Watch and Match: Supercharging Imitation with Regularized Optimal Transport

Siddhant Haldar
New York University
sh6474@nyu.edu

Denis Yarats
New York University
denisyarats@cs.nyu.edu

Lerrel Pinto
New York University
lerrel@cs.nyu.edu

Abstract

Imitation learning holds tremendous promise in learning policies efficiently for complex decision making problems. Current state-of-the-art algorithms often use inverse reinforcement learning (IRL), where given a set of expert demonstrations, an agent alternatively infers a reward function and the associated optimal policy. However, such IRL approaches often require substantial online interactions particularly for complex control problems. In this work, we present Regularized Optimal Transport (ROT), a new imitation learning algorithm that builds on recent advances in optimal transport based state-matching. Our key technical insight is that adaptively combining state-matching rewards with behavior cloning can significantly accelerate imitation even without task-specific rewards. Our experiments on 19 tasks across the DeepMind Control Suite, the OpenAI Robotics Suite, and the Meta-World Benchmark, demonstrate an average of $7.8\times$ faster imitation to reach 90% of expert performance compared to prior state-of-the-art methods.

1 Introduction

Learning policies with the fewest possible interactions is a challenging problem in machine learning [1–3]. Over the last few decades, research in Imitation Learning (IL) has shown that it is not only among the most efficient learning methodologies, but can also operate without explicit reward functions. This is especially true for practical applications ranging from self-driving [4] to robotic manipulation [5], where online interactions are costly.

IL has a rich history that can be categorized into two broad paradigms, Behavior Cloning (BC) [1] and Inverse Reinforcement Learning (IRL) [6]. BC uses supervised learning to obtain a policy that maximizes the likelihood of taking the demonstrated action given an observation in the demonstration. While this allows for training without online interactions, it suffers from distributional mismatch during online rollouts [7]. IRL, on the other hand, infers the underlying reward function from the demonstrated trajectories, followed by using RL to optimize a policy through online environment rollouts. This results in a policy that can robustly solve demonstrated tasks even in the absence of task-specific rewards [8, 9].

Although powerful, IRL methods suffer from a significant drawback – expensive and numerous online interactions with the environment. There are three reasons for this: (a) The inferred reward function is often highly non-stationary, which compromises the learning of the associated behavior policy [9]. (b) Even when the rewards are stationary, policy learning still requires effective exploration to maximize rewards [10]. (c) Third, when strong priors such as pretraining with BC are applied to accelerate policy learning, ensuing updates to the policy cause a distribution shift that destabilizes training [11, 12]. Combined, these issues manifest themselves on empirical benchmarks, where IRL methods often have poorer efficiency than vanilla RL methods on hard control tasks [13]. Since this defeats the very reason one may want to do imitation, it begs the question: How can we make IRL more sample-efficient?

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

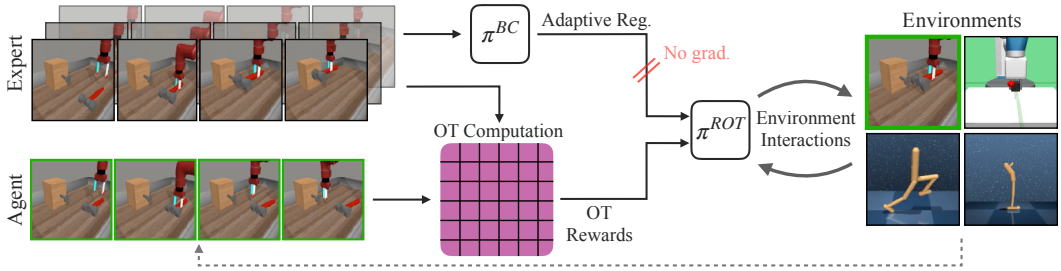

Figure 1: We present ROT, an imitation learning algorithm that adaptively combines offline behavior cloning with online Optimal Transport (OT) based IRL to achieve state-of-the-art imitation on a variety of control and manipulation environments.

