Train Offline, Test Online: A Real Robot Learning Benchmark

Anonymous Author(s)
Affiliation
Address
email

Abstract: Three challenges limit the progress of robot learning research: robots are expensive (few labs can participate), everyone uses different robots (findings do not generalize across labs), and we lack internet-scale robotics data. We take on these challenges via a new benchmark: Train Offline, Test Online (TOTO). TOTO provides remote users access to shared robots for evaluating methods on common tasks and an open-source dataset of these tasks for offline training. Its manipulation task suite requires challenging generalization to unseen objects, positions, and lighting. We present initial results on TOTO comparing five pretrained visual representations and four offline policy learning baselines, remotely contributed by five institutions. The real promise of TOTO, however, lies in the future: we release the benchmark for additional submissions from any user, enabling easy, direct comparison to several methods without the need to obtain hardware or collect data.

Keywords: Benchmarking, Robot Learning, Datasets

1 Introduction

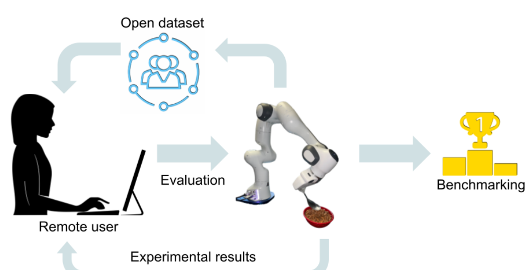

Figure 1: **Train Offline, Test Online**: Our benchmark lets remote users test offline learning methods on shared robots.

One of the biggest drivers of success in machine learning research is arguably the availability of benchmarks. From GLUE [1] in natural language processing to ImageNet [2] in computer vision, benchmarks have helped identify fundamental advances in many areas. On the other hand, robotics as a field struggles to establish common benchmarks due to the physical nature of evaluation. The experimental conditions, objects of interest, and even hardware vary across labs, often making algorithms sensitive to implementation details. Finally, the difficulties of purchasing, building, and installing hardware and software infrastructure make it challenging for newcomers to contribute to the field.

For robotics research to advance, we clearly need a common way to evaluate and benchmark different algorithms. A good benchmark will not only be fair to all algorithms but also have low participation barrier: setup to evaluation time should be as low as possible. Efforts like YCB [3] and RB2 [4] aim to standardize objects and tasks, but the onus of setting up infrastructure still lies with each lab. A simple way to overcome this is the use of a common physical evaluation site, as the Amazon Picking Challenge [5] and DARPA Robotics Challenges [6, 7, 8] have. However, the barrier is still high since participants must set up their own training infrastructure. Both of the above frameworks leave the method development phase unspecified and struggle to provide apples-to-apples comparisons.

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

Many robot learning algorithms do online training, where a policy is learned concurrently with data collection. One way to standardize online training is with simulation [9, 10, 11, 12]. While simulation mitigates issues with variation across labs, the findings from simulated benchmarks may not transfer to the real world. On the other hand, if we conduct online training in the real world, comparison across labs becomes difficult due to physical differences. In recent years, larger datasets have surfaced in robotics [13, 14, 15], and with them the rise of offline training algorithms. From imitation learning to offline RL, these algorithms can be trained on the same data and tested on common hardware.

Inspired by this observation, we propose a new robotics benchmark: **TOTO (Train Offline, Test Online)**. TOTO has two key components: (a) a large-scale offline manipulation dataset to train imitation learning and offline RL algorithms; (b) a shared hardware setup where users can evaluate their methods now and going forward. Because all participants train using the same publicly-released dataset and evaluate on shared hardware, the benchmark provides a fair apples-apples comparison.

TOTO paves a path forward for robot learning by lowering the entry barrier: when designing a new method, a researcher can train their policy on our dataset, evaluate it on our hardware, and directly compare it to the existing baselines for our benchmark. TOTO means no more time devoted to setting up hardware, collecting data, or tuning baselines. In this paper, we lay out the TODO design and present initial methods contributed by benchmark beta testers across the country. Our results show that our benchmark is challenging yet possible, providing room for growth as TOTO users iterate.

