A Framework for Efficient Robotic Manipulation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent advances in unsupervised representation learning significantly improved the sample efficiency of training Reinforcement Learning policies in simulated environments. However, similar gains have not yet been seen for real-robot learning. In this work, we focus on enabling data-efficient real-robot learning from pixels. We present a **F**ramework for **E**fficient **R**obotic **M**anipulation (FERM), a method that utilizes data augmentation and unsupervised learning to achieve sample-efficient training of real-robot arm policies from sparse rewards. While contrastive pre-training, data augmentation, and demonstrations are alone insufficient for efficient learning, our main contribution is showing that the combination of these disparate techniques results in a simple yet data-efficient method. We show that, given only 10 demonstrations, a single robotic arm can learn sparse-reward manipulation policies from pixels, such as reaching, picking, moving, pulling a large object, flipping a switch, and opening a drawer in just 30 minutes of mean real-world training time.

1 Introduction

Recent advances in deep reinforcement learning (RL) have given rise to unprecedented capabilities in autonomous decision making. Notable successes include learning to solve a diverse set of challenging video games Mnih et al. (2015); Berner et al. (2019); Vinyals et al. (2019); Badia et al. (2020), mastering complex classical games like Go, Chess, Shogi, and Hanabi Silver et al. (2016, 2017); Schrittwieser et al. (2019), and learning autonomous robotic control policies in both simulated Schulman et al. (2015, 2017); Laskin et al. (2020); Hafner et al. (2020) and real-world settings Levine et al. (2015); Kalashnikov et al. (2018). In particular, deep RL has been an effective method for learning diverse robotic manipulation policies such as grasping Pinto and Gupta (2016); Mahler et al. (2016); Levine et al. (2016); Gupta et al. (2018) and dexterous in-hand manipulation of objects Andrychowicz et al. (2018).

However, to date, general purpose RL algorithms have been extremely sample inefficient, which has limited their widespread adoption in the field of robotics. State-of-the-art RL algorithms for discrete Hessel et al. (2017) and continuous Lillicrap et al. (2015) control often require tens of millions of environment interactions to learn effective policies from image input Tassa et al. (2018), while training the Dota5 agent Berner et al. (2019) to perform competitively to human experts required an estimated 180 human-years of game play. Even when the underlying proprioceptive state is accessible, sparse reward robotic manipulation still needs millions of training samples Andrychowicz et al. (2017), an estimated 2 weeks of training in real time, to achieve reliable success rates on fundamental tasks such as reaching, picking, pushing, and placing objects.

Another common approach to learned robotic control is through imitation learning Zhang et al. (2018); Ho and Ermon (2016); Duan et al. (2017); Finn et al. (2017); Young et al. (2020), where a large number of expert demonstrations are collected and the policy is extracted through supervised learning by regressing onto the expert trajectories. However, imitation learning usually needs hundreds or

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

thousands of expert demonstrations, which are laborious to collect, and the resulting policies are bounded by the quality of expert demonstrations. It would be more desirable to learn the optimal policy required to solve a particular task autonomously.

In this work, rather than relying on transferring policies from simulation or labor intensive human input through imitation learning or environment engineering, we investigate how pixel-based RL applied to real robots can be made data-efficient. Recent progress in unsupervised representation learning Laskin et al. (2020); Stooke et al. (2020) and data augmentation Laskin et al. (2020); Kostrikov et al. (2020) has significantly improved the efficiency of learning with RL in simulated robotic Tassa et al. (2018) and video game Bellemare et al. (2013) environments. The primary strength of these methods is learning high quality representations from image input either explicitly through unsupervised learning or implicitly via data augmentation.

Building on these advances, we propose **F**ramework for **E**fficient **R**obotic **M**anipulation (FERM). FERM utilizes off-policy RL with data augmentation along with unsupervised pre-training to learn efficiently with a simple three-staged procedure. First, a small number of (10) demonstrations are collected and stored in a replay buffer. Second, the convolutional encoder weights are initialized with unsupervised contrastive pre-training on the demonstration data. Third, an off-policy RL algorithm is trained with augmented images on both data collected online during training and the initial demonstrations. Our core contribution is the novel combination of contrastive pre-training, online data augmentations, and utilizing a small number of demonstrations that together enable efficient real-robot learning from pixels. In contrast, prior leading algorithms that utilize these components individually are unable to learn efficiently on real robots.

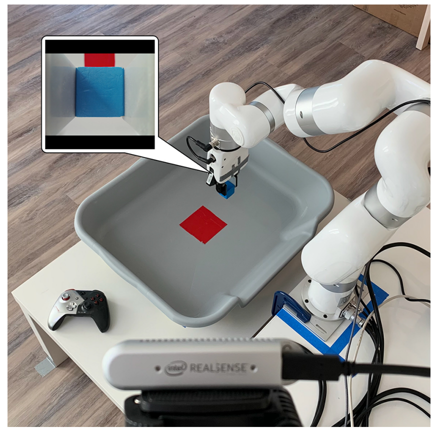


Figure 1: **F**ramework for **E**fficient **R**obotic **M**anipulation (FERM)enables robotic agents to learn skills directly from pixels in less than one hour of training. Our setup requires a robotic arm, two cameras, and a joystick to provide 10 demonstrations.

