IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Identifying Expert Behavior in Offline Training Datasets Improves Behavioral Cloning of Robotic Manipulation Policies

Qiang Wang[1], Robert McCarthy[2], David Cordova Bulens[1], Francisco Roldan Sanchez[3], Kevin McGuinness[3], Noel E. O'Connor[3], and Stephen J. Redmond[1]

***Abstract*—This paper presents our solution for the Real Robot Challenge III[1], aiming to address dexterous robotic manipulation tasks through learning from offline data. In this competition, participants were given two types of datasets for each task: expert and mixed. Each expert dataset is collected by a high-skill policy, whereas the mixed dataset is collected using both expert and non-expert policies. We found that the vanilla behavioural cloning (BC) can learn a very proficient policy with minimal human intervention when trained on expert datasets. Notably, BC outperformed even the most advanced offline reinforcement learning (RL) algorithms. However, when applied to mixed datasets, the performance of BC deteriorates; similarly, the performance of offline RL algorithms is also less than satisfactory. Upon examining the provided datasets, it was apparent that each mixed dataset contained a significant proportion of expert data, which should enable the training of a proficient BC agent. However, the expert data is not labelled in the datasets. As a result, we propose a classifier to identify the pattern of the expert behaviour within a mixed dataset and then utilize it to isolate the expert data. To further boost the BC performance, we take advantage of the geometric symmetry of the arena to augment the training dataset through mathematical transformations. Ultimately, our submission outperformed that of all other participants. Our solution scripts are available on our website: https://github.com/wq13552463699/Real-Robot-Challenge-III-Winning-Solution.**

***Index Terms*—Imitation Learning; Reinforcement Learning; Data Sets for Robot Learning**

I. INTRODUCTION

DATA-DRIVEN learning methods are emerging as a promising approach for dexterous robotic manipulation tasks and have begun to surpass conventional control methods in certain scenarios [1]–[3]. However, their application in the physical world remains limited. This is because these methods often require extensive data acquisition from the environment, which is typically costly and time-consuming in physical settings. These issues can be mitigated by making use of available pre-collected data. The Real Robot Challenge (RRC) III sought to encourage the development of offline policy learning algorithms that can make efficient use of such pre-existing real-world data [4], and thus improve the performance of these learning methods when deployed in practical real-world scenarios.

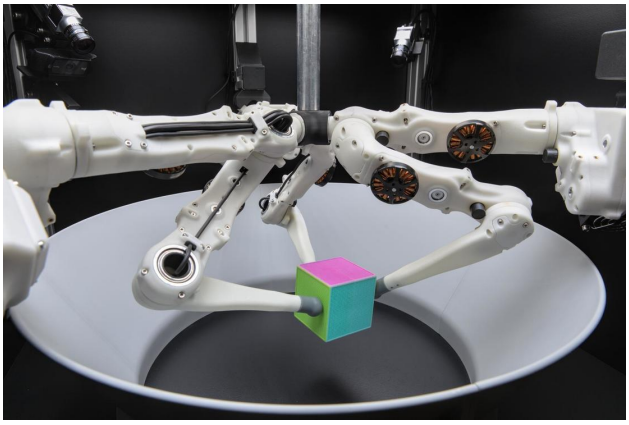


Figure 1: Illustration of the physical TriFinger robot from the RRC III competition. Images sourced from the RRC website[1].

A. Real Robot Challenge III

The RRC III robotic platform, as seen in Fig. 1, features three identical robotic fingers symmetrically positioned at 120° intervals around a circular arena. The coloured cube is the object to be moved. In this competition, participants aim to solve two tasks: the *push* task and the *lift* task (please visit the RRC III website to access the demonstration videos). The objective of the *push* task is to relocate the cube to a target 2D position on the arena floor. The *lift* task, on the other hand, presents a more demanding challenge, necessitating the cube to be elevated and maintain a stable target pose (3D position and orientation). The action in the task consists of commanding the torques sent to the actuators in the 3 joints of the 3 fingers. The state of the task includes: the angular positions, velocity, and torques applied to the robot's 9 joints; the forces, Cartesian coordinates, and velocity of the robot's 3 fingertips; the robot's ID; the Cartesian positions, quaternion orientations,

Manuscript received July 12, 2023; Revised October 6, 2023; Accepted November 5, 2023. This paper was recommended for publication by Editor Aleksandra Faust upon evaluation of the Associate Editor and Reviewers' comments. This publication has emanated from research conducted with the financial support of China Scholarship Council under grant number CSC202006540003 and of Science Foundation Ireland under grant numbers 17/FRL/4832 and SFI/12/RC/2289_P2. *(Corresponding author: Stephen J. Redmond)*

[1]Qiang Wang, David Cordova Bulens, and Stephen J. Redmond are with School of Electrical and Electronic Engineering, University College Dublin, Ireland (e-mail: qiang.wang@ucdconnect.ie; {david.cordovabulens, stephen.redmond}@ucd.ie).

[2]Robert McCarthy is with CeADAR - Ireland's Centre for Applied AI, University College Dublin, Ireland (e-mail: robert.mccarthy.22@ucl.ac.uk).

[3]Francisco Roldan Sanchez, Kevin McGuinness and Noel E. O'Connor are with School of Electronic Engineering, Dublin City University, Ireland (e-mail: francisco.sanchez@insight-centre.org; {kevin.mcguinness, noel.oconnor}@dcu.ie).

Digital Object Identifier (DOI): see top of this page.

