2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Real World Offline Reinforcement Learning with Realistic Data Source

Gaoyue Zhou[1]*, Liyiming Ke[24]*, Siddhartha Srinivasa[4], Abhinav Gupta[1], Aravind Rajeswaran[3], Vikash Kumar[3]

***Abstract*— Offline reinforcement learning (ORL) holds great promise for robot learning due to its ability to learn from arbitrary pre-generated experience. However, current ORL benchmarks are almost entirely in simulation and utilize contrived datasets like replay buffers of online RL agents or sub-optimal trajectories, and thus hold limited relevance for real-world robotics. In this work (Real-ORL), we posit that data collected from safe operations of closely related tasks are more practical data sources for real-world robot learning. Under these settings, we perform an extensive (6500+ trajectories collected over 800+ robot hours and 270+ human labor hour) empirical study evaluating generalization and transfer capabilities of representative ORL methods on four real-world tabletop manipulation tasks. Our study finds that ORL and imitation learning prefer different action spaces, and that ORL algorithms can generalize from leveraging offline heterogeneous data sources and outperform imitation learning. We release our dataset and implementations at URL: `https://sites.google.com/view/real-orl`**

I. Introduction

Despite rapid advances, the applicability of Deep Reinforcement Learning (DRL) algorithms [1]–[8] to real-world robotics tasks is limited due to sample inefficiency and safety considerations. The emerging field of offline reinforcement learning (ORL) [9], [10] has the potential to overcome these challenges, by learning only from logged or pre-generated offline datasets, thereby circumventing safety and exploration challenges. This makes ORL well suited for applications with large datasets (e.g. recommendation systems) or those where online interactions are scarce and expensive (e.g. robotics). However, comprehensive benchmarking and empirical evaluation of ORL algorithms is significantly lagging behind the burst of algorithmic progress [11]–[21]. Widely used ORL benchmarks [22], [23] are entirely in simulation and use contrived data collection protocols that do not capture fundamental considerations of physical robots. In this work (Real-ORL), we aim to bridge this gap by outlining practical offline dataset collection protocols that are representative of real-world robot settings. Our work also performs a comprehensive empirical study spanning **6500+ trajectories collected over 800+ robot hours and 270+ human labor hour**, to benchmark and analyze three representative ORL algorithms thoroughly. We release all the datasets and implementations from this paper.

*: equal contribution.
[1]: Carnegie Mellon University. `gaoyuez@andrew.cmu.edu`
[2]: Work conducted during an internship at Meta AI
[3]: Meta AI; [4]: University of Washington

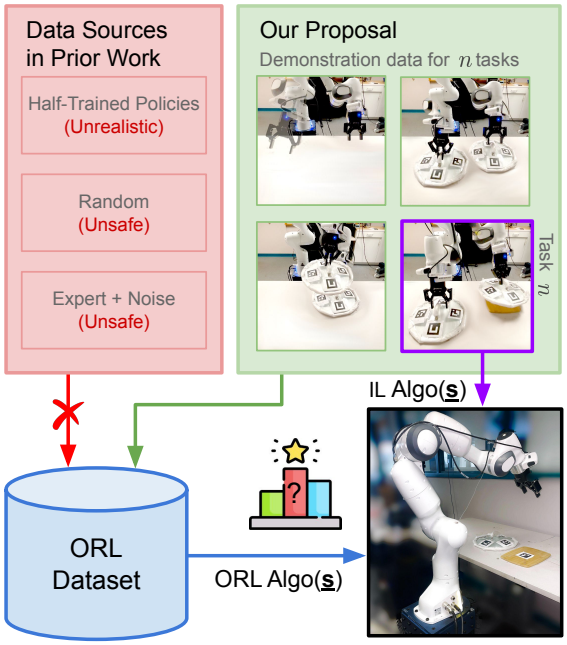

Fig. 1: Realistic data sources for ORL in real world tasks. The *Our Proposal* section displays four canonical tasks that we consider: `reach`, `slide`, `lift`, and `PnP`, arranged from top to bottom and left to right.

In principle, ORL can consume and train policies from arbitrary datasets. This has prompted the development of simulated ORL benchmarks [22]–[24] that utilize data sources like expert policies trained with *online* RL, exploratory policies, or even replay buffers of *online* RL agents. However, simulated dataset may fail to capture the challenges in real world: hardware noises coupled with varying reset conditions lead to covariate shift and violate the i.i.d. assumption about state distributions between train and test time. Further, such datasets are not feasible on physical robots and defeat the core motivation of ORL in robotics – to avoid the use of *online* RL due to poor sample efficiency and safety! Recent works [24], [25] suggest that dataset composition and distribution dramatically affect the relative performance of algorithms. In this backdrop, we consider the pertinent question: *What is a practical instantiation of the ORL setting for physical robots, and can existing ORL algorithms learn successful policies in such a setting?*

In this work, we envision *practical* scenarios to apply ORL for real-world robotics. Towards this end, our first

insight (Figure 1) is that real-world offline datasets are likely to come from well-behaved policies that abide by safety and monetary constraints, in sharp contrast to simulator data collected from exploratory or partially trained policies, as used in simulated benchmarks [22]–[24]. Such trajectories can be collected by user demonstrations or through hand-scripted policies that are partially successful but safe. It is more realistic to collect large volumes of data for real robots using multiple successful policies designed under expert supervision for specific tasks than using policies that are unsuccessful or without safety guarantees. Secondly, the goal of any learning (including ORL) is broad generalization and transfer. It is therefore critical to study whether a learning algorithm can leverage task-agnostic datasets, or datasets intended for a source task, to make progress on a new target task. In this work, we collect offline datasets consistent with these principles and evaluate representative ORL algorithms on a set of canonical table-top tasks as illustrated in Figure 1.