2 The TOTO Benchmark

Our benchmark focuses on manipulation due to lack of benchmarking in this area. The robots (Appendix Section 5.2) are set in environments that enable a set of benchmark manipulation tasks described in Section 2.1. We collect an initial dataset on these tasks, detailed in Section 2.2. Finally, in Section 2.3, we present the evaluation protocol for all policies contributed to our benchmark.

2.1 Tasks

We use two everyday manipulation tasks: pouring and scooping, similar to those introduced in prior work [4, 16]. Example observations are shown in Fig. 4 of Appendix Section 5.3. To see the original task designs, please refer to RB2: `https://rb2.info`. Our tasks differ from those in RB2 in a few ways. We randomize the robot's pose at the start of each episode, apply more noise to target object locations, and use a variety of objects for each task based on availability. Lastly, we do not normalize the reward: the reward is the weight in grams of the material successfully scooped or poured.

Scooping. The training set includes all combinations of three target bowls, three materials, and six target bowl locations (front left, front center, front right, back left, back center, and back right).

Pouring. The training set includes all combinations of four target cups, two materials, and six target cup locations (same locations as scooping). The cup in the robot gripper is the same in all experiments (clear plastic, enabling better perception of the material remaining in the cup).

2.2 Dataset

A key pillar of our benchmark is the release of a manipulation dataset, which includes 1895 scooping trajectories and 1003 pouring trajectories collected with a mix of teleoperation, behavior cloning rollouts, and replay with noise. Each trajectory includes RGB-D video, actions (joint angle targets), joint states (joint angles), and task metrics (rewards). More details are in Appendix Section 5.3.

2.3 Evaluation Protocol

We evaluate using a variety of test settings. We use two unseen objects (bowls and cups) and one unseen material (mixed nuts for scooping and Starburst candies for pouring). We evaluate three object locations seen during training (front left, front center, front right) and three unseen locations. We evaluate three training seeds per method. The robot starts each trajectory at random pose based on the random seed. 2 objects, 1 material, 3 locations, and 3 seeds means that methods are evaluated across 18 trials each for train and test locations. We report mean and variance of these 18 trials.

3 Baselines

We highlight TOTO's importance with two sets of experiments: (a) what is a good visual representation for manipulation? and (b) what is a good offline algorithm for policy learning? To test TOTO infrastructure, we have solicited baseline implementations for both experiments from several labs.

3.1 Visual Representation Baselines

A core unanswered question, due to the lack of benchmarking, is what is a good visual representation for manipulation? Is ResNet trained on ImageNet great or do self-supervised approaches outperform supervised models? We evaluate five visual representations provided by TOTO users from multiple labs. Two are trained on our data (in-domain) and three are generically pretrained.

`Resnet50` refers to the model trained with supervised learning on ImageNet [17].

Momentum Contrast (`MoCo`) is trained on ImageNet [18], which we call `MoCo (Generic)`. This is distinguished from `MoCo (In-Domain)` which is trained on our data with crop augmentations [19].

Reusable Representations for Robot Manipulation (`R3M`) [20] is trained on Ego4D [21] with time-contrastive learning and video-language alignment. `R3M`, `MoCo`, and `Resnet50` use the 2048-dimensional embedding vector following the fifth convolutional layer.

Bootstrap Your Own Latent (`BYOL`) [22] is a self-supervised representation learning method trained on our dataset. The BYOL representation embedding size is 512.

These representations performed the best among a larger set of vision models on which we ran an initial brief analysis (including offline visualizations and BC rollouts). Additional representations that performed less well included CLIP [23] and a third-layer MoCo model (instead of fifth-layer).

