Never Stop Learning: The Effectiveness of Fine-Tuning in Robotic Reinforcement Learning

Ryan Julian[†‡] **Benjamin Swanson**[†] **Gaurav S. Sukhatme**[‡]

Sergey Levine[†§] **Chelsea Finn**[†¶] **Karol Hausman**[†]

Abstract: One of the great promises of robot learning systems is that they will be able to learn from their mistakes and continuously adapt to ever-changing environments. Despite this potential, most of the robot learning systems today produce static policies that are not further adapted during deployment, because the algorithms which produce those policies are not designed for continual adaptation. We present an adaptation method, and empirical evidence that it supports a robot learning framework for continual adaption. We show that this very simple method–fine-tuning off-policy reinforcement learning using offline datasets–is robust to changes in background, object shape and appearance, lighting conditions, and robot morphology. We demonstrate how to adapt vision-based robotic manipulation policies to new variations using less than 0.2% of the data necessary to learn the task from scratch. Furthermore, we demonstrate that this robustness holds in an episodic continual learning setting. We also show that pre-training via RL is essential: training from scratch or adapting from supervised ImageNet features are both unsuccessful with such small amounts of data. Our empirical conclusions are consistently supported by experiments on simulated manipulation tasks, and by 60 unique fine-tuning experiments on a real robotic grasping system pre-trained on $580,000$ grasps. For video results and an overview of the methods and experiments in this study, see the project website at https://ryanjulian.me/continual-fine-tuning.

Keywords: reinforcement learning, fine-tuning, continual learning, manipulation

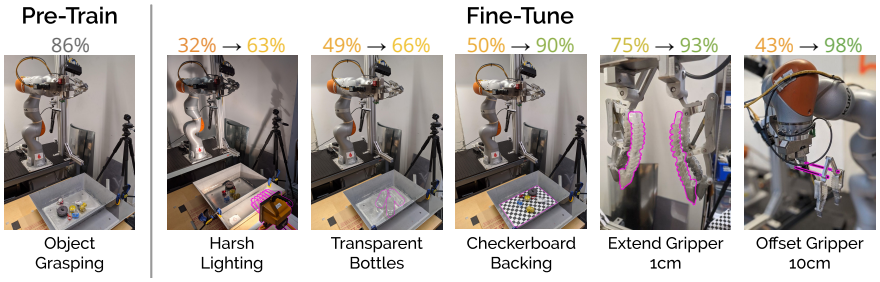


Figure 1: **Left:** Original robot configuration used for pre-training. **Right:** Adaptation challenges (highlighted in pink) studied in this work. **Top:** Associated performance improvements obtained using our fine-tuning method.

1 Introduction

Surviving and existing in the real world inevitably requires nearly every living being to constantly learn, adapt, and evolve. Similarly, to thrive in the real world, robots should be able to continuously learn and adapt throughout their lifetime to the ever-changing environments in which they are deployed. This is a widely recognized requirement. In fact, there is an entire academic sub-field of lifelong learning [1] which studies how to create agents that never stop learning. Despite the wide interest in this ability, most of the intelligent agents deployed today are not tested for their adaptation capabilities. Even though reinforcement learning theoretically provides the ability to perpetually

[†] Google Research, Robotics at Google Team
[‡] Department of Computer Science, University of Southern California
[§] Department of Electrical Engineering and Computer Sciences, University of California, Berkeley
[¶] Department of Computer Science, Stanford University

4th Conference on Robot Learning (CoRL 2020), Cambridge MA, USA.

learn from trial and error, this is not how it is typically evaluated. Instead, the predominant method of acquiring a new task with reinforcement learning is to initialize a policy from scratch, collect entirely new data in a stationary environment, and evaluate a static policy that was trained with this data. This static paradigm does not evaluate the robot's capability to adapt, and traps robotic reinforcement learning in the worst-case regime for sample efficiency: the cost to acquire a new task is dominated by the *complexity* of the task.

Most machine learning models successfully deployed in the real world, such as those used for computer vision and natural language processing (NLP) do not live in this regime. For instance, the predominant method of acquiring a new computer vision task is to start learning the new task with a pre-trained model for a related task, acquired from a pre-collected data set, and *fine-tune* that model to achieve the new task [2, 3, 4]. This changes the sample efficiency regime of the learning process from one which is dominated by *task complexity* to one that is dominated by *task novelty*, i.e., the differences between the new task and the task on which the model was pre-trained. While a number of works have studied how to use pre-trained ImageNet [5] features for robotics [6, 7, 8], there are remarkably few works that study how to adapt motor skills themselves.

Our work attempts to bridge this gap: instead of focusing on the robot's performance in the environment in which it was trained, we purposefully modify the robot and its environment to mimic the persistent change of the real world, and investigate its ability to adapt. Likewise, rather than proposing a new adaptation algorithm, with new complexity and caveats, we show how to successfully adapt robotic policies to substantial changes, using only the most basic components of existing off-policy reinforcement learning algorithms. The main contributions of this work are (1) a careful real-world study of the problem of end-to-end skill adaptation for a continually-learning robot, and (2) evidence that a very simple fine-tuning method can achieve that adaptation. Additionally, we plan to open source the models and datasets in this work, to aid others studying offline RL and identifying better off-policy evaluation metrics. To our knowledge, this work is the first to demonstrate that simple fine-tuning of off-policy reinforcement learning can successfully adapt to substantial task, robot, and environment variations which were not present in the training distribution (*i.e.* off-distribution).

2 Related Work

Reinforcement learning is a long-standing approach for enabling robots to autonomously acquire skills [9] such as locomotion [10, 11], pushing objects [12, 13], ball-in-cup manipulation [14], peg insertion [15, 16, 17, 18, 19], throwing objects [20, 21], or grasping [22, 23]. We specifically focus on the problem of deep reinforcement learning from raw pixel observations [16], as it allows us to place very few restrictions on state representation. A number of works have also considered this problem setting [24, 20, 13, 19, 25, 26]. However, a key challenge with deep RL methods is that they typically learn each skill from scratch, disregarding previously-learned skills. If we hope for robots to generalize to a broad range of real world environments, this approach is not practical.

We instead consider how we might transfer knowledge for efficient learning in new conditions [27, 28, 29], a widely-studied problem particularly outside the domain of robotics [2, 3, 4, 30, 31]. Prior works in robotics have considered how we might transfer information from models trained with supervised learning on ImageNet [5] by fine-tuning [16, 24, 32, 22] or other means [33, 34]. Our experiments show that transfer from pre-trained conditions is significantly more successful than transfer from ImageNet. Other works have leveraged experience in simulation [35, 36, 37, 38, 39, 40, 41, 42, 43] or representations learned with auxiliary losses [44, 45, 46] for effective transfer. While successful, these approaches either require significant engineering effort to construct an appropriate simulation or significant supervision. Most relevantly, recent work in model-based RL has used predictive models for fast transfer to new experimental set-ups [47, 48], *i.e.* by fine-tuning predictive models [49], via online search of a pre-learned representation of the space models, policies, or high-level skills [50, 51, 52, 53], or by learning physics simulation parameters from real data [54, 55]. We show how fine-tuning is successful with a model-free RL approach, and show how a state-of-the-art grasping system can be adapted to new conditions.