Evaluation studies on physical robots are sparse in the field due to time and resource constraints, but they are vital to furturing our understanding. Our real robot results corroborate and validate intuitions from simulated benchmarks [26] but also enable novel discoveries. We find that (1) even for scenarios with sufficiently high-quality data, some ORL algorithms could outperform behavior cloning (BC) [27] on specific tasks, (2) for scenarios that require generalization or transfer to new tasks with low data support, ORL agents generally outperform BC. (3) in cases with overlapping data support, ORL algorithms can leverage heterogeneous task-agnostic data to improve their own performance, and in some cases even surpass the best in-domain agent.

Our empirical evaluation is unique as it focuses on ORL algorithms ability to leverage more realistic, multi-task data sources, spans over several tasks that are algorithm-agnostic, trains various ORL algorithms on the same settings and evaluates them directly in the real world. In summary, we believe Real-ORL establishes the effectiveness of offline RL algorithms in leveraging out of domain high-quality heterogeneous data for generalization and transfer in robot-learning, which is representative of real world applications.

II. Preliminaries and Related Work

a) Markov Decision Process: We model the environment as a Markov Decision Process (MDP): $M = \langle S, A, R, T, \rho_0, H \rangle$ where $S \subseteq \mathbb{R}^n$ is the state space, $A \subseteq \mathbb{R}^m$ is the action space, $R : S \times A \to \mathbb{R}$ is the reward function, $T : S \times A \times S \to \mathbb{R}_+$ is the (stochastic) transition dynamics, $\rho_0 : S \to \mathbb{R}_+$ is the initial state distribution, and H is the maximum trajectory horizon. Here we assume access to R and a pre-generated dataset of the form: $\mathbb{D} = \{\tau_1, \tau_2, \dots \tau_N\}$, where each $\tau_i = (s_0, a_0, s_1, a_1, \dots s_H)$ is a trajectory collected using a behavioral policy or a mix of policies $\pi_b : S \times A \to \mathbb{R}_+$.

The goal in ORL is to use the offline dataset $\mathbb{D}$ to learn a near-optimal policy,

$$\pi^* := \arg\max_{\pi} \; \mathbb{E}_{M,\pi} \left[\sum_{t=0}^{H} r(s_t, a_t) \right].$$

b) Offline RL Algorithms: Recent years have seen tremendous interests in offline RL and the development of new ORL algorithms. Most of these algorithms incorporate some form of regularization or conservatism. This can take many forms, such as regularized policy gradients or actor critic algorithms [11]–[13], [28]–[30], approximate dynamic programming [14]–[17], and model-based RL [18], [31]–[33]. We select a representative ORL algorithms from each category: AWAC [12], IQL [17], and MOREL [18]. In this work, we do not propose new algorithms for ORL; rather we study a spectrum of representative ORL algorithms and evaluate their assumptions and effectiveness on a physical robot under realistic usage scenarios.

c) Offline RL Benchmarks and Evaluation: Existing offline RL benchmarks are predominantly captured with simulation [22], [23], [34], [35] using datasets with idealistic coverage of the state space, i.i.d. samples, and synchronous execution. Most of these assumptions are invalid in real world which is stochastic and has operational delays. Prior works investigating offline RL for these settings on physical robots are limited. For instance, IQL [17] did not conduct real robot evaluation, which we provide in this work; [36], [37] evaluate performance on a specialized Arm-Farm; [38] evaluate on a single drawer closing task; [16], [39] evaluate only one algorithm (COG, CQL, respectively). [40] evaluate BCQ and CQL alongside BC on three real robotics tasks, but their evaluations consider only in-domain setting: that the agents were trained only on the specific task data, without giving them access to a pre-generated, offline dataset. Thus, it is unclear whether insights from simulated benchmarks, limited hardware or in-domain evaluations can generalize broadly. Our work aims to bridge this gap by studying representative ORL algorithms on a suite of real-world robot learning tasks with an emphasize on transfer learning and out-domain generalization.

d) Imitation Learning (IL): IL [41] is an alternate approach to training control policies for robotics. Unlike RL, which learns policies by optimizing rewards (or costs), IL learns by mimicking expert demonstrations and typically requires no reward function. IL has been studied in both the offline setting [42], [43], where the agent learns from a fixed set of expert demonstrations, and the online setting [44], [45], where the agent performs additional environmental interactions. A combination of RL and IL has also been explored in prior work [46], [47]. Our offline dataset consists of trajectories from a heuristic hand-scripted policy collected under expert supervision, which represents a dataset of reasonably *high quality*. Therefore we consider offline IL and, BC in particular, as a baseline in our evaluation.

