Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning

Tianhe Yu[*1], **Deirdre Quillen**[*2], **Zhanpeng He**[*3], **Ryan Julian**[4], **Karol Hausman**[5], **Chelsea Finn**[1], **Sergey Levine**[2]
Stanford University[1], UC Berkeley[2], Columbia University[3],
University of Southern California[4], Robotics at Google[5]

Abstract

Meta-reinforcement learning algorithms can enable robots to acquire new skills much more quickly, by leveraging prior experience to learn how to learn. However, much of the current research on meta-reinforcement learning focuses on task distributions that are very narrow. For example, a commonly used meta-reinforcement learning benchmark uses different running velocities for a simulated robot as different tasks. When policies are meta-trained on such narrow task distributions, they cannot possibly generalize to more quickly acquire entirely new tasks. Therefore, if the aim of these methods is to enable faster acquisition of entirely new behaviors, we must evaluate them on task distributions that are sufficiently broad to enable generalization to new behaviors. In this paper, we propose an open-source simulated benchmark for meta-reinforcement learning and multi-task learning consisting of 50 distinct robotic manipulation tasks. Our aim is to make it possible to develop algorithms that generalize to accelerate the acquisition of entirely new, held-out tasks. We evaluate 6 state-of-the-art meta-reinforcement learning and multi-task learning algorithms on these tasks. Surprisingly, while each task and its variations (e.g., with different object positions) can be learned with reasonable success, these algorithms struggle to learn with multiple tasks at the same time, even with as few as ten distinct training tasks. Our analysis and open-source environments pave the way for future research in multi-task learning and meta-learning that can enable meaningful generalization, thereby unlocking the full potential of these methods.[1].

1 Introduction

While reinforcement learning (RL) has achieved some success in domains such as assembly [30], ping pong [35], in-hand manipulation [1], and hockey [7], state-of-the-art methods require substantially more experience than humans to acquire only one narrowly-defined skill. If we want robots to be broadly useful in realistic environments, we instead need algorithms that can learn a wide variety of skills reliably and efficiently. Fortunately, in most specific domains, such as robotic manipulation or locomotion, many individual tasks share common structure that can be reused to acquire related tasks more efficiently. For example, most robotic manipulation tasks involve grasping or moving objects in the workspace. However, while current methods can learn to individual skills like screwing on a bottle cap [30] and hanging a mug [33], we need algorithms that can efficiently learn shared structure across many related tasks, and use that structure to learn new skills quickly, such as screwing a jar lid or hanging a bag. Recent advances in machine learning have provided unparalleled generalization capabilities in domains such as images [28] and speech [11], suggesting that this should be possible; however, we have yet to see such generalization to diverse tasks in reinforcement learning settings.

∗ denotes equal contribution

[1]Videos of the benchmark tasks are on the project page: `meta-world.github.io`. Our open-sourced codes are available at: `https://github.com/rlworkgroup/metaworld`

3rd Workshop on Meta-Learning at NeurIPS 2019, Vancouver, Canada.

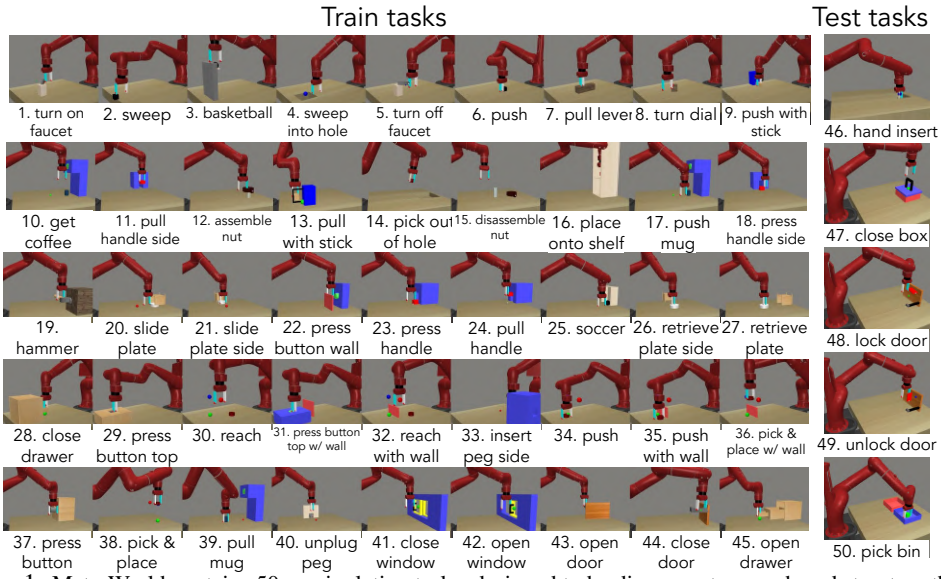

Figure 1: Meta-World contains 50 manipulation tasks, designed to be diverse yet carry shared structure that can be leveraged for efficient multi-task RL and transfer to new tasks via meta-RL. In the most difficult evaluation, the method must use experience from 45 training tasks (left) to quickly learn distinctly new test tasks (right).

Recent works in meta-learning and multi-task reinforcement learning have shown promise for addressing this gap. Multi-task RL methods aim to learn a single policy that can solve multiple tasks more efficiently than learning the tasks individually, while meta-learning methods train on many tasks, and optimize for fast adaptation to a new task. While these methods have made progress, the development of both classes of approaches has been limited by the lack of established benchmarks and evaluation protocols that reflect realistic use cases. On one hand, multi-task RL methods have largely been evaluated on disjoint and overly diverse tasks such as the Atari suite [23], where there is little efficiency to be gained by learning across games [39]. On the other hand, meta-RL methods have been evaluated on very narrow task distributions. For example, one popular evaluation of meta-learning involves choosing different running directions for simulated legged robots [17], which then enables fast adaptation to new directions. While these are technically distinct tasks, they are a far cry from the promise of a meta-learned model that can adapt to any new task within some domain. In order to study the capabilities of current multi-task and meta-reinforcement learning methods and make it feasible to design new algorithms that actually generalize and adapt quickly on meaningfully distinct tasks, we need evaluation protocols and task suites that are broad enough to enable this sort of generalization, while containing sufficient shared structure for generalization to be possible.

The key contributions of this work are a suite of 50 diverse simulated manipulation tasks and an extensive empirical evaluation of how previous methods perform on sets of such distinct tasks. We contend that multi-task and meta reinforcement learning methods that aim to efficiently learn many tasks and quickly generalize to new tasks should be evaluated on distributions of tasks that are diverse and exhibit shared structure. To this end, we present a benchmark of simulated manipulation tasks with everyday objects, all of which are contained in a shared, table-top environment with a simulated Sawyer arm. By providing a large set of distinct tasks that share common environment and control structure, we believe that this benchmark will allow researchers to test the generalization capabilities of the current multi-task and meta RL methods, and help to identify new research avenues to improve the current approaches. Our empirical evaluation of existing methods on this benchmark reveals that, despite some impressive progress in multi-task and meta-reinforcement learning over the past few years, current methods are generally not able to learn diverse task sets, much less generalize successfully to entirely new tasks. We provide an evaluation protocol with evaluation modes of varying difficulty, and observe that current methods only show success in the easiest modes. This opens the door for future developments in multi-task and meta reinforcement learning: instead of focusing on further increasing performance on current narrow task suites, we believe that it is essential for future work in these areas to focus on increasing the capabilities of algorithms to handle highly diverse task sets. By doing so, we can enable meaningful generalization across many tasks and achieve the full potential of meta-learning as a means of incorporating past experience to make it possible for robots to acquire new skills as quickly as people can.

2 Related Work

Previous works that have proposed benchmarks for reinforcement learning have largely focused on single task learning settings [2, 9, 53]. One popular benchmark used to study multi-task learning is the Arcade Learning Environment, a suite of dozens of Atari 2600 games [31]. While having a tremendous impact on the multi-task reinforcement learning research community [39, 44, 23, 14, 52], the Atari games included in the benchmark have significant differences in visual appearance, controls, and objectives, making it challenging to acquire any efficiency gains through shared learning. In fact, many prior multi-task learning methods have observed substantial negative transfer between the Atari games [39, 44]. In contrast, we would like to study a case where positive transfer between the different tasks should be possible. We therefore propose a set of related yet diverse tasks that share the same robot, action space, and workspace.