We summarize our key contributions and benefits of the FERM algorithm: *(1) Data-efficiency:* FERM enables learning optimal policies on 6 diverse manipulation tasks such as reaching, pushing, moving, pulling a large object, flipping a switch, drawer opening **in 15-50 minutes of total training time** for each task. *(2) Real-robot deployment:* FERM trains efficiently on real robotic hardware while prior related approaches that were successful in simulation Laskin et al. (2020,?) fail to learn robust real-robot policies. *(3) Simplicity:* FERM is a novel combination of existing ideas such as contrastive pre-training, data augmentation, and demonstrations that results in a simple and easy to reproduce algorithm. *(4) General & lightweight setup:* Our setup requires a robot, one GPU, two cameras, a handful of demonstrations, and a sparse reward function. These requirements are quite lightweight relative to setups that rely on Sim2Real, motion capture, multiple robots, or engineering dense rewards. To the best of our knowledge, this work is the first to show how recent advances in contrastive learning and data augmentation can enable efficient real-robot reinforcement learning from pixels.

2 Background

Soft Actor Critic: The Soft Actor Critic (SAC) Haarnoja et al. (2018) is an off-policy RL algorithm that jointly learns an action-conditioned state value function through Q learning and a stochastic policy by maximizing expected returns. SAC is a state-of-the-art model-free RL algorithm for continuous control from state Haarnoja et al. (2018) and, in the presence of data augmentations, from pixels as well Laskin et al. (2020); Kostrikov et al. (2020). In simulated benchmarks, such as DeepMind control Tassa et al. (2018), SAC is as data-efficient from pixels as it is from state. For this reason, we utilize it as our base RL algorithm for sparse-reward manipulation in this work. As an actor-critic method, SAC learns an actor policy π_θ and an ensemble of critics Q_{ϕ_1} and Q_{ϕ_2}.

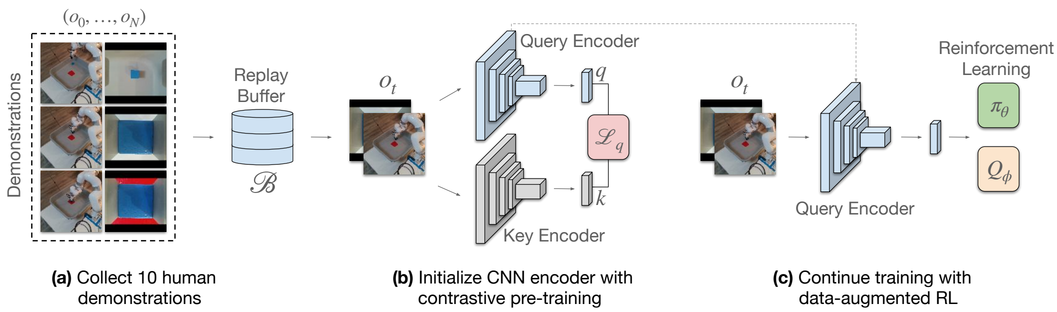


Figure 2: The FERM architecture. **(a)** Demonstrations are collected, and stored in a replay buffer. **(b)** The observations from the demonstrations are used to pre-train the encoder with a contrastive loss. **(c)** The encoder and replay buffer are then used to train an RL agent using an off-policy data-augmented RL algorithm.

To learn the actor policy, samples are collected stochastically from π_θ such that $a_\theta(o, \xi) \sim \tanh\left(\mu_\theta(o) + \sigma_\theta(o) \odot \xi\right)$, where $\xi \sim \mathcal{N}(0, I)$ is a sample from a normalized Gaussian noise vector, and then trained to maximize the expected return as shown in eq. 1.

$$\mathcal{L}(\theta) = \mathbb{E}_{a \sim \pi}\left[Q^\pi(o, a) - \alpha \log \pi_\theta(a|o)\right] \tag{1}$$

Simultaneously to learning the policy, SAC also trains the critics Q_{ϕ_1} and Q_{ϕ_2} to minimize the Bellman equation in Equation 2. Here, a transition $t = (o, a, o', r, d)$ is sampled from the replay buffer $\mathcal{B}$, where (o, o') are consecutive timestep observations, a is the action, r is the reward, and d is the terminal flag.

$$\mathcal{L}(\phi_i, \mathcal{B}) = \mathbb{E}_{t \sim \mathcal{B}}\left[\left(Q_{\phi_i}(o, a) - (r + \gamma(1 - d)Q_{\text{targ}})\right)^2\right] \tag{2}$$

The function Q_{targ} is the target value that the critics are trained to match, defined in Equation 3. The target is the entropy regularized exponential moving average (EMA) of the critic ensemble parameters, which we denote as $\bar{Q}_\phi$.

$$Q_{\text{targ}} = \left(\min_{i=1,2} \bar{Q}_{\phi_i}(o', a') - \alpha \log \pi_\theta(a'|o')\right) \tag{3}$$