Other works have aimed to share and transfer knowledge across tasks and conditions by simultaneously learning across multiple goals and tasks [56, 57]. For example, prior works in model-based RL [13, 58, 59] and in goal-conditioned RL [25, 60, 61, 62, 63] have shared data and representations across multiple goals and objects. Along a similar vein, prior work in robotic meta-learning has aimed to learn representations that can be quickly adapted to new dynamics [64, 65, 66] and objects [67, 68, 69, 70]. We consider adaptation to a broad class of changes including dynamics,

object classes, and visual observations. We include conditions that shift substantially from the training conditions, and we do not require the full set of conditions to be represented during the initial training phase.

3 Identifying Adaptation Challenges

To study the problem of continual adaptation, we evaluate a pre-trained grasping policy in five different conditions that were not encountered during pre-training. In this section, we will describe the pre-training process and test the robustness of the pre-trained policy to various robot and environment modifications. We choose these modifications to reflect changes we believe a learning robot would experience, and should be expected to a adapt to, when deployed "on the job" in the real world. In Section 4, we will describe a simple fine-tuning based adaptation process, and we evaluate it using these modifications in Sections 4 and 5.

3.1 Pre-training process

We pre-train the grasping policy, which we refer to as the "base policy," using the QT-Opt algorithm in two stages, as described in [23]. This procedure yields a final base policy that achieves 96% accuracy on a set of previously-unseen test objects. We use a challenging subset of six of these test objects for most experiments in this work. On this set, our base model achieves a **success rate of 86% on the baseline grasping task**.

3.2 Robustness of the pre-trained policy

We begin by choosing a set of significant modifications to the robot and environment, which we believe are characteristic of a real-world continual learning scenario. We then evaluate the performance of the base policy on increasingly-severe versions of these modifications. This process allows us to assess the robustness of policies trained using the pre-training method. Once we find a modification that is severe enough to compromise the base policy's performance in each category, we use it to define a "Challenge Task" for our study of adaptation methods. Next, we describe these challenges and the corresponding performance of the base policy.

Background: We introduce a black-white $1\,\mathrm{inch}$ checkerboard pattern that we glue to the bottom of the robot's workspace (see Fig. 1, fourth from left). This often fools the robot into grasping at checkerboard edges rather than objects.

Lighting conditions: We introduce a high-intensity halogen light source parallel with the workspace (see Fig. 1, second from left), creating a bright spot in the robot's camera view, and intense light-dark contrasts along the plane of the workspace.

Gripper shape: We extend the parallel gripper attached to the robot by $1\,\mathrm{cm}$ and significantly narrow its width and compliance in the process (see Fig. 1, fifth from left). This changes the robot's kinematics (lengthening the gripper in the distal direction), while also lowering the relative pose of the robot with respect to the workspace surface by $1\,\mathrm{cm}$.

Robot morphology: We translate the gripper laterally by $10\,\mathrm{cm}$ (approximately a full gripper or arm link width) (see Fig. 1, far-right).

Unseen objects: We introduce completely-transparent plastic beverage bottles (see Fig. 1, third from left) that were not present in the training set. This causes the robot to often grasp where two bottles are adjacent, as though it cannot differentiate which parts of the scene are inside vs outside a bottle.

See Table 1 for a summary of the modification experiments, and their effect on base policy performance.

Challenge Task	Type	Base Policy	Δ
Checkerboard Backing	Background	50%	-36%
Harsh Lighting	Lighting conditions	31%	-55%
Extend Gripper $1\,\mathrm{cm}$	Gripper shape	76%	-10%
Offset Gripper $10\,\mathrm{cm}$	Robot morphology	47%	-39%
Transparent Bottles	Unseen objects	49%	-37%

Table 1: Summary of modifications to the robot and environment, and their effect on the performance of the base policy. Changing the background lighting, morphology, and objects leads to substantial degradation in performance compared to the original training conditions.

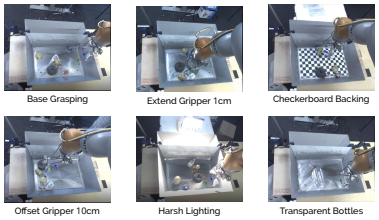


Figure 2: Views of from the robot camera for each of our six Challenge Tasks and the base grasping task.

4 Large-Scale Experimental Evaluation

We define then evaluate a simple technique for offline fine-tuning.

Our experiments model an "on the job" adaptation scenario, where a robot is initially trained to perform a general task (in our case, grasping diverse objects), and then the conditions of the task change in a drastic and substantial way as the robot performs the task, *e.g.* significantly brighter lighting, or a peculiar and unexpected type of object. The robot must adapt to this change quickly in order to recover a proficient policy. Handling these changes reflects what we expect to be a common requirement of reinforcement learning policies deployed in the real world: since an RL policy can learn from all of the experience that it has collected, there is no need to separate learning into clearly distinct training and deployment phases. Instead, it is likely desirable to allow the policy to simply continue learning "on the job" so as to adapt to these changes.

We define a very simple fine-tuning procedure for off-policy RL, as follows (Fig. 3).

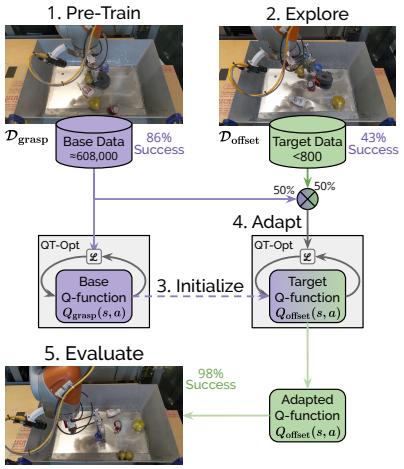


Figure 3: Schematic of the simple method we test in Sections 4 and 5. We pre-train a policy using the old data from the pre-training task, which is then adapted using the new data from the fine-tuning task.

4.1 A very simple fine-tuning method

First, we (1) pre-train a general grasping policy, as described in Section 3.1 and [23]. To fine-tune a policy onto a new target task, we (2) use the pre-trained policy to collect an exploration dataset of attempts on the target task. We then (3) initialize the same off-policy reinforcement learning algorithm used for pre-training (QT-Opt, in our case) with the parameters of the pre-trained policy, and both the target task and base task datasets[6] as the data sources (*e.g.* replay buffers). Using this training algorithm, we (4) update the policy, using a reduced learning rate, and sample training examples with equal probability from the base and target task datasets, for some number of update steps. Finally, we (5) evaluate the fine-tuned policy on the target task.

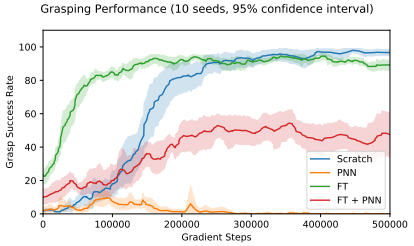
Our method is offline, *i.e.* it uses a single dataset of target task attempts and requires no robot interaction after the initial dataset collection to compute a fine-tuned policy, which may then be deployed onto a robot.

4.2 Evaluating fine-tuning in simulation

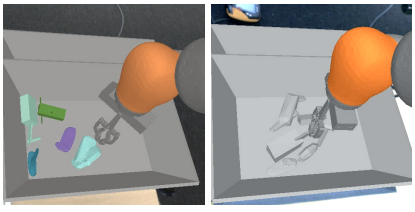
We evaluate the simple fine-tuning method we introduced in Section 4.1 on simulated adaptation challenge, and compare its performance to simpler and more-complex state-of-the-art alternatives (Fig. 4).

In this simulation experiment, we first pre-train a policy to grasp randomly-generated opaque objects (Fig. 4b), using a total of 3500 grasp attempts. We then change the task by turning the objects transparent (Fig. 4c, similar to the "Transparent Bottles" Challenge Task in Sec. 3), and challenge state-of-the-art methods to adapt to the transparent objects using only 75 grasp attempts, which is 0.5% of the data used to train from scratch.