3.2 Policy Learning Baselines

Remote users have contributed the policy learning baselines detailed below. These methods span the spectrum from nearest neighbor querying to BC to offline reinforcement learning (RL). They were selected according to each TOTO contributor's expertise with approach coverage in mind. All methods pass RGB image observations through frozen vision representations before passing them to a policy. BC, IQL, and DT use the `MoCo (In-Domain)` model, while VINN uses BYOL.

Behavior Cloning (`BC`) learns to mirror actions in the training data. Closed-loop BC predicts a new action every timestep, while open-loop BC predicts a sequence of actions to execute without re-planning. Our BC baseline is *quasi* open-loop: training trajectories are split into 50-step action sequences, and the policy is trained to predict such a sequence. During evaluation, these 50 actions are executed between each prediction step. We find that this performs better than closed-loop or open-loop alone: closed-loop struggles without history, and open-loop is challenging with our variable-length tasks. We filter out zero-reward trajectories from the training data [24].

Implicit Q-learning (`IQL`) [25] uses the open-source implementation from `d3rlpy` [26]. We concatenate frozen image embeddings with the robot's joint angles as the input state to the model.

Visual Imitation through Nearest Neighbors (`VINN`) [27] is a nearest neighbor policy using an image encoder trained with `BYOL` [22]. BYOL maps the high-dimensional observation space to a low dimension to obtain a robust policy. VINN was originally closed-loop, but in this work we mirror the 50-step quasi open-loop approach used in the BC baseline (described above).

Decision Transformers (`DT`) [24] recasts offline RL as a (conditional) sequence modeling task. It is trained to predict the action in the dataset, but also conditions on the trajectory history and a target return (desired level of performance). We use the Hugging Face DT implementation. DT uses a sub-sampling period of 8 and a history window of 10 frames. For evaluation, the target return prompt is chosen as the mean return from the top 10% of trajectories in the dataset for each task.

4 Experimental Results

Visual Representation Results. Our first experiments compare the vision representations detailed in Section 3.1 combined with BC policies and evaluated according to Section 2.3. The success rates for all representations and tasks are visualized in Fig. 2, and the numerical rewards are presented in Appendix Table 2. Finetuning the MoCo model on our data outperforms the generic version, as expected. `MoCo (In-Domain)` achieves the highest success rate and average reward on both scooping and pouring, followed by `BYOL`, the other in-domain model. The relative performance between models is mostly consistent across scooping and pouring. Resnet50 and `MoCo (Generic)` perform slightly better on pouring than on scooping.

Fig. 2 also visualizes performance differences due to object locations. Locations seen during training perform better, as expected, but performance does not degrade significantly, suggesting that the representations have a generalizable notion of where the target object is. Surprisingly, the two representations trained on our data (`MoCo (In-Domain)` and `BYOL`) perform equally good or even slightly better on unseen locations for scooping.

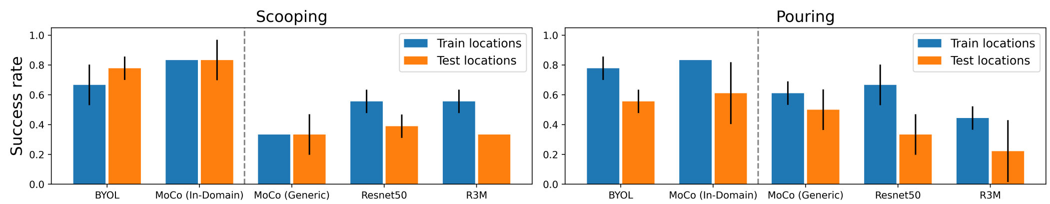

Figure 2: **Vision representation comparison with BC.** Models trained on our data (left of dashed line) perform better than generic ones (right), and object train locations work better than unseen ones.