Meta-reinforcement learning methods have been evaluated on a number of different problems, including maze navigation [13, 55, 34], continuous control domains with parametric variation across tasks [17, 43, 40, 16], bandit problems [55, 13, 34, 42], levels of an arcade game [38], and locomotion tasks with varying dynamics [36, 45]. Complementary to these evaluations, we aim to develop a testbed of tasks and an evaluation protocol that are reflective of the challenges in applying meta-learning to robotic manipulation problems, including both parameteric and non-parametric variation in tasks.

There is a long history of robotics benchmarks [5], datasets [29, 18, 58, 6, 21, 32, 51], competitions [10] and standardized object sets [4, 8] that have played an important role in robotics research. Similarly, there exists a number of robotics simulation benchmarks including visual navigation [47, 27, 3, 46, 57], autonomous driving [12, 56, 41], grasping [24, 26, 20], single-task manipulation [15], among others. In this work, our aim is to continue this trend and provide a large suite of tasks that will allow researchers to study multi-task learning, meta-learning, and transfer in general. Further, unlike these prior simulation benchmarks, we particularly focus on providing a suite of many diverse manipulation tasks and a protocol for multi-task and meta RL evaluation.

3 The Multi-Task and Meta-RL Problem Statements

Our proposed benchmark is aimed at making it possible to study generalization in meta-RL and multi-task RL. In this section, we define the meta-RL and multi-task RL problem statements, and describe some of the challenges associated with task distributions in these settings.

We use the formalism of Markov decision processes (MDPs), where each task $\mathcal{T}$ corresponds to a different finite horizon MDP, represented by a tuple (S, A, P, R, H, γ), where $s \in S$ correspond to states, $a \in A$ correspond to the available actions, $P(s_{t+1}|s_t, a_t)$ represents the stochastic transition dynamics, $R(s, a)$ is a reward function, H is the horizon and γ is the discount factor. In standard reinforcement learning, the goal is to learn a policy $\pi(a|s)$ that maximizes the expected return, which is the sum of (discounted) rewards over all time. In multi-task and meta-RL settings, we assume a distribution of tasks $p(\mathcal{T})$. Different tasks may vary in any aspect of the Markov decision process, though efficiency gains in adaptation to new tasks are only possible if the tasks share some common structure. For example, as we describe in the next section, the tasks in our proposed benchmark have the same action space and horizon, and structurally similar rewards and state spaces.[2]

Multi-task RL problem statement. The goal of multi-task RL is to learn a single, task-conditioned policy $\pi(a|s, z)$, where z indicates an encoding of the task ID. This policy should maximize the average expected return across all tasks from the task distribution $p(\mathcal{T})$, given by $\mathbb{E}_{\mathcal{T}\sim p(\mathcal{T})}[\mathbb{E}_{\pi}[\sum_{t=0}^{T} \gamma^t R_t(s_t, a_t)]]$. The information about the task can be provided to the policy in various ways, e.g. using a one-hot task identification encoding z that is passed in addition to the current state. There is no separate test set of tasks, and multi-task RL algorithms are typically evaluated on their average performance over the *training* tasks.

Meta-RL problem statement. Meta-reinforcement learning aims to leverage the set of training task to learn a policy $\pi(a|s)$ that can quickly adapt to new test tasks that were not seen during training, where both training and test tasks are assumed to be drawn from the same task distribution $p(\mathcal{T})$.

[2] In practice, the policy must be able to read in the state for each of the tasks, which typically requires them to at least have the same dimensionality. In our benchmarks, some tasks have different numbers of objects, but the state dimensionality is always the same, meaning that some state coordinates are unused for some tasks.

Typically, the training tasks are referred to as the *meta-training* set, to distinguish from the adaptation (training) phase performed on the (meta-) test tasks. During meta-training, the learning algorithm has access to M tasks $\{\mathcal{T}_i\}_{i=1}^{M}$ that are drawn from the task distribution $p(\mathcal{T})$. At meta-test time, a new task $\mathcal{T}_j \sim p(\mathcal{T})$ is sampled that was not seen during meta-training, and the meta-trained policy must quickly adapt to this task to achieve the highest return with a small number of samples. A key premise in meta-RL is that a sufficiently powerful meta-RL method can meta-learn a model that effectively implements a highly efficient reinforcement learning procedure, which can then solve entirely new tasks very quickly – much more quickly than a conventional reinforcement learning algorithm learning from scratch. However, in order for this to happen, the meta-training distribution $p(\mathcal{T})$ must be sufficiently broad to encompass these new tasks. Unfortunately, most prior work in meta-RL evaluates on very narrow task distributions, with only one or two dimensions of parametric variation, such as the running direction for a simulated robot [17, 43, 40, 16].

4 Meta-World

If we want meta-RL methods to generalize effectively to entirely new tasks, we must meta-train on broad task distributions that are representative of the range of tasks that a particular agent might need to solve in the future. To this end, we propose a new multi-task and meta-RL benchmark, which we call Meta-World. In this section, we motivate the design decisions behind the Meta-World tasks, discuss the range of tasks, describe the representation of the actions, observations, and rewards, and present a set of evaluation protocols of varying difficulty for both meta-RL and multi-task RL.

4.1 The Space of Manipulation Tasks: Parametric and Non-Parametric Variability

Meta-learning makes two critical assumptions: first, that the meta-training and meta-test tasks are drawn from the same distribution, $p(\mathcal{T})$, and second, that the task distribution $p(\mathcal{T})$ exhibits shared structure that can be utilized for efficient adaptation to new tasks. If $p(\mathcal{T})$ is defined as a family of variations within a particular control task, as in prior work [17, 40], then it is unreasonable to hope for generalization to entirely new control tasks. For example, an agent has little hope of being able to quickly learn to open a door, without having ever experienced doors before, if it has only been trained on a set of meta-training tasks that are homogeneous and narrow. Thus, to enable meta-RL methods to adapt to entirely new tasks, we propose a much larger suite of tasks consisting of 50 qualitatively-distinct manipulation tasks, where continuous parameter variation cannot be used to describe the differences between tasks.

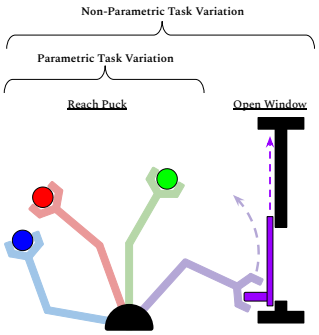

Figure 2: Parametric/non-parametric variation: all "reach puck" tasks (left) can be parameterized by the puck position, while the difference between "reach puck" and "open window" (right) is non-parametric.

With such non-parametric variation, however, there is the danger that tasks will not exhibit enough shared structure, or will lack the task overlap needed for the method to avoid memorizing each of the tasks. Motivated by this challenge, we design each task to include parametric variation in object and goal positions, as illustrated in Figure 2. Introducing this parametric variability not only creates a substantially larger (infinite) variety of tasks, but also makes it substantially more practical to expect that a meta-trained model will generalize to acquire entirely new tasks more quickly, since varying the positions provides for wider coverage of the space of possible manipulation tasks. Without parametric variation, the model could for example memorize that any object at a particular location is a door, while any object at another location is a drawer. If the locations are not fixed, this kind of memorization is much less likely, and the model is forced to generalize more broadly. With enough tasks and variation within tasks, pairs of qualitatively-distinct tasks are more likely to overlap, serving as a catalyst for generalization. For example, closing a drawer and pushing a block can appear as nearly the same task for some initial and goal positions of each object.

Note that this kind of parametric variation, which we introduce *for each task*, essentially represents the entirety of the task distribution for previous meta-RL evaluations [17, 40], which test on single tasks (e.g., running towards a goal) with parametric variability (e.g., variation in the goal position). Our full task distribution is therefore substantially broader, since it includes this parametric variability *for each of the* 50 *tasks*.