III. Experiment Scope and Setup

To investigate the effectiveness of ORL algorithms on real-world robot learning tasks, we adhere to a few guiding principles: (1) we strive to be fair to all baselines by providing them their best chance; (2) we prioritize reproducibility and conduct statistical significance tests and seed sweeping for all

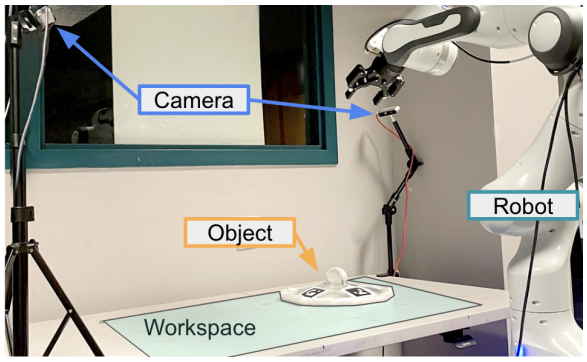

Fig. 2: Our setup consists of a Franka arm, a RobotiQ parallel gripper, and two Intel Realsense 435 cameras.

conclusions we draw; (3) we open-source our data, camera images along with our training and evaluation codebase.

a) Hardware Setup: Hardware plays a seminal role in robotic capability. For reproducibility and extensibility, we selected a hardware platform that is well-established, non-custom, and commonly used in the field. After an exhaustive literature survey [48]–[53], we converged on a table-top manipulation setup, shown in Figure 2. It consists of a table-mounted Franka panda arm that uses a RobotiQ parallel gripper as its end effector, accompanied by two Intel 435 RGBD cameras. Our robot has 8 DOF, uses factory-supplied default controller gains, accepts position commands at 15 Hz, and runs a low-level joint position controller at 1000 Hz. To perceive the object to interact with, we exact the position of the AprilTags attached to the object from RGB images. Our robot states consist of joint positions, joint velocities, and positions of the object to interact with (if applicable). Our policies compute actions (desired joint pose) using robot proprioception, tracked object locations, and goal location.

b) Canonical Tasks: We consider four classic manipulation tasks common in literature: `reach`, `slide`, `lift`, and `pick-n-place (PnP)` (see Figure 1). `reach` requires the robot to move from a randomly sampled configuration to another sampled config. The other three tasks involve a heavy glass lid with a handle, which is initialized randomly on the table. `slide` requires the robot to hold and push the lid along the table to a specified goal location. `lift` requires the robot to grasp and lift the lid 10 cm off the table. `PnP` requires the robot to grasp, lift, move and place the lid at a designated goal position i.e. the chopping board. The four tasks constitute a representative range of common tabletop manipulation challenges: `reach` focuses on free movements while the other three tasks involve intermittent interaction dynamics between the table, the lid, and the parallel grippers. We model each canonical task as a MDP with an unique reward function: Table. I.

Task	$r(s)$
`reach`	$1 - dis(g - x)$
`slide`	$1 - (2 * dis(g - t) + dis(t - x))$
`lift`	$min(1, 0.57 - dis(t - x) + height(t))$
`PnP`	$1 - (dis(g - t) + 2 * dis(t - x)) * 0.9 + height(t)$

TABLE I: Reward functions. x = position of the end-effector. g = goal position. t = position of the object.

c) Data Collection: For each canonical task, we design a scripted policy under expert supervision to collect (dominantly) successful trajectories. Previous works [22], [35], [40] have injected noise to expert policies, used half-trained RL policies or collected human demonstrations with varying qualities to highlight the performance gain of ORL algorithms over compromised, suboptimal datasets. We posit that such data sources are not representative of robotics domains, where noisy or random behaviors can be inefficient, unsafe or detrimental to hardware's stability. Instead of infusing noise or failure data points to serve as negative examples, we believe that mixing data collected from various tasks offers a more realistic setting in which to apply ORL on real robots for three reasons: (1) collecting such "random/roaming/explorative" data on a real robot autonomously would require comprehensive safety constraints, expert supervision and oversight, (2) engaging experts to record such random data in large quantities makes less sense than utilizing experts to collect meaningful trajectories on a real task, and (3) designing task-specific strategies and stress testing ORL's ability against practical dataset is more viable than using a compromised dataset. In Real-ORL, we collected demonstrations using heuristic strategies designed with reasonable efforts and mix data from different tasks to serve as offline dataset.

d) Dataset: Our Real-ORL dataset consists of around 3000 trajectories and the characteristics of our dataset is shown in Table. II. To validate the quality of our datasets, we use the Top-K% portion of dataset and trained behavior cloning for each task ($K \in \{100, 90\}$), observing that using the full datasets allowed behavior cloning to achieve the best performance on all tasks. To avoid biases favoring task/algorithm, we frozen the dataset ahead of time.

Task	Reach	Slide	Lift	Pick-n-place
# Traj	1000	731	609	616
# Samples	99752	244422	178515	327478
Avg Score	0.96	0.819	0.948	0.875
Max Score	0.99	0.93	1	1.09
Human Score	0.963	0.834	1	0.924
Theoretical Best Score	1	1	1	1.15
BC *using full dataset*	.924	.681	.823	.818
BC *using Top-90%*	.899	.659	.784	.789

TABLE II: Collected data. Each trajectory's score is the max reward in the trajectory. *Avg Score* shows the average scores per trajectories, *Max Score* shows the max reward achieved by trajectories in our dataset, *Human Score* shows the max reward of a human teleoperator and *Theoretical Best Score* denotes the theoretical max possible reward determined by our reward function.