Policy Learning Results. Fig. 3 visualizes the policy learning comparison (described in 3.2) evaluated on TOTO, and the numerical rewards are in Appendix Table 3. Due to compute constraints, we have 1 and 2 seeds for `DT` and `IQL` respectively. We compensate by duplicating the evaluation of these seeds to keep the number of trials consistent. We find that `VINN` performs best in train locations. We also note that offline-RL approaches (especially IQL) achieve some success unlike in RB2[4]. Our dataset is larger and more diverse than RB2, likely contributing to better offline RL performance.

We found that scooping proves challenging due to a non-markovian aspect: the spoon is above the bowl both before and after scooping. Thus we would expect open-loop methods (`BC`, `VINN`) and those with history (`DT`) to perform better than others. While `BC` and `VINN` achieve competitive performance on scooping, `DT` only achieves moderate success on scooping and does not see any positive rewards on pouring. Meanwhile, `IQL` provides decent performance without history on a non-markovian task.

Comparing the train and test location results for policy learning proves interesting. `VINN` performs the best on train locations but struggles on unseen locations, since it selects actions using the nearest neighbor from the training data. All other methods also experience some level of degradation when moving to unseen locations, leaving one clear direction for method improvement using TOTO.

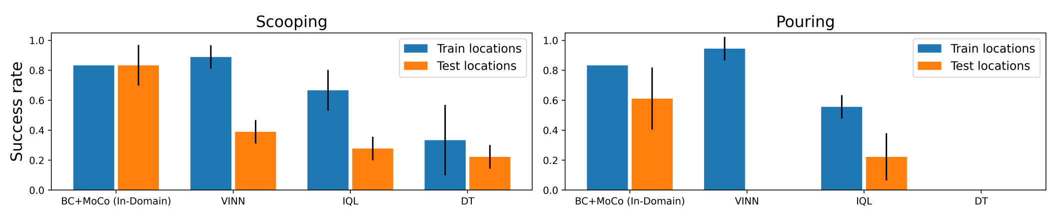

Figure 3: **Evaluating offline policy learning results.** `VINN` has the best performance on train locations but degrades on unseen locations, as does the performance of other methods.

4.1 Discussion

The main goal of this work is to introduce TOTO, our robotics benchmark. We presented a broad initial set of vision representations and policy learning baselines, which can be built off of by future users. Notably, these baselines were contributed in the same way that TOTO will be used in the future: by collaborators who locally train policies and submit them for remote evaluation on shared hardware. This shows the feasibility of our user workflow. The initial baseline results show the challenging nature of our tasks, especially with respect to generalization. By using TOTO as a community, we can more quickly iterate on ideas and make progress on the real-world bottlenecks to robot learning.

References

[1] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[3] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015.

[4] S. Dasari, J. Wang, J. Hong, S. Bahl, Y. Lin, A. S. Wang, A. Thankaraj, K. S. Chahal, B. Calli, S. Gupta, et al. Rb2: Robotic manipulation benchmarking with a twist. In *NeurIPS Datasets and Benchmarks Track*, 2021.

[5] N. Correll, K. E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Okada, A. Rodriguez, J. M. Romano, and P. R. Wurman. Analysis and observations from the first amazon picking challenge. *IEEE Transactions on Automation Science and Engineering*, 15(1):172–188, 2016.

[6] M. Buehler, K. Iagnemma, and S. Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer, 2009.

[7] E. Krotkov, D. Hackett, L. Jackel, M. Perschbacher, J. Pippine, J. Strauss, G. Pratt, and C. Orlowski. The darpa robotics challenge finals: Results and perspectives. *Journal of Field Robotics*, 34(2):229–240, 2017.

[8] G. Seetharaman, A. Lakhotia, and E. P. Blasch. Unmanned vehicles come of age: The darpa grand challenge. *Computer*, 39(12):26–29, 2006.

[9] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[10] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020.

[11] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

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

[13] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019.

[14] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[15] J. Collins, J. McVicar, D. Wedlock, R. Brown, D. Howard, and J. Leitner. Benchmarking simulated robotic manipulation through a real world dataset. *IEEE Robotics and Automation Letters*, 5(1):250–257, 2019.