[1]Featured in the NeurIPS 2022 Competition Track, more details see https://real-robot-challenge.com/

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

and keypoints (Cartesian coordinates of the 8 vertices of the object) of the object; the delay in object tracking; confidence in estimates of the object's state; the achieved and desired goal (Cartesian coordinates of object center points for the *push* task and keypoints for the *lift* task); and the action at the last time point. The reward is determined using a logistic kernel, $k(x) = (b+2)(exp(a\,\|x\|) + b + exp(-a\,\|x\|))^{-1}$, where x represents the distance between the desired and achieved goal. The parameters a regulate the length scale over which the reward diminishes, while b controls the sensitivity of the reward to small distances. The cube's initial position and orientation are randomized to ensure variability in the starting setup.

For each task, we are provided two datasets: one obtained from an expert policy (denoted as expert dataset), and the other collected from multiple policies exhibiting varying skill levels (denoted as mixed dataset). The specific configurations of each dataset are outlined in TABLE I. Moreover, participants are provided with a cluster of six real TriFinger robots and a simulation environment with the same configuration to evaluate their trained policies. It is crucial to emphasize that, in the spirit of offline learning, any data collected during the evaluation, whether from simulation or the real robot, cannot be utilized for refining the policy further. For more detailed information, please refer to the RRC III website provided above.

Table I: Information about the datasets provided.

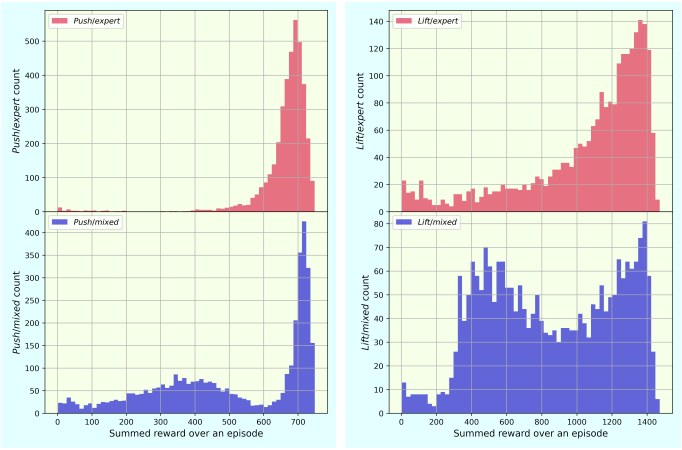
Dataset	Mean return	Number of transitions	Episodic length
Push/expert	660	2.8×10^6	750
Push/mixed	429	2.8×10^6	750
Lift/expert	1064	3.6×10^6	1500
Lift/mixed	851	3.6×10^6	1500

B. Dataset investigations

We explored the provided datasets during the early stage of the competition. We employed cutting-edge offline policy learning algorithms to train multiple agents, specifically including PLAS [5], TD3+BC [6] and BC [7]. The evaluated results are presented in TABLE II. We found that agents trained on the expert dataset consistently outperformed those trained on the mixed datasets. Notably, BC demonstrated superior performance on the expert datasets, surpassing all sophisticated offline RL algorithms. Furthermore, when training with mixed datasets, none of the algorithms were able to achieve satisfactory performance, especially for the relatively challenging *lift* task. Therefore, we believed that better utilizing the mixed datasets would be key to outperforming other competitors.

To gain insight into the composition of the mixed datasets, we plot their reward distributions and compare them with those of the expert datasets in Fig. 2. It is evident that a significant portion of episodes in the mixed datasets achieve expert-level scores. Hence, we formulated a hypothesis that the mixed dataset contains a large proportion of expert data. Motivated by the remarkable performance of BC on the expert dataset, we set out to filter out a subset of expert data from the mixed dataset and subsequently utilize this subset for BC training.

A simple method to identify expert data within a mixed dataset is to use the rewards, as experts tend to earn higher rewards when performing a task compared to less effective



(a) *Push*/expert and *Push*/mixed (b) *Lift*/expert and *Lift*/mixed

Figure 2: The histograms of the accumulated reward four datasets, calculated by summing the rewards across all time steps in each episode (each manipulation trajectory). The expert dataset consists mostly of successful episodes achieving large cumulative reward values, and it has one distinct peak. The mixed dataset appears to consist of two distinct peaks.

policies. Hence, in our investigations, we extracted the top 10% and 50% of rewarded episodes for BC training inspired by [8]. The evaluated results are shown in TABLE II. This method basically addresses the *push*/mixed dataset as there is a substantial performance disparity between the expert and the weaker policy that are assumed to have generated the dataset (see Fig. 2(a)). However, the effectiveness of this filtering approach diminishes when applied to the *lift*/mixed dataset. This is because the expert and weaker policies exhibit relatively similar performance on the *lift*/mixed task, resulting in significant overlap in cumulative rewards (see Fig. 2(b)). Indeed, using the top 10% reward episodes, a high threshold ensures mostly expert data extraction but results in an insufficient subset for robust policy training.

C. Our method

In this paper, we propose a learning-based filter method to address the above problem; it is essentially a binary classifier that can learn the patterns of expert behavior and use the acquired knowledge to filter expert data from the mixed dataset.

Considering the reduction of the amount of data left for policy learning after filtering, we utilize the rotational symmetry of the RRC III robot platform geometry to augment the training data. Essentially, this method triples the dataset size through a mathematical transformation. Moreover, we propose a theory-to-real transfer training method, enabling the trained policy derived from the theoretically augmented dataset to adapt to the actual environment.

II. RELATED WORK

A. Real-world dexterous robotic manipulation

Dexterous robotic manipulation refers to enabling robots to control and manipulate objects with human-like precision and strength. Generally, such a system utilizes artificial sensors to

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

gather information about the state of the world. It incorporates a controller processing sensor inputs and generating control signals, which are then transmitted to robotic manipulators capable of executing delicate movements akin to those performed by humans [9], such as pick-and-place operations [10], rotation tasks [11], and the utilization of tools [12].