In addition to testing our simple fine-tuning method ("FT") and motivated by our desire to find an adaptation method for a continual learning setting, we test a 2-task instantiation of a Progressive Neural Networks [57] ("PNN") model, and the combination of the PNN model with our simple fine-tuning method ("FT + PNN"). For comparison, we also train a policy on the target task from scratch, using 3500 new attempts.



(a) Performance



(b) Base Task (c) Target Task

Figure 4: Grasping performance (a) of four transfer methods on a simulated fine-tuning scenario, in which the robot is trained to grasp opaque objects (b) using 3500 attempts, but must adapt to transparent objects (c) using only 75 attempts. **Scratch:** Train a new policy from scratch (3500 attempts). **PNN:** Progressive Neural Networks. **FT:** Fine-tuning method from Sec. 4.1. **FT + PNN:** PNN model trained using the FT method.

[6] We assume this dataset was saved during training of the base policy

Challenge Task	Original Policy	Ours (exploration grasps)							Comparisons	
		25	50	100	200	400	800	**Best** (Δ)	Scratch	ImageNet
Checkerboard Backing	50%	67%	48%	71%	47%	89%	90%	**90% (+40)**	0%	0%
Harsh Lighting	32%	23%	16%	52%	44%	58%	63%	**63% (+31)**	4%	2%
Extend Gripper 1 cm	75%	93%	67%	80%	51%	90%	69%	**93% (+18)**	0%	14%
Offset Gripper 10 cm	43%	73%	50%	60%	56%	91%	98%	**98% (+55)**	37%	47%
Transparent Bottles	49%	46%	43%	65%	65%	58%	66%	**66% (+17)**	27%	20%
Baseline Grasping Task	86%	98%	81%	84%	78%	93%	89%	**98% (+12)**	0%	12%

Table 2: Summary of grasping success rates ($N \geq 50$) for the experiments by challenge task, fine-tuning method, and number of exploration grasps.

The simple fine-tuning method is able to achieve the baseline performance using the 75 exploration grasps in about $200,000$ gradient steps. The PNN model is unable to adapt to the new task with access to so few grasp attempts, and never exceeds 10% success rate, and believe this is likely due to the additional sample complexity of training its randomly-initialized "adapter" layers. Applying our simple offline fine-tuning method to the PNN model allows it to converge to a much higher 50% success rate, but still never achieve baseline performance.

Note that an important feature of the PNN model is that it is guaranteed to never experience performance degradation in the base task due to adaptation (*i.e.* "forgetting" or negative backward transfer). Though our much simpler method has no such guarantee, we find no evidence of forgetting in both simulated and real-world fine-tuning and continual learning experiments. See the Appendix for more details.

4.3 Evaluating offline fine-tuning for real-world grasping

We now turn our attention to evaluating this simple method's effectiveness as an adaptation procedure for end-to-end robot learning, and perhaps continual learning. Our goal is to determine whether the method is sample efficient, whether it works over a broad range of possible variations, and to determine whether it performs better than simpler ways of acquiring the target tasks.

With this goal in mind, we conduct a large panel of ablation experiments experiments on a real 7 DoF Kuka arm. These experiments evaluate the performance of our method across the diverse range of previously-defined Challenge Tasks and a continuum of target task dataset sizes, and compare this performance to two comparison methods.

For videos of our experimental results, see the project website.[7]

Collect exploration datasets First, we collect a dataset of 800 grasp attempts for each of our 5 challenge tasks (see Table 1) plus the base grasping task. We then partitioned each dataset into 6 tiers of difficulty by number of exploration grasps (25, 50, 100, 200, 400, and 800 grasp attempts), yielding 36 individual datasets.

Adapt policies using fine-tuning We train a fine-tuned policy for each of these 36 datasets using the procedure described above. We execute the fine-tuning algorithm for 500,000 gradient steps (see Sec. 6 for more information on how we chose this number) and use a learning rate of 10^{-4}, which is 25% of the learning rate used for pre-training. This yields 36 fine-tuned policies, each trained with a different combination of target task and target dataset size. This set of 36 policies includes 6 policies fine-tuned on data from the base grasping task, for validation.

Train comparison policies To provide points of comparison, we train two additional policies for each challenge task and the base grasping task, yielding 12 additional policies, for a total of 54.

The first comparison ("Scratch") is a policy trained using the aforementioned fine-tuning procedure and an 800-grasp data set, but using a randomly-initialized Q-function rather than the Q-function obtained from pre-training. The purpose of this comparison is to help us assess the contribution of the pre-trained parameters to the fine-tuning process' performance.

The second comparison ("ImageNet") is also trained using an identical fine-tuning procedure and the 800-grasp dataset, but initialized with the weights obtained by training the network to classify images from the ImageNet dataset [5]. The purpose of this comparison is to provide a comparison to a strong alternative to end-to-end RL for obtaining pre-training parameters. It uses a modified Q-function architecture in which we replace the convolutional trunk of the network with that of the popular ResNet50 architecture [71]. Refer to to Fig. 8 for a diagram of the Q-function architecture, and the Appendix for more details.

[7] For video results, see https://ryanjulian.me/continual-fine-tuning

Evaluate performance Finally, we evaluate all 54 policies on their target task by deploying them to the robot and executing 50 or more grasp attempts to calculate the policy's final performance. The full experiment required more than 15,000 grasp attempts and 14 days of real robot time, and was conducted over approximately one month.

The experiments are very challenging. For example, the "Transparent Bottles" task presents a major challenge to most grasping systems: the transparent bottles generally confuse depth-based sensors and, especially in cluttered bins, require the robot to singulate individual items and position the gripper in the right orientation for grasping. Although our base policy uses only RGB images, it is still not able to grasp the transparent bottles reliably, because they differ so much from the objects it observed during training. However, after fine-tuning with only 1 hour (100 grasp attempts) of experience, we observe that the transparent bottles can be picked up with a success rate of 66%, 20% better than the base policy. Similarly, the "Checkerboard Backing" challenge task asks the robot to differentiate edges associated with real objects from edges on an adversarial checkerboard pattern. It never needed this capability to succeed during pre-training, where the background is always featureless and grey, and all edges can be assumed to be associated with a graspable object. After 1 hour (100 grasp attempts) of experience, using our method the robot can grasp objects on the checkerboard background with a 71% success rate, 21% better than the base policy, and this success rate reaches 90% after 8 hours of experience (800 grasp attempts).

We present a full summary of our results in Table 2. Across the board, we observe substantial benefits from fine-tuning, suggesting that the robot can indeed adapt to drastically new condition with a modest amount of data: our most data-intensive experiment uses just 0.2% of the data used train the base grasping policy to similar performance. Our method consistently outperforms both the "ImageNet" and "Scratch" comparison methods. We provide more detailed analysis of this experiment in Section 6).

5 Evaluating Offline Fine-Tuning for Continual Learning

Now that we have defined and evaluated a simple method for offline fine-tuning, we evaluate its suitability for use in continual learning, which could allow us to achieve the goal of a robot that adapts to ever-changing environments and tasks. To do so, we define a simple continual learning challenge as follows (Fig. 5).