To provide shared structure, the 50 environments require the same robotic arm to interact with different objects, with different shapes, joints, and connectivity. The tasks themselves require the robot to execute a combination of reaching, pushing, and grasping, depending on the task. By recombining these basic behavioral building blocks with a variety of objects with different shapes and articulation properties, we can create a wide range of manipulation tasks. For example, the **open door** task involves pushing or grasping an object with a revolute joint, while the **open drawer** task requires pushing or grasping an object with a sliding joint. More complex tasks require a combination of these building blocks, which must be executed in the right order. We visualize all of the tasks in Meta-World in Figure 1, and include a description of all tasks in Appendix B.

All of the tasks are implemented in the MuJoCo physics engine [54], which enables fast simulation of physical contact. To make the interface simple and accessible, we base our suite on the Multiworld interface [37] and the OpenAI Gym environment interfaces [2], making additions and adaptations of the suite relatively easy for researchers already familiar with Gym.

4.2 Actions, Observations, and Rewards

In order to represent policies for multiple tasks with one model, the observation and action spaces must contain significant shared structure across tasks. All of our tasks are performed by a simulated Sawyer robot, with the action space corresponding to 3D end-effector positions. For all tasks, the robot must either manipulate one object with a variable goal position, or manipulate two objects with a fixed goal position. The observation space is represented as a 3-tuple of either the 3D Cartesian positions of the end-effector, the object, and the goal, or the 3D Cartesian positions of the end-effector, the first object, and the second object, and is always 9 dimensional.

Designing reward functions for Meta-World requires two major considerations. First, to guarantee that our tasks are within the reach of current single-task reinforcement learning algorithms, which is a prerequisite for evaluating multi-task and meta-RL algorithms, we design well-shaped reward functions for each task that make each of the tasks at least individually solvable. More importantly, the reward functions must exhibit shared structure across tasks. Critically, even if the reward function admits the same optimal policy for multiple tasks, varying reward scales or structures can make the tasks appear completely distinct for the learning algorithm, masking their shared structure and leading to preferences for tasks with high-magnitude rewards [23]. Accordingly, we adopt a structured, multi-component reward function for all tasks, which leads to effective policy learning for each of the task components. For instance, in a task that involves a combination of reaching, grasping, and placing an object, let $o \in \mathbb{R}^3$ be the object position, where $o = (o_x, o_y, o_z)$, $h \in \mathbb{R}^3$ be the position of the robot's gripper, $z_{\text{target}} \in \mathbb{R}$ be the target height of lifting the object, and $g \in \mathbb{R}^3$ be goal position. With the above definition, the multi-component reward function R is the additive combination of a reaching reward R_{reach}, a grasping reward R_{grasp} and a placing reward R_{place}, or subsets thereof for simpler tasks that only involve reaching and/or pushing. With this design, the reward functions across all tasks have similar magnitude and conform to similar structure, as desired. The full form of the reward function and a list of all task rewards is provided in Appendix C.

4.3 Evaluation Protocol

With the goal of providing a challenging benchmark to facilitate progress in multi-task RL and meta-RL, we design an evaluation protocol with varying levels of difficulty, ranging from the level of current goal-centric meta-RL benchmarks to a setting where methods must learn distinctly new, challenging manipulation tasks based on diverse experience across 45 tasks. We hence divide our evaluation into five categories, which we describe next. We then detail our evaluation criteria.

Meta-Learning 1 (ML1): Few-shot adaptation to goal variation within one task. The simplest evaluation aims to verify that previous meta-RL algorithms can adapt to new object or goal configurations on only one type of task. ML1 uses single Meta-World Tasks, with the meta-training "tasks" corresponding to 50 random initial object and goal positions, and meta-testing on 10 held-out positions. This resembles the evaluations in prior works [17, 40]. We evaluate algorithms on three individual tasks from Meta-World: reaching, pushing, and pick and place, where the variation is over reaching position or goal object position. The goal positions are not provided in the observation, forcing meta-RL algorithms to adapt to the goal through trial-and-error.

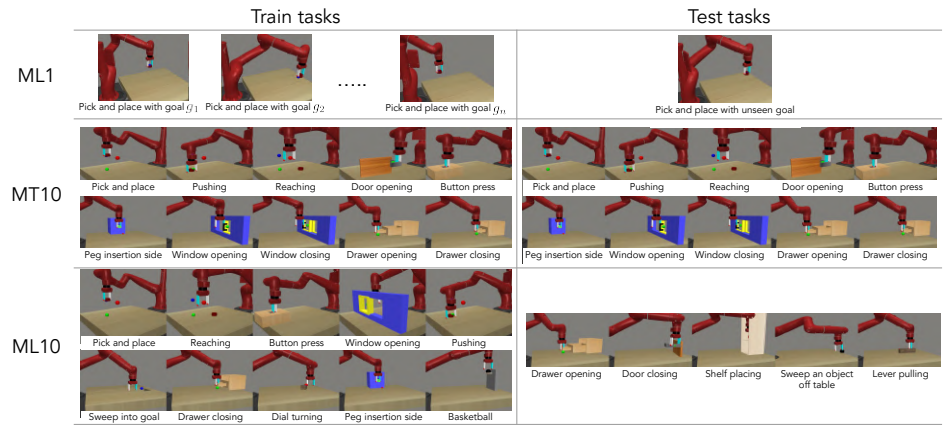

Figure 3: Visualization of three of our multi-task and meta-learning evaluation protocols, ranging from within task adaptation in ML1, to multi-task training across 10 distinct task families in MT10, to adapting to new tasks in ML10. Our most challenging evaluation mode ML45 is shown in Figure 1.

Multi-Task 10, Multi-Task 50 (MT10, MT50): Learning one multi-task policy that generalizes to 10 and 50 training tasks. A first step towards adapting quickly to distinctly new tasks is the ability to train a single policy that can solve multiple distinct training tasks. The multi-task evaluation in Meta-World tests the ability to learn multiple tasks at once, without accounting for generalization to new tasks. The MT10 evaluation uses 10 tasks: reach, push, pick and place, open door, open drawer, close drawer, press button top-down, insert peg side, open window, and open box. The larger MT50 evaluation uses all 50 Meta-World tasks. The policy is provided with a one-hot vector indicating the current task. The positions of objects and goal positions are fixed in all tasks in this evaluation, so as to focus on acquiring the distinct skills, rather than generalization and robustness.

Meta-Learning 10, Meta-Learning 45 (ML10, ML45): Few-shot adaptation to new test tasks with 10 and 50 meta-training tasks. With the objective to test generalization to new tasks, we hold out 5 tasks and meta-train policies on 10 and 45 tasks. We randomize object and goals positions and intentionally select training tasks with structural similarity to the test tasks. Task IDs are not provided as input, requiring a meta-RL algorithm to identify the tasks from experience.

Success metrics. Since values of reward are not directly indicative how successful a policy is, we define an interpretable success metric for each task, which will be used as the evaluation criterion for all of the above evaluation settings. Since all of our tasks involve manipulating one or more objects into a goal configuration, this success metric is based on the distance between the task-relevant object and its final goal pose, i.e. $\|o - g\|_2 < \epsilon$, where ϵ is a small distance threshold such as 5 cm. For the complete list of success metrics and thresholds for each task, see Appendix C.

5 Experimental Results and Analysis

The first, most basic goal of our experiments is to verify that each of the 50 presented tasks are indeed solveable by existing single-task reinforcement learning algorithms. We provide this verification in Appendix D. Beyond verifying the individual tasks, the goals of our experiments are to study the following questions: (1) can existing state-of-the-art meta-learning algorithms quickly learn qualitatively new tasks when meta-trained on a sufficiently broad, yet structured task distribution, and (2) how do different multi-task and meta-learning algorithms compare in this setting? To answer these questions, we evaluate various multi-task and meta-learning algorithms on the Meta-World benchmark. We include the training curves of all evaluations in Figure 8 in the Appendix E. Videos of the tasks and evaluations, along with all source code, are on the project webpage[3].