In this work, we present Regularized Optimal Transport (ROT) for imitation learning, a new method that is conceptually simple, is compatible with high-dimensional observations, and requires minimal additional hyperparameters compared to standard IRL approaches. To address the challenge of reward non-stationarity in IRL, ROT builds on recent advances in using Optimal Transport (OT) [14, 15, 13] for reward computation that use stationary trajectory-matching functions. To alleviate the challenge of exploration, we pretrain the IRL behavior policy using BC on the expert demonstrations. This reduces the need for our imitation agent to explore from scratch.

However, even with OT-based reward computation and pretrained policies, we only obtain marginal gains in empirical performance. The key reason for this is that the high-variance of IRL policy gradients [16, 17] often wipe away the progress made by the offline BC pretraining. This phenomenon has been observed in both online RL [18] and offline RL [11] methods that are guided by demonstrations. Inspired by solutions presented in these works, we stabilize the online learning process by regularizing the IRL policy to stay close to the pretrained BC policy. To enable this, we have developed a new and simple adaptive weighing scheme called soft Q-filtering that automatically sets the regularization – prioritizing to stay close to the BC policy in the beginning of training while prioritizing exploration later on. In contrast to prior policy regularization schemes [18, 19], soft Q-filtering does not require hand-specification of decay schedules.

To demonstrate the effectiveness of ROT, we run extensive experiments on 19 tasks across DM Control [20], OpenAI Robotics [21], and Meta-world [22]. Our main findings can be summarized as:

1. ROT outperforms prior state-of-the-art imitation methods, reaching 90% of expert performance $7.8\times$ faster than our strongest baselines (Section 5.1).
2. On difficult control tasks, ROT exceeds the performance of state-of-the-art RL trained with rewards, while coming close to methods that augment RL with demonstrations (Section 5.3).
3. Ablation studies demonstrate the importance of every component in ROT, particularly the role that soft Q-filtering plays in stabilizing training (Section 5.4).

2 Background

Before describing our method, we first provide a brief background to imitation learning with optimal transport, which serves as the backbone of our method. Formalism related to RL follows the convention in prior work [10, 13] and is described in Appendix A.1.

Imitation Learning The goal of imitation learning is to learn a behavior policy π^b given access to either the expert policy π^e or trajectories derived from the expert policy $\mathcal{T}^e$. While there are a multitude of settings with differing levels of access to the expert [23], this work operates in the setting where the agent only has access to observation-based trajectories, i.e. $\mathcal{T}^e \equiv \{(o_t, a_t)_{t=0}^T\}_{n=0}^N$. Here N and T denotes the number of trajectory rollouts and episode timesteps respectively. We choose this specific setting since obtaining observations and actions from expert or near-expert demonstrators is feasible in real-world settings [24, 25] and falls in line with recent work in this area [15, 8, 9].

Inverse Reinforcement Learning (IRL) IRL [6, 26] tackles the IL problem by inferring the reward function r^e based on expert trajectories $\mathcal{T}^e$. Then given the inferred reward r^e, policy optimization is used to derive the behavior policy π^b. Prominent algorithms in IRL [9, 8] requires alternating the inference of reward and optimization of policy in an iterative manner, which is practical for restricted model classes [26]. For compatibility with more expressive deep networks, techniques such as adversarial learning [8, 9] or optimal-transport [14, 15, 13] are needed. Adversarial learning based approaches tackle this problem by learning a discriminator that models the gap between the expert trajectories $\mathcal{T}^e$ and behavior trajectories $\mathcal{T}^b$. The behavior policy π^b is then optimized to minimize this gap through gap-minimizing rewards r^e. Such a training procedure is prone to instabilities since r^e is updated at every iteration and is hence non-stationary for the optimization of π^b.