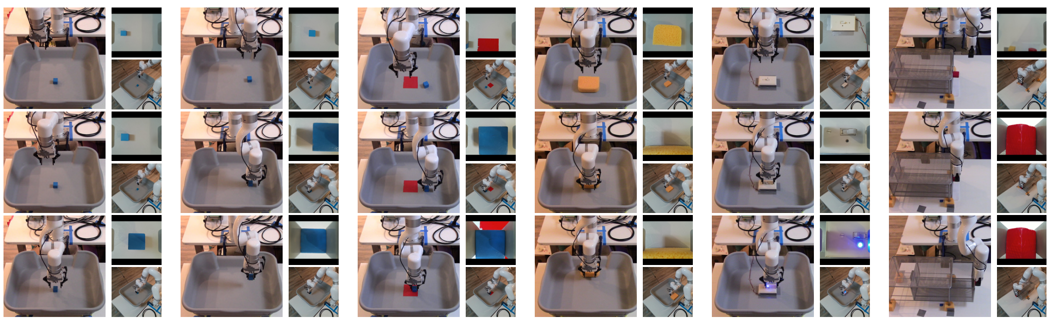
where (a', o') are the consecutive timestep action and observation, and α is a positive action-entropy coefficient. A non-zero action-entropy term improves exploration – the higher the value of α to more entropy maximization is prioritized over optimizing the value function.

Unsupervised Contrastive Pretraining: Contrastive learning Hadsell et al. (2006); LeCun et al. (2006); van den Oord et al. (2018); Wu et al. (2018); He et al. (2019); Chen et al. (2020); He et al. (2020); Hénaff et al. (2019) aims to maximize agreement between positive examples in data while minimizing agreement between negative examples. Contrastive methods require the specification of *query-key* pairs, also known as *anchors* and *positives*, which are similar data pairs whose agreement needs to be maximized. Given a query q and a key k , we seek to maximize the score $f_{\text{score}}(q, k)$ between them while minimizing them between the query q and negative examples in the dataset k_-. The score function is most often represented as an inner product, such as a dot product $f_{\text{score}}(q, k) = q^T k$ Wu et al. (2018); He et al. (2019) or a bilinear product $f_{\text{score}}(q, k) = q^T W k$ van den Oord et al. (2018); Hénaff et al. (2019), while other Euclidean metrics are also available Schroff et al. (2015); Wang and Gupta (2015). Modern contrastive approaches Chen et al. (2020); He et al. (2020); Hénaff et al. (2019); Laskin et al. (2020) employ the InfoNCE loss van den Oord et al.

(2018), which is described in Equation 4 and can also be interpreted as a multi-class cross entropy classification loss with K classes.

$$\mathcal{L}_q = \log \frac{\exp(q^T W k)}{\exp\left(\sum_{i=0}^{K} \exp(q^T W k_i)\right)} \tag{4}$$

In the computer vision setting, a simple and natural choice of query-key specification is to define queries and keys as two data augmentations of the same image. This approach, called instance discrimination, is used in most of the state-of-the-art representation learning methods for static images Chen et al. (2020); He et al. (2020) as well as RL from pixels Laskin et al. (2020). In the minibatch setting, which we also employ in this work, the InfoNCE loss is computed by sampling $K = \{x_1, \ldots, x_K\}$ images from the dataset, generating queries $Q = \{q_1, \ldots, q_K\}$ and keys $K = \{k_1, \ldots, k_K\}$ with stochastic data augmentations $q_i, k_i = \text{aug}(x_i)$, and using each augmented datapoint x_i as positives while the rest of the images are negatives.

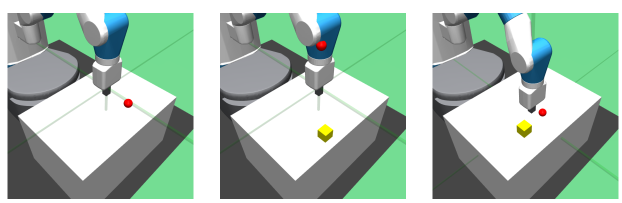


(a) Reach (b) Pickup (c) Move (d) Pull (e) Light Switch (f) Drawer Open

Figure 3: The set of real world tasks used in this work, along with their pixel observations. Each column shows initial, intermediate, and completion states of a rollout during evaluation of our optimal policy. The right two images comprise the processed camera image input, which are concatenated and used as the observational input for the RL agent. The sparse reward is only given when the robot completes the task. FERM is able to solve all 6 tasks within an hour, using only 10 demonstrations.

3 Method

Our proposed framework, shown in Figure 2, combines demonstrations, unsupervised pre-training, and off-policy model-free RL with data augmentation into one holistic Framework. FERM has three distinct steps – (i) minimal collection of demonstrations (ii) encoder initialization with unsupervised pre-training and (iii) online policy learning through RL with augmented data – which we describe in detail below.



(a) FetchReach (b) FetchPickAndPlace (c) FetchPush

Figure 4: Simulated environments from OpenAI gym Brockman et al. (2016) used in addition to our real robot experiments. We use the Fetch Gym Suite to investigate the core components of FERM .