IV. Experiment Design

Our Real-ORL experiments aim to answer the following questions. **(1)** Are ORL algorithms sensitive to, or show a preference for, any specific state and action space parameterization? **(2)** How do they perform against the standard methods for in-domain tasks? **(3)** How do they perform in

out-of-domain tasks requiring (a) generalization, and (b) re-targeting? To ensure fair evaluation, we now outline our choice of candidate algorithms and performance metrics.

Algorithms: We compare four algorithms: Behavior Cloning (BC) [27], Model-based Offline REinforcement Learning (MOREL) [18], Advantage-Weighted Actor Critic (AWAC) [12] and Implicit Q-Learning (IQL) [17]. BC is a model-free IL algorithm that remains a strong baseline for real robot experiments due to its simplicity and practicality [54]. AWAC is a variant of actor critic algorithms that uses KL divergence to constrain the resulting policy to be close to the given policy distribution. IQL is an approximate dynamic programming algorithm that leverages expectile regression to avoid querying the value function for any out-of-distribution query. MOREL is distinct since it is a model-based approach: it recovers a dynamics model from offline data that allows it directly apply policy gradient RL algorithms. [1]

Training: Since neural network agents are empirically sensitive to parameters and seeds, we (1) used the same fixed random seed (123) for all our experiments with additional seed sweeping to strengthen the reproducibility of our results and (2) conducted equal amount of efforts for hyperparameter tuning efforts for all algorithms. Unlike traditional supervised learning, we cannot simply select the agents with the best validation loss for tuning the hyperparameters, because we cannot know the performance of an agent unless testing it on a real robot [40]. We thus keep our tuning simple and fair: starting with the default parameters and training 5 agents in 3 rounds, trying to make the agent converge. We observe that certain agents cannot converge after exhausting the allocated trials and report these results with a (*) marker, signaling the challenge in tuning parameters for such algorithms.

Evaluation: Real robot evaluations can have high variance due to reset conditions and hardware noise. For each agent, we collect 12 trajectories and report their mean and standard deviation of scores. To confirm the reproducibility of our results and robustness to seed sweeping, for agents that contributed to our conclusions (usually the best and the second-best agents) we report performance swept over three consecutive random seeds (122, 124 in addition to the fixed seed of 123). To verify the statistical significance when comparing performance between agents, we conduct pairwise p-value test.

A. In-domain Ablations We note the distinction between "in-domain" training and "out-domain" training, where the former leverages only data that were collected for the test task and the later allow incorporating heterogeneous data from different tasks. We first train all agents using in-domain data (i.e., we train a `slide` agent by feeding only `slide` data) to test ORL algorithms' sensitivity to varying data representation and inspect: (1) whether it is worth including velocity information in the state space (**Vel** versus **NoVel**); for simulator experiments, it is almost always a gain to include velocity, but velocity sensors on real robots are notoriously noisy; (2) whether to use the policy output joint position (**Abs**) vs the change in joint position (**Delta**) as action. Most BC literature uses the former, whereas RL prefers the latter. We use the outcome of the ablations and the best-performing setting for each algorithm to study generalization and transfer in the following three scenarios:

B. Generalization: Lacking data support The data collection may not cover the task space uniformly. For example, imagine that a robot trained to wipe clean a table but now cleans a bigger table. Empirically, a policy trained with BC would have trouble predicting actions for states when there is less data support. Can ORL algorithms, by learning a value function or model, generalize to a task space that lacks data support? We create a new dataset from our slide task by dividing the task space to three regions: left, center, right. We remove any trajectory where the object was initially placed in the center region from the collected dataset. We train all agents and gather evaluation trajectories asking them to slide an object initially placed in the left, center and right regions.

C. Generalization: Re-targeting data for dynamic tasks For the slide task, our collected demonstrations has static goal positions. We test agents trained with such static data in a dynamic setting by updating the goal at a fixed frequency, and asking agents to slide the lid following some predetermined curves. We collected goal trajectories via human teleoperation. This task can be viewed as a simplified version of daily tasks, including drawing, wiping, and cleaning, which require possibly repeated actions and a much longer horizon than usual IL and ORL tasks. We select a variety of trajectories: circle, square, and the numbers 3, 5, 6, 8, which have different combinations of smooth curves and corners.

D. Transfer: Reusing data from different tasks We investigate whether we can reuse hetergenuous data collected from previous tasks to train a policy for a new task. For example, would combining data from two canonical tasks (e.g., `slide+lift`) helps the agent perform better on either of these tasks? When aggregating data collected for multiple tasks, ORL algorithms can use the reward function for the test task to relabel the offline dataset. Evaluating ORL algorithms on such out-domain, transfer-learning settings is practical and relevant: instead of collecting random explorative data which demands careful setup of safety constraints on a real robot, we want to leverage offline datasets collected from different tasks to improve ORL performance. We train our algorithm with different combinations of canonical task demonstrations ("train-data") and evaluate each agent on each individual task.

V. Results and Discussion

A. In-domain Tasks

Table III summarizes all agents' performance for *in-domain* tasks. Interestingly, two of the ORL agent seemed to win over BC on two tasks. After verifying statistical significance, we confirmed that even with abundant, in-domain real robot demonstrations, IQL outperformed BC on two tasks ($p = 0.041, p = 0.001$). For the other tasks: on the

[1] We use implementations of BC and MOREL from the MOREL author implementation. For the later, we add a weighted behavior cloning loss to its policy training step to serve as a regularizer, inspired by [30]. We use AWAC and IQL implemented in the open sourced d3rlpy library [55].

