jie Tan, et al. Datasets and benchmarks for offline safe reinforcement learning. *arXiv preprint arXiv:2306.09303*, 2023.

Yecheng Ma, Dinesh Jayaraman, and Osbert Bastani. Conservative offline distributional reinforcement learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 19235–19247. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/a05d886123a54de3ca4b0985b718fb9b-Paper.pdf.

Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pp. 879–893. PMLR, 2018.

Jorge A Mendez and Eric Eaton. How to reuse and compose knowledge for a lifetime of tasks: A survey on continual learning and functional composition. *Transactions on Machine Learning Research (TMLR)*, 2023.

Jorge A. Mendez, Marcel Hussing, Meghna Gummadi, and Eric Eaton. CompoSuite: A compositional reinforcement learning benchmark. In *1st Conference on Lifelong Learning Agents*, 2022a.

Jorge A. Mendez, Harm van Seijen, and Eric Eaton. Modular lifelong reinforcement learning via neural composition. In *10th International Conference on Learning Representations (ICLR-22)*, 2022b.

Eric Mitchell, Rafael Rafailov, Xue Bin Peng, Sergey Levine, and Chelsea Finn. Offline meta-reinforcement learning with advantage weighting. In *Proceedings of the 38th International Conference on Machine Learning (ICML-21)*, pp. 7780–7791, 2021.

Tongzhou Mu, Jiayuan Gu, Zhiwei Jia, Hao Tang, and Hao Su. Refactoring policy for compositional generalizability using self-supervised object proposals. In *Advances in Neural Information Processing Systems 33, NeurIPS-20*, pp. 8883–8894, 2020.

Ashvin Nair, Murtaza Dalal, Abhishek Gupta, and Sergey Levine. Accelerating online reinforcement learning with offline datasets. *CoRR*, abs/2006.09359, 2020.

Geraud Nangue Tasse, Steven James, and Benjamin Rosman. A Boolean task algebra for reinforcement learning. In *Advances in Neural Information Processing Systems 33, NeurIPS-20*, pp. 9497–9507, 2020.

Liam Paull, Jacopo Tani, Heejin Ahn, Javier Alonso-Mora, Luca Carlone, Michal Cap, Yu Fan Chen, Changhyun Choi, Jeff Dusek, Yajun Fang, Daniel Hoehener, Shih-Yuan Liu, Michael Novitzky, Igor Franzoni Okuyama, Jason Pazis, Guy Rosman, Valerio Varricchio, Hsueh-Cheng Wang, Dmitry Yershov, Hang Zhao, Michael Benjamin, Christopher Carr, Maria Zuber, Sertac Karaman, Emilio Frazzoli, Domitilla Del Vecchio, Daniela Rus, Jonathan How, John Leonard, and Andrea Censi.

Duckietown: An open, inexpensive and flexible platform for autonomy education and research. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1497–1504, 2017. doi: 10.1109/ICRA.2017.7989179.

Daniel Pickem, Paul Glotfelter, Li Wang, Mark Mote, Aaron Ames, Eric Feron, and Magnus Egerstedt. The robotarium: A remotely accessible swarm robotics research testbed. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1699–1706, 2017. doi: 10.1109/ICRA.2017.7989200.

Vitchyr H Pong, Ashvin V Nair, Laura M Smith, Catherine Huang, and Sergey Levine. Offline meta-reinforcement learning with online self-supervision. In *Proceedings of the 39th International Conference on Machine Learning (ICML-22)*, pp. 17811–17829, 2022.

Rong-Jun Qin, Xingyuan Zhang, Songyi Gao, Xiong-Hui Chen, Zewen Li, Weinan Zhang, and Yang Yu. NeoRL: A near real-world benchmark for offline reinforcement learning. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Yun Qu, Boyuan Wang, Jianzhun Shao, Yuhang Jiang, Chen Chen, Zhenbin Ye, Lin Liu, Yang Jun Feng, Lin Lai, Hongyang Qin, Minwen Deng, Juchao Zhuo, Deheng Ye, QIANG FU, YANG GUANG, Yang Wei, Lanxiao Huang, and Xiangyang Ji. Hokoff: Real game dataset from honor of kings and its offline reinforcement learning benchmarks. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Noah Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavior modelling priors for offline reinforcement learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rke7geHtwH.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999a.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211, 1999b.

Michita Imai Takuma Seno. d3rlpy: An offline deep reinforcement library. In *NeurIPS 2021 Offline Reinforcement Learning Workshop*, December 2021.

Chen Tessler, Shahar Givony, Tom Zahavy, Daniel Mankowitz, and Shie Mannor. A deep hierarchical approach to lifelong learning in Minecraft. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI-17*, 2017.

Tristan Tomilin, Meng Fang, Yudi Zhang, and Mykola Pechenizkiy. COOM: A game benchmark for continual reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

Benjamin Van Niekerk, Steven James, Adam Earle, and Benjamin Rosman. Composing value functions in reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML-19)*, pp. 6401–6409, 2019.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. FeUdal networks for hierarchical reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML-17*, pp. 3540–3549, 2017.

Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John P. Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy P. Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. StarCraft II: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

Brian Yang, Jesse Zhang, Vitchyr Pong, Sergey Levine, and Dinesh Jayaraman. Replab: A reproducible low-cost arm benchmark platform for robotic learning. *arXiv preprint arXiv:1905.07447*, 2019.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning (CoRL-20)*, pp. 1094–1100, 2020.

Tianhe Yu, Aviral Kumar, Yevgen Chebotar, Karol Hausman, Sergey Levine, and Chelsea Finn. Conservative data sharing for multi-task offline reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS-21)*, 2021.

Tony Z. Zhao, Jianlan Luo, Oleg Sushkov, Rugile Pevceviciute, Nicolas Heess, Jon Scholz, Stefan Schaal, and Sergey Levine. Offline meta-reinforcement learning for industrial insertion. In *2022 International Conference on Robotics and Automation (ICRA-22)*, pp. 6386–6393, 2022.

Gaoyue Zhou, Victoria Dean, Mohan Kumar Srirama, Aravind Rajeswaran, Jyothish Pari, Kyle Beltran Hatch, Aryan Jain, Tianhe Yu, Pieter Abbeel, Lerrel Pinto, Chelsea Finn, and Abhinav Gupta. Train offline, test online: A real robot learning benchmark. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.

Yuke Zhu, Josiah Wong, Ajay Mandlekar, and Roberto Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

A Computational requirements

We ran our experiments using both server-grade (e.g., NVIDIA RTX A6000s) and consumer-grade (e.g., NVIDIA RTX 3090) GPUs, depending on the number of tasks we consider. Large experiment's training on 224 tasks can be run within two days on a single NVIDIA A6000 GPU, but require up to 256GB of RAM. Smaller experiments with up to 64 training tasks can be trained within less than one day on a single RTX 3090 and 70GB of RAM. For evaluation, we used consumer-grade AMD CPUs with 16 cores and a single RTX 3090 for model inference.

B Hyperparameters

With the exception of the batch size, hyperparameters were left at the default values used in d3rlpy. Table 3 contains the hyperparameters used to generate the BC results, Table 4 contains those for IQL. Compositional BC/IQL used the same hyperparamters as BC/IQL, with the exception of the neural network architecture, which is described in detail in Appendix D. For the standard BC and IQL training, each neural network (all policies, Q-functions, and value functions) is encoded as a multi-layer perceptron (MLP) with 2 hidden layers and 256 hidden units per layer.

Table 3: Hyperparameters for Behavioral Cloning

Optimizer	Adam
Adam β_1	0.9
Adam β_2	0.999
Adam ε	1e-8
Learning Rate	1e-3
Batch Size	#Tasks $\times 256$

Table 4: Hyperparameters for Implicit Q-Learning

Optimizer	Adam
Adam β_1	0.9
Adam β_2	0.999
Adam ε	$1e-8$
Actor Learning Rate	$4e-3$
Critic Learning Rate	$4e-3$
Batch Size	#Tasks $\times 256$
n_steps	1
γ	0.99
τ	0.005
n_critics	2
expectile	0.7
weight_temp	3.0
max_weight	100

C Metrics

The metrics we report follow the original CompoSuite publication (Mendez et al., 2022a). The two metrics are given by:

- per-task cumulative returns: $\overline{R} = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{t=1}^{H} R_i(s_t, a_t)$, and
- per-task success rate: $\overline{S} = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \max_{t \in [1,H]} \mathbb{1}[R_i(s_t, a_t) = 1]$,

where N is the number of tasks, M is the number of evaluation trajectories, the length of each trajectory is H, and $\mathbb{1}$ is the indicator function. A success is defined as reaching the maximum reward of 1 per step in a single step during evaluation. Note that the success metric counts trajectories in which the agent is in a successful state at *any* time. In consequence, if the agent receives the maximum step reward once but then moves to a non-successful configuration, the trajectory is still counted as successful. We evaluate these two metrics separately over the training tasks and the (remaining) zero-shot tasks.

D Details on Compositional Policy

Our compositional policies use the same neural network architecture as used by Mendez et al. (2022a;b), which follows a graph structure that exploits the compositional relations across CompoSuite tasks. The full network consists of 16 MLP modules, each of which corresponds to a single element in CompoSuite—four obstacle modules, four object modules, four objective modules, and four robot modules. The graph is constructed hierarchically by passing the output of the previous module as (part of the) input to the next module. Each module operates in three stages: 1) a pre-processing MLP that consumes the module-specific component of the state as input (e.g., the robot module processes only the proprioceptive state features), 2) a concatenation layer that combines the output of the pre-processing module and the output of the previous module, and 3) a post-processing MLP that consumes the concatenated input and produces the module's output. The order of the hierarchy is `obstacle` $\rightarrow$ `object` $\rightarrow$ `objective` $\rightarrow$ `robot`. The `obstacle` modules have a single stage (since they are the first module), which is an MLP with a single hidden layer of size 32. The MLPs of the `object` module have each one hidden layer of 32 units. The first stage of the `objective` modules is an MLP with two hidden layers of 64 units each, and the second stage is an MLP with a single hidden layer of size 64. The `robot` module's first stage MLP has three hidden layers of size 64 and the second stage is the policy's output layer of dimension 8 for the 8 actions.