Minimal Collection of Demonstrations: We initialize the replay buffer with a small number of expert demonstrations (we found 10 to be sufficient) for each task. Demonstrations are collected with a joystick controller, shown in Figure 1. Our goal is to minimize the total time required to acquire a skill for an RL agent, including both policy training as well as time required to collect demonstrations. While collecting a larger number of demonstrations certainly improves training speed, we find 10 demonstrations is already sufficient to learn skills quickly (see Fig. 7). For real world experiments, collecting 10 expert demonstrations can be done within 10 minutes which includes the time needed to reset the environment after every demonstration.

Unsupervised Encoder Pre-training: After initializing the replay buffer with 10 demonstrations, we pre-train the convolutional encoder with instance-based contrastive learning, using stochastic random crop Laskin et al. (2020) to generate query-key pairs. The key encoder is an exponentially moving average of the query encoder He et al. (2020), and the similarity measure between query-key pairs is the bi-linear inner product van den Oord et al. (2018) shown in Equation 4. Note that the bi-linear inner product is only used to pre-train the encoder. After pre-training, the weight matrix in the bi-linear measure is discarded.

Reinforcement Learning with Augmented Data: After pre-training the convolutional encoder on offline demonstration data, we train a SAC Haarnoja et al. (2018) agent with data augmentation Laskin et al. (2020) as the robot interacts with the environment. Since the replay buffer was initialized with demonstrations and SAC is an off-policy RL algorithm, during each minibatch update the agent receives a mix of demonstration observations and observations collected during training when performing gradient updates. The image augmentation used during training is random crop – the same augmentation used during contrastive pre-training.

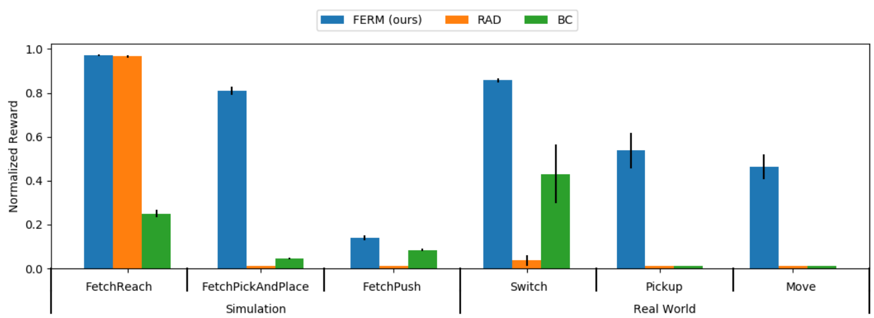


Figure 5: Baseline Comparisons. Shown are normalized rewards of the agent at the end of training for the simulated as well as the real robot results, as well as standard error. While FERM is able to learn all the tasks, the baseline RL agent (RAD) is unable to learn one sparse reward task without demonstrations. Conversely, with access to only 10 demonstrations, behavior cloning is unable to learn in the more difficult environments, and only succeeds on the simpler tasks (FetchReach, Light Switch). FERM and RAD are trained for 200k environment steps in simulated tasks, and until convergence for real world tasks (30 episodes for Switch and Pickup, 60 episodes for Move). BC is trained over the dataset for 200 epochs for both simulated and real world tasks. Simulated tasks are evaluated over 100 episodes, while real world tasks are evaluated over 30 episodes.

4 Experimental Evaluation

4.1 Experimental Setup

Real robot: We use the xArm xar robot for all real-world experiments. The end effector, a parallel two-jaw gripper, is position controlled with three degrees of freedom. At each step, the robot takes in an action containing the end effector and gripper aperture displacement.

Operation space: The range of motion of the gripper is confined to a 25 cm-high imaginary box above the manipulation surface. For majority of the tasks, objects are contained in a plastic tray approximately 40×34 cm in size measured at its bottom. Sponge padding was placed below the tray to absorb minor collisions between the gripper and the objects.

Input: We use two RGB cameras, one positioned over the shoulder for maximal view of the arm, and the other located within the gripper to provide a local object-level view. The over-the-shoulder camera is an Intel Realsense D415, with native resolution of 1280×720. Specifically, we only utilize the RGB frames during both training as testing. Inside the gripper, we use an Arducam 8MP camera module configured to 640×480 in resolution. Image frames from both cameras are cropped and down-sampled to 100×100 pixels for use in our training algorithm.

Demonstrations: Using a Xbox controller xbo, we teleoperate the robot by supplying the end effector and gripper aperture displacement. Collecting demonstrations for each task requires less than **10 minutes**, including resetting the environment.

4.2 Environments and Baselines

Environments: We evaluate FERM on six real-robotic manipulation tasks - reaching an object, picking up a block, moving a block to a target destination, pulling a large deformable object, flipping a switch, and opening a drawer. The block manipulation tasks (reach, pickup, move) are real-world adaptations of tasks from the OpenAI Gym Fetch suite Brockman et al. (2016). We utilize these three OpenAI gym environments for simulated environment experiments. Since our method uses demonstrations, we include pull, which has been used in prior work on imitation learning Rahmatizadeh et al. (2017); Florence et al. (2020). Flipping a switch is included as it demands precision, while drawer opening is a common task in existing simulated robotic benchmarks Yu et al. (2019). Details of task setup are provided in the supplementary material.