[16] S. Bahl, A. Gupta, and D. Pathak. Hierarchical neural dynamic policies. *arXiv preprint arXiv:2107.05627*, 2021.

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

[18] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[19] S. Parisi, A. Rajeswaran, S. Purushwalkam, and A. K. Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *ICML*, 2022.

[20] 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.

[21] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[22] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf.

[23] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[24] 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. *Advances in neural information processing systems*, 34:15084–15097, 2021.

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

[26] T. Seno and M. Imai. d3rlpy: An offline deep reinforcement learning library. *arXiv preprint arXiv:2111.03788*, 2021.

[27] J. Pari, N. M. Shafiullah, S. P. Arunachalam, and L. Pinto. The surprising effectiveness of representation learning for visual imitation. *CoRR*, abs/2112.01511, 2021. URL https://arxiv.org/abs/2112.01511.

[28] V. Dean, Y. G. Shavit, and A. Gupta. Robots on demand: A democratized robotics research cloud. In *Conference on Robot Learning*, pages 1769–1775. PMLR, 2022.

[29] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt. The robotarium: A remotely accessible swarm robotics research testbed. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1699–1706. IEEE, 2017.

[30] A. Kumar, T. Buckley, J. B. Lanier, Q. Wang, A. Kavelaars, and I. Kuzovkin. Offworld gym: open-access physical robotics environment for real-world reinforcement learning benchmark and research. *arXiv preprint arXiv:1910.08639*, 2019.

[31] Y. Sun, J. Falco, M. A. Roa, and B. Calli. Research challenges and progress in robotic grasping and manipulation competitions. *IEEE robotics and automation letters*, 7(2):874–881, 2021.

[32] Z. Liu, W. Liu, Y. Qin, F. Xiang, M. Gou, S. Xin, M. A. Roa, B. Calli, H. Su, Y. Sun, et al. Ocrtoc: A cloud-based competition and benchmark for robotic grasping and manipulation. *IEEE Robotics and Automation Letters*, 7(1):486–493, 2021.

[33] N. Funk, C. Schaff, R. Madan, T. Yoneda, J. U. De Jesus, J. Watson, E. K. Gordon, F. Widmaier, S. Bauer, S. S. Srinivasa, et al. Benchmarking structured policies and policy optimization for real-world dexterous object manipulation. *arXiv preprint arXiv:2105.02087*, 2021.

[34] L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[35] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018.

[36] P. Sharma, L. Mohan, L. Pinto, and A. Gupta. Multiple interactions made easy (mime): Large scale demonstrations data for imitation. In *Conference on robot learning*, pages 906–915. PMLR, 2018.

[37] V. Kumar and E. Todorov. Mujoco haptix: A virtual reality system for hand manipulation. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 657–663. IEEE, 2015.

[38] 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 *International Conference on Robotics and Automation (ICRA)*, 2021.

5 Appendix

5.1 Related Work

For a thorough description of work related to remote robotics benchmarking, we refer to the Robotics Cloud concept paper [28]. Here we describe related work specific to our instantiation of a robotics cloud (TOTO).

Shared Tasks and Environments A necessary step in comparing method performance is evaluation on a common task. Common tasks might mean a standard object set such as YCB [3], which can be distributed to remote labs, allowing for shared metrics like grasp success on these objects. The Ranking-Based Robotics Benchmark (RB2) [4] provides four common manipulation tasks (similar to those we use, described in Section 2.1) as well as a framework for comparing and ranking methods across results from multiple labs. Another route is sharing the environment itself, as the Amazon Picking Challenge [5] and DARPA Robotics Challenges [6, 7, 8] have done. Sharing tasks or environments gives metrics by which we can compare approaches. However, users must still develop the approach on their own hardware in their own lab, and recreating identical environment setups is quite challenging.