Traditional robotic manipulation: This entails the manual coordination of individual servo systems and their integration with the environment. The objective of these approaches is to empower the robot system to carry out predetermined actions in an anticipated state. Several control objectives are known to improve the compliance of traditional manipulation. These objectives include position control [13], force control [14], hybrid position/force control [15] and visual servoing control [16]. Traditional approaches were utilized in previous years of the RRC and have achieved success [17]. However, these methods often require extensive manual tailoring, which limits their generalizability and robustness.

Data-driven robotic manipulation: This aims to facilitate robot movements through the collection and analysis of extensive data. By leveraging machine learning technologies, its objective is to enable robots to autonomously learn and refine their manipulation skills. Deep RL [18] has emerged as a popular data-driven approach in the past decade, showcasing remarkable achievements in various complex robotic control domains [1], [2]. Notably, it was employed in previous RRC solutions [19], outperforming traditional approaches and showcasing impressive results.

B. Offline policy learning

Deep RL often needs many environment interactions per training, making it impractical in many real-world cases given multiple attempts are usually required. A promising solution is using a pre-collected dataset to shift policy learning to an offline paradigm. This research mainly encompasses two methods: offline imitation learning (IL) and offline RL.

Offline IL: BC is a straightforward approach to imitation learning (IL) [7]. It seeks to discover a policy that replicates the behavior employed to accomplish a specific task. Typically, the desired behavior for cloning is obtained from an expert source, such as a human [20], or a proficient scripted agent [21]. BC can be seen as a form of supervised regression, as it learns a policy that maps states from the dataset to the corresponding actions. BC demonstrates high efficiency when trained with high-quality expert data, leading to outstanding performance of the trained agent [22]. In comparison to more complex IL methods like GAIL [23] and inverse RL [24], BC generally achieves superior performance. But standard BC, as a form of supervised learning, has certain limitations. Firstly, to avoid regression ambiguity, the action conditioned on a state should be drawn from a unimodal distribution, or the target action mode must constitute the majority proportion in the dataset [25]. Secondly, a reasonably large training dataset is necessary to mitigate the covariate shift issue [26], which refers to the compounded error arising from unseen data during deployment. Lastly, BC's performance is typically constrained by the capabilities of the demonstrator [27].

Offline RL: This is similar to standard online RL paradigms [28], the objective is to discover a policy that maximizes the expected sum of discounted rewards. While most off-policy RL algorithms can be applied offline, the lack of exploration leads to a mismatch between the actual training batch and the expected state-action visitation under the current policy. Consequently, during the policy evaluation phase, state-action pairs that are not present in the dataset would be inaccurately estimated; during the policy improvement stage, the policy may learn to overestimate out-of-distribution (OOD) actions [21]. The OOD issue often results in subpar performance of policies learned through classical off-policy RL algorithms in pure offline settings. To address this challenge, various methods have been proposed, including policy regularization [5], [6], [29] and the use of conservative value estimates [30], [31]. These approaches aim to minimize the discrepancy between the learned policy and the policy employed by the agent responsible for generating the dataset.

Our method can be benchmarked against [32]–[34], which are generally filter-based BC algorithms and taxonomized as *imitation learning* in the offline RL survey paper in [35]; therefore, our method falls into the same taxonomy. These baseline algorithms involve estimating advantage functions using available rewards and utilizing the learn function to perform weighted regression for BC, either with hard or soft weights. In contrast to other algorithms, our algorithm does not use reward signals to update policies directly. Instead, we employ a classifier to identify expert-like behaviors and then use supervised BC to mimic these behaviors. As a result, our algorithm outperforms the state-of-the-art offline RL algorithms in all dexterous manipulation tasks in the RRC III competition.

III. METHODS

A. Background

The problem of offline policy learning can be formulated within the context of a Markov decision process (MDP), denoted as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma)$. In this formulation, $\mathcal{S}$ represents the state space, $\mathcal{A}$ corresponds to the action space, $\mathcal{R}$ is the reward function, $\mathcal{P}$ represents the dynamics of the environment, and γ denotes the discount factor. In each time step t, the agent observes a state $s_t \in \mathcal{S}$ and selects an action $a_t \in \mathcal{A}$ based on a policy $\pi(a_t \mid s_t)$. After executing the chosen action in the environment, the agent receives a reward $r_t = \mathcal{R}(s_t, a_t)$, and the environment transitions to a new state s_{t+1}. In the context of a RRC III, the dataset provided for each task can be represented as $\mathcal{D} = (s_t, a_t, r_t, s_{t+1})_{t=1\ldots i}$, where i denotes the number of time steps in the episode.

B. Filtering expert data from mixed datasets

We propose a learning-based binary classifier to filter expert data from the mixed datasets; the classifier can learn the behavior pattern of the expert by comparing the expert's behaviour with that of a non-expert. The data filtering process is semi-supervised, where the trained classifier is continuously used to separate more training samples from the mixed dataset for the next training iteration. This continues until the composition of the training samples converges. The algorithm can be decomposed into the following steps:

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

1) Generate training samples: First, we manually label a small section of data from the mixed dataset, assuming it contains expert demonstrations; and this subset serves as positive training examples for the binary classifier. As previously mentioned, the highest-rewarding trajectories in the mixed dataset are highly likely to have been generated by an expert. Therefore, in this context, we consider the top 10% most rewarded episodes as the expert data. The determination of this percentage value also involved referencing Fig. 2(b), which shows a decreasing trajectory amount trend for weaker policies in the mixed dataset as the episodic reward increases. Intuitively, fewer episodes from weaker policies fall in the highest cumulative reward range. Our tests show that selecting this percentage can vary between 5% to 15% without significant impact, converging to a similar outcome.