Baselines: We compare FERM to RAD Laskin et al. (2020), a leading supervised RL algorithm in simulated environments, and behavior cloning for our main results in Fig. 5. In the ablations section, we investigate each individual component of FERM . We investigate the contribution of each component of the FERM algorithm by removing one component - demonstrations, contrastive pre-training, or data augmentation - while keeping others fixed.

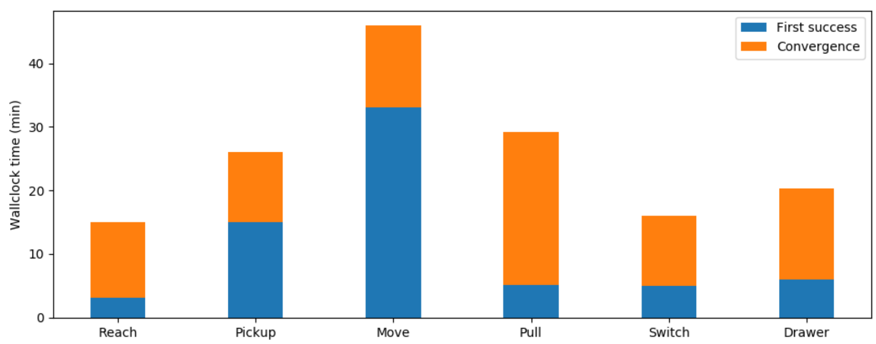


Figure 6: The speed at which our agents learn to complete the tasks. Plotted above are the times at which the policy first achieves a success, as well as when an optimal policy is learnt. Our method starts to complete the tasks in around 30 minutes of training, and as little as 3 minutes for simple tasks such as Reach.

Table 1: The success rates when evaluating the final policy learned by FERM over 30 episodes. Our method is able to achieve perfect success rate on the simpler tasks (Reach, Pickup, Light Switch, Drawer Open), and high success rates on the harder tasks (Move, Pull).

TASKS	REACH	PICKUP	MOVE	PULL	LIGHT SWITCH	DRAWER OPEN
# SUCCESSES (/30)	30	30	26	28	30	30
SUCCESS RATE (%)	100	100	86	93	100	100

4.3 Results

The main results of our investigation, including the time required to train an optimal policy as well the first successful task completion, are shown in Figure 5 and Table 1. We summarize the key findings below:

(i) On average, FERM enables a single robotic arm to learn optimal policies across all 6 tasks tested **within 30 minutes of training time** with a range of 15-50 minutes, which corresponds to to 20-80 episodes of training. (see Fig. 6 and Table 1).

(ii) When evaluated on 3 simulated and 3 real-robot tasks, FERM substantially outperforms RAD and behavior cloning baselines (see Fig. 5).

(iii) The time to first successful task completion is **on average 11 minutes** with a range of 3-33 minutes. The final policies achieve an **average success rate of 96%** with a range of 86-100% across the tasks tested, suggesting that they have converged to near-optimal solutions to the tasks.

(iv) Collecting demonstrations and contrastive pre-training does not introduce significant overhead in terms of time. Collecting 10 expert demonstrations with a joystick requires 10 minutes of human operation. Contrastive pre-training completes within one minute on a single NVIDIA 2080Ti GPU.

(v) FERM solves all 6 tasks using the **same hyperparameters** and without altering the camera setup, which demonstrates the ease of use and generality of the framework.

Altogether, an RL agent trained with FERM is able to learn optimal policies for the 6 tasks extremely efficiently. While prior work was able to solve dexterous manipulation tasks using RL with demonstrations in 2-3 hours of training Zhu et al. (2019), it also utilized dense rewards and more demonstrations. To the best of our knowledge, FERM is the first reinforcement learning method to solve a diverse set of sparse-reward robotic manipulation tasks directly from pixels in less than one hour.

4.4 Ablations

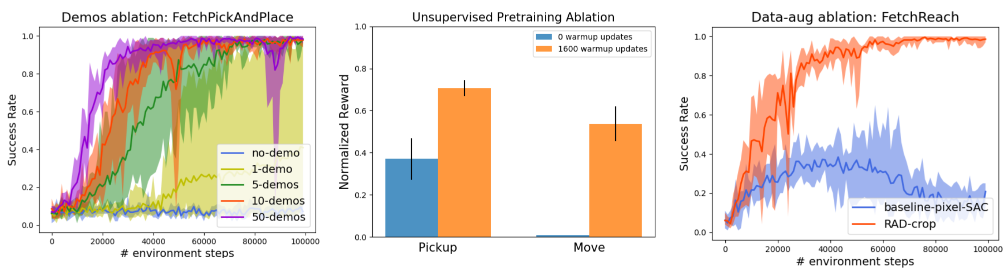


Figure 7: **Left:** We ablate the number of demonstrations required by FERM , and find that although the agent fails to learn with zero demonstrations, it can learn the PickAndPlace task efficiently using only 10 demonstrations. **Center:** We compare the performance of the move task with and without the use of pre-training on the real xArm robot. The plotted episode returns at convergence show that the contrastive pre-training substantially boosts performance. **Right:** Policy performance is measured by evaluation success rate. Using data augmentation, the agent achieves successful performance. Using non-augmented observations, the agent fails to learn the task.