Shared, Remote Robots Going one step further, remotely-accessible robots can be shared across the community, enabling method development and evaluation without users acquiring their own hardware. Georgia Tech's Robotarium [29] allows for remote experimentation of multi-agent methods on a physical robotic swarm, which has been extensively used not just in research but also in education. OffWorld Gym [30] provides remote access to navigation tasks using a mobile robot, with closely mirrored simulated and physical instances of the same environment. A recent survey paper [31] provides an overview of robotic grasping and manipulation competitions, including some that involve remotely-accessible, shared robots like [32]. Finally, most closely related to our work, the Real Robot Challenge [33] runs a tri-finger manipulation competition on cube reorientation tasks. The success of the Real Robot Challenge framework inspires our work, which also allows for the evaluation of manipulation tasks on shared robots. Our work, however, is designed to evaluate robot *learning* through challenging variations (lighting, unseen test objects, etc.) and an image-based dataset (as opposed to assuming ground-truth state access).

Open-Source Robotics Datasets Collecting real-world robotics data is challenging and expensive due to physical constraints like environment resets and hardware failures. Thus open-source datasets serve an important role in the field by enabling larger-scale offline robot learning. Some work has improved the way we collect robotics data, such as self-supervised grasping [34] and further parallelization of robots [35]. RoboTurk [14] provides a system for simple teleoperated data collection which can be executed remotely. Much work in robot learning has introduced datasets more generally, such as MIME [36] (8260 demonstrations over 20 tasks), RoboNet [13] (162,000 trajectories collected across 7 robots), and Bridge Data (7,200 demonstrations across 10 environments). However, it is hard to understand the value of these datasets without a common evaluation platform, something that Collins et al. [15] addresses by using simulation to replicate a real-world dataset. In contrast, we address this issue with real-world evaluation that matches the domain of the data collection. Our initial dataset is 2,898 trajectories, but this will grow over time as we add evaluation trajectories collected from users' policies.

Offline Robot Learning Our benchmark focuses on offline robot learning, including imitation learning and offline RL. Our initial baselines are described and contextualized in Section 3.2.

5.2 Hardware

Our hardware includes a Franka Emika Panda robot arm and workstation for real-time inference. We use a simple and common joint position control stack that runs at 30 Hz. Actions are specified as joint targets, which are translated into motor control signals using an underlying high-frequency PD controller. We use joint position control because end effector control using X,Y, Z positions alone is not feasible to solve our tasks: for example, the orientation of the gripper must change as the robot pours. We use an Intel D435 RealSense camera for recording RGB-D image observations.

We allow users to opt for a lower control frequency if desired. The training data can be subsampled by taking one of N frames since the actions are in absolute joint angles. We decrease the test time control frequency accordingly.

5.3 Task Details

Exmaple image observations for each task, pouring and scooping, are shown in Fig. 4. We also list relevant statistics of our dataset in Table. 1.

Table 1: Dataset overview. Pouring data collection using replay and BC proved challenging to reset (unsuccessful trials require more cleanup), so it was nearly all collected with teloperation.

Task	Trials	Length	Success	Teleop	BC	Replay
`Scooping`	1895	495	0.690	41%	33%	26%
`Pouring`	1003	324	0.977	99%	0%	1%

5.4 Data Collection

To improve diversity, our dataset were collected with three techniques: teleoperation, behavior cloning rollouts, and trajectory replay. Details of each collection method are described below.

Teleoperation We collected the majority of trajectories with teleoperation using Puppet [37]. The human controls the robot in an intuitive end effector space using an HTC Vive virtual reality headset and controller. While this teleoperation is theoretically possible to use remotely, we collect the data with the human and robot in the same room, giving the human direct perception of the scene. Our multiple teleoperators have different dominant hands, leading to more diverse data. Most teleoperation trials are successful.

Behavior cloning rollouts After teleoperation trajectories are collected, we train simple, state-based behavior cloning (BC) policies on each target location, so no visual perception is required. We roll out these trajectories with some noise added to actions at each timestep. The amount of noise varies across trajectories for additional diversity.