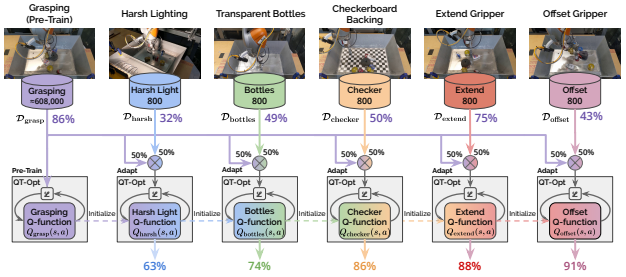


Figure 5: Flow chart of the continual learning experiment, in which we fine-tune on a sequence of conditions. Every transition to a new scenario happens after 800 grasps.

As in the fine-tuning experiments, we begin with a base policy pre-trained for general object grasping. Likewise, we also use our fine-tuning method to adapt the base policy to a target task, in this case "Harsh Lighting." We then use this *adapted* policy – *not* the base policy – as the initialization for another iteration of our fine-tuning algorithm, this time targeting "Transparent Bottles." We repeat this process until we have run out of new tasks, ending at the task "Offset Gripper 10cm," at which point we evaluate the policy on the last task.

We perform this experiment using 800 exploration-grasp datasets for each Challenge Task from our ablation study of online fine-tuning with real robots. We summarize the results in Table 3.

Challenge Task	Continual Learning	Δ	
		Base	Single
Harsh Lighting	63%	+32%	-
Transparent Bottles	74%	+25%	+8%
Checkerboard Backing	86%	+36%	−4%
Extend Gripper 1 cm	88%	+12%	−5%
Offset Gripper 10 cm	91%	+44%	−7%

Table 3: Summary of grasping success rates ($N \geq 50$) for the continual learning experiment by challenge task, and comparison to single-step fine-tuning.

Recall that our goal for this experiment is to determine whether continual fine-tuning incurs a significant performance penalty compared to the single-step variant, because we are interested in finding a building block for continual learning algorithms. We find that continual fine-tuning does not impose a drastic performance penalty compared to single-step fine-tuning. The continual fine-tuning policies for the "Checkerboard Backing," "Extend Gripper 1 cm," and "Offset Gripper 10 cm" challenges succeeded in grasping between 4% and 7% less often than their single-step fine-tuning counterparts, whereas the policy for the challenging "Transparent Bottles" case actually succeeded 8% *more* often. These small deltas are within the margin-of-error

of our evaluation procedure, so we conclude that the effect of continual fine-tuning on the performance compared to single-step fine-tuning is very small. Additionally, we find that the continually fined-tuned policies do not experience forgetting of the base task. This experiment demonstrates that our method can perform continual adaptation, and may serve as the basis for a continual end-to-end robot learning method.

6 Empirical Analysis

In this section, we aim to further investigate the efficiency, performance, and characteristics of our large-scale real-world adaptation experiments.

6.1 Performance and sample efficiency of fine-tuning

Figure 6 shows the success rates for our method from Table 2 against the amount of data used to achieve that success rate for selected tasks. The data indicate that a simple fine-tuning method can adapt policies to many new tasks using modest amounts of data. For instance, "Extend Gripper 1cm" and "Offset Gripper 10cm" both needed only 25 exploration grasps to achieve substantial gains in performance (+18% and +30%, respectively). All policies attain substantial performance gains over the base policy as they are fine-tuned with increasing amounts of data, but the relationship between data and performance is not linear. All policies experience a substantial improvement in performance after 100 or fewer exploration grasps. However, we observe that these performance improvements in the very low-data regime (*e.g.* ≤ 200 grasp attempts) are also unstable. We attribute both of these phenomenon to the early stopping problem, which we did not solve in this work, and discuss in detail below.

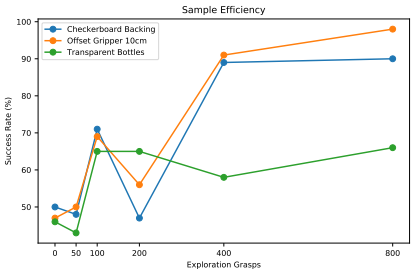


Figure 6: Sample efficiency of our fine-tuning method on selected real-robot challenge tasks.

6.2 Limitation: the early stopping problem

Our results indicate that offline fine-tuning can adapt robotic policies to substantial performance improvements with modest amounts of data An additional benefit of offline methods such as this one is that they are not limited by the need to preserve an always-sufficient exploration policy, as with online methods. However, we identify one significant drawback to the method compared to online fine-tuning.

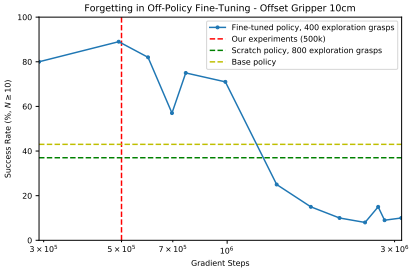


Figure 7: Evaluation performance of a single offline fine-tuning experiment at different numbers of gradient steps (optimization epochs).

A pure offline fine-tuning method has no built-in evaluation step which would inform us when the robot's performance on the target task has stopped improving, and therefore when we should stop fine-tuning with a fixed set of target task data. This is a subset of the off-policy evaluation problem [72]. Knowing when the policy stops improving is important: fine-tuning exists in a low-data regime, and repeatedly updating a neural network model with small amounts of data leads to overfitting onto that data. Not only does this degrade the performance on the target task, but also the ability of the network to adapt to new tasks later (*i.e.* for continual learning).

We can see this phenomenon in Figure 7 showing a real robot's performance on the "Offset Gripper 10cm" target task at different numbers of steps into an offline fine-tuning process that uses 400 exploration grasps. Performance quickly rises until around 500,000 gradient steps. Past this point, it precipitously drops and never recovers, dropping below even the initial performance of the base policy from which it was trained, as the initialization is being overwritten by overfitting to the target samples. The point at which overfitting begins is a function of the initialized model, target dataset, learning algorithm, and many other factors, and is not necessarily stable or easily predictable.

For the purposes of our large-scale fine-tuning study, we use this experiment and several others to determine that 500,000 gradient steps was an acceptable choice for the real-world experiments. The variance in the results in Table 2 and Figure 6, however, shows that this choice was not necessarily optimal for all of our tasks and datasets. In addition to off-policy evaluation metrics, we believe one practical solution to this problem for a continual learning robot is to use a mix of offline fine-tuning and online evaluation. The point at which performance stops improving represents when the

training process has exhausted the fine-tuning dataset of new information, and the robot must return to exploring online to continue improving.

6.3 Comparing initializing with RL to initializing with supervised learning

In order to answer the question of whether RL is better suited than supervised learning for pre-training a continually-learning robotic agent, in Section 4 we compared our results to an ImageNet-pretrained baseline. As shown in Table 2, the best performing ImageNet-based agent achieves a success rate of 47% on "Offset Gripper 10cm," a 4% improvement over the base policy performance. This result confirms our hypothesis that our RL-based pre-training is crucial for subsequent fine-tuning. Note that we first attempted to fine-tune these ImageNet-based policies while holding the ImageNet feature layers constant, but this procedure failed to achieve any non-zero success rates. This suggests that, unlike adapting computer vision networks to new visual tasks, adapting end-to-end robot learning to new sensorimotor tasks may require changing the features used to represent the problem, and not just the post-processing of said features.