In this section, we investigate how the three core components of FERM – demonstrations, contrastive pre-training, and data augmentation – contribute to the overall efficiency of the framework.

How many demonstrations are needed?

While sparse rewards are simpler to define, they pose an exploration challenge since the robot is unlikely to randomly stumble on a reward state. We address this issue by providing demonstrations to the RL agent. We ablate the number of demonstrations required to learn efficiently on the simulated pick and place task in Figure 7. We find that while the agent fails entirely with zero demonstrations, it is able to start learning the task with just one demonstration. While more demonstrations improve learning efficiency and reduce the variance of the policy, ten demonstrations suffice to learn quickly. We then evaluate the effectiveness of the 10 demonstrations by comparing our method to training behavior cloning. As shown in Figure 5, the 10 demonstrations are not enough to learn an effective policy. Refer to the supplementary material for further details.

How important is unsupervised contrastive pre-training? We next study the role of contrastive pre-training in FERM . We ablate our method with and without contrastive pre-training on the real world pickup and move task, shown in Figure 7, where we compare with (0), and without (1600) iterations of pre-training to initialize the encoder. With 1600 contrastive iterations, the agent is able to learn a successful policy while the other runs fail to learn. In the case of no pre-training at all, the agent is only able to succeed once during the entire hour of training.

Is online data augmentation necessary? To justify the use of data augmentation during online RL training, we compare the performance of SAC with and without data augmentation for a simple, dense reward reaching task. In the FetchReach environment, we use the dense reward $r = -d$ where d is the Euclidean distance between the gripper and the goal. As shown in Figure 7, without data augmentation, the RL agent is unable to learn the simple task, and asymptotically collapses. This motivates us to use data augmentation for our sparse reward tasks, which encounter even less learning signal.

5 Related Work

Imitation Learning: Imitation learning is a framework for learning autonomous skills from demonstrations. One of the simplest and perhaps most widely used forms of imitation learning is behavior cloning (BC) where an agent learns a skill by regressing onto demonstration data. BC has been successfully applied across diverse modalities including video games Ross et al. (2011), autonomous navigation Pomerleau (1988); Bojarski et al. (2016), autonomous aviation Giusti et al. (2016), locomotion Nakanishi et al. (2004); Kalakrishnan et al. (2009), and manipulation Duan et al. (2017); Zhang et al. (2018); Young et al. (2020); Rahmatizadeh et al. (2017). Other imitation learning approaches include Dataset Aggregation Ross et al. (2010), Inverse Reinforcement Learning Ng and Russell (2000); Abbeel and Ng (2004), and Generative Adversarial Imitation Learning Ho and Ermon (2016). A general limitation of imitation learning approaches is the requirement for a large number of demonstrations for each task Sharma et al. (2018). Although recent advancements have shown that imitation learning can learn with a much more modest amount of demonstrations Zhang et al. (2018); Rahmatizadeh et al. (2017); Florence et al. (2020), FERM can learn in the same number of episodes, of which the majority are spent with reinforcement learning.

Reinforcement Learning: Reinforcement Learning (RL) has been a promising approach for robotic manipulation due to its ability to learn skills autonomously, but has not achieved widespread adoption in real-world robotics. Recently, deep RL methods excelled at playing video games from pixels Mnih et al. (2015); Berner et al. (2019) as well as learning robotic manipulation policies from visual input Levine et al. (2015); Finn and Levine (2017); Haarnoja et al. (2018); Nair et al. (2018a). However, widespread adoption of RL in real-world robotics has been bottle-necked due to the data-inefficiency of the method, among other factors such as safety. Though there exist prior frameworks for efficient position controlled robotic manipulation Zhu et al. (2019), they still require hours of training per task and provide additional information such as a dense reward function. FERM is most closely related to other methods that use RL with demonstrations. Prior methods Nair et al. (2018b); Rajeswaran et al. (2017); Vecerík et al. (2017) solve robotic manipulation tasks from coordinate state input, rather than image input, by initializing the replay buffer of an RL algorithm with demonstrations to overcome the exploration problem in the sparse reward setting.

Data Augmentation: Image augmentation refers to stochastically altering images through transformations such as cropping, rotating, or color-jittering. It is widely used in computer vision architectures including seminal works such as LeNet Lecun et al. (1998) and AlexNet Krizhevsky et al. (2017). Data augmentation has played a crucial role in unsupervised representation learning in computer vision Hénaff et al. (2019); He et al. (2020); Chen et al. (2020), while other works investigated automatic generation of data augmentation strategies Cubuk et al. (2019). Data augmentation has also been utilized in prior real robot RL methods Kalashnikov et al. (2018); however, the extent of its significance for efficient training was not fully understood until recent works Laskin et al. (2020,?); Kostrikov et al. (2020), which showed that carefully implemented data augmentation makes RL policies from pixels as efficient as those from coordinate state. Finally, data augmentation has also been shown to improve performance in imitation learning Young et al. (2020). In this work, data augmentation comprises one of three components of a general framework for efficient learning.