Optimal Transport for Imitation Learning (OT) To alleviate the non-stationary reward problem with adversarial IRL frameworks, a new line of OT-based approaches have been recently proposed [14, 15, 13]. Intuitively, the closeness between expert trajectories $\mathcal{T}^e$ and behavior trajectories $\mathcal{T}^b$ can be computed by measuring the optimal transport of probability mass from $\mathcal{T}^b \rightarrow \mathcal{T}^e$. Similar to [13], we use the entropic Wasserstein distance with cosine cost as our OT metric, which for two discrete distributions $\mu_b = \frac{1}{T}\sum_{t=1}^{T}\delta_{x_t^b}$ and $\mu_e = \frac{1}{T}\sum_{t=1}^{T}\delta_{x_t^e}$ is given by

$$\mathcal{W}^2(\mu_b, \mu_e) = \underset{\mu \in \mathcal{M}}{min} \sum_{t,t'=1}^{T} C_{t,t'}\,\mu_{t,t'} \tag{1}$$

where $M = \{\mu \in \mathbb{R}^{T\times T} : \mu\mathbf{1} = \mu^T\mathbf{1} = \frac{1}{T}\mathbf{1}\}$ is the set of coupling matrices and $C_{t,t'} = c(o_t^{\boldsymbol{b}}, o_{t'}^{\boldsymbol{e}})$ is the cost matrix obtained using a cost function $c : \mathcal{O} \times \mathcal{O} \rightarrow \mathbb{R}$. Using the cost matrix C and the optimal alignment μ^* obtained by optimizing Eq. 1, a reward signal can be computed for each observation using the equation

$$r^{OT}(o_t^{\boldsymbol{b}}) = -\sum_{t'=1}^{T} C_{t,t'}\,\mu^*_{t,t'} \tag{2}$$

Intuitively, this reward encourages the imitating agent to produce trajectories that closely match demonstrated trajectories. Since solving Eq. 1 is computationally expensive, approximate solutions such as the Sinkhorn algorithm [27, 14] are used instead. Further details on the OT formulation for IRL can be found in Appendix C.

Actor-Critic based reward maximization Given, rewards obtained through OT computation in Eq. 9, efficient maximization of the reward can be achieved through off-policy actor-critic learning [9]. For this work, we use Deep Deterministic Policy Gradient (DDPG) [28] as our base RL optimizer which is an actor-critic algorithm that concurrently learns a deterministic policy π_ϕ and a Q-function Q_θ. Vanilla DDPG uses Q-learning [29] to learn Q_θ by minimizing a one step Bellman residual. To accelerate learning we use a recent n-step version of DDPG from Yarats et al. [10].

3 Challenges in Online Finetuning from a Pretrained Policy

In this section, we study the challenges with finetuning a pretrained policy with online interactions in the environment. Fig. 2 illustrates a task where an agent is supposed to navigate the environment from the top left to the bottom right, while dodging obstacles in between. The agent has access to a single expert demonstration, which is used to learn a BC policy for the task. Fig. 2 (a) shows that this BC policy, though close to the expert demonstration, performs suboptimally due to accumulating errors on out-of-distribution states during online rollouts [7]. Further, Fig. 2 (b) uses this BC policy as an initialization and naively finetunes it with OT rewards (described in Section 2). Such naive finetuning of a pretrained policy (or actor) with an untrained critic in an actor-critic framework exhibits a forgetting behavior in the actor, resulting in performance degradation as compared to the pretrained policy. This phenomenon has also been reported by Nair et al. [11] and we provide a detailed discussion in Appendix B. In this paper, we propose ROT which addresses this issue by adaptively keeping the policy close to the behavior data during the initial phase of finetuning and reduces this dependence over time. Fig. 2 (c) demonstrates the performance of our approach on such finetuning. It can be clearly seen that even though the BC policy is suboptimal, our proposed adaptive regularization scheme quickly improves and solves the task by driving it closer to the expert