Task	Agent	Representations			
		AbsNoVel	**AbsVel**	**DeltaNoVel**	**DeltaVel**
Reach	BC	.86 ± .07	.77 ± .12	.91 ± .03	**.92 ± .05**
	Morel	.80 ± .09	.58 ± .11	.86 ± .07	.92 ± .04
	AWAC	.77 ± .11	.71 ± .16	.92 ± .03	**.93 ± .05**
	IQL	.84 ± .15	.87 ± .10	.90 ± .03	.89 ± .07
Slide	BC	.62 ± .17	**.68 ± .15**	.55 ± .20	.55 ± .10
	Morel	.36 ± .19	.12 ± .24	.53 ± .15	.63 ± .16
	AWAC	.55 ± .17 *	.59 ± .15 *	.57 ± 0.14	.73 ± .11
	IQL	.67 ± .14	.59 ± .17	.71 ± .14	**.77 ± .07**
Lift	BC	.76 ± .18	**.82 ± .18**	.72 ± .23	.61 ± .14
	Morel	.46 ± .19	.15 ± .09	.68 ± .19	.65 ± .16
	AWAC	.52 ± .08 *	<0 *	.86 ± .15 *	.82 ± .12
	IQL	.68 ± .16	<0	.84 ± .14	**.88 ± .15**
PnP	BC	.63 ± .12	**.82 ± .19**	.56 ± .05	.68 ± .20
	Morel	<0	<0	**.75 ± .20**	.75 ± .22
	AWAC	.45 ± .16 *	<0	.63 ± .23 *	.74 ± .18 *
	IQL	.47 ± .14	<0	.55 ± .16	.60 ± .23

TABLE III: Performance of all algorithms on varying representations. We trained all agents on four settings: to include velocity in state or not (**Vel** versus **NoVel**); to use absolute or delta action space (**Abs** versus **Delta**). For each task, the best BC agent and the best ORL agent are highlighted and bolded. Agents that could not converge during training time are marked with (*). Some agents triggered violent crashes on robot (<0). Underline scores are swept over 3 seeds.

simplest task `reach`, the best version of all agents reached comparable performance ($p = 0.84$). On the hardest task `PnP`, BC outperformed the best ORL agents ($p = 0.016$). We thus recommend considering both BC and IQL as baselines for classic imitation learning.

In terms of sensitivity to representations, BC demonstrated empirical robustness to different state and action spaces, whereas ORL agents had high-variance performance. In 3 of 4 tasks considered, BC performed the best when using absolution joint position as the action space and including velocity in the state space (**AbsVel**). On all tasks, ORL agents performed better using delta action space (**Delta**) than joint position. Intuitively, using the delta action space is equivalent to restricting the policy to move in a unit ball centered around the current state. Such constraints could benefit RL policies that need exploration and sampling in action space more than it helped BC, which simply learns the mapping from states to actions. Additionally, we observed that our best agents all included velocity in their state space, despite that velocities on real robot fluctuate with hardware noise.

B. Generalization and Transfer

Generalization to regions that lack data support. Table IV trains agent using a carved-out dataset and compares the agents' performance on regions with more data support versus the region with less data support (`Center`). We also train all agents using the full dataset (without carved-out) and evaluate them on the `Center` region (`Center(FULL)`). We discovered that: (1) on the region with abundant support (`Left` and `Right`), BC/IQL performed better than AWAC/MOREL ($p = 0.062$), aligning with our previous observation that BC/IQL performed better on in-domain tasks, (2) on regions that have less data support, AWAC and

Start Position	BC	MOREL	AWAC	IQL
`Left`	.79 ± .06	.57 ± .06	.70 ± .12	.70 ± .07
`Right`	.77 ± .02	.80 ± .03	.71 ± .14	.81 ± .02
`Center`	.76 ± .01	.79 ± .02	.83 ± .03	.81 ± .01
`Center (FULL)`	.79 ± .02	.78 ± .02	.81 ± .02	.81 ± .05

TABLE IV: Training agents using a carved-out dataset to see how they perform when *generalizing* to a task region that lacks data support (the `Center` region, highlighted in Gray). For comparison, we also train all agents using full dataset and evaluate them on the `Center` region, producing `Center(FULL)` results.

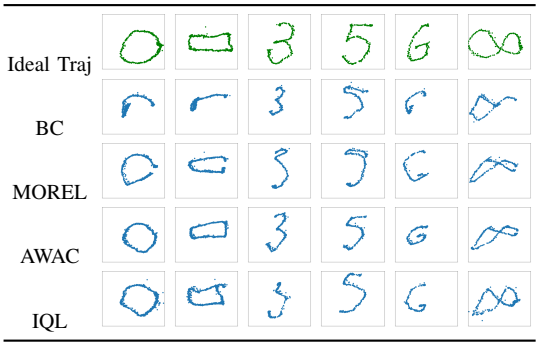

TABLE V: Trajectory tracking. Green: the ideal demo trajectories, followed by each agent's tracking trajectories.