On the contrary, it is necessary to have negative samples that contrast with the positive expert samples in order to generate supervised signals to train the classifier. However, using reward alone is not an effective method for directly identifying valid negative samples from the mixed dataset, particularly for the *lift* task. Fig. 2(b) illustrates that a certain portion of the expert data receives lower scores, resembling a weaker policy. This can be attributed to the complexity and randomness of the environment, which can lead to failures even with strong policies. Thus, we propose an alternative approach, in which we combine states and actions from different sources. The generated negative sample set involves: 1) Pairing states from the positive sample subset with randomly generated actions; 2) Pairing randomly generated states with actions from the positive sample subset; and 3) Pairing randomly-generated states with randomly-generated actions. The new state-action pairs are unlikely to resemble the expert's behavior and this mixture of different sources helps introduce novel and diverse examples into the classifier training process. Actually, any state-action pair that deviates from expert behavior and is generated using a different approach than those we proposed could work here.

2) Train the classifier: We feed state-action pairs into the neural network to train the classifier. It is worth noting that within the RRC III setting, there is a considerable dimensional discrepancy between the state and action spaces. Therefore, we initially deploy a fully-connected encoder to condense the state dimension. Subsequently, we concatenate the action with the reduced-dimension state tensor. This combined tensor is then inputted into a fully-connected predictor. The network's final layer uses a softmax function, producing probabilistic outputs, with binary cross-entropy as the loss function.

3) Employ the classifier: We feed the state-action pairs of each time step of an episode in the mixed dataset into a classifier to estimate the probability of that state-action pair being collected by the expert. The RRC III robotic data is organized into episodes, each consisting of a sequence of consecutive time steps. We average the probabilities from each timestep in an episode for a comprehensive prediction. This average is termed the confidence (***conf***) subsequently.

We set a confidence threshold, $\boldsymbol{\theta_{conf}}$, to binarize each episode's ***conf***. This binary value indicates if the episode is expert-derived, used for further filter classifier training. Once the membership of filtered positive samples converges, it serves as the training set for BC. We regard $\boldsymbol{\theta_{conf}}$ as a tunable hyperparameter and optimize it by observing the evaluated performance of the policy trained using BC. Our experiments indicate that the optimal value of $\boldsymbol{\theta_{conf}}$ is 0.95 for the *lift*/mixed dataset and 0.96 for the *push*/mixed dataset.

C. Symmetry-based data augmentation

As previously discussed, one of the main drawbacks of BC is its susceptibility to the covariate shift issue due to being a supervised learning method; note, this issue could be mitigated by increasing the quantity or diversity of the training data [36], [37]. Hence, we used data augmentation techniques during the RRC III competition to further improve the performance of BC. In prior work, data has been augmented for offline policy learning by editing the state vector to improve the robustness of the learned policy, using techniques such as adding noise, scaling, dimensional dropout, state-switch, state mix-up, and adversarial transformation [38]; however, these approaches have a limited ability to diversify the dataset, as these operations are anchored around the same state-action pair. We use the robot arena's rotational symmetry to create new state-action pairs by transforming state and action components. This adds variations to the dataset and enriches training sample diversity.

1) Rotational transformation: The three fingers of the robot are evenly spaced around the center of the circular arena, with an angle difference between two adjacent fingers of 120°. Since the structure of each finger is theoretically identical, the correctness of the data, including the states of the object and robot, should remain unchanged after rotating clockwise or counterclockwise around the central point of the arena by 120°.

In effect, we will spatially rotate the entire experiment (robot, arena, and object) around the centre of the arena by integer multiples of 120° in the world frame, but the indexes of each finger do not move and still reference the same location in the world frame. Since different transformations are required to perform this rotation, depending on whether we are transforming a robot state or a spatial pose of the object, we split the state vector into robot and object state subvectors ($\mathbf{s} = [\mathbf{s}^{robot^T}, \mathbf{s}^{obj^T}]^T$). Simultaneously, we perform a permutation on the robot state subvector $\mathbf{s}^{robot}$ and its associated action vector $\mathbf{a}$; and we perform spatial rotation on the object state subvector $\mathbf{s}^{obj}$:

$$\mathbf{s}^{robot}_{aug}(\alpha) = \mathbf{s}^{robot}((\alpha + k \cdot 120^\circ) \ \% \ 360^\circ) \tag{1}$$

$$\mathbf{a}_{aug}(\alpha) = \mathbf{a}((\alpha + k \cdot 120^\circ) \ \% \ 360^\circ) \tag{2}$$

$$\begin{bmatrix} s^{obj}_{x,aug} \\ s^{obj}_{y,aug} \end{bmatrix} = \begin{bmatrix} \cos(k \cdot 120^\circ) & -\sin(k \cdot 120^\circ) \\ \sin(k \cdot 120^\circ) & \cos(k \cdot 120^\circ) \end{bmatrix} \cdot \begin{bmatrix} s^{obj}_{x} \\ s^{obj}_{y} \end{bmatrix}, \tag{3}$$

where $\alpha \in (0^\circ, 120^\circ, 240^\circ)$ and $k \in (0, 1, 2)$, $\mathbf{s}^{robot}(\alpha)$ and $\mathbf{a}(\alpha)$ represents the state and action subvectors for the finger of the robot located at an angle of α degrees, and s^{obj}_x and s^{obj}_y represent the x and y coordinates of the object. For example, the entire experiment will be spatially rotated 120° around the z of the world frame when $k = 1$ (anticlockwise, looking top-down). The z coordinate of the object remains unchanged. The data after these rotational transformations are concatenated with the original dataset to form a larger augmented dataset.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Table II: The evaluated scores comparing our method to comparative methods, where task-specific scores are given as mean ± SD. In the evaluation stage, the policy is randomly deployed on one of six available physical robots, and the goal (position and/or orientation of a cube) is randomly generated. The evaluation for each task lasts 15 episodes.