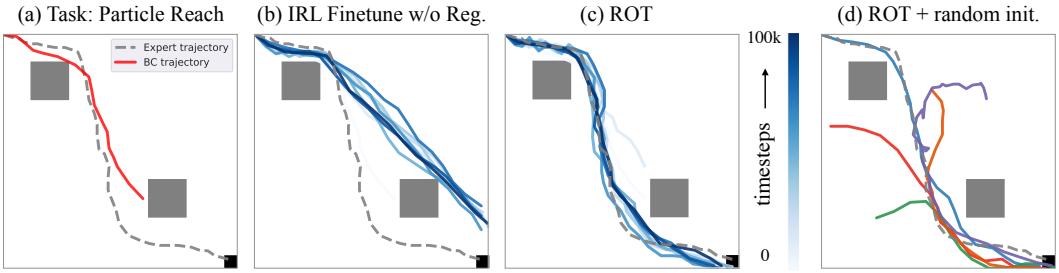

Figure 2: Given a single demonstration to avoid the grey obstacle and reach the black square, BC is unable to solve the task (a). Finetuning from this BC policy with OT-based reward also fails to solve the task (b). ROT, with adaptive regularization of OT-based IRL with BC successfully solves the task (c). Even when the ROT agent is initialized randomly, it is able to solve the task (d).

demonstration. In Fig. 2 (d), we demonstrate that even if the agent was initialized at points outside the expert trajectory, the agent is still able to learn quickly and complete the task. This generalization to starting states would not be possible with regular BC.

4 Regularized Optimal Transport

A fundamental challenge in imitation learning is to balance the ability to mimic demonstrated actions along with the ability to recover from states outside the distribution of demonstrated states. Behavior Cloning (BC) specializes in mimicking demonstrated actions through supervised learning, while Inverse Reinforcement Learning (IRL) specializes in obtaining policies that can recover from arbitrary states. Regularized Optimal Transport (ROT) combines the best of both worlds by adaptively combining the two objectives. This is done in two phases. In the first phase, a randomly initialized policy is trained using the BC objective on expert demonstrated data. This 'BC-pretrained' policy then serves as an initialization for the second phase. In the second phase, the policy is allowed access to the environment where it can train using an IRL objective. To accelerate the IRL training, the BC loss is added to the objective with an adaptive weight. Details of each component are described below, while the algorithm block can be found in Appendix D.

4.1 Phase 1: BC Pretraining

BC corresponds to solving the maximum likelihood problem shown in Eq. 3. Here $\mathcal{T}^e$ refers to expert demonstrations.

$$\mathcal{L}^{BC} = \mathbb{E}_{(s^e,a^e)\sim\mathcal{T}^e}\|a^e - \pi^{BC}(s^e)\|^2 \tag{3}$$

When parameterized by a normal distribution with fixed variance, the objective can be framed as a supervised learning regression problem where, given inputs s^e, π^{BC} needs to output a^e. After training, it enables π^{BC} to mimic the actions corresponding to the observations seen in the demonstrations. However, during rollouts in an environment, small errors in action prediction can lead to the agent visiting states not seen in the demonstrations [7]. This distributional mismatch often causes π^{BC} to fail on empirical benchmarks [18, 13] (See Fig. 2 (a)).

4.2 Phase 2: Online Finetuning with IRL

Given a pretrained π^{BC} model, we now begin online 'finetuning' of the policy $\pi^b \equiv \pi^{ROT}$ in the environment. Since we are operating without explicit task rewards, we use rewards obtained through OT-based trajectory matching, which is described in Section 2. Given the OT-based rewards r^{OT} from Eq.9, we can use standard RL optimizers to maximize cumulative reward from $\pi^b \equiv \pi^{ROT}$. In this work we use n-step DDPG [28], a deterministic actor-critic based method that provides high-performance in continuous control [10].