MOREL could match BC's performance ($p > 0.1$, no significant difference) despite their initial disadvantage; and (3) ORL agents trained with carved-out dataset and evaluated on carved-out region performed no worse than them trained with full dataset, in contrast to BC agent, which performed significantly worse after carving-out.

Generalization to dynamic tasks. Table V lists the ideal curves and the curves traced by each model. Each dot represents the location of the object at a time step. BC had the worst performance among all models since it failed to trace the circle, square, and number 8 which require a larger range of motion by seemingly getting stuck during execution. Meanwhile, ORL methods largely succeeded tracing the entire curve following the time-varying goals, demonstrating stronger generalizing ability for this dynamic task.

Transfer learning by leveraging heterogeneous dataset. Table VI evaluates the performance of ORL algorithms when trained with different combinations of datasets from multiple tasks. We observe that:

1) The performance changes to ORL agents after leveraging offline data from different tasks can vary, due to the characteristics of the algorithm, the nature of the task, design of the reward function and the data distribution.

2) We observed all ORL agents *could* improve their own performance using some task/data combinations. Noticeably, MOREL achieved higher or comparable performance on all tasks after leveraging more offline data. On `lift`, MOREL performance increased $0.61 \rightarrow 0.73 \rightarrow 0.90$ ($p = 0.003$) by including data from `slide` and `PnP`. Intuitively, MOREL's dynamic model training process could benefit from any realistic data, regardless of whether the data was

in-domain or out-of-domain.

3) Certain task-agnostic data could provide overlapping data support and enable effective transfer learning, allowing some ORL agents to surpass imitation learning and even the best in-domain agents. On `slide` and `lift`, all ORL algorithms managed to surpass BC ($p < 0.1$). On `PnP`, AWAC achieved comparable performance as BC ($p > 0.1$) but with a slightly higher mean using a combo of `slide` and `lift` data. With our extensive ablations, we observe that the final best agent for each task is either an ORL algorithm or a tie between ORL and BC.

4) ORL algorithms are not guaranteed to increase performance by including more data. For instance, both AWAC and IQL agents have not gained performance on `lift`when using `slide+lift+PnP` than using only `slide+lift` data ($p > 0.1$). Surprisingly, training IQL for `PnP` using `slide` or `slide+lift` data yielded even better results than including `PnP` data ($0.84 > 0.6$). Qualitatively we observe that IQL agents trained with `slide` data were better at grasping the object than the ones trained with `PnP` data, completing this first part of the task (grasp) with more success while claiming distance-to-goal reward bonus.

C. Reproducibility and comparisong to previous work

To further demonstrate the reproducibility of our results, we conducted random seed sweeping for our best and (optionally) the second-best agents over 3 consecutive random seeds ($122, 124$ in addition to the original fixed seed 123) and reported their scores using an underline. These additional 360 trajectories showed that the seed2seed variation for our experiments is low, providing statistical significance to our observations: comparing with scores computed from a single-seed, $\sim$60% of newly trained agents change score by less than 1%, $\sim$90% of agents change by less than 2%, and the maximum change was 6% from one agent (whose score change does not affect the conclusion drawn).

Some of our *in-domain* conclusions align with previous works [40], [41]: that behavior cloning demonstrates strong robustness to varying representations and tasks, serving as a competitive baseline in all four tasks tested. Even when BC is not the best, it has reasonable performance that is no worse than 85% of the best in-domain agents. Our findings also provide empirical verification to one of [35]'s observations that ORL could outperform BC for tasks where the initial state distributions change during deployment, a common condition for real robotic task, or when the environment has a few "critical" states, as seen in our manipulation tasks. In contrast to previous works, however, we highlight that (1) IQL can be a competitive baseline for settings that were traditionally favoring behavior cloning, as it turns out to be the best in-domain agent on 2 out of 4 tasks we tested, despite the lack of real robotic evaluation for IQL [17], (2) our extensive ablations on out-domain transfer learning are unique and allow us to verify several ORL algorithms' capability in generalizing to task region with less data-support (Table. IV) and to dynamic tasks (Table. V), (3) we observe that leveraging heterogeneous data has enabled all ORL algorithms to improve their own performance on at least one of the tasks, allowing some to even surpass the best in-domain agents, which suggest that ORL can be an interesting paradigm for real-world robotic learning.

Agent	Train Data	Test Task `slide`	Test Task `lift`	Test Task `PnP`
BC	in-domain	.68 ± .15	.82 ± .18	.82 ± .19
	slide	.68 ± .15	.58 ± .06	.61 ± .08
	slide+lift	.60 ± .13	.58 ± .05	.61 ± .12
	slide+lift+PnP	.61 ± .14	.61 ± .08	.64 ± .14
MOREL	in-domain	.63 ± .16	.68 ± .19	.75 ± .20
	slide	.63 ± .16	.61 ± .06	.74 ± .17
	slide+lift	.62 ± .15	.73 ± .18	.64 ± .17
	slide+lift+PnP	.72 ± .13	**.90 ± .13**	.75 ± .18
AWAC	in-domain	.73 ± .11	.86 ± .15 *	.74 ± .18 *
	slide	.73 ± .11	.64 ± .06	.77 ± .11*
	slide+lift	.73 ± .11 *	.90 ± .15	.81 ± .12
	slide+lift+PnP	.64 ± .14 *	.73 ± .20 *	.76 ± .19 *
IQL	in-domain	**.77 ± .07**	.88 ± .12	.60 ± .23
	slide	.76 ± .10	.26 ± .03	.81 ± .11
	slide+lift	.70 ± .14	.86 ± .17	**.84 ± .11**
	slide+lift+PnP	.64 ± .14	.68 ± .16	.83 ± .18

TABLE VI: Training agents trained with different combinations of offline data. The best in-domain agent, transfer learning agents that improves over their in-domain counterparts are colored. The **best agent for each task** is bold. Agents that could not converge during training time are marked with (*). Some agents triggered violent crashes on robot (<0). Underline scores are swept over 3 seeds.