In the multi-task evaluation, we evaluate the following RL algorithms: **multi-task proximal policy optimization (PPO)** [50]: a policy gradient algorithm adapted to the multi-task setting by providing the one-hot task ID as input, **multi-task trust region policy optimization (TRPO)** [49]: an on-policy policy gradient algorithm adapted to the multi-task setting using the one-hot task ID as input, **multi-task soft actor-critic (SAC)** [22]: an off-policy actor-critic algorithm adapted to the multi-task setting using the one-hot task ID as input, **multi-task multi-head soft actor-critic (SAC)** [22]: an

[3] Videos are on the project webpage, at `meta-world.github.io`

off-policy actor-critic algorithm similar to **multi-task SAC** but using a multi-head policy with one head per task, and an on-policy version of **task embeddings (TE)** [25]: a multi-task reinforcement learning algorithm that parameterizes the learned policies via shared skill embedding space. For the meta-RL evaluation, we study three algorithms: **RL**^{2} [13, 55]: an on-policy meta-RL algorithm that corresponds to training a LSTM network with hidden states maintained across episodes within a task and trained with PPO, **model-agnostic meta-learning (MAML)** [17, 43]: an on-policy gradient-based meta-RL algorithm that embeds policy gradient steps into the meta-optimization, and is trained with PPO, and **probabilistic embeddings for actor-critic RL (PEARL)** [40]: an off-policy actor-critic meta-RL algorithm, which learns to encode experience into a probabilistic embedding of the task that is fed to the actor and the critic.

We show results of the simplest meta-learning evaluation mode, ML1, in Figure 7. We find that there is room for improvement even in this very simple setting. Next, we look at results of multi-task learning across distinct tasks, starting with MT10 in the top left of Figure 5 and in Table 1. We find that multi-task multi-head SAC is able to learn the MT10 task suite well, achieving around 88% success rate averaged across tasks, while multi-task SAC that has a single head can only solve around 40% of the tasks, indicating that adopting a multi-head architecture can greatly improve multi-task learning performance. On-policy methods such as task embeddings, multi-task PPO, and multi-task TRPO perform significantly worse, achieving less than 30% success across tasks. However, as we scale to 50 distinct tasks with MT50 (Figure 5, bottom left, and average results in Table 1), we find that multi-task multi-head SAC achieves only 35.85% average performance across the 50 tasks, while the other four methods have less than 30% success rates, indicating significant room for improvement.

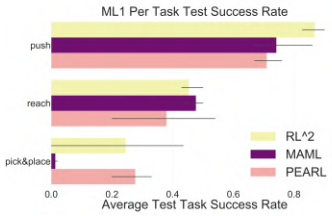

Figure 4: Comparison on our simplest meta-RL evaluation, ML1.

Finally, we study the ML10 and ML45 meta-learning benchmarks, which require learning the meta-training tasks and generalizing to new meta-test tasks with small amounts of experience. From Figure 5 and Table 1, we find that the prior meta-RL methods, MAML and RL^2 reach 36% and 10% success on ML10 test tasks, while PEARL is unable to generalize to new tasks on ML10. On ML45, PEARL manages to accomplish around 30% success rate on the test set, which suggests that having more meta-training tasks is conducive for PEARL to learn the underlying shared structure and adapt to unseen tasks. MAML and RL^2 solve around 20% of the meta-test tasks, potentially due to the additional optimization challenges in this regime. Note that, on both ML10 and ML45, the meta-training performance of all methods also has considerable room for improvement, suggesting that optimization challenges are generally more severe in the meta-learning setting. The fact that some methods nonetheless exhibit meaningful generalization suggests that the ML10 and ML45 benchmarks are solvable, but challenging for current methods, leaving considerable room for improvement in future work.

Methods	MT10	MT50
Multi-task PPO	25%	8.98%
Multi-task TRPO	29%	22.86%
Task embeddings	30%	15.31%
Multi-task SAC	39.5%	28.83%
Multi-task multi-head SAC	**88%**	**35.85%**

Methods	ML10		ML45	
	meta-train	meta-test	meta-train	meta-test
MAML	25%	**36%**	21.14%	23.93%
RL^2	**50%**	10%	**43.18%**	20%
PEARL	42.78%	0%	11.36%	**30%**

Table 1: Average success rates over all tasks for MT10, MT50, ML10, and ML45. The best performance in each benchmark is bolden. For MT10 and MT50, we show the average training success rate and multi-task multi-head SAC outperforms other methods. For ML10 and ML45, we show the meta-train and meta-test success rates. RL^2 achieves best meta-train performance in ML10 and ML45, while MAML and PEARL get the best generalization performance in ML10 and ML45 meta-test tasks respectively.

6 Conclusion and Directions for Future Work

We proposed an open-source benchmark for meta-reinforcement learning and multi-task learning, which consists of a large number of simulated robotic manipulation tasks. Unlike previous evaluation benchmarks in meta-RL, our benchmark specifically emphasizes generalization to distinctly new tasks, not just in terms of parametric variation in goals, but completely new objects and interaction

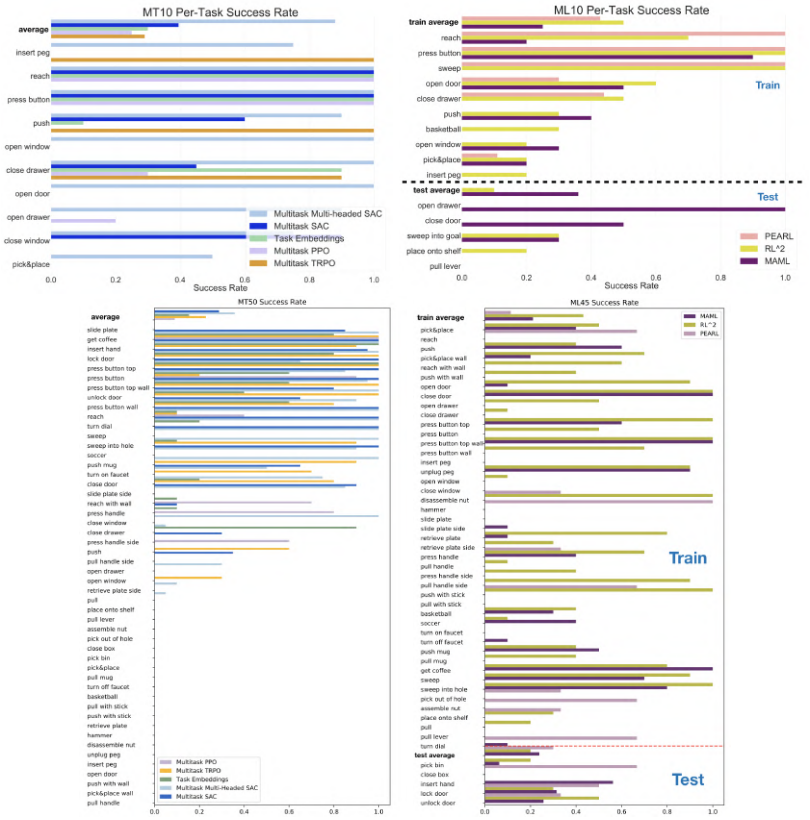

Figure 5: Full quantitative results on MT10, MT50, ML10, and ML45. Note that, even on the challenging ML10 and ML45 benchmarks, current methods already exhibit some degree of generalization, but meta-training performance leaves considerable room for improvement, suggesting that future work could attain better performance on these benchmarks. We also show the average success rates for all benchmarks in Table 1.

scenarios. While meta-RL can in principle make it feasible for agents to acquire new skills more quickly by leveraging past experience, previous evaluation benchmarks utilize very narrow task distributions, making it difficult to understand the degree to which meta-RL actually enables this kind of generalization. The aim of our benchmark is to make it possible to develop new meta-RL algorithms that actually exhibit this sort of generalization. Our experiments show that current meta-RL methods in fact cannot yet generalize effectively to entirely new tasks and do not even learn the meta-training tasks effectively when meta-trained across multiple distinct tasks. This suggests a number of directions for future work, which we describe below.