Trajectory replay Finally, we replay individual teleoperated trajectories with added noise. While these might seem overly similar to the original teleoperated trajectories, keep in mind that conditions like lighting also vary with time of day, so this replay still expands the dataset in other ways.

5.5 Benchmark Use

Here we introduce the framework for our benchmark. TOTO is designed to make the user workflow (Section 5.5.1) easy for newcomers with well-documented software infrastructure (Section 5.5.2) including examples and tests.

5.5.1 User Workflow

We provide a real-world dataset (Section 2.2) collected using our hardware setup (Section 5.2). Participants optionally use our software starter kit (Section 5.5.2) and locally train policies of their choosing using this data.

Users submit policies through Google Drive for evaluation on our real-world setup. They do not receive the low-level data from these evaluation trials; they simply receive a reward and high-level video to guide algorithm development, but not enough data to be used effectively for online training.

We run the real-world evaluations while an engineer is present to supervise; thus the evaluation turnaround time is currently around 12 hours (depending on the time of day submitted). Our goal is to place the emphasis on offline learning and prevent overfitting, thus removing the need for real-time results or large quantities of evaluation.

As new users evaluate methods after the paper release, we will post (anonymous) evaluation scores for each attempt on a website leaderboard. We will also periodically add data collected by the users' policies to the original dataset.

5.5.2 Software Infrastructure

Our software starter kit includes documented code and instructions for policy formatting and dataset usage. We have open-sourced baseline code, trajectory data, and pretrained models (see our website). These components ensure that TOTO is easily accessible to a broad portion of the robotics, ML, and even computer vision communities.

We adapt the agent format from Ke et al. [38], which requires a `predict` function taking in the observation and returning the action. We also use a standard config format and require an `init_agent_from_config` function to create the agent.

We provide users with code for training an example image-based BC agent and a docker environment which wraps the minimum required dependencies to run this code. Users can optionally extend the docker containers with additional dependencies. We also provide a stub environment which users can use to locally evaluate whether the agent's predictions are compatible with our robot environment. This setup allows resolution of all agent format and library dependency issues before users submit their agents for evaluation.

5.6 Experimental Results

We present the numerical rewards achieved by each method for visual policy comparison (Table. 2) and policy learning (Table. 3).

Table 2: Performance of vision representations with BC across train and test locations.

	Model	**Scooping**		**Pouring**	
		Reward	Success %	Reward	Success %
In Domain	`BYOL`	4.39	72.2%	20.22	66.6%
	`MoCo`	**7.42**	**83.3%**	**22.86**	**72.2%**
Out of Domain	`MoCo`	2.11	33.3%	14.89	55.5%
	`ResNet50`	2.83	47.2%	18.86	50.0%
	`R3M`	2.97	44.4%	6.94	33.3%

Table 3: TOTO policy learning results across train and test locations.

Model	**Scooping**		**Pouring**	
	Reward	Success %	Reward	Success %
`BC + MoCo`	7.42	**83.3%**	**22.86**	**72.2%**
`VINN`	**7.89**	63.9%	21.75	47.2%
`IQL`	6.08	47.2%	9.86	38.9%
`DT`	2.83	27.8%	0.00	0.0%

5.7 Limitations and Future Work

The evaluation protocol currently has manual steps: we measure the material transferred during pouring and scooping to compute rewards and reset by returning the material to the original object. We do see future potential to automate reward measurements and resets, such as by adding a scale beneath the target object and using an additional robot to reset the transferred materials. Spills of the transferred material, however, might still require manual intervention.

We plan to expand the evaluation setup to include additional robots. This would help us meet the increasing demand in evaluations as more users adopt the benchmark. One challenge will be visual differences across robots, but we plan to collect additional demonstrations on new robots, and this would be an opportunity to expand the set of tasks as well (we could designate one robot per task).

As user demand further grows, we will implement an evaluation job queue that prioritizes evaluation requests from different users and schedules the jobs based on the number of robots currently available.