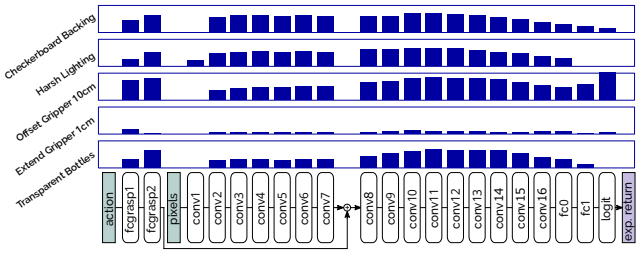


Figure 8: Analysis of parameter changes induced by different fine-tuning target tasks. This plot portrays the cosine distance between the parameters of the pre-trained and fine-tuned networks for our 5 fine-tuning target tasks. The bar heights are normalized by the magnitude of parameter changes induced in the Q-function network by fine-tuning the baseline grasping task.

Figure 8 highlights some of the changes that happen during the RL-based fine-tuning in greater detail. While it is unsurprising that primarily-visual challenges such as "Checkerboard Backing" and "Harsh Lighting" induce large changes in the parameters of the convolutional parts of the network, we observe that even 'Offset Gripper 10cm," a purely-morphological change to the robot, induces substantial changes to the network's image-processing parameters (e.g. layers conv2-conv7). We attribute this to the successful agent's need for hand-eye coordination to complete the task: offsetting the gripper not only changes robot morphology, it changes the location of the robot in its own visual field drastically. In order to perform effective visual servoing with a new morphology, both the image and action-processing parts of the network must be updated.

7 Conclusion and Future Work

For robots to be able to operate in unconstrained environments, they must be able to continuously adapt to new situations. We empirically studied this challenge by evaluating a state-of-the-art vision-based robotic grasping system, and testing its robustness to a range of new conditions such as varying backgrounds, lighting conditions, the shape and appearance of objects, and robot morphologies. We found that these new conditions degraded performance of the trained grasping system substantially. Motivated by this initial study, we explored how to adapt vision-based robotic manipulation policies by fine-tuning with off-policy reinforcement learning.

Our large-scale study shows that combining off-policy RL with a very simple fine-tuning procedure is an effective adaptation method, and this method is capable of achieving remarkable improvements in robot performance on new tasks with very little new data. Furthermore, our continual learning experiment shows that using this simple method in a continual setting imposes very little performance penalty compared to the single-step setting. This suggests that the combination of off-policy RL and fine-tuning can serve as a building block for future continual learning methods.

Our results comparing supervised learning-based initialization to initialization with RL highlight a familiar truism about robotics: that robotic agents must do more than perceive the world, they must also act in it. The ability to learn the combination of these two capabilities is what makes RL well-suited for creating continually-learning robots.

While our work demonstrated promising results on a real-world robotic grasping system under a wide range of scenarios, both perceptual and physical, further work is needed to understand how such adaptation performs on a broader range of robotic manipulation tasks. In future work, we would like to further assess our method's suitability for continual adaptation, by using longer continual learning sequences, and measuring forward and backward transfer during continual fine-tuning. We also plan to open source the models and datasets used in this work, to aid research in offline RL and off-policy evaluation metrics.

Acknowledgments

The authors thank Noah Brown and Ivonne Fajardo for their superb and unyielding support with real robot experiments. We also thank Alex Irpan and Eric Jang for their help with robot learning software, Yevgen Chebotar for his advice on early revisions of this work and always-insightful discussions, Dmitry Kalashnikov and Jake Varley for their help with QT-Opt, K.R. Zentner for her help with artwork for this paper, and Chad Richards for his help with copy-editing and presentation.

References

[1] S. Thrun. *Lifelong Learning Algorithms*, page 181–209. Kluwer Academic Publishers, USA, 1998. ISBN 0792380479.

[2] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

[3] J. Howard and S. Ruder. Universal language model fine-tuning for text classification, 2018.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] 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.

[6] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[7] M. Huh, P. Agrawal, and A. A. Efros. What makes imagenet good for transfer learning? *arXiv preprint arXiv:1608.08614*, 2016.

[8] S. Kornblith, J. Shlens, and Q. V. Le. Do better imagenet models transfer better? In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[9] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32 (11):1238–1274, 2013.

[10] N. Kohl and P. Stone. Machine learning for fast quadrupedal locomotion. In *The Nineteenth National Conference on Artificial Intelligence*, pages 611–616, July 2004.

[11] R. L. Tedrake. *Applied optimal control for dynamically stable legged locomotion*. Thesis, Massachusetts Institute of Technology, 2004. URL https://dspace.mit.edu/handle/1721.1/28742.

[12] S. Mahadevan and J. Connell. Automatic programming of behavior-based robots using reinforcement learning. *Artificial intelligence*, 55 (2-3):311–365, 1992.

[13] C. Finn and S. Levine. Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2786–2793. IEEE, 2017.

[14] J. Kober and J. R. Peters. Policy search for motor primitives in robotics. In *Advances in neural information processing systems*, pages 849–856, 2009.

[15] V. Gullapalli, J. A. Franklin, and H. Benbrahim. Acquiring robot skills via reinforcement learning. *IEEE Control Systems Magazine*, 14 (1):13–24, 1994.

[16] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[17] G. Schoettler, A. Nair, J. Luo, S. Bahl, J. A. Ojea, E. Solowjow, and S. Levine. Deep reinforcement learning for industrial insertion tasks with visual inputs and natural reward signals. In *International Conference on Learning Representations*, 2019.

[18] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg. Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks. *arXiv preprint arXiv:1810.10191*, 2018.

[19] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning. *arXiv preprint arXiv:1803.09956*, 2018.

[20] A. Ghadirzadeh, A. Maki, D. Kragic, and M. Björkman. Deep predictive policy training using reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2351–2358. IEEE, 2017.

[21] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *arXiv preprint arXiv:1903.11239*, 2019.

[22] 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.

[23] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673, 2018.

[24] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 512–519. IEEE, 2016.

[25] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in neural information processing systems*, pages 5074–5082, 2016.

[26] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[27] M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10 (Jul):1633–1685, 2009.

[28] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[29] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu. A survey on deep transfer learning. *Lecture Notes in Computer Science*, page 270–279, 2018. ISSN 1611-3349. doi:10.1007/978-3-030-01424-7_27. URL http://dx.doi.org/10.1007/978-3-030-01424-7_27.

[30] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu. Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning*, pages 193–200, 2007.

[31] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766, 2007.

[32] A. Gupta, A. Murali, D. Gandhi, and L. Pinto. Robot learning in homes: Improving generalization and reducing dataset bias, 2018.

[33] P. Sermanet, K. Xu, and S. Levine. Unsupervised perceptual rewards for imitation learning. *Proceedings of Robotics: Science and Systems (RSS)*, 2017. URL http://arxiv.org/abs/1612.06699.

[34] M. Hazara and V. Kyrki. Transferring generalizable motor primitives from simulation to real world. *IEEE Robotics and Automation Letters*, 4(2):2172–2179, 2019.

[35] F. Sadeghi and S. Levine. Cad2rl: Real single-image flight without a single real image. *Robotics: Science and Systems XIII*, Jul 2017. doi:10.15607/rss.2017.xiii.034. URL http://dx.doi.org/10.15607/RSS.2017.XIII.034.

[36] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep 2017. doi:10.1109/iros.2017.8202133. URL http://dx.doi.org/10.1109/IROS.2017.8202133.

[37] F. Sadeghi, A. Toshev, E. Jang, and S. Levine. Sim2real viewpoint invariant visual servoing by recurrent control. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4691–4699, 2018. doi:10.1109/CVPR.2018.00493. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Sadeghi_Sim2Real_Viewpoint_Invariant_CVPR_2018_paper.html.

[38] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *Robotics: Science and Systems XIV*, Jun 2018. doi:10.15607/rss.2018.xiv.010. URL http://dx.doi.org/10.15607/RSS.2018.XIV.010.