Future directions for algorithm design. The main conclusion from our experimental evaluation with our proposed benchmark is that current meta-RL algorithms generally struggle in settings where the meta-training tasks are highly diverse. This issue mirrors the challenges observed in multi-task RL, which is also challenging with our task suite, and has been observed to require considerable additional algorithmic development to attain good results in prior work [39, 44, 14]. A number of recent works have studied algorithmic improvements in the area of multi-task reinforcement learning, as well as potential explanations for the difficulty of RL in the multi-task setting [23, 48]. Incorporating some of these methods into meta-RL, as well as developing new techniques to enable meta-RL algorithms to train on broader task distributions, would be a promising direction for future work to enable meta-RL methods to generalize effectively across diverse tasks, and our proposed benchmark suite can provide future algorithms development with a useful gauge of progress towards the eventual goal of broad task generalization.

Future extensions of the benchmark. While the presented benchmark is significantly broader and more challenging than existing evaluations of meta-reinforcement learning algorithms, there are a number of extensions to the benchmark that would continue to improve and expand upon its applicability to realistic robotics tasks. We leave the discussion to Appendix A.

7 Acknowledgments

We thank Suraj Nair for feedback on a draft of the paper. This research was supported in part by the National Science Foundation under IIS-1651843, IIS-1700697, and IIS-1700696, the Office of Naval Research, ARL DCIST CRA W911NF-17-2-0181, DARPA, Google, Amazon, and NVIDIA.

References

[1] Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *arXiv:1808.00177*, 2018.

[2] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv:1606.01540*, 2016.

[3] Simon Brodeur, Ethan Perez, Ankesh Anand, Florian Golemo, Luca Celotti, Florian Strub, Jean Rouat, Hugo Larochelle, and Aaron Courville. Home: A household multimodal environment. *arXiv:1711.11017*, 2017.

[4] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *International Conference on Advanced Robotics (ICAR)*, 2015.

[5] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols. *arXiv:1502.03143*, 2015.

[6] Yevgen Chebotar, Karol Hausman, Zhe Su, Artem Molchanov, Oliver Kroemer, Gaurav Sukhatme, and Stefan Schaal. Bigs: Biotac grasp stability dataset. In *ICRA 2016 Workshop on Grasping and Manipulation Datasets*, 2016.

[7] Yevgen Chebotar, Karol Hausman, Marvin Zhang, Gaurav Sukhatme, Stefan Schaal, and Sergey Levine. Combining model-based and model-free updates for trajectory-centric reinforcement learning. In *ICML*, 2017.

[8] Young Sang Choi, Travis Deyle, Tiffany Chen, Jonathan D Glass, and Charles C Kemp. A list of household objects for robotic retrieval prioritized by people with als. In *International Conference on Rehabilitation Robotics*, 2009.

[9] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. *arXiv:1812.02341*, 2018.

[10] Nikolaus Correll, Kostas E Bekris, Dmitry Berenson, Oliver Brock, Albert Causo, Kris Hauser, Kei Okada, Alberto Rodriguez, Joseph M Romano, and Peter R Wurman. Analysis and observations from the first amazon picking challenge. *IEEE Transactions on Automation Science and Engineering*, 15(1):172–188, 2016.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. *arXiv:1711.03938*, 2017.

[13] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl$ˆ2$: Fast reinforcement learning via slow reinforcement learning. *CoRR*, abs/1611.02779, 2016.

[14] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv:1802.01561*, 2018.

[15] Linxi Fan, Yuke Zhu, Jiren Zhu, Zihua Liu, Orien Zeng, Anchit Gupta, Joan Creus-Costa, Silvio Savarese, and Li Fei-Fei. Surreal: Open-source reinforcement learning framework and robot manipulation benchmark. In *Conference on Robot Learning*, 2018.

[16] Chrisantha Fernando, Jakub Sygnowski, Simon Osindero, Jane Wang, Tom Schaul, Denis Teplyashin, Pablo Sprechmann, Alexander Pritzel, and Andrei Rusu. Meta-learning by the baldwin effect. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 1313–1320. ACM, 2018.

[17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

[18] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, pages 64–72, 2016.

[19] Chelsea Finn, Aravind Rajeswaran, Sham Kakade, and Sergey Levine. Online meta-learning. *ICML*, 2019.

[20] Corey Goldfeder, Matei Ciocarlie, Hao Dang, and Peter K Allen. The columbia grasp database. 2008.

[21] Abhinav Gupta, Adithyavairavan Murali, Dhiraj Prakashchand Gandhi, and Lerrel Pinto. Robot learning in homes: Improving generalization and reducing dataset bias. In *Advances in Neural Information Processing Systems*, pages 9112–9122, 2018.

[22] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[23] Matteo Hessel, Hubert Soyer, Lasse Espeholt, Wojciech Czarnecki, Simon Schmitt, and Hado van Hasselt. Multi-task deep reinforcement learning with popart. *CoRR*, abs/1809.04474, 2018.

[24] Daniel Kappler, Jeannette Bohg, and Stefan Schaal. Leveraging big data for grasp planning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4304–4311. IEEE, 2015.

[25] Ziyu Wang Nicolas Heess Martin Riedmiller Karol Hausman, Jost Tobias Springenberg. Learning an embedding space for transferable robot skills. *International Conference on Learning Representations*, 2018.

[26] Alexander Kasper, Zhixing Xue, and Rüdiger Dillmann. The kit object models database: An object model database for object recognition, localization and manipulation in service robotics. *IJRR*, 2012.

[27] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv:1712.05474*, 2017.

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

[29] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *IJRR*, 2015.

[30] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research (JMLR)*, 2016.

[31] Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew J. Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *CoRR*, abs/1709.06009, 2017.

[32] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. *arXiv:1811.02790*, 2018.

[33] Lucas Manuelli, Wei Gao, Peter R. Florence, and Russ Tedrake. kpam: Keypoint affordances for category-level robotic manipulation. *CoRR*, abs/1903.06684, 2019.

[34] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. *arXiv:1707.03141*, 2017.

[35] Katharina Mülling, Jens Kober, Oliver Kroemer, and Jan Peters. Learning to select and generalize striking movements in robot table tennis. *IJRR*, 2013.

[36] Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv:1803.11347*, 2018.

[37] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In *Advances in Neural Information Processing Systems*, 2018.

[38] Alex Nichol, Vicki Pfau, Christopher Hesse, Oleg Klimov, and John Schulman. Gotta learn fast: A new benchmark for generalization in rl. *arXiv:1804.03720*, 2018.

[39] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv:1511.06342*, 2015.

[40] Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *arXiv:1903.08254*, 2019.

[41] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2213–2222, 2017.

[42] Samuel Ritter, Jane X Wang, Zeb Kurth-Nelson, Siddhant M Jayakumar, Charles Blundell, Razvan Pascanu, and Matthew Botvinick. Been there, done that: Meta-learning with episodic recall. *arXiv preprint arXiv:1805.09692*, 2018.

[43] Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. Promp: Proximal meta-policy search. *arXiv:1810.06784*, 2018.

[44] Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv:1511.06295*, 2015.

[45] Steindór Sæmundsson, Katja Hofmann, and Marc Peter Deisenroth. Meta reinforcement learning with latent variable gaussian processes. *arXiv:1803.07551*, 2018.

[46] Manolis Savva, Angel X Chang, Alexey Dosovitskiy, Thomas Funkhouser, and Vladlen Koltun. Minos: Multimodal indoor simulator for navigation in complex environments. *arXiv:1712.03931*, 2017.

[47] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. *arXiv:1904.01201*, 2019.

[48] Tom Schaul, Diana Borsa, Joseph Modayil, and Razvan Pascanu. Ray interference: a source of plateaus in deep reinforcement learning. *arXiv preprint arXiv:1904.11455*, 2019.

[49] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

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

[51] Pratyusha Sharma, Lekha Mohan, Lerrel Pinto, and Abhinav Gupta. Multiple interactions made easy (mime): Large scale demonstrations data for imitation. *arXiv preprint arXiv:1810.07121*, 2018.