Finetuning with Regularization As seen in Fig. 2 (a), π^{BC} is susceptible to distribution shift due to accumulations of errors during online rollouts [7]. Directly finetuning π^{BC} also leads to subpar performance as seen in Fig. 2 (b). To address this, we build upon prior work in guided RL [18] and offline RL [11], and regularize the training of π^{ROT} by combining it with a BC loss as seen in Eq. 4.

$$\pi^{ROT} = \underset{\pi}{\operatorname{argmax}} \left[(1-\lambda_1(i))\mathbb{E}_{(s,a)\sim\mathcal{D}_\beta}[Q(s,a)] - \lambda_0\lambda_1(i)\mathbb{E}_{(s^e,a^e)\sim\mathcal{T}^e}\|a^e - \pi^{BC}(s^e)\|^2\right] \quad (4)$$

Here, $Q(s,a)$ represents the Q-value from the critic used in actor-critic policy optimization, while $\mathcal{L}^{BC}$ represents the BC loss obtained from Eq. 3. λ_0 is a fixed weight, while $\lambda_1(i)$ is a time-varying adaptive weight that controls the contributions of the two loss terms, where i denotes the cumulative training time. $\mathcal{D}_\beta$ refers to the replay buffer for online rollouts.

Adaptive Regularization with Soft Q-filtering While prior work [18, 19] use hand-tuned schedules for $\lambda_1(i)$, we propose a new adaptive scheme that removes the need for tuning. This is done by comparing the performance of the current policy π^{ROT} and the pretrained policy π^{BC} on a batch of data sampled from an expert replay buffer $\mathcal{D}_\beta$. More precisely, given a behavior policy $\pi^{BC}(s)$, the current policy $\pi^{ROT}(s)$, the Q-function $Q(s,a)$ and the replay buffer $\mathcal{D}_\beta$, we set $\lambda_1(i)$ as:

$$\lambda_1(i) = \mathbb{E}_{(s,\cdot)\sim\mathcal{D}_\beta}\left[\mathbb{1}_{Q(s,\pi^{BC}(s))>Q(s,\pi^{ROT}(s))}\right] \quad (5)$$

The strength of the BC regularization hence depends on the performance of the current policy with respect to the behavior policy. This filtering strategy is inspired by Nair et al. [30], where we augment binary hard assignment with a soft continuous weight. Experimental comparisons with hand-tuned decay strategies are presented in Section 5.2.

4.3 Implementation details

Considerations for image-based observations Since we are interested in using ROT with high-dimensional visual observations, additional machinery is required to ensure compatibility. Following prior work in image-based RL and imitation [10, 13], we perform data augmentations on visual observations and then feed it into a CNN encoder. Similar to Cohen et al. [13], we use a target encoder with Polyak averaging to obtain representations for OT reward computation. This is necessary to reduce the non-stationarity caused by learning the encoder alongside the ROT imitation process.

Algorithm and training procedure Our model consists of 3 primary neural networks - the encoder, the actor and the critic. During the BC pretraining phase, the encoder and the actor are trained using a mean squared error (MSE) on the expert demonstrations. Next, for finetuning, weights of the pretrained encoder and actor are loaded from memory and the critic is initialized randomly. We observed that the performance of the algorithm is not very sensitive to the value of λ_0 and we set it to 0.03 for all experiments in this paper. A copy of the pretrained encoder and actor are stored with fixed weights to be used for computing $\lambda_1(i)$ for soft Q-filtering. More details on our implementation can be found in Appendix D.

5 Experiments

Our experiments are designed to answer the following questions: (a) How efficient is ROT for imitation learning? (b) Does soft Q-filtering improve imitation? (c) How does ROT compare to standard RL? (d) How important are the IRL design choices in ROT?

Environments We experiment with 10 tasks from the DeepMind Control suite [20, 31], 3 tasks from the OpenAI Robotics suite [32], and 6 tasks from the Meta-world suite [33]. Full environment details can be found in Appendix E.

Expert demonstrations For DeepMind Control tasks, we train expert policies using DrQ-v2 [10] and collect 10 demonstrations for each task using this policy. For OpenAI Robotics tasks, we train a state-based DrQ-v2 with hindsight experience replay [34] and collect 50 demonstrations for each