	BC	10% BC	50% BC	CRR	TD3+BC	PLAS	C-Aug	Ablation 1	Ablation 2	**Ours**
Push/expert	626 ± 101	- ± -	- ± -	611 ± 127	623 ± 99	618 ± 92	619 ± 74	626 ± 101	641 ± 47	**662 ± 87**
Push/mixed	497 ± 88	541 ± 90	623 ± 43	599 ± 93	604 ± 111	595 ± 121	532 ± 81	618 ± 83	627 ± 88	**636 ± 126**
Lift/expert	928 ± 205	- ± -	- ± -	792 ± 227	852 ± 401	874 ± 359	861 ± 187	928 ± 205	1077 ± 199	**1130 ± 193**
Lift/mixed	489 ± 282	503 ± 117	492 ± 219	606 ± 312	698 ± 362	707 ± 350	495 ± 214	917 ± 237	980 ± 280	**1038 ± 305**
Average	635 ± 169	- ± -	- ± -	652 ± 190	694 ± 243	699 ± 231	622 ± 139	772 ± 157	831 ± 154	**867 ± 178**

2) Theory-to-real transfer: Similar to the well-known sim-to-real gap problem (which describes how policies learned in ideal simulations often underperform in the real-world due to lack of consideration of variances in physical properties), trying to leverage spatial symmetry to perform data augmentation can also fall foul of similar assumptions of ideal physical properties. For example, we might assume that all three fingers of the robot are identical in every way, but we know that this is unlikely to be true, and they might vary in ways such as having different frictional properties, the motors generating different torques in response to a given command, or the sensitivity/calibration of the tactile sensors differing across fingers.

Therefore, we began by training a BC policy on the augmented dataset, utilizing a higher learning rate and longer training length to establish a more general policy. Subsequently, we fine-tuned this policy using the original, non-augmented dataset with a lower learning rate and reduced training length. This process ensures that the policy deployed aligns more closely with the data distribution observed from the real robots.

D. Method summary

In our final submission, we utilize BC as the control algorithm across all datasets. The training objective of BC:

$$\min_{\pi} \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{D}_{expert}} [-\log \pi(a_t \mid s_t)]\,. \tag{4}$$

For the *lift*/mixed and *push*/mixed datasets, we first employed the data filtering method to separate out the expert data (we assume). Then, we performed augmentation on these expert data. Finally, we trained the BC policy using the method proposed in Sec.III-C2. For two expert datasets, the method was basically the same, except the filtering step was omitted.

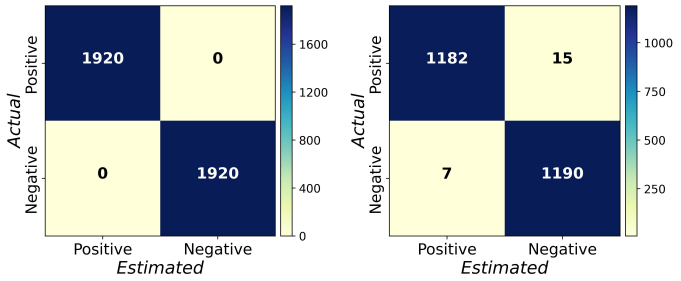
IV. Experiments and Results

This section will showcase the filtering accuracy of our method in diffrentiating the expert data from mixed datasets. Furthermore, we will emphasize the advantages of incorporating our filtering and augmentation techniques in offline policy learning and compare them with other baseline methods.

A. Classifier filtering accuracy

We reached out to the organizers of RRC III to inquire about the composition of the mixed datasets after the competition, establishing a reliable ground truth for evaluating our classifier. The evaluation results are displayed in Fig. 3.

Our filtering method achieves exceptional accuracy, even when applied to the *lift*/mixed dataset, where the reward-based performances of expert and weaker policies are notably similar



(a) *Push*/mixed dataset (b) *Lift*/mixed dataset

Figure 3: Confusion matrices showing the performance of our filtering method in recognising expert-generated episodes among a larger mixed dataset from the RRC III.

on many episodes. Remarkably, our method achieves 100% accuracy on the comparatively simpler *push*/mixed dataset. These results substantiate the reliability and effectiveness of our method for practical applications.

B. Comparative performance evaluation

1) Baseline algorithms: We compare the performance of our approach with other relevant baseline algorithms, including:

- **CRR** [32]: CRR learns the Q-function to construct the advantage function:

$$\hat{A}(s_t, a_t) = Q_\theta(s_t, a_t) - \frac{1}{m}\sum_{j=1}^{m} Q_\theta(s_t, a^j), \tag{5}$$

  where $a^j \sim \pi(\cdot \mid s_t)$, $\pi(\cdot \mid s_t)$ is the learned policy, and $Q_\theta(\cdot,\cdot)$ refers to the learned critic. The advantage of a specific state-action pair relative to the dataset can be obtained through this function and then be used to weight the importance of the training samples for BC. This method shares similarities with our approach, as both aim to partially focus the BC on the more promising transitions in the dataset.
- **PLAS** [5]: A variational autoencoder (VAE) is trained on the raw dataset to capture the underlying distribution of actions. In the policy improvement stage, the policy function outputs a latent action, which is subsequently fed into the VAE's decoder to produce an action that aligns with the distribution of actions in the raw dataset. This method is employed to mitigate the generation of OOD actions.
- **TD3+BC** [6]: This method introduces BC into the TD3 algorithm [39] as the regularization term:

$$\pi = \mathop{argmax}_{\pi} \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{D}} [\lambda Q(s, \pi(s)) - (\pi(s) - a)^2], \tag{6}$$