[52] Sahil Sharma and Balaraman Ravindran. Online multi-task learning using active sampling. 2017.

[53] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv:1801.00690*, 2018.

[54] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems*, 2012.

[55] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Rémi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn, 2016. *arXiv:1611.05763*.

[56] Bernhard Wymann, Eric Espié, Christophe Guionneau, Christos Dimitrakakis, Rémi Coulom, and Andrew Sumner. Torcs, the open racing car simulator. *Software available at http://torcs. sourceforge. net*, 4(6), 2000.

[57] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Computer Vision and Pattern Recognition*, 2018.

[58] Kuan-Ting Yu, Maria Bauza, Nima Fazeli, and Alberto Rodriguez. More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing. In *IROS*, 2016.

A Future Extensions of the Benchmark

First, in many situations, the poses of objects are not directly accessible to a robot in the real world. Hence, one interesting and important direction for future work is to consider image observations and sparse rewards. Sparse rewards can be derived already using the success metrics, while support for image rendering is already supported by the code. However, for meta-learning algorithms, special care needs to be taken to ensure that the task cannot be inferred directly from the image, else meta-learning algorithms will memorize the training tasks rather than learning to adapt. Another natural extension would be to consider including a breadth of compositional long-horizon tasks, where there exist combinatorial numbers of tasks. Such tasks would be a straightforward extension, and provide the possibility to include many more tasks with shared structure. Another challenge when deploying robot learning and meta-learning algorithms is the manual effort of resetting the environment. To simulate this case, one simple extension of the benchmark is to significantly reduce the frequency of resets available to the robot while learning. Lastly, in many real-world situations, the tasks are not available all at once. To reflect this challenge in the benchmark, we can add an evaluation protocol that matches that of online meta-learning problem statements [19]. We leave these directions for future work, either to be done by ourselves or in the form of open-source contributions. To summarize, we believe that the proposed form of the task suite represents a significant step towards evaluating multi-task and meta-learning algorithms on diverse robotic manipulation problems that will pave the way for future research in these areas.

B Task Descriptions

In Table 2, we include a description of each of the 50 Meta-World tasks.

C Task Rewards and Success Metrics

The form of the reward function is shared across tasks. In particular, the multi-component reward function R is a combination of a reaching reward R_{reach}, a grasping reward R_{grasp} and a placing reward R_{place} as follows:

$$\begin{aligned} R &= R_{\text{reach}} + R_{\text{grasp}} + R_{\text{place}} \\ &= \underbrace{-\|h - o\|_2}_{R_{\text{reach}}} + \underbrace{\mathbb{I}_{\|h-o\|_2<\epsilon} \cdot c_1 \cdot \min\{o_z, z_{\text{target}}\}}_{R_{\text{grasp}}} + \underbrace{\mathbb{I}_{|o_z - z_{\text{target}}|<\epsilon} \cdot c_2 \cdot \exp\{\|o-g\|_2^2/c_3\}}_{R_{\text{place}}} \end{aligned}$$

where ϵ, c_1, c_2, c_3 are constant for all tasks. For tasks that involve reaching and pushing, the reward R can be formed as a combination of a reaching reward R_{reach} and a pushing reward R_{push}:

$$\begin{aligned} R &= R_{\text{reach}} + R_{\text{push}} \\ &= \underbrace{-\|h - o\|_2}_{R_{\text{reach}}} + \underbrace{\mathbb{I}_{\|h-o\|_2<\epsilon} \cdot c_2 \cdot \exp\{\|o-g\|_2^2/c_3\}}_{R_{\text{push}}} \end{aligned}$$

With this design, the reward functions across all tasks have similar magnitude and conform to similar structure, as desired. In Table 3, we include a complete list of reward functions of each of the 50 Meta-World tasks. In Table 4, we include a complete list of success metrics of each of the 50 Meta-World tasks.

D Benchmark Verification with Single-Task Learning

In this section, we aim to verify that each of the benchmark tasks are individually solvable provided enough data. To do so, we consider two state-of-the-art single task reinforcement learning methods, proximal policy optimization (PPO) [50] and soft actor-critic (SAC) [22]. This evaluation is purely for validation of the tasks, and not an official evaluation protocol of the benchmark. Details of the hyperparameters are provided in Appendix F. The results of this experiment are illustrated in Figure 6. We indeed find that SAC can learn to perform all of the 50 tasks to some degree, while PPO can solve a large majority of the tasks.

Task	Description
turn on faucet	Rotate the faucet counter-clockwise. Randomize faucet positions
sweep	Sweep a puck off the table. Randomize puck positions
assemble nut	Pick up a nut and place it onto a peg. Randomize nut and peg positions
turn off faucet	Rotate the faucet clockwise. Randomize faucet positions
push	Push the puck to a goal. Randomize puck and goal positions
pull lever	Pull a lever down 90 degrees. Randomize lever positions
turn dial	Rotate a dial 180 degrees. Randomize dial positions
push with stick	Grasp a stick and push a box using the stick. Randomize stick positions.
get coffee	Push a button on the coffee machine. Randomize the position of the coffee machine
pull handle side	Pull a handle up sideways. Randomize the handle positions
basketball	Dunk the basketball into the basket. Randomize basketball and basket positions
pull with stick	Grasp a stick and pull a box with the stick. Randomize stick positions
sweep into hole	Sweep a puck into a hole. Randomize puck positions
disassemble nut	pick a nut out of the a peg. Randomize the nut positions
place onto shelf	pick and place a puck onto a shelf. Randomize puck and shelf positions
push mug	Push a mug under a coffee machine. Randomize the mug and the machine positions
press handle side	Press a handle down sideways. Randomize the handle positions
hammer	Hammer a screw on the wall. Randomize the hammer and the screw positions
slide plate	Slide a plate into a cabinet. Randomize the plate and cabinet positions
slide plate side	Slide a plate into a cabinet sideways. Randomize the plate and cabinet positions
press button wall	Bypass a wall and press a button. Randomize the button positions
press handle	Press a handle down. Randomize the handle positions
pull handle	Pull a handle up. Randomize the handle positions
soccer	Kick a soccer into the goal. Randomize the soccer and goal positions
retrieve plate side	Get a plate from the cabinet sideways. Randomize plate and cabinet positions
retrieve plate	Get a plate from the cabinet. Randomize plate and cabinet positions
close drawer	Push and close a drawer. Randomize the drawer positions
press button top	Press a button from the top. Randomize button positions
reach	reach a goal position. Randomize the goal positions
press button top wall	Bypass a wall and press a button from the top. Randomize button positions
reach with wall	Bypass a wall and reach a goal. Randomize goal positions
insert peg side	Insert a peg sideways. Randomize peg and goal positions
pull	Pull a puck to a goal. Randomize puck and goal positions
push with wall	Bypass a wall and push a puck to a goal. Randomize puck and goal positions
pick out of hole	Pick up a puck from a hole. Randomize puck and goal positions
pick&place w/ wall	Pick a puck, bypass a wall and place the puck. Randomize puck and goal positions
press button	Press a button. Randomize button positions
pick&place	Pick and place a puck to a goal. Randomize puck and goal positions
pull mug	Pull a mug from a coffee machine. Randomize the mug and the machine positions
unplug peg	Unplug a peg sideways. Randomize peg positions
close window	Push and close a window. Randomize window positions
open window	Push and open a window. Randomize window positions
open door	Open a door with a revolving joint. Randomize door positions
close door	Close a door with a revolvinig joint. Randomize door positions
open drawer	Open a drawer. Randomize drawer positions
insert hand	Insert the gripper into a hole.
close box	Grasp the cover and close the box with it. Randomize the cover and box positions
lock door	Lock the door by rotating the lock clockwise. Randomize door positions
unlock door	Unlock the door by rotating the lock counter-clockwise. Randomize door positions
pick bin	Grasp the puck from one bin and place it into another bin. Randomize puck positions

Table 2: A list of all of the Meta-World tasks and a description of each task.