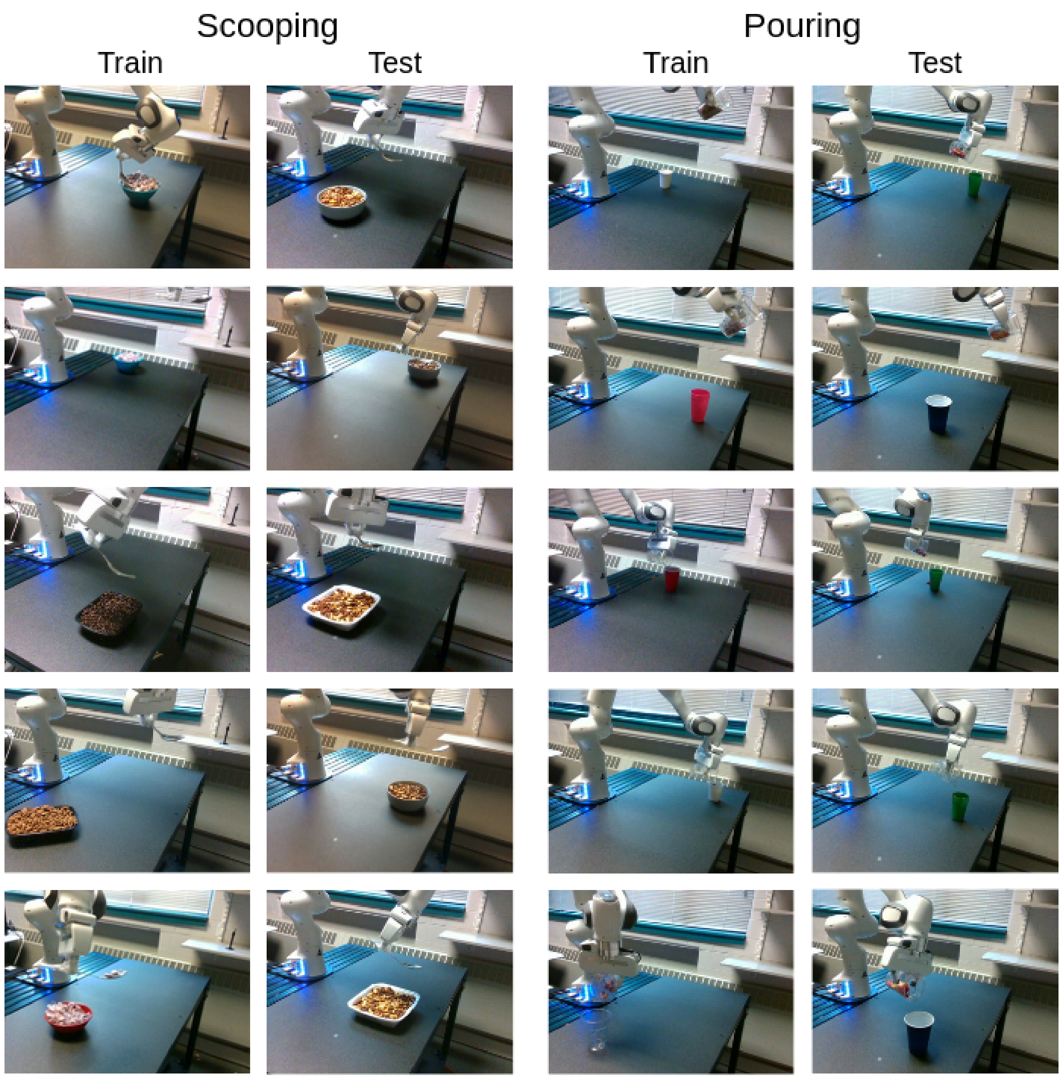

Figure 4: **TOTO Task Suite.** Our pouring and scooping tasks involve challenging variations in objects, position, lighting, and more.