[39] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang. Solving rubik's cube with a robot hand, 2019.

[40] A. A. Rusu, M. Vecerík, T. Rothörl, N. M. O. Heess, R. Pascanu, and R. Hadsell. Sim-to-real robot learning from pixels with progressive nets. In *CoRL*, 2016.

[41] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1–8. IEEE, 2018.

[42] J. C. G. Higuera, D. Meger, and G. Dudek. Adapting learned robotics behaviours through policy adjustment. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5837–5843. IEEE, 2017.

[43] A. Hämäläinen, K. Arndt, A. Ghadirzadeh, and V. Kyrki. Affordance learning for end-to-end visuomotor robot control. *arXiv preprint arXiv:1903.04053*, 2019.

[44] M. A. Riedmiller, R. Hafner, T. Lampe, M. Neunert, J. Degrave, T. V. de Wiele, V. Mnih, N. M. O. Heess, and J. T. Springenberg. Learning by playing solving sparse reward tasks from scratch. In *ICML*, 2018.

[45] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016.

[46] A. Sax, B. Emi, A. R. Zamir, L. J. Guibas, S. Savarese, and J. Malik. Mid-level visual representations improve generalization and sample efficiency for learning visuomotor policies. In *Conference on Robot Learning*, 2019.

[47] K. Chatzilygeroudis and J.-B. Mouret. Using parameterized black-box priors to scale up model-based policy search for robotics. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9. IEEE, 2018.

[48] D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, pages 2450–2462, 2018.

[49] 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, 2019.

[50] K. Chatzilygeroudis, V. Vassiliades, and J.-B. Mouret. Reset-free trial-and-error learning for robot damage recovery. *Robotics and Autonomous Systems*, 100:236–250, 2018.

[51] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

[52] R. Kaushik, P. Desreumaux, and J.-B. Mouret. Adaptive prior selection for repertoire-based online adaptation in robotics. *Frontiers in Robotics and AI*, 6:151, 2020.

[53] J. Merel, S. Tunyasuvunakool, A. Ahuja, Y. Tassa, L. Hasenclever, V. Pham, T. Erez, G. Wayne, and N. Heess. Reusable neural skill embeddings for vision-guided whole body movement and object manipulation. *arXiv preprint arXiv:1911.06636*, 2019.

[54] D. Rastogi, I. Koryakovskiy, and J. Kober. Sample-efficient reinforcement learning via difference models.

[55] R. Jeong, J. Kay, F. Romano, T. Lampe, T. Rothorl, A. Abdolmaleki, T. Erez, Y. Tassa, and F. Nori. Modelling generalized forces with reinforcement learning for sim-to-real transfer. *arXiv preprint arXiv:1910.09471*, 2019.

[56] S. Ruder. An overview of multi-task learning in deep neural networks, 2017.

[57] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[58] L. Yen-Chen, M. Bauza, and P. Isola. Experience-embedded visual foresight. *arXiv preprint arXiv:1911.05071*, 2019.

[59] A. Nagabandi, K. Konoglie, S. Levine, and V. Kumar. Deep dynamics models for learning dexterous manipulation. *arXiv preprint arXiv:1909.11652*, 2019.

[60] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine. Visual reinforcement learning with imagined goals. In *Advances in Neural Information Processing Systems*, pages 9191–9200, 2018.

[61] D. Pathak, P. Mahmoudieh, G. Luo, P. Agrawal, D. Chen, F. Shentu, E. Shelhamer, J. Malik, A. A. Efros, and T. Darrell. Zero-shot visual imitation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun 2018. doi:10.1109/cvprw.2018.00278. URL http://dx.doi.org/10.1109/CVPRW.2018.00278.

[62] V. H. Pong, M. Dalal, S. Lin, A. Nair, S. Bahl, and S. Levine. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

[63] T. Yu, G. Shevchuk, D. Sadigh, and C. Finn. Unsupervised visuomotor control through distributional planning networks. *arXiv preprint arXiv:1902.05542*, 2019.

[64] A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. 2018.

[65] F. Alet, T. Lozano-Pérez, and L. P. Kaelbling. Modular meta-learning. *arXiv preprint arXiv:1806.10166*, 2018.

[66] A. Nagabandi, C. Finn, and S. Levine. Deep online learning via meta-learning: Continual adaptation for model-based rl. *arXiv preprint arXiv:1812.07671*, 2018.

[67] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. *arXiv preprint arXiv:1709.04905*, 2017.

[68] S. James, M. Bloesch, and A. J. Davison. Task-embedded control networks for few-shot imitation learning. In *Conference on Robot Learning*, pages 783–795, 2018.

[69] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. In *International Conference on Learning Representations*, 2018.

[70] A. Bonardi, S. James, and A. J. Davison. Learning one-shot imitation from humans without humans. *arXiv preprint arXiv:1911.01103*, 2019.

[71] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[72] A. Irpan, K. Rao, K. Bousmalis, C. Harris, J. Ibarz, and S. Levine. Off-policy evaluation via off-policy classification. In *Advances in Neural Information Processing Systems*, pages 5438–5449, 2019.

[73] J. Howard and S. Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

A Project Website and Experiment Videos

Please see our anonymous project website at https://ryanjulian.me/continual-fine-tuning for experiment videos and overview of the method.

To access the experiment videos directly, access the video directly at https://youtu.be/pPDVewcSpdc. Experiment videos start at 2:52.

B Additional Experiments on Forgetting

Figure 9 presents the opaque-to-transparent object grasping simulation experiment from Figure 4a, but includes a lower pane showing the performance of the fine-tuned policy on the base task (opaque object grasping) at every step of training.

The simple fine-tuning method presented in Section 4.1 ("FT") retains base task performance, even as it is adapted to solve the target task. When trained with our method, the target task column of the PNN model ("FT + PNN") starts at around 25% performance, drops to below 10% performance, then recovers base task performance in about $300,000$ gradient steps.

As the training process does not modify the parameters of the policy for the base task, the base task column of the PNN model ("PNN") retains its original performance on the base task.