E Learning curves

In evaluating meta-learning algorithms, we care not just about performance but also about efficiency, i.e. the amount of data required by the meta-training process. While the adaptation process for all algorithms is extremely efficient, requiring only 10 trajectories, the meta-learning process can be very inefficient, particularly for on-policy algorithms such as MAML, RL^2. In Figure 7, we show full learning curves of the three meta-learning methods on ML1. In Figure 8, we show full

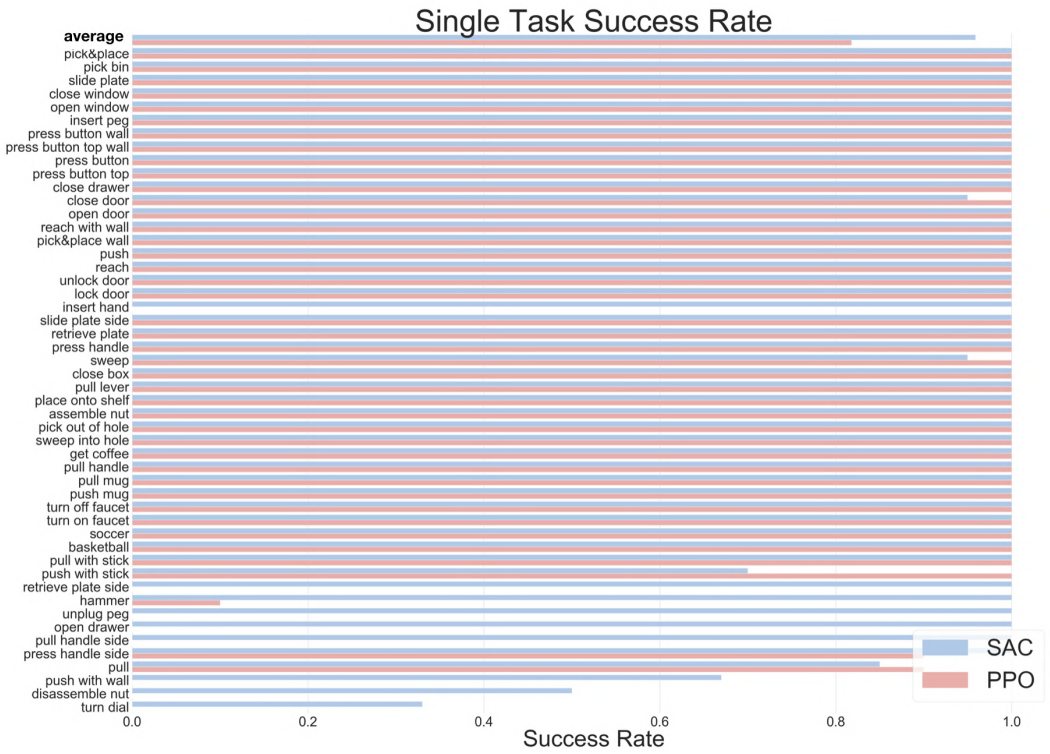

Figure 6: Performance of independent policies trained on individual tasks using soft actor-critic (SAC) and proximal policy optimization (PPO). We verify that SAC can solve all of the tasks and PPO can also solve most of the tasks.

learning curves of MT10, ML10, MT50 and ML45. The MT10 and MT50 learning curves show the efficiency of multi-task learning, a critical evaluation metric, since sample efficiency gains are a primary motivation for using multi-task learning. Unsurprisingly, we find that off-policy algorithms such as soft actor-critic and PEARL are able to learn with substantially less data than on-policy algorithms.

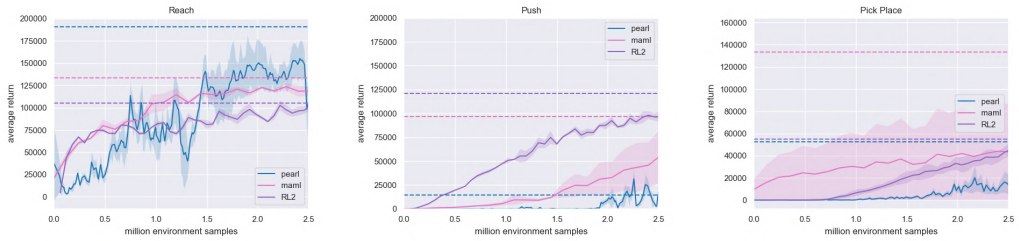

Figure 7: Comparison of PEARL, MAML, and RL^2 learning curves on the simplest evaluation, ML1, where the methods need to adapt quickly to new object and goal positions within the one meta-training task.

F Hyperparameter Details

In this section, we provide hyperparameter values for each of the methods in our experimental evaluation.

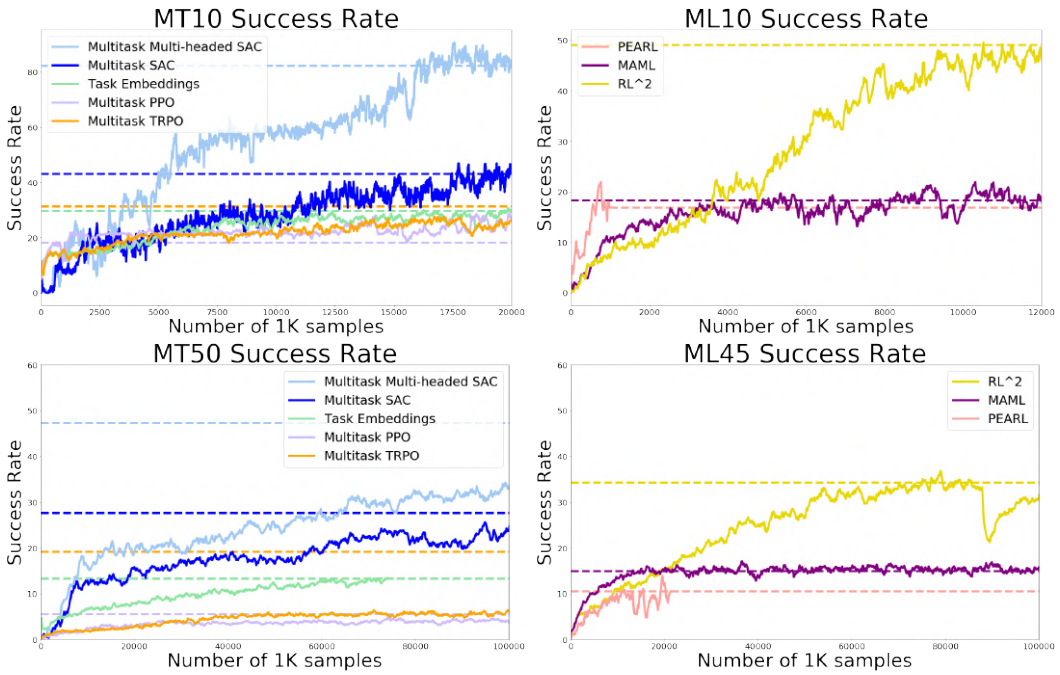

Figure 8: Learning curves of all methods on MT10, ML10, MT50, and ML45 benchmarks. Y-axis represents success rate averaged over tasks in percentage (%). The dashed lines represent asymptotic performances. Off-policy algorithms such as multi-task SAC and PEARL learn much more efficiently than off-policy methods, though PEARL underperforms MAML and RL^2.

