On the Effectiveness of Fine-tuning Versus Meta-RL for Robot Manipulation

Anonymous Author(s)
Affiliation
Address
email

Abstract: It is often said that robots should have the ability to leverage knowledge from previously learned tasks in order to learn new ones quickly and efficiently. Meta-learning approaches have emerged as a popular solution to achieve this. However, these approaches have mainly been studied in either supervised learning settings or in full-state, reinforcement learning settings with shaped rewards and narrow task distributions. Moreover, the necessity of meta learning over simpler, pretraining setups, have been called into question within the supervised learning domain. We investigate meta-learning approaches in a vision-based, sparse-reward robot manipulation setting, where evaluations are made on completely novel tasks. Our findings show that, when meta-learning approaches are evaluated on different tasks (rather than different variations), multi-task pretraining with fine-tuning on new tasks can perform equally as well as meta-pretraining with meta test-time adaptation. This is both enlightening and encouraging for future research in pretraining for robot learning, as multi-task learning tends to be simpler and computationally cheaper than meta-reinforcement learning.

Keywords: Multi-task Pretraining, Meta-RL, Vision-based robot manipulation

1 Introduction

One of the major gaps between human and machine intelligence is the sample efficiency of learning. In contrast to how humans can leverage past knowledge to learn a new task from a few examples, current machine learning systems often require a large amount of data and heavy supervision to achieve even a single task. To bridge this gap, meta-learning has become a popular approach — it uses many tasks to meta-train an optimal learning strategy, which enables few-shot generalization on a test task. Efficient adaptation is particularly desirable in robot learning: it could significantly save on the cost of data collection, real-world exploration, etc. when learning a new task.

Meta-learning methods have had the most success in supervised learning settings [1, 2, 3], specifically few-shot image classification, where the goal is to learn a classifier to recognize unseen classes during a test-time training phase with limited labeled data. Recent work has found that variations of simple pretraining and fine-tuning can perform equally as well as more complex meta-learning approaches [4, 5, 6, 7].

One popular line of approach to introduce meta-learning to robot learning systems is meta-reinforcement learning (meta-RL), where an agent is trained and adapts using a base reinforcment learning algorithm. In contrast to few-shot classification, simple pretraining and fine-tuning is not known to out perform meta-RL — our hypothesis for this intriguing discrepancy in literature is simple: the computer vision (CV) community evaluates their approaches on distinct test tasks (e.g. classifying dogs, cats, and birds), while the meta-RL community evaluates on *variations* of the same train-time tasks; for example, varying transition dynamics (e.g. different friction parameters) or varying reward functions (e.g. running forward v.s. running backward) are better categorized

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

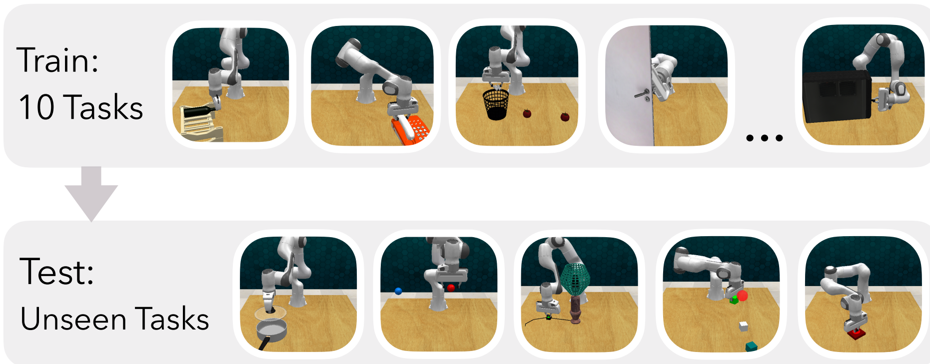

Figure 1: We study a challenging setup in vision-based, sparse-rewarded robot manipulation, where training and testing use strictly disjoint sets of tasks. We compare across different meta-reinforcement learning (meta-RL) algorithms and multi-task pretraining with fine-tuning. Our investigation concludes that, fine-tuning on novel tasks performs equally as well as meta test-time adaptation, can overcome sparse rewards on unseen test tasks, and perform significantly better than training from scratch.

as variations rather than different tasks, as discussed in recent work [8, 9]. *Variation* adaptation is inherently easier than *task* adaptation, and does not paint a full picture of the shortcomings of meta-RL. Moreover, most meta-RL methods (with a few exceptions, discussed in 4.2) have been studied in fully observable settings or with shaped rewards [10, 11, 12], neglecting more realistic real-world scenarios of robot learning, where rewards are often sparse, and observations are high-dimensional (e.g. images, point-clouds, etc).