VI. Conclusion

In Real-ORL, we conducted an empirical study of representative ORL algorithms on real-world robotic learning tasks. The study encompassed three representative ORL algorithms (along with BC), four table-top manipulation tasks with a Franka-Panda robot arm, `3000+` train trajectories, `3500+` evaluation trajectories, and `270+` human labor hours. Through our extensive ablation studies, we find that (1) even for in-domain tasks with abundant amount of high-quality data, IQL can be a competitive baseline against the best BC policy, (2) for out-domain tasks, ORL algorithms were able to generalize to task regions with low data-support and to dynamic tasks, (3) the performance changes of ORL after leveraging heterogeneous data are likely to vary by agents, the design of the task, and the characteristics of the data, (4) certain heterogeneous task-agnostic data could provide overlapping data support and enable transfer learning, allowing ORL agents to improve their own performance and, in some cases, even surpass the best in-domain agents. Overall, (5) the best agent for each task is either an ORL algorithm or a tie between ORL and BC. Our rigorous evaluations indicate that even in out-of-domain multi-task data regime, (more realistic in real world setting) offline RL is an effective paradigm to leverage out of domain data.

Acknowledgement This work was (partially) funded by the National Science Foundation NRI (#2132848) & CHS (#2007011), DARPA RACER, the Office of Naval Research, Honda Research Institute, and Amazon.

REFERENCES

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

[2] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust Region Policy Optimization," *CoRR*, vol. abs/1502.05477, 2015.

[3] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *CoRR*, vol. abs/1707.06347, 2017.

[4] 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," *arXiv preprint arXiv:1509.02971*, 2015.

[5] M. Janner, J. Fu, M. Zhang, and S. Levine, "When to trust your model: Model-based policy optimization," in *NeurIPS*, 2019.

[6] A. Rajeswaran, I. Mordatch, and V. Kumar, "A Game Theoretic Framework for Model-Based Reinforcement Learning," in *ICML*, 2020.

[7] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. P. Lillicrap, K. Simonyan, and D. Hassabis, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play," *Science*, vol. 362, pp. 1140–1144, 2018.

[8] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. J. Dudzik, J. Chung, D. H. Choi, R. W. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. P. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, pp. 1–5, 2019.

[9] S. Lange, T. Gabel, and M. A. Riedmiller, "Batch Reinforcement Learning," in *Reinforcement Learning*. Springer, 2012, vol. 12.

[10] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[11] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision Transformer: Reinforcement Learning via Sequence Modeling," 2021.

[12] A. Nair, M. Dalal, A. Gupta, and S. Levine, "Accelerating online reinforcement learning with offline datasets," *arXiv preprint arXiv:2006.09359*, 2020.

[13] Z. Wang, A. Novikov, K. Zolna, J. S. Merel, J. T. Springenberg, S. E. Reed, B. Shahriari, N. Siegel, C. Gulcehre, N. Heess *et al.*, "Critic regularized regression," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7768–7778, 2020.

[14] S. Fujimoto, D. Meger, and D. Precup, "Off-Policy Deep Reinforcement Learning without Exploration," *CoRR*, vol. abs/1812.02900, 2018.

[15] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative Q-Learning for Offline Reinforcement Learning," *ArXiv*, vol. abs/2006.04779, 2020.

[16] A. Singh, A. Yu, J. Yang, J. Zhang, A. Kumar, and S. Levine, "Cog: Connecting new skills to past experience with offline reinforcement learning," *arXiv preprint arXiv:2010.14500*, 2020.

[17] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," *arXiv preprint arXiv:2110.06169*, 2021.

[18] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, "MOReL : Model-Based Offline Reinforcement Learning," in *NeurIPS*, 2020.

[19] W. Zhou, S. Bajracharya, and D. Held, "Plas: Latent action space for offline reinforcement learning," *arXiv preprint arXiv:2011.07213*, 2020.

[20] K. Konyushova, Y. Chen, T. Paine, C. Gulcehre, C. Paduraru, D. J. Mankowitz, M. Denil, and N. de Freitas, "Active offline policy selection," *Advances in Neural Information Processing Systems*, vol. 34, pp. 24 631–24 644, 2021.

[21] T. Z. Zhao, J. Luo, O. Sushkov, R. Pevceviciute, N. Heess, J. Scholz, S. Schaal, and S. Levine, "Offline meta-reinforcement learning for industrial insertion," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6386–6393.

[22] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4RL: Datasets for Deep Data-Driven Reinforcement Learning," *ArXiv*, vol. abs/2004.07219, 2020.