  It can guide the policy to prioritize actions present in the dataset, effectively avoiding OOD actions.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Table III: The final official ranking of RRC III competition. The organisers pre-defined several goals in the official evaluation protocol and evenly distributed them among the six robots. Our team is named *excludedrice*. *superiordinosaur* and *jealousjaguar* shared joint third place. The baseline score is the mean episodic return of the dataset. More details refer to the official leaderboard: https://real-robot-challenge.com/leaderboard

	Baseline score	660	429	1064	851	751
#	Team name	Push/expert	Push/mixed	Lift/expert	Lift/mixed	Average
1	**excludedrice**	624 ± 144	**635 ± 137**	**956 ± 431**	**923 ± 442**	**784**
2	decimalcurlew	**639 ± 112**	613 ± 134	841 ± 415	717 ± 383	703
3	superiordinosaur	618 ± 143	575 ± 191	856 ± 452	571 ± 346	655
	jealousjaguar	**639 ± 121**	561 ± 178	855 ± 392	506 ± 348	640

We implemented these baseline algorithms using the d3rlpy [40] library, using the recommended hyperparameters.

2) Baseline augmentation method: We secondly compare our augmentation method with the classical approach presented in [38], which involves the addition of Gaussian noise to the state: $s_{aug} = s + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 3 \times 10^{-4})$. This technique can simulate real-world variations and uncertainties; as a result, it should enhance the adaptability and robustness of the policy trained by BC, in theory. For conciseness, we will refer to it as *C-aug* in the subsequent text.

3) Ablated algorithms: We finally conduct an ablation study that dissects the contribution of each component to the overall performance of BC, including:

- **Ablation 1**: Only the data from the raw dataset is utilized for training the BC model here, while augmented datasets are not used. Specifically, for mixed datasets, we employ filtered subsets for BC model training without augmentation. For expert datasets, we directly use the raw datasets for BC training without augmentation as well.
- **Ablation 2**: BC is exclusively trained on the augmented dataset without subsequent fine-tuning. Specifically, for the mixed datasets, we augment the filtered subsets for BC training. For the expert datasets, we directly augment the raw datasets for BC training. Both experiments are conducted without additionally fine-tuning on the raw data.

4) Results and analysis: We present the outcomes of the baselines, ablation studies and our method in TABLE II. It is evident that CRR, PLAS, and TD3+BC exhibit suboptimal performance on RRC III tasks, particularly on the demanding *lift*/mixed dataset. We propose a hypothesis that offline learning alone lacks the essential capability to accurately approximate the action-value function required for the RRC III environment and tasks due to its complexity.

Traditional data augmentation techniques failed to enhance the performance of BC in the context of RRC III settings (see results of C-aug in TABLE II). Unexpectedly, it even detracts the overall performance of BC. This decline may result from the inherent complex state-action relationship. Hence, the introduction of noise via augmentation appears to overtax the neural network's learning process instead of facilitating it.

From the two sets of ablation experiments conducted, it is also evident that our data augmentation approach effectively enhances the performance of BC in addressing the four manipulation tasks. Moreover, additional fine-tuning of BC using raw data subsequently yields further improvement in the model's efficacy. Importantly, these optimization and fine-tuning strategies not only progressively improve performance, but also manage to do so without unduly increasing the model's complexity or computational burden during training; And it do not incur any additional cost during deployment.

C. Extending to D4RL datasets

In this section, we extend our filtering method to four D4RL datasets [41]: "halfcheetah-medium-expert-v2", "hopper-medium-expert-v2", "walker2d-medium-expert-v2" and "ant-medium-expert-v2". These datasets, similar to the RRC III, consist of a mix of data generated by expert and weaker policies. We benchmark our approach against the baseline algorithms detailed in IV-B1, demonstrating the generalizability and effectiveness of our method. Similar to the approach we employed using RRC III, as detailed in III-B1, we initiate the filter with the top 10% of the highest rewarded episodes. Through experimentation, we have determined that setting $\boldsymbol{\theta_{conf}}$ to 0.95 yields satisfactory performance.

As illustrated in Fig. 4, our method consistently outperforms all other algorithms. When compared to CRR and PLAS, our approach exhibits superior performance. Moreover, when evaluated against the relatively robust TD3+BC algorithm, our method exhibits notably quicker convergence and more stable policy checkpoints.