In this work, we are hence motivated to study meta-RL across a truly diverse set of tasks that are better aligned with realistic robot learning challenges. We use RLBench [8], a simulation benchmark that provides numerous vision-based and sparse-rewarded manipulation tasks. We train and test on strictly disjoint sets of tasks: for example, an agent could be trained to pick up cups, take a USB out of a computer, and reach target locations, while at test time, adaptation would be evaluated on completely unseen tasks, such as lifting blocks and pushing buttons.

We investigate two representative meta-RL algorithms of differing paradigms: Reptile [11] — a gradient-based method, and PEARL [12] — a context-based method. Results from this study are enlightening: multi-task pretraining, followed by fine-tuning on novel tasks, performs equally as well as the meta-RL algorithms, while being much simpler and less computationally expensive to train. In light of this, we advocate for future research in pretraining for robot learning to shift towards more challenging benchmarks, and involve multi-task pretraining with fine-tuning as a simple, yet strong baseline.

2 Experiments

2.1 Task Setup

We use RLBench [8], a vision-based manipulation benchmark and learning environment with sparse rewards. The environment has more than 100 diverse, real-world inspired tasks, and provides easy access to expert demonstrations for all tasks, which has been shown as vital for overcoming the exploration problem imposed by the benchmark's sparse rewards [13, 14].

To ensure the experiment results do not get affected by arbitrary task selection, we design a comprehensive set of train-test task splits that resemble cross-validation in the supervised learning setting. Specifically, we use a fixed set of 11 RLBench tasks and create 5 splits. Each split uses a (randomly selected) held-out task and trains an agent on the remaining 10 tasks.

2.2 Training Setup

We use C2F-ARM [14] as the base off-policy RL algorithm. This was chosen because more widely-used RL algorithm, such as DDPG [15], TD3 [16], SAC [17], and DrQ [18] are known to fail [13] in RLBench due to the challenging setup. C2F-ARM [14] is a vision-based robot manipulation algorithm that can learn sparse-reward reinforcement learning tasks by using a small number of initial demonstrations. C2F-ARM is described in more detail in Section 4. Note that RL^2 is excluded from this section because it is on-policy.

Reptile-C2F-ARM modifies the off-policy batch update in C2F-ARM with an inner- and outer-loop proposed in Reptile [11]. At the beginning of training, each task is given a separate replay buffer, which is initialized with transitions collected from 5 demonstration trajectories and continuously appended with the agent's online experiences. During training, for multiple steps in the inner loop, the agent draws a batch from the replay buffer of a randomly sampled task and performs updates to the Q-attention. In the outer loop, the network gets a soft update to mix the parameters from before and after the inner loop updates.

PEARL-C2F-ARM conditions a context embedding to the Q-attention network. To obtain the context for a task, a batch of transitions is drawn from a window of recent agent experiences, and a separate convolution encoder is used to first encode the image observations individually. Then, each image embedding is concatenated with the action and reward, and together encoded into a single vector. Finally, the context embeddings are sampled as proposed in [12]. The context encoder is additionally trained with a KL loss.

MT-C2F-ARM jointly trains C2F-ARM on all training tasks. During each replay batch update, both MT-C2F-ARM draw samples from multiple task replay buffers. During each replay update, a fixed number of tasks (less or equal to the total number of available training tasks) are randomly selected, then an equal number of samples are drawn for each task to construct the replay batch.

2.3 Test-time Adaptation Setup

Both MT-C2F-ARM and Reptile-C2F-ARM use the same C2F-ARM update and adapt the agent parameters to the new task via gradient descent. Adaptation for PEARL-C2F-ARM is done by gathering rollout samples in the new environment and re-computing the context embeddings, hence running only inference on the agent's policy model.