[23] C. Gulcehre, Z. Wang, A. Novikov, T. L. Paine, S. G. Colmenarejo, K. Zolna, R. Agarwal, J. Merel, D. Mankowitz, C. Paduraru *et al.*, "Rl unplugged: Benchmarks for offline reinforcement learning," *arXiv preprint arXiv:2006.13888*, 2020.

[24] D. Yarats, D. Brandfonbrener, H. Liu, M. Laskin, P. Abbeel, A. Lazaric, and L. Pinto, "Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning," *ArXiv*, vol. abs/2201.13425, 2022.

[25] R. Qin, S. Gao, X. Zhang, Z. Xu, S. Huang, Z. Li, W. Zhang, and Y. Yu, "Neorl: A near real-world benchmark for offline reinforcement learning," *ArXiv*, vol. abs/2102.00714, 2021.

[26] A. Kumar, J. Hong, A. Singh, and S. Levine, "When should we prefer offline reinforcement learning over behavioral cloning?" *ArXiv*, vol. abs/2204.05618, 2022.

[27] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, vol. 1, 1988.

[28] Y. Wu, G. Tucker, and O. Nachum, "Behavior Regularized Offline Reinforcement Learning," *CoRR*, vol. arXiv:1911.11361, 2019.

[29] N. Jaques, A. Ghandeharioun, J. H. Shen, C. Ferguson, À. Lapedriza, N. Jones, S. Gu, and R. W. Picard, "Way Off-Policy Batch Deep Reinforcement Learning of Implicit Human Preferences in Dialog," *CoRR*, vol. abs/1907.00456, 2019.

[30] S. Fujimoto and S. S. Gu, "A minimalist approach to offline reinforcement learning," in *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[31] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma, "Mopo: Model-based offline policy optimization," *ArXiv*, vol. abs/2005.13239, 2020.

[32] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn, "Combo: Conservative offline model-based policy optimization," in *NeurIPS*, 2021.

[33] A. Argenson and G. Dulac-Arnold, "Model-based offline planning," *ArXiv*, vol. abs/2008.05556, 2021.

[34] R. Qin, S. Gao, X. Zhang, Z. Xu, S. Huang, Z. Li, W. Zhang, and Y. Yu, "Neorl: A near real-world benchmark for offline reinforcement learning," *arXiv preprint arXiv:2102.00714*, 2021.

[35] A. Kumar, J. Hong, A. Singh, and S. Levine, "When should we prefer offline reinforcement learning over behavioral cloning?" *arXiv preprint arXiv:2204.05618*, 2022.

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

[37] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman, "Mt-opt: Continuous multi-task robotic reinforcement learning at scale," *ArXiv*, vol. abs/2104.08212, 2021.

[38] R. Rafailov, T. Yu, A. Rajeswaran, and C. Finn, "Offline reinforcement learning from images with latent space models," in *L4DC*, 2021.

[39] A. Kumar, A. Singh, S. Tian, C. Finn, and S. Levine, "A workflow for offline model-free robotic reinforcement learning," in *CoRL*, 2021.

[40] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," *arXiv preprint arXiv:2108.03298*, 2021.

[41] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[42] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5628–5635.

[43] L. Ke, J. Wang, T. Bhattacharjee, B. Boots, and S. Srinivasa, "Grasping with chopsticks: Combating covariate shift in model-free imitation learning for fine manipulation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6185–6191.

[44] J. Chang, M. Uehara, D. Sreenivas, R. Kidambi, and W. Sun, "Mitigating covariate shift in imitation learning via offline data without great coverage," *ArXiv*, vol. abs/2106.03207, 2021.

[45] R. Rafailov, T. Yu, A. Rajeswaran, and C. Finn, "Visual adversarial imitation learning using variational models," in *NeurIPS*, 2021.

[46] W. Sun, J. A. Bagnell, and B. Boots, "Truncated horizon policy search: Combining reinforcement learning & imitation learning," *arXiv preprint arXiv:1805.11240*, 2018.
[47] S. Reddy, A. D. Dragan, and S. Levine, "Sqil: Imitation learning via reinforcement learning with sparse rewards," *arXiv preprint arXiv:1905.11108*, 2019.
[48] O. Kroemer, S. Niekum, and G. D. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *Journal of machine learning research*, vol. 22, no. 30, 2021.
[49] M. Shridhar, L. Manuelli, and D. Fox, "Cliport: What and where pathways for robotic manipulation," in *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.
[50] M. Bhardwaj, B. Sundaralingam, A. Mousavian, N. D. Ratliff, D. Fox, F. Ramos, and B. Boots, "STORM: An integrated framework for fast joint-space model-predictive control for reactive manipulation," 2021.
[51] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3389–3396.
[52] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," in *Conference on Robot Learning*. PMLR, 2022, pp. 158–168.
[53] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," *arXiv preprint arXiv:2203.12601*, 2022.
[54] S. Dasari, J. Wang, J. Hong, S. Bahl, Y. Lin, A. Wang, A. Thankaraj, K. Chahal, B. Calli, S. Gupta *et al.*, "Rb2: Robotic manipulation benchmarking with a twist," *arXiv preprint arXiv:2203.08098*, 2022.
[55] M. I. Takuma Seno, "d3rlpy: An offline deep reinforcement library," in *NeurIPS 2021 Offline Reinforcement Learning Workshop*, December 2021.