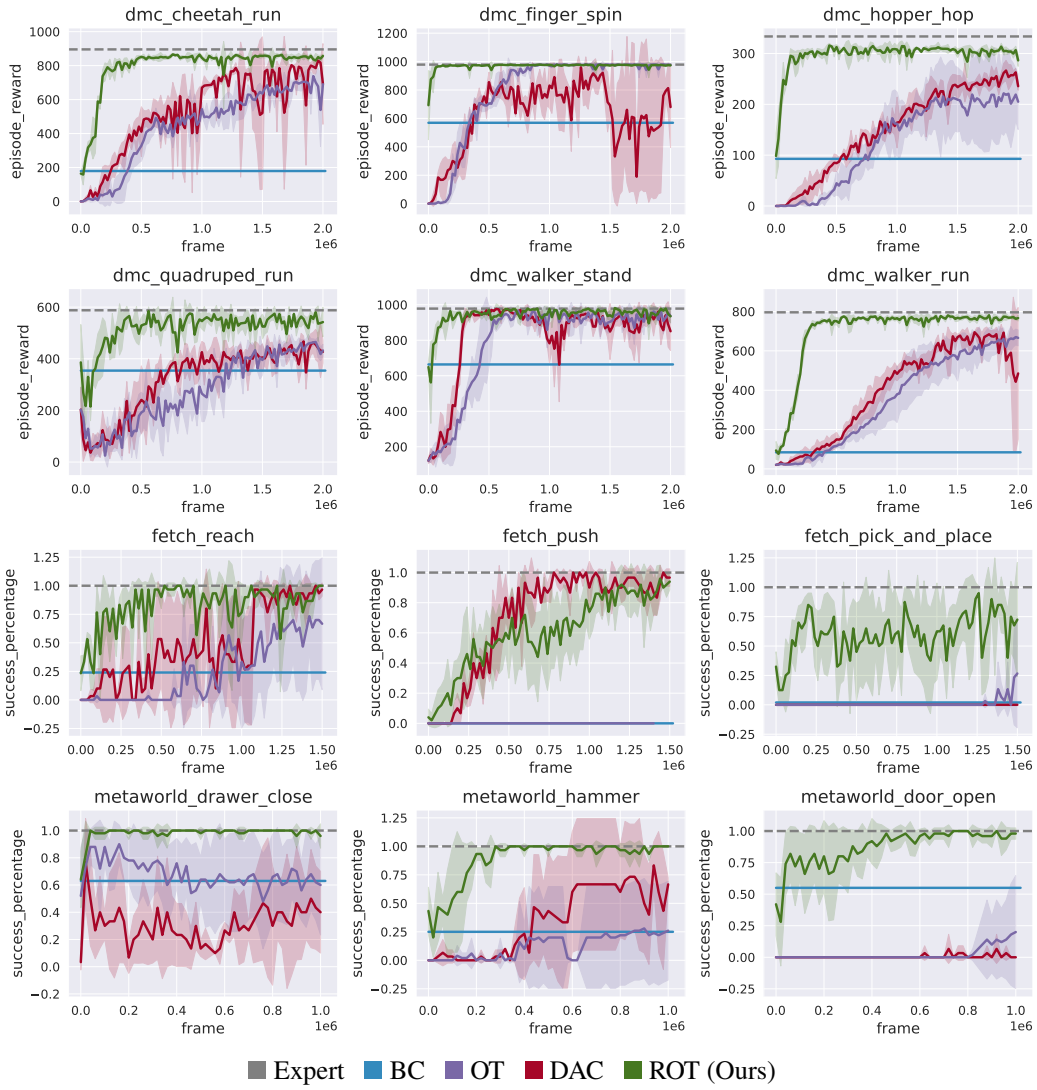

Figure 3: Pixel-based continuous control learning on 12 selected environments. Shaded region represents ± 1 standard deviation across 5 seeds. We notice that ROT is significantly more sample efficient compared to prior work.

task. For Meta-world tasks, we use a single hard-coded expert demonstration from their open-source implementation [33]. Details on the variations in demonstrations and initialization conditions can be found in Appendix F.

Primary baselines We compare ROT with baselines against several prominent imitation learning methods. While a full description of our baselines are in Appendix G, a brief description of the two strongest ones are as follows:

(a) **Adversarial IRL (DAC):** Discriminator Actor Critic [9] is a state-of-the-art adversarial imitation learning method [8, 35, 9]. Since DAC outperforms prior work such as GAIL [8] and AIRL [36] it serves as our primary adversarial imitation baseline.