2.4 Results

The first set of evaluations are the most challenging for adaptation: an unseen **test-time task** given **0 demonstrations**. The agent is expected to leverage knowledge and skills gained in the 10 training tasks and perform intelligent exploration on the test task, without any guidance from demonstrations. Results for this setup are presented in the top row of Figure 2. Across all 5 test tasks, multi-task fine-tuning performs equally as well as Reptile while performing significantly better than both PEARL and training from scratch.

We next investigate the effect of reward sparsity on test-time performance. We now provide test-time demonstrations of each of the methods, as an aid for exploration under sparse reward. Results in the second and third row of Figure 2 show how the methods behave when given 1 and 2 *test-time demonstrations*. The fact that increasing the number of demonstrations improves training from scratch performance is unsurprising, however, one intriguing observation is that this effect is less apparent for MT-C2F-ARM and Reptile-C2F-ARM methods. This is encouraging evidence that **fine-tuning significantly reduces (or even omit) the need for demonstrations in sparse rewarded tasks, with little loss to performance.** We further investigate the various properties of fine-tuning C2F-ARM in Section 4.

Apparent from Figure 2 is that PEARL does not seem equipped to handle such a disjoint train-test split. Recall that PEARL adapts without model parameter updates, and the only way to understand a

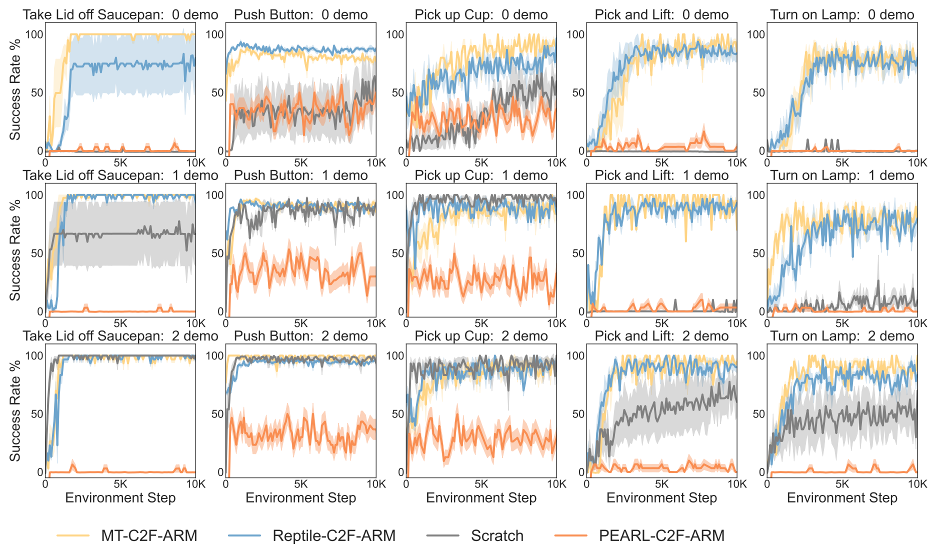

Figure 2: When varying the number of test-time demonstrations (from 0-2 trajectories), does multi-task pretraining and fine-tuning outperform meta-RL methods on unseen tasks? We perform a "cross-validation" style evaluation: from a set of 11 RLBench tasks, 1 is held out for test-time evaluation, while the other 10 are used for training (and are given 5 demos). This is done for each of the 5 tasks above. Multi-task pretraining and fine-tuning perform equally as well or better than meta-RL. Unsurprisingly, fine-tuning (from either the Multi-task or Reptile agent) requires fewer demonstrations than training from scratch. Solid lines are average over 5 seeds, with shaded regions representing standard errors

new task is via aggregating new experiences into the context. However, the context encoder clearly fails at providing a useful context for the unseen tasks. we hypothesize this is due to our tasks setup: the training tasks are so visually disjoint that the agent never needs to learn high-quality context embeddings to infer which task it should do. This is different from the original experiment setup in PEARL, where *variations* are treated as "tasks", meaning that the observations from different "tasks" are similar or even identical; in order to disentangle the correct task to perform, the network is heavily motivated to read the context. Further evidence towards this hypothesis can be seen by looking at the zero-shot performance of the PEARL agent (i.e., environment steps $= 0$), where it starts with the same performance as multi-task and Reptile agents but doesn't improve. This suggests the meaningful performance that PEARL does achieve should be attributed to the pretraining and not test-time adaptation.