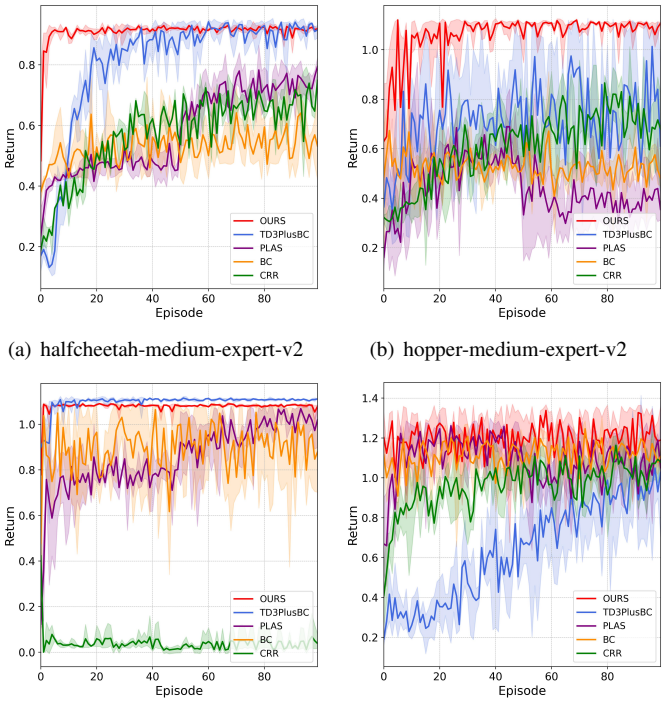
V. Discussion

Overall, our methods significantly enhance the performance of the vanilla BC algorithm, allowing it to achieve expert-level performance on both mixed datasets. In contrast, complex offline RL algorithms struggle to learn an effective policy, particularly when faced with the high-complexity *lift*/mixed dataset[2]. TABLE III presents the final official evaluated scores of the RRC III competition, where our team achieved the notable accomplishment of surpassing the baseline in the *lift*/mixed task. The control policy employed by team *decimalcurlew* is TD3+BC [6]. They utilized a method known as spatial smoothing to handle noisy data sourced from the physical environment. Similarly, team *superiordinosaur* adopted a BC-based policy learning approach, incorporating feature selection to eliminate superfluous features. Team *jealousjaguar* implemented IQL [29] for their control policy, augmenting their data with conventional techniques. Their reports are accessible on the leaderboard page of the RRC III website.

Indeed, our filtering method has certain limitations, including cases where only a few expert demonstrations are available, potentially resulting in increased computational expense in the semi-supervised classification scheme, or cases where it is unable to obtain the initial positive data subset at all.

[2]A demo video can be found at https://youtu.be/-segGw0o8XM

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.



(a) halfcheetah-medium-expert-v2 (b) hopper-medium-expert-v2

(c) walker2d-medium-expert-v2 (d) ant-medium-expert-v2

Figure 4: Score curves over 100 checkpoints compare our method against baseline algorithms on the D4RL datasets. Each checkpoint is evaluated over 20 episodes to get an average value and then normalized using $s_{norm} = (s - s_r)/(s_e - s_r)$, where s_r and s_e denote the performance benchmarks for a random policy and an expert policy, respectively, as specified in [41]. We used 3 different random seeds for the tests. Solid curves show the average scores over seeds, and shaded areas indicate value range across seeds.

Secondly, our method intentionally disregards non-expert data; it is indeed valid that non-expert demonstrations may still hold value. Therefore, in future work, we may explore soft-weighted filtering methods rather than the binary-weighted approach presented in this paper, or transform non-expert data to better resemble expert data [42]. Thridly, the labor-intensive process of tuning the confidence threshold $\boldsymbol{\theta_{conf}}$ in our technique was noted. In fact, another option could involve tuning $\boldsymbol{\theta_{conf}}$ in a simulator (if available), rather than on a real robot, especially if the evaluation on the real robot is costly. Since $\boldsymbol{\theta_{conf}}$ is not an overly complex hyperparameter, the trends observed in the simulator can generally mirror the impact of $\boldsymbol{\theta_{conf}}$ on a real robot. For the tasks reported in the paper, it was not a particularly costly activity to evaluate a small number candidate policies (no more than 5 candidate policies for each dataset/task, obtained by varying the value of $\boldsymbol{\theta_{conf}}$). The sim-to-real transfer risk, when tuning θ_{conf} in a simulator, can be partly mitigated by using this final online evaluation step to tune θ_{conf}, but performing the major task of learning a policy for a given θ_{conf} value in an offline manner. This labor-intensive process also prompt us to consider the possibility of using an adaptive $\boldsymbol{\theta_{conf}}$. Lastly, since our approach hinges on supervised classification to perform the filtering operation, it presents us with the opportunity to further improve its effectiveness using strategies such as ensemble techniques. In the process of preparing this paper for publication, we conducted further investigations and analyses of our filtering method to address the concerns mentioned above, as reported in [43]. Our proposed filtering BC method is still in its early stages of development, and we plan to continue refining and expanding this method in future research, as it holds untapped potential. This paper serves to describe the foundational version of this methodology, which was used to win the RRC III competition, and which is the first a series of papers that will explore this filtering concept for offline RL. Importantly, the primary objective of this paper is to present a comprehensive pipeline for addressing robotic manipulation tasks through offline learning. This pipeline includes data filtering, data augmentation, and theory-to-real-world transfer learning; all of which played a significant role in the success of the proposed method.

In the physical world, a multitude of systems exhibit spatial symmetries and invariances. For instance, consider the case of bimanual robots, which typically possess two arms and often exhibit left-right mirror symmetry, where a simple mathematical augmentation transformations can efficiently enhance the dataset to achieve an improved policy. When combined with our proposed theory-to-real transfer training method, these spatial symmetries can prove highly valuable in the pursuit of more generalized policies. These policies can subsequently undergo fine-tuning using physically-consistent real-world data, enhancing their real-world applicability.

Prior studies suggest that decreasing the learning rate of BC leads to improved policies for physical robots [44]. To verify that the performance improvement from our "train-and-tune" approach was not merely due to a reduced learning rate in the tuning phase, we carried out an extensive investigation of this hypothesis. The findings positively confirmed the efficacy of our "train-and-tune" method. Due to space constraints, we have placed the detailed results and discussion on our website (see ***Abstract***). We confirm that the website is exclusively associated with this paper and will be permanently accessible.

VI. Conclusion

To summarize, this paper comprehensively describes and evaluates our solution for the RRC III. We propose an effective strategy for recognizing behaviors produced by a specific expert policy. This technique enables a learning algorithm to exclude behaviors associated with lower-skilled agents, thereby enhancing the effectiveness of the policy learned. In addition, we introduce a geometric data augmentation method capable of significantly boosting the performance of the policy; coupled with our theory-to-real transfer, our training approach yields effective real-world results.

ACKNOWLEDGMENT

We gratefully acknowledge the Max Planck Institute for Intelligent Systems in Tübingen/Stuttgart, Germany, for their invaluable support and exceptional organization of the RRC III. We sincerely appreciate the reviewers for their invaluable suggestions and insights that improved this paper. Finally, we extend our heartfelt gratitude to Dr Kevin McGuinness for his invaluable contributions and expertise to this research. It is with deep sorrow that we note he did not live to see the completion of

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

this work. His exceptional insights and unwavering dedication will forever be treasured in our memories.