F.1 Single Task SAC

Hyperparameter	Hyperparameter values
batch size	128
non-linearity	ReLU
policy initialization	standard Gaussian
exploration parameters	run a uniform exploration policy 1000 steps
# of samples / # of train steps per iteration	1 env step / 1 training step
policy learning rate	3e-4
Q function learning rate	3e-4
optimizer	Adam
discount	.99
horizon	150
reward scale	1.0
temperature	learned

F.2 Single Task PPO

Hyperparameter	Hyperparameter values
non-linearity	ReLU
batch size	4096
policy initial standard deviation	2.
entropy regularization coefficient	1e-3
baseline	linear feature baseline

F.3 Multi-Task SAC

Hyperparameter	Hyperparameter values
network architecture	feedforward network
network size	three fully connected layers with 400 units
batch size	$128 \times$ number_of_tasks
non-linearity	ReLU
policy initialization	standard Gaussian
exploration parameters	run a uniform exploration policy 1000 steps
# of samples / # of train steps per iteration	number_of_tasks env steps / 1 training step
policy learning rate	3e-4
Q function learning rate	3e-4
optimizer	Adam
discount	.99
horizon	150
reward scale	1.0
temperature	learned and disentangled with tasks

F.4 Multi-Task Multi-Headed SAC

Hyperparameter	Hyperparameter values
network architecture	multi-head (one head per task)
network size	three fully connected layers with 400 units
batch size	$128 \times$ number_of_tasks
non-linearity	ReLU
policy initialization	standard Gaussian
exploration parameters	run a uniform exploration policy 1000 steps
# of samples / # of train steps per iteration	number_of_tasks env steps / 1 training step
policy learning rate	3e-4
Q function learning rate	3e-4
optimizer	Adam
discount	.99
horizon	150
reward scale	1.0
temperature	learned and disentangled with tasks

F.5 Multi-Task PPO

Hyperparameter	Hyperparameter values
batch size	# of tasks * 10 * 150
policy initial standard deviation	2.
entropy regularization coefficient	0.002
baseline	linear feature baseline fit with observations and returns

F.6 Multi-Task TRPO

Hyperparameter	Hyperparameter values
batch size	# of tasks * 10 * 150
policy initial standard deviation	2.
step size	0.01
baseline	linear feature baseline fit with observations and returns

F.7 Task Embeddings

Hyperparameter	Hyperparameter values
nonlinearity	tanh
batch size	# of tasks * 10 * 150
latent dimension	6
inference window length	20
embedding maximum standard deviation	2.
baseline	Gaussian MLP, fit with observations, latent variables and returns

F.8 PEARL

Hyperparameter	Hyperparameter values
policy learning rate	3e-4
Q function learning rate	3e-4
discount	.99
horizon	150
# of samples / # of train steps per iteration	22500 env steps / 4,000 training steps
KL loss weight	.1
nonlinearity	relu

F.9 RL^2

Hyperparameter	Hyperparameter values
nonlinearity	tanh
policy initialization	Gaussian with $\sigma = 2.0$
baseline	linear-fit with polynomial features $(n = 2)$ of observation and time step
meta batch size	40
# roll-outs per meta task	10
horizon	150
optimizer	Adam
learning rate	1e-3
discount	.99
batch size	# of tasks * 10 * 150

F.10 MAML

Hyperparameter	Hyperparameter values
nonlinearity	tanh
meta batch size	20
# roll-outs per meta task	10
inner gradient step learning rate	0.05
discount	.99

Task	Reward
turn on faucet	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
sweep	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pick out of hole	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
turn off faucet	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
push with stick	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
get coffee	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pull handle side	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
basketball	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pull with stick	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
sweep into hole	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
disassemble nut	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
assemble nut	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
place onto shelf	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
push mug	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
press handle side	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
hammer	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
slide plate	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
slide plate side	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
press button wall	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
press handle	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pull handle	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
soccer	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
retrieve plate side	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
retrieve plate	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
close drawer	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
reach	$1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
press button top wall	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
reach with wall	$1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
insert peg side	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
push	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
push with wall	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pick&place w/ wall	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
press button	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
press button top	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pick&place	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pull	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pull mug	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
unplug peg	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
turn dial	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pull lever	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
close window	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
open window	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
open door	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
close door	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
open drawer	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
insert hand	$1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
close box	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
lock door	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
unlock door	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$
pick bin	$-\|h-o\|_2 + \mathbb{I}_{\|h-o\|_2<0.05} \cdot 100 \cdot \min\{o_z, z_{\text{target}}\} + \mathbb{I}_{\|o_z-z_{\text{target}}\|<0.05} \cdot 1000 \cdot \exp\{\|h-g\|_2^2/0.01\}$

Table 3: A list of reward functions used for each of the Meta-World tasks.

Task	Success Metric
turn on faucet	$\mathbb{I}_{\|o-g\|_2<0.05}$
sweep	$\mathbb{I}_{\|o-g\|_2<0.05}$
pick out of hole	$\mathbb{I}_{\|o-g\|_2<0.08}$
turn off faucet	$\mathbb{I}_{\|o-g\|_2<0.05}$
push	$\mathbb{I}_{\|o-g\|_2<0.07}$
push with stick	$\mathbb{I}_{\|o-g\|_2<0.08}$
get coffee	$\mathbb{I}_{\|o-g\|_2<0.02}$
pull handle side	$\mathbb{I}_{\|o-g\|_2<0.04}$
basketball	$\mathbb{I}_{\|o-g\|_2<0.08}$
pull with stick	$\mathbb{I}_{\|o-g\|_2<0.08}$
sweep into hole	$\mathbb{I}_{\|o-g\|_2<0.05}$
disassemble nut	$\mathbb{I}_{\|o-g\|_2<0.08}$
assemble nut	$\mathbb{I}_{\|o-g\|_2<0.08}$
place onto shelf	$\mathbb{I}_{\|o-g\|_2<0.08}$
push mug	$\mathbb{I}_{\|o-g\|_2<0.07}$
press handle side	$\mathbb{I}_{\|o-g\|_2<0.04}$
hammer	$\mathbb{I}_{\|o-g\|_2<0.05}$
slide plate	$\mathbb{I}_{\|o-g\|_2<0.07}$
slide plate side	$\mathbb{I}_{\|o-g\|_2<0.07}$
press button wall	$\mathbb{I}_{\|o-g\|_2<0.02}$
press handle	$\mathbb{I}_{\|o-g\|_2<0.02}$
pull handle	$\mathbb{I}_{\|o-g\|_2<0.04}$
soccer	$\mathbb{I}_{\|o-g\|_2<0.07}$
retrieve plate side	$\mathbb{I}_{\|o-g\|_2<0.07}$
retrieve plate	$\mathbb{I}_{\|o-g\|_2<0.07}$
close drawer	$\mathbb{I}_{\|o-g\|_2<0.08}$
reach	$\mathbb{I}_{\|o-g\|_2<0.05}$
press button top wall	$\mathbb{I}_{\|o-g\|_2<0.02}$
reach with wall	$\mathbb{I}_{\|o-g\|_2<0.05}$
insert peg side	$\mathbb{I}_{\|o-g\|_2<0.07}$
push with wall	$\mathbb{I}_{\|o-g\|_2<0.07}$
pick&place w/ wall	$\mathbb{I}_{\|o-g\|_2<0.07}$
press button	$\mathbb{I}_{\|o-g\|_2<0.02}$
press button top	$\mathbb{I}_{\|o-g\|_2<0.02}$
pick&place	$\mathbb{I}_{\|o-g\|_2<0.07}$
pull	$\mathbb{I}_{\|o-g\|_2<0.07}$
pull mug	$\mathbb{I}_{\|o-g\|_2<0.07}$
unplug peg	$\mathbb{I}_{\|o-g\|_2<0.07}$
turn dial	$\mathbb{I}_{\|o-g\|_2<0.03}$
pull lever	$\mathbb{I}_{\|o-g\|_2<0.05}$
close window	$\mathbb{I}_{\|o-g\|_2<0.05}$
open window	$\mathbb{I}_{\|o-g\|_2<0.05}$
open door	$\mathbb{I}_{\|o-g\|_2<0.08}$
close door	$\mathbb{I}_{\|o-g\|_2<0.08}$
open drawer	$\mathbb{I}_{\|o-g\|_2<0.08}$
insert hand	$\mathbb{I}_{\|o-g\|_2<0.05}$
close box	$\mathbb{I}_{\|o-g\|_2<0.08}$
lock door	$\mathbb{I}_{\|o-g\|_2<0.05}$
unlock door	$\mathbb{I}_{\|o-g\|_2<0.05}$
pick bin	$\mathbb{I}_{\|o-g\|_2<0.08}$

Table 4: A list of success metrics used for each of the Meta-World tasks. All units are in meters.