3 Conclusion

We study the setting of meta-RL on vision-based robot manipulation with sparse rewards, across a *truly* diverse set of tasks. We showed that when meta-RL is tested on truly diverse robot reinforcement learning tasks, simple multi-task RL followed by finetuning can perform equally as well, while being simpler and less expensive to train. Our conclusion is consistent with findings within the CV community, and we hope it is an initial step towards understanding the subtleties between meta-RL and multi-task pretraining on robot learning systems. Our study lies within the manipulation domain, and calls for future investigations on other robotic tasks and settings.

References

[1] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. *Advances in Neural Information Processing Systems*, 29:3630–3638, 2016.

[2] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. *Intl. Conference on Learning Representations*, 2017.

[3] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Intl. Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

[4] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang. A closer look at few-shot classification. *Intl. Conference on Learning Representations*, 2019.

[5] G. S. Dhillon, P. Chaudhari, A. Ravichandran, and S. Soatto. A baseline for few-shot image classification. *Intl. Conference on Learning Representations*, 2020.

[6] Y. Tian, Y. Wang, D. Krishnan, J. B. Tenenbaum, and P. Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, pages 266–282. Springer, 2020.

[7] Y. Chen, Z. Liu, H. Xu, T. Darrell, and X. Wang. Meta-baseline: exploring simple meta-learning for few-shot learning. In *Intl. Conference on Computer Vision*, pages 9062–9071, 2021.

[8] S. James, Z. Ma, D. Rovick Arrojo, and A. J. Davison. RLBench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

[9] Z. Mandi, F. Liu, K. Lee, and P. Abbeel. Towards more generalizable one-shot visual imitation learning. *arXiv preprint arXiv:2110.13423*, 2021.

[10] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. In *Conference on Robot Learning*, pages 357–368. PMLR, 2017.

[11] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[12] K. Rakelly, A. Zhou, D. Quillen, C. Finn, and S. Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables, 2019.

[13] S. James and A. J. Davison. Q-attention: Enabling Efficient Learning for Vision-based Robotic Manipulation. *IEEE Robotics and Automation Letters*, 2022.

[14] S. James, K. Wada, T. Laidlow, and A. J. Davison. Coarse-to-Fine Q-attention: Efficient Learning for Visual Robotic Manipulation via Discretisation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[15] 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. *Intl. Conference on Learning Representations*, 2015.

[16] S. Fujimoto, H. Van Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. *Intl. Conference on Machine Learning*, 2018.

[17] 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.

[18] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

[19] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel. rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[20] J. X. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Z. Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[21] R. Fakoor, P. Chaudhari, S. Soatto, and A. J. Smola. Meta-q-learning, 2020.

[22] Z. Xu, H. van Hasselt, and D. Silver. Meta-gradient reinforcement learning. *arXiv preprint arXiv:1805.09801*, 2018.

[23] F. Sung, L. Zhang, T. Xiang, T. Hospedales, and Y. Yang. Learning to learn: Meta-critic networks for sample efficient learning. *arXiv preprint arXiv:1706.09529*, 2017.

[24] R. Houthooft, R. Y. Chen, P. Isola, B. C. Stadie, F. Wolski, J. Ho, and P. Abbeel. Evolved policy gradients. *arXiv preprint arXiv:1802.04821*, 2018.

[25] J. Rothfuss, D. Lee, I. Clavera, T. Asfour, and P. Abbeel. Promp: Proximal meta-policy search. *Intl. Conference on Learning Representations*, 2019.

[26] A. Raghu, M. Raghu, S. Bengio, and O. Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of maml, 2020.

[27] C. Packer, P. Abbeel, and J. E. Gonzalez. Hindsight task relabelling: Experience replay for sparse reward meta-rl. *Advances in Neural Information Processing Systems*, 34, 2021.

[28] L. M. Zintgraf, L. Feng, C. Lu, M. Igl, K. Hartikainen, K. Hofmann, and S. Whiteson. Exploration in approximate hyper-state space for meta reinforcement learning. In *International Conference on Machine Learning*, pages 12991–13001. PMLR, 2021.