REFERENCES

[1] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26):eaau5872, 2019.

[2] Herke Van Hoof, Tucker Hermans, Gerhard Neumann, and Jan Peters. Learning robot in-hand manipulation with tactile features. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 121–127. IEEE, 2015.

[3] Qiang Wang, Francisco Roldan Sanchez, Robert McCarthy, David Cordova Bulens, Kevin McGuinness, Noel O'Connor, Manuel Wüthrich, Felix Widmaier, Stefan Bauer, and Stephen J Redmond. Dexterous robotic manipulation using deep reinforcement learning and knowledge transfer for complex sparse reward-based tasks. *Expert Systems*, page e13205, 2022.

[4] 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*, 2022.

[5] Wenxuan Zhou, Sujay Bajracharya, and David Held. Plas: Latent action space for offline reinforcement learning. In *Conference on Robot Learning*, pages 1719–1735. PMLR, 2021.

[6] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:20132–20145, 2021.

[7] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pages 103–129, 1995.

[8] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[9] Vikash Kumar. *Manipulators and Manipulation in high dimensional spaces*. PhD thesis, University of Washington, Seattle, 2016.

[10] S Saravana Perumaal and N Jawahar. Automated trajectory planner of industrial robot for pick-and-place task. *International Journal of Advanced Robotic Systems*, 10(2):100, 2013.

[11] F Basile, F Caccavale, P Chiacchio, J Coppola, and C Curatella. Task-oriented motion planning for multi-arm robotic systems. *Robotics and Computer-Integrated Manufacturing*, 28(5):569–582, 2012.

[12] Cota Nabeshima, Yasuo Kuniyoshi, and Max Lungarella. Adaptive body schema for robotic tool-use. *Advanced Robotics*, 20(10):1105–1126, 2006.

[13] Prabin Kumar Padhy, Takeshi Sasaki, Sousuke Nakamura, and Hideki Hashimoto. Modeling and position control of mobile robot. In *2010 11th IEEE International Workshop on Advanced Motion Control (AMC)*, pages 100–105. IEEE, 2010.

[14] Qiang Wang, Pablo Martinez Ulloa, Robert Burke, David Cordova Bulens, and Stephen J. Redmond. Robust learning-based incipient slip detection using the papillarray optical tactile sensor for improved robotic gripping. *arXiv preprint arXiv:2307.04011*, 2023.

[15] MH Reibert. Hybrid position/force control of manipulators. *ASME, J. of Dynamic Systems, Measurement, and Control*, 103:2–12, 1981.

[16] Darius Burschka and Gregory Hager. Vision-based control of mobile robots. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, volume 2, pages 1707–1713. IEEE, 2001.

[17] Takuma Yoneda, Charles Schaff, Takahiro Maeda, and Matthew Walter. Grasp and motion planning for dexterous manipulation for the real robot challenge. *arXiv preprint arXiv:2101.02842*, 2021.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[19] Robert McCarthy, Francisco Roldan Sanchez, Qiang Wang, David Cordova Bulens, Kevin McGuinness, Noel O'Connor, and Stephen J Redmond. Solving the real robot challenge using deep reinforcement learning. *arXiv preprint arXiv:2109.15233*, 2021.

[20] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

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

[22] Josh Merel, Leonard Hasenclever, Alexandre Galashov, Arun Ahuja, Vu Pham, Greg Wayne, Yee Whye Teh, and Nicolas Heess. Neural probabilistic motor primitives for humanoid control. *arXiv preprint arXiv:1811.11711*, 2018.

[23] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in Neural Information Processing Systems*, 29, 2016.

[24] Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[25] S. Levine. Supervised Learning of Behaviors, 2022. (Accessed 2022, Oct 10).

[26] Jonathan Chang, Masatoshi Uehara, Dhruv Sreenivas, Rahul Kidambi, and Wen Sun. Mitigating covariate shift in imitation learning via offline data with partial coverage. *Advances in Neural Information Processing Systems*, 34:965–979, 2021.

[27] Daniel S Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on Robot Learning*, pages 330–359. PMLR, 2020.

[28] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[29] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[30] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

[31] Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in Neural Information Processing Systems*, 34:7436–7447, 2021.

[32] Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33:7768–7778, 2020.

[33] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[34] Gerhard Neumann and Jan Peters. Fitted q-iteration by advantage weighted regression. *Advances in neural information processing systems*, 21, 2008.

[35] Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[36] Mingming Gong, Kun Zhang, Tongliang Liu, Dacheng Tao, Clark Glymour, and Bernhard Schölkopf. Domain adaptation with conditional transferable components. In *International Conference on Machine Learning*, pages 2839–2848. PMLR, 2016.

[37] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

[38] Samarth Sinha, Ajay Mandlekar, and Animesh Garg. S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics. In *Conference on Robot Learning*, pages 907–917. PMLR, 2022.

[39] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

[40] Takuma Seno and Michita Imai. d3rlpy: An offline deep reinforcement learning library. *The Journal of Machine Learning Research*, 23(1):14205–14224, 2022.

[41] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[42] Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. Rvs: What is essential for offline rl via supervised learning? *arXiv preprint arXiv:2112.10751*, 2021.

[43] Qiang Wang, Robert McCarthy, David Cordova Bulens, Kevin McGuinness, Noel E O'Connor, Francisco Roldan Sanchez, Nico Gürtler, Felix Widmaier, and Stephen J Redmond. Improving behavioural cloning with positive unlabeled learning. *arXiv preprint arXiv:2301.11734v2*, 2023.

[44] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.