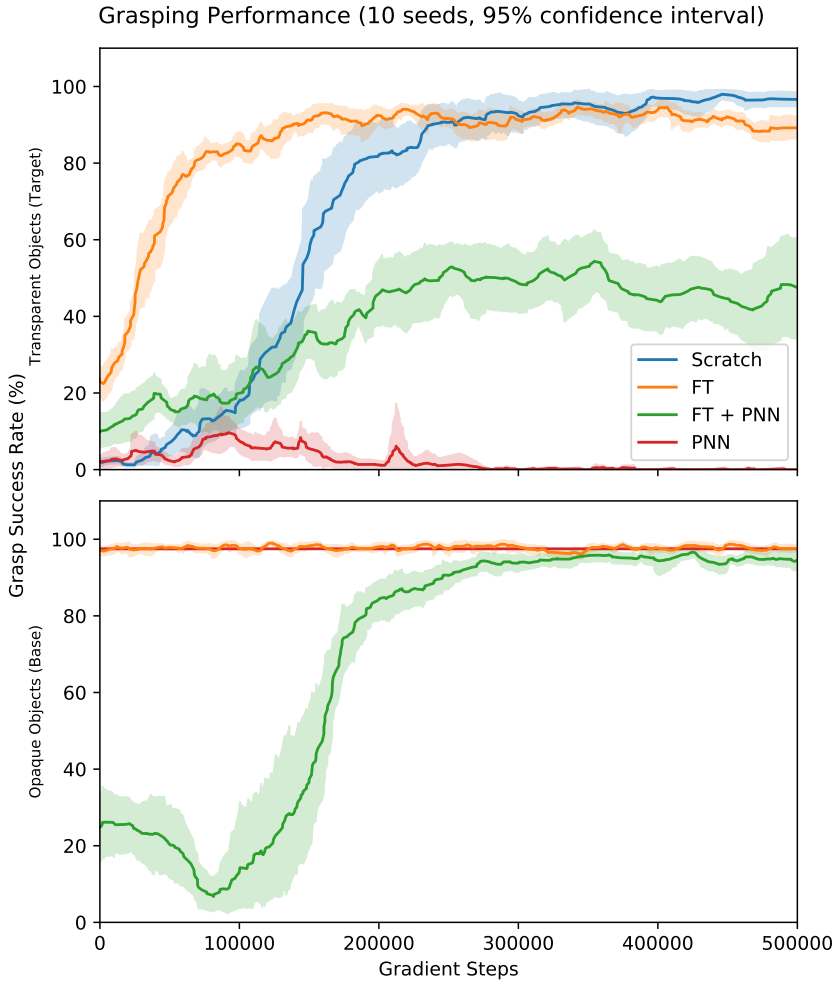


Figure 9: Grasping performance of four transfer methods on a simulated fine-tuning scenario, in which the robot is trained to grasp opaque objects using 3500 attempts, but must adapt to transparent objects using only 75 attempts. **Top:** Performance on the target task (grasping transparent objects). **Bottom:** Performance on the base task (grasping opaque objects). **Scratch:** Train a new policy from scratch (3500 attempts). **PNN:** Progressive Neural Networks. **FT:** Fine-tuning method from Sec. 4.1. **FT + PNN:** PNN model trained using the FT method.

C Assessing Fine-Tuning Techniques for End-to-End Robotic Reinforcement Learning

We propose a conceptual framework for fine-tuning algorithms, and use simulation experiments to assess the suitability of some algorithm variations for end-to-end robot learning.

"Fine-tuning" refers to a family of transfer learning techniques in which we seek to acquire a neural network for one task (which we will refer to as the "target" task) by making use of some or all of a network trained on a related task (the "base" task). This is a very common technique for quickly acquiring new tasks in computer vision [2, 7, 8] and natural language processing [73]. As collecting new robot experience data is expensive, our goal is to use as little target task data as possible. In this section, we first describe the general algorithmic sketch for fine-tuning, then enumerate some of the most common fine-tuning techniques. We evaluate the suitability of these techniques for end-to-end robot learning in Sections 4, 5, and 6.

C.1 Fine-Tuning: Conceptual Framework

We can organize fine-tuning for end-to-end reinforcement learning into four essential steps (See Fig. 3 for a detailed schematic). Different fine-tuning techniques change the details of one of these steps.

1. **Pre-training**: *Pre-train a policy to perform some base task, which is related to our target task.* In the experiments in this work, the base task is always indiscriminate object grasping. In computer vision and NLP, this step can often by skipped by making use of one of many pre-trained and publicly-available state-of-the-art vision and language models. We hope for a future in which this is possible in robotics.
2. **Exploration**: *Explore in the new target task, to collect data for adaptation.* In principle, any policy may be used for exploration in off-policy reinforcement learning. In our study, however, we always use the pre-trained policy for exploration, which we believe to be most representative of a real-world continual learning scenario.
3. **Initialization**: *Initialize the policy for the target task using some or all of the weights from the pre-trained policy.* The standard implementation of this step is to start with the entire pre-trained network. Some techniques may choose to use only a subset of the pre-trained network (*e.g.* truncating the last few layers of a CNN).
4. **Adaptation**: *Use the exploration data update the initialized policy to perform the new task.* The standard version of this step continues updating the entire initialized policy with the same algorithm and hyperparameters as was used for the pre-training process, but with the target task data. There are many variations on this step, including which parts of the network to update, at what learning rate, with what data, with which optimization algorithm, whether to add additional network layers, etc.
5. **Evaluation**: *Assess performance of the fine-tuned network on the new task.* If this step only happens once, we refer to such a technique as "offline fine-tuning," because the adaptation step never uses data from an updated policy. If this step happens repeatedly (*e.g.* exploration and evaluation are one-and-the-same), and its result is used for further adaptation to the same target task, we refer to a technique as "online fine-tuning." We explore both variations in our experiments.

Using this fine-tuning framework, we consider several variations of fine-tuning, and assess their suitability for end-to-end robotic RL. Notably, we neglect an analysis of pre-training techniques for fine-tuning reinforcement learning (i.e. (1)), which has a large and rapidly-growing body of research in the meta- and multi-task RL communities (see Sec. 2). Instead, we focus on initialization (2) and adaptation (3). All of our experiments use end-to-end off-policy reinforcement learning of an indiscriminate object grasping task for their pre-training step. Refer to Section 3.1 for details on our pre-training process.

C.2 Experiments in simulation

We use simulation experiments to evaluate the suitability of some fine-tuning variations, along the axes we defined in Section C.1.

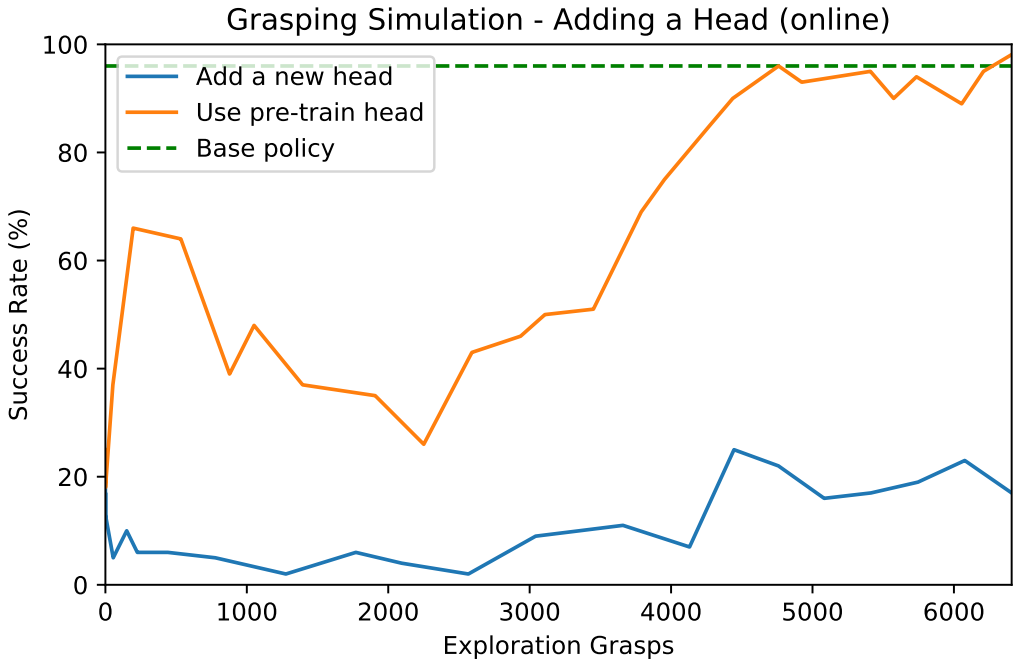


Figure 10: Comparison of fine-tuning performance for a policy which uses all base parameters, and a policy which initializes the head parameters from scratch. Re-initializing parameters has a negative effect on sample efficiency for fine-tuning.