Unsupervised Representation Learning: The goal of unsupervised representation learning is to extract representations of high-dimensional unlabeled data that can then be used to learn downstream tasks efficiently. Most relevant to our work is contrastive learning, which is a framework for learning effective representations that satisfy similarity constraints between a pair of points in dataset. In contrastive learning, latent embeddings are learned by minimizing the latent distance between similar data points and maximizing them between dissimilar ones. Recently, a number of contrastive learning methods Hénaff et al. (2019); He et al. (2019); Chen et al. (2020) have achieved state-of-the-art label-efficient training in computer vision. A number of recent investigations in robotics have leveraged contrastive losses to learn viewpoint invariant representations from videos Sermanet et al. (2018), manipulate deformable objects Yan et al. (2020), and learn object representations Pirk et al. (2019). In this work, we focus on instance-based contrastive learning Wu et al. (2018) similar to how it is used in vision He et al. (2020); Chen et al. (2020) and RL on simulated benchmarks Laskin et al. (2020); Stooke et al. (2020).

6 Limitations

Although FERM enables data-efficient deployment of RL onto real robots, the method also has a number of limitations. First, like most RL algorithms, FERM may require assistance for resets, and FERM policies can only solve the tasks that they were trained on and while they may display some degree of generalization to small changes such as object shape or perturbations, we do not expect FERM policies to generalize to qualitatively different tasks that were unseen during training. Second, while the tasks considered in this paper are standard robotics evaluation tasks, they all have relatively short horizons. Since FERM relies on a sparse reward signal to learn, we do not expect this framework to succeed in long-horizon sparse reward tasks, where random interaction with the reward is unlikely. Finally, we expect the performance of FERM to degrade if the visual conditions of the scene change substantially, which is likely in non-lab settings with frequent background distractors and lighting changes. Rather than addressing generalization to new tasks and visual settings or long-horizon settings, this paper focuses on the data-efficiency problem of training RL policies on real robots. We believe that data-efficient generalization and long-horizon problem solving are important open problem in robot learning that we leave for future work.

7 Conclusion and Future Work

We present FERM , a framework that combines demonstrations, unsupervised learning, and RL, to efficiently learn complex tasks in the real world. Using image input, our method is able to successfully solve a diverse set of tasks, all using the same hyperparameters, and from sparse reward. Due to the limited amount of supervision required, our work presents exciting avenues for applying RL to real robots in a quick and efficient manner.

References

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, 2015.

C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019. doi: 10.1038/s41586-019-1724-z. URL `https://doi.org/10.1038/s41586-019-1724-z`.

A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, D. Guo, and C. Blundell. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*, 2020.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–503, 2016. URL `http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html`.

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. Driessche, T. Graepel, and D. Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 10 2017. doi: 10.1038/nature24270.

J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *arXiv preprint arXiv:1911.08265*, 2019.

J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust region policy optimization. In *ICML*, pages 1889–1897, 2015.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas. Reinforcement learning with augmented data. *arXiv preprint arXiv:2004.14990*, 2020.

D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-End Training of Deep Visuomotor Policies. *CoRR*, abs/1504.00702, 2015. URL `http://arxiv.org/abs/1504.00702`.

D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. *ICRA*, 2016.

J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *ICRA*, 2016.

S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *ISER*, 2016.

A. Gupta, A. Murali, D. P. Gandhi, and L. Pinto. Robot learning in homes: Improving generalization and reducing dataset bias. In *Advances in Neural Information Processing Systems*, pages 9094–9104, 2018.

M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *arXiv preprint arXiv:1808.00177*, 2018.

M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning, 2017.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba. Hindsight experience replay. In *NeurIPS*, 2017.

T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Goldberg, and P. Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2018.

J. Ho and S. Ermon. Generative Adversarial Imitation Learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4565–4573. Curran Associates, Inc., 2016. URL `http://papers.nips.cc/paper/6391-generative-adversarial-imitation-learning.pdf`.

Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning, 2017.

C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-Shot Visual Imitation Learning via Meta-Learning. volume 78 of *Proceedings of Machine Learning Research*, pages 357–368. PMLR, 13–15 Nov 2017. URL `http://proceedings.mlr.press/v78/finn17a.html`.

S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto. Visual Imitation Made Easy. *CoRR*, abs/2008.04899, 2020. URL `https://arxiv.org/abs/2008.04899`.

M. Laskin, A. Srinivas, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, 2020.

A. Stooke, K. Lee, P. Abbeel, and M. Laskin. Decoupling representation learning from reinforcement learning, 2020.

I. Kostrikov, D. Yarats, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A tutorial on energy-based learning. 2006.

A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Z. Wu, Y. Xiong, S. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance-level discrimination. *arXiv preprint arXiv:1805.01978*, 2018.

K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, 2020.

K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

O. J. Hénaff, A. Srinivas, J. De Fauw, A. Razavi, C. Doersch, S. Eslami, and A. v. d. Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019.

F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015.

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

xArm 7. URL `https://store.ufactory.cc/products/xarm-7-2020`.