[29] E. Z. Liu, A. Raghunathan, P. Liang, and C. Finn. Decoupling exploration and exploitation for meta-reinforcement learning without sacrifices. *arXiv preprint arXiv:2008.02790*, 2020.

[30] J. Zhang, J. Wang, H. Hu, T. Chen, Y. Chen, C. Fan, and C. Zhang. Metacure: Meta reinforcement learning with empowerment-driven exploration. In *International Conference on Machine Learning*, pages 12600–12610. PMLR, 2021.

[31] R. Mendonca, X. Geng, C. Finn, and S. Levine. Meta-reinforcement learning robust to distributional shift via model identification and experience relabeling. *arXiv preprint arXiv:2006.07178*, 2020.

[32] L. Kirsch, S. Flennerhag, H. van Hasselt, A. Friesen, J. Oh, and Y. Chen. Introducing symmetries to black box meta reinforcement learning. *arXiv preprint arXiv:2109.10781*, 2021.

[33] 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. *arXiv preprint arXiv:1803.11347*, 2018.

[34] 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. PMLR, 2018.

[35] A. Bonardi, S. James, and A. J. Davison. Learning one-shot imitation from humans without humans. *IEEE Robotics and Automation Letters*, 5(2):3533–3539, 2020.

[36] J. Oh, M. Hessel, W. M. Czarnecki, Z. Xu, H. P. van Hasselt, S. Singh, and D. Silver. Discovering reinforcement learning algorithms. *Advances in Neural Information Processing Systems*, 33:1060–1070, 2020.

[37] L. Kirsch, S. van Steenkiste, and J. Schmidhuber. Improving generalization in meta reinforcement learning using learned objectives. *arXiv preprint arXiv:1910.04098*, 2019.

[38] Z. Xu, H. P. van Hasselt, M. Hessel, J. Oh, S. Singh, and D. Silver. Meta-gradient reinforcement learning with an objective discovered online. *Advances in Neural Information Processing Systems*, 33:15254–15264, 2020.

[39] J. D. Co-Reyes, Y. Miao, D. Peng, E. Real, S. Levine, Q. V. Le, H. Lee, and A. Faust. Evolving reinforcement learning algorithms. *arXiv preprint arXiv:2101.03958*, 2021.

[40] R. Houthooft, Y. Chen, P. Isola, B. Stadie, F. Wolski, O. Jonathan Ho, and P. Abbeel. Evolved policy gradients. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/7876acb66640bad41f1e1371ef30c180-Paper.pdf.

[41] Z. Xiong, L. M. Zintgraf, J. Beck, R. Vuorio, and S. Whiteson. On the practical consistency of meta-reinforcement learning algorithms. *ArXiv*, abs/2112.00478, 2021.

[42] R. Yang, H. Xu, Y. Wu, and X. Wang. Multi-task reinforcement learning with soft modularization. *ArXiv*, abs/2003.13661, 2020.

[43] S. Sodhani, A. Zhang, and J. Pineau. Multi-task reinforcement learning with context-based representations, 2021.

[44] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale, 2021.

[45] V. Kurin, A. De Palma, I. Kostrikov, S. Whiteson, and M. P. Kumar. In defense of the unitary scalarization for deep multi-task learning. *arXiv preprint arXiv:2201.04122*, 2022.

[46] K. Gao and O. Sener. Modeling and optimization trade-off in meta-learning. *Advances in Neural Information Processing Systems*, 33:11154–11165, 2020.

[47] T. Yu, D. Quillen, Z. He, R. Julian, A. Narayan, H. Shively, A. Bellathur, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning, 2021.

[48] A. Anand, J. Walker, Y. Li, E. Vértes, J. Schrittwieser, S. Ozair, T. Weber, and J. B. Hamrick. Procedural generalization by planning with self-supervised world models. *arXiv preprint arXiv:2111.01587*, 2021.

4 Appendix

4.1 Ablation Experiments

We investigate whether it is better to fine-tune an unseen task in isolation, or together with other tasks (in a multi-task setup). The intuition for the former is that train-time tasks (where we have access to demos), can be used to learn good representations and exploration strategies; while the latter intuition is that mixing with train-task data can act as auxiliary tasks, and the test-time task is treated as the main task. As shown in Figure 3, fine-tuning in isolation is superior to training in a multi-task setting. The hypothesis here is that the agent can keep the representations and skills that are useful to the fine-tune task, while forgetting non-useful ones; whereas training with other tasks requires that the network have the capacity to remember all skills.