C.2.1 Adding a new head and other selective initialization techniques

Selective-initialization techniques start the fine-tuning process with a policy that has some of its parameters initialized to random. For example, a popular variant is to "add a head" to a pre-trained neural network by omitting its last few layers from initialization, so that the new head can be trained to perform on the target task.

Figure 10 portrays a study of partial initialization for online fine-tuning using a simulated grasping experiment. In this experiment, the base task is "grasp opaque blocks" and the target task "grasp semi-transparent blocks." The base policy performance is 98% when trained from scratch on 43,000 grasp attempts. Both fine-tuned policies begin with low performance, around 15%. After 5000 exploration grasps (12% of the data used for the base policy), the performance of the full initialization policy has reached the base policy performance, while the policy with a new head has barely reached 30%. This gap shows that the combination of off-policy RL and selective initialization is unsuitable for sample-efficient fine-tuning.

Our experiments immediately make apparent the downsides of selective initialization for fine-tuning. In particular, online fine-tuning requires to maintain a policy that can competently explore the target task at all times. Any method which compromises the performance of such a policy – even temporarily – has a high risk of failing as a sample-efficient fine-tuning technique. The resulting performance gap, once created, is hard to recover. As a consequence, we find in simulation experiments that online fine-tuning with selective re-initialization takes a significant fraction of the pre-training samples to converge to baseline performance, making this family of fine-tuning methods sample inefficient.

C.3 Training with a mix of data from the base and target tasks

We experiment with mixing data from the pre-training task into the fine-tuning process (Fig. 11), and find this has a predictable relationship with sample efficiency in simulation: higher shares of target task data allow the fine-tuning policy to achieve higher performance faster.

Our goal is to design a fine-tuning algorithm for real robots which might be used for continual learning, and our conclusion from this brief study is that online fine-tuning is a poor fit for for this

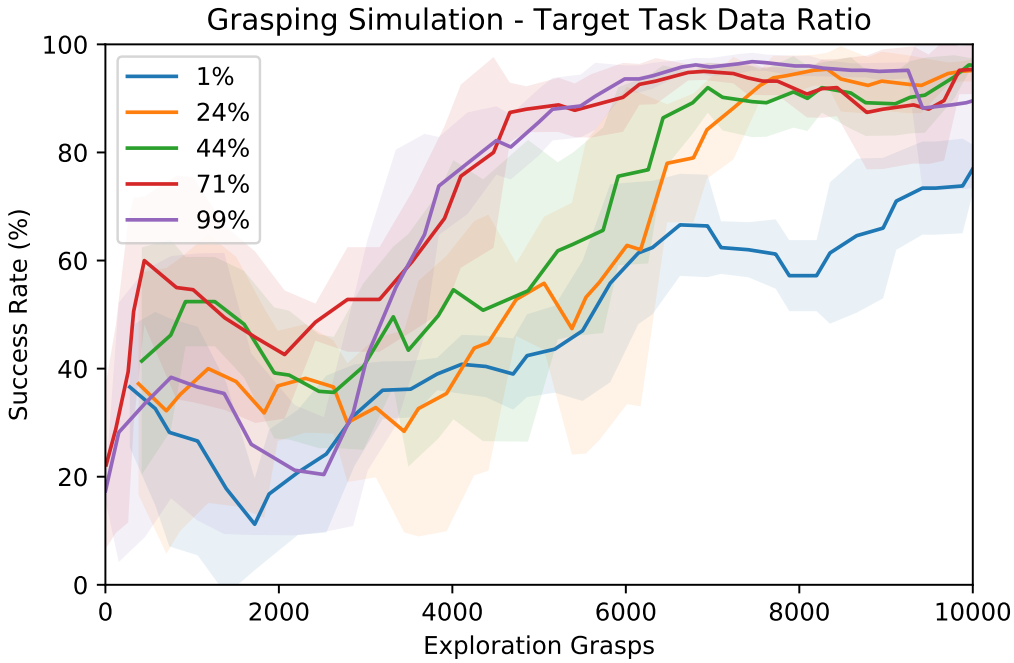


Figure 11: Performance curve for an online fine-tuning simulation experiment. The base policy is pre-trained to grasp opaque colored blocks, and the target task is to grasp semi-transparent blocks. Each curve represents a different fraction of target task data, and the remaining data is sampled from the base task. In simulation, the amount of target task data has a straightforward relationship with sample efficiency.

goal. In particular the experiments with selective re-initialization highlight the primary challenge of online fine-tuning: it only allows us to use algorithms which preserve the exploration ability of the policy at all times. We also believe that offline fine-tuning is more practical than online fine-tuning, due to the inherent complexity of placing a robot in the loop of a reinforcement learning algorithm. If used as part of a continual learning method, an offline method would also allow a robot to collect data on a new task piecemeal, and only attempt to adapt to that new task when it has collected enough data to succeed.

D Additional Experiment Details

D.1 Pre-Training Procedure

First, we train a Q-function network offline using data from 580,000 real grasp attempts over a corpus of 1,000 visually and physically diverse objects. Second, we continue training this network online[8] over the course of 28,000 real grasp attempts on the same corpus of objects. That is, we use a real robot to collect trials using the current network, update the network using these new trials, deploy the updated network to the real robot, and repeat.

D.2 Robustness Experiments with the Pre-Trained Policy

Background: We observe that conventional variations in the workspace surface, such as uniform changes in color or specularity, have no effect on the base policy's performance.

Lighting conditions: The base policy was trained in standard indoor lighting conditions, with no exposure to natural light or significant variation. We observe that mild perturbations in lighting conditions (i.e., those created by standard-intensity household lights) have no effect on the base policy's performance.

Gripper shape: No additional details.

Robot morphology: Note that during training this policy experienced absolutely no variation in robot morphology. We observe that translating the gripper laterally by up to $5\,\mathrm{cm}$ has no impact on performance.

Unseen objects: Based on our experiments, the system is robust to a broad variety of previously-unseen objects, as long as they have significant opaque components. For example, even though there are no drinking bottles in the training set, we find the system is able to pick up labeled drink bottles with 98% success rate. Success rates for other novel, opaque objects are similarly consistent with the baseline performance on the test set.

D.3 Comparison Methods from Section 4

The experiments "Scratch" and "ResNet 50 + ImageNet" both use 800 exploration grasps and the same update process as the other experiments.

D.3.1 "Scratch"

"Scratch" starts the grasping network with randomly-initialized parameters.

D.3.2 "ResNet 50 + ImageNet"

"ResNet 50 + ImageNet" refers to training a grasping network with an equivalent architecture to the other experiments, but with its convolutional layers replaced with a ResNet 50 architecture and pre-loaded with ImageNet features. We initialize the remaining fully-connected layers with random parameters, and concatenate the action input features at the end of the CNN (rather than the adding them in middle of the CNN, as in the original architecture). Note that in this comparison, the fine-tuning process still updates all parameters, including those of the ResNet50 sub-network.

D.4 Data Collection and Performance Evaluation

To reduce the variance of our evaluation statistics, we shuffle the contents of the bin between each trial by executing a randomly-generated sequence of sweeping movements with the end-effector.

D.5 Continual Learning Experiment

"Base" refers to the baseline grasping policy before fine-tuning, and "Single" refers to the best performance from the single-step fine-tuning experiment in Table 2. Note that because it is the first

[8] Following the example set by [23], we refer to this procedure as "online" rather than "on-policy" as the policy is still updated by the off-policy reinforcement learning algorithm.

step of the continual learning experiment, the policy for “Harsh Lighting” is identical to that of the 800-grasp variant of the single-step experiment.