Xbox wireless controller. URL `https://www.xbox.com/en-US/accessories/controllers/xbox-wireless-controller`.

R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine. Vision-Based Multi-Task Manipulation for Inexpensive Robots Using End-To-End Learning from Demonstration. *CoRR*, abs/1707.02920, 2017. URL `http://arxiv.org/abs/1707.02920`.

P. R. Florence, L. Manuelli, and R. Tedrake. Self-Supervised Correspondence in Visuomotor Policy Learning. *IEEE Robotics Autom. Lett.*, 5(2):492–499, 2020. doi: 10.1109/LRA.2019.2956365. URL `https://doi.org/10.1109/LRA.2019.2956365`.

T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning. In *Conference on Robot Learning (CoRL)*, 2019. URL `https://arxiv.org/abs/1910.10897`.

H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, and V. Kumar. Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3651–3657, 2019. doi: 10.1109/ICRA.2019.8794102.

S. Ross, G. J. Gordon, and D. Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. In G. J. Gordon, D. B. Dunson, and M. Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, volume 15 of *JMLR Proceedings*, pages 627–635. JMLR.org, 2011. URL `http://proceedings.mlr.press/v15/ross11a/ross11a.pdf`.

D. Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 1, [NIPS Conference, Denver, Colorado, USA, 1988]*, pages 305–313. Morgan Kaufmann, 1988. URL `http://papers.nips.cc/paper/95-alvinn-an-autonomous-land-vehicle-in-a-neural-network`.

M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to End Learning for Self-Driving Cars. *CoRR*, abs/1604.07316, 2016. URL `http://arxiv.org/abs/1604.07316`.

A. Giusti, J. Guzzi, D. C. Ciresan, F. He, J. P. Rodriguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. D. Caro, D. Scaramuzza, and L. M. Gambardella. A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots. *IEEE Robotics Autom. Lett.*, 1 (2):661–667, 2016. doi: 10.1109/LRA.2015.2509024. URL `https://doi.org/10.1109/LRA.2015.2509024`.

J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato. Learning from demonstration and adaptation of biped locomotion. *Robotics Auton. Syst.*, 47(2-3):79–91, 2004. doi: 10.1016/j.robot.2004.03.003. URL `https://doi.org/10.1016/j.robot.2004.03.003`.

M. Kalakrishnan, J. Buchli, P. Pastor, and S. Schaal. Learning locomotion over rough terrain using terrain templates. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 11-15, 2009, St. Louis, MO, USA*, pages 167–172. IEEE, 2009. doi: 10.1109/IROS.2009.5354701. URL `https://doi.org/10.1109/IROS.2009.5354701`.

S. Ross, G. Gordan, and A. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *arXiv preprint arXiv:1011.0686*, 2010.

A. Y. Ng and S. J. Russell. Algorithms for inverse reinforcement learning. In P. Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pages 663–670. Morgan Kaufmann, 2000.

P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In C. E. Brodley, editor, *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, volume 69 of *ACM International Conference Proceeding Series*. ACM, 2004. doi: 10.1145/1015330.1015430. URL `https://doi.org/10.1145/1015330.1015430`.

P. Sharma, L. Mohan, L. Pinto, and A. Gupta. Multiple interactions made easy (mime): Large scale demonstrations data for imitation. *arXiv preprint arXiv:1810.07121*, 2018.

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.

A. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine. Visual Reinforcement Learning with Imagined Goals. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 9209–9220, 2018a. URL `http://papers.nips.cc/paper/8132-visual-reinforcement-learning-with-imagined-goals`.

A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel. Overcoming Exploration in Reinforcement Learning with Demonstrations. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 6292–6299. IEEE, 2018b. doi: 10.1109/ICRA.2018.8463162. URL `https://doi.org/10.1109/ICRA.2018.8463162`.

A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

M. Vecerík, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. A. Riedmiller. Leveraging Demonstrations for Deep Reinforcement Learning on Robotics Problems with Sparse Rewards. *CoRR*, abs/1707.08817, 2017. URL `http://arxiv.org/abs/1707.08817`.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, 2017. doi: 10.1145/3065386. URL `http://doi.acm.org/10.1145/3065386`.

E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. AutoAugment: Learning Augmentation Strategies From Data. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 113–123. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00020. URL `http://openaccess.thecvf.com/content_CVPR_2019/html/Cubuk_AutoAugment_Learning_Augmentation_Strategies_From_Data_CVPR_2019_paper.html`.

P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine. Time-Contrastive Networks: Self-Supervised Learning from Video. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 1134–1141. IEEE, 2018. doi: 10.1109/ICRA.2018.8462891. URL `https://doi.org/10.1109/ICRA.2018.8462891`.

W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto. Learning Predictive Representations for Deformable Objects Using Contrastive Estimation. *CoRR*, abs/2003.05436, 2020. URL `https://arxiv.org/abs/2003.05436`.

S. Pirk, M. Khansari, Y. Bai, C. Lynch, and P. Sermanet. Online Object Representations with Contrastive Learning. *CoRR*, abs/1906.04312, 2019. URL `http://arxiv.org/abs/1906.04312`.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]