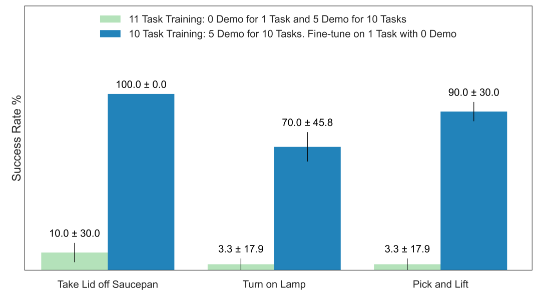

Figure 3: Is it is better to fine-tune an unseen task in isolation, or together with other tasks? We use a set of 11 RLBench tasks, where 1 is held out for test-time evaluation (and given 0 demos), while the other 10 are used for pretraining (and are given 5 demos). This is done for each of the 3 tasks above. Each color bar represents the average evaluation over 30 episodes while the error bars represent the standard deviations.

4.2 Related Work

Meta-Reinforcement Learning Meta-RL aims to find the best learning strategy that enables fast adaptation to a new task via reinforcement learning. This often relies on meta-training with a distribution of tasks and exploiting their shared structures. Two main approaches include context-based methods and gradient-based methods. Context-based methods are trained to use recent rollout experiences from a new task to form a context that can be used to distinguish what task the policy is solving. Previously, this context has been formed implicitly via an LSTM [19, 20], or explicitly, by passing trajectories through a separate encoder, whose output is given to a context-conditioned policy [12, 21]. Gradient-based methods perform test-time optimisation of hyperparameters [22], loss functions [23, 24], or network parameters [3, 25, 26].

The meta-RL approaches above have only been studied in fully observable state settings with shaped rewards; neglecting more realistic real-world scenarios, where rewards are often sparse, and observations are high-dimensional (e.g. images, point clouds, etc). There is limited work that study these issues: for example, hindsight relabeling is used to aid in sparse reward setups e.g. [27], but uses fully observable states. Other approaches to sparse reward and partial observability include HyperX [28], DREAM [29], and MetaCure [30]. Out-of-distribution variation adaptation within the same task is another challenging setup, where recent methods include model-identification, experience relabeling, [31], and adding symmetries [32].

Beyond context-based and gradient-based methods built on model-free RL algorithms, other lines of work include: **model-based meta-RL**, via meta-training a dynamics model and has seen success in enabling adapting to different hardware or terrain conditions on a legged millirobot [33]; **meta-imitation learning**, has been applied to vision-based robot manipulation [10, 34, 35]; **meta-learn RL algorithms**, which aims to discover RL objectives or update rules that can be transferred across different task environments [36, 37, 38, 39, 40].

Although in our experiments, we follow the original designs pf PEARL and RL^2 which don't allow gradient updates during test time, recent work [41] has looked into the theoretical limitations of context-based meta-RL algorithms in out-of-distribution variation adaptation setting [41], and shown that adding gradient updates (i.e. finetuning) at test time helps improve adaptation.

Multi-task Reinforcement Learning The pretraining procedure in our experiments is multi-task reinforcement learning (MTRL), where the training objective is simply finding a single best policy across multiple tasks. The main challenge in multi-task learning in general lies in multi-objective optimization, and has been investigated in MTRL [42, 43] and applied to robotics [44]. Recently, *Kurin et al.* [45] demonstrated that joint training with proper regularization achieves competitive performance with the more complicated multi-task algorithms. This observation aligns with our multi-task training results.

Meta- v.s. Multi-task pretraining in RL Multi-variation pretraining followed by fine-tuning, also called domain random search (DRS), is also shown to achieve comparable performance to meta-RL on existing state-based benchmarks [46]. Our work expands on this setup by training on more distinct **tasks** instead of variations, and excluding the test-time task from training. Notably, the meta-learning suite (e.g. ML10, ML45) in the MetaWorld [47] benchmark also poses such task generalization challenges, and finetuning is recently shown to be better than meta-RL algorithms such as RL^2 and MAML [48].