(b) **State-matching IRL (OT):** Sinkhorn Imitation Learning [14, 15] is a state-of-the-art state-matching imitation learning method [37] that approximates OT matching through the Sinkhorn Knopp algorithm. Since ROT is derived from similar OT-based foundations, we use SIL as our primary state-matching imitation baseline.

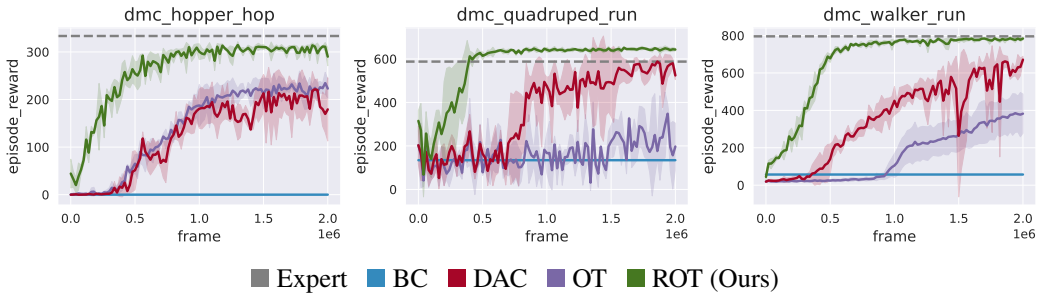

Figure 4: State-based continuous control learning on 3 representative environments. We observe similar gains in performance as our image-based experiments.

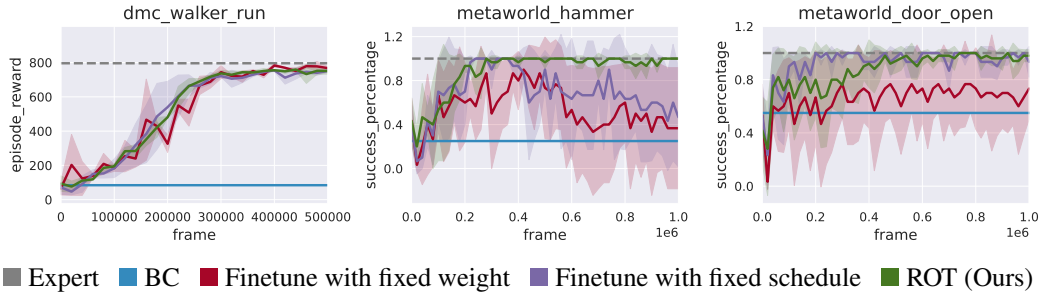

Figure 5: Effect of varying BC regularization schemes on 3 selected environments. We observe that our adaptive soft-Q filtering regularization is more stable compared to prior hand-tuned regularization schemes.

5.1 How efficient is ROT for imitation learning?

Performance of ROT for image-based imitation is depicted on select environments in Fig. 3. On all but one task, ROT trains significantly faster than prior work. To reach 90% of expert performance, ROT is on average $8.7\times$ faster on DeepMind Control tasks, $2.1\times$ faster on Fetch Robotics tasks, and $8.9\times$ faster on Meta-world tasks. We also find that the improvements of ROT are most apparent on the harder tasks (rightmost column in Fig. 3). Finally, the improvements from ROT hold on state-based observations as well (see Fig. 4). Learning curves for all tasks are depicted in Appendix H.

5.2 Does soft Q-filtering improve imitation?

To understand the importance of soft Q-filtering, we compare ROT against two variants of our proposed regularization scheme: (a) A tuned fixed BC regularization weight (ignoring $\lambda_1(i)$ in Eq. 4); (b) A carefully designed linear-decay schedule for $\lambda_1(i)$, where it varies from 1.0 to 0.0 in the first 20000 environment steps [18]. As demonstrated in Fig. 5, ROT is on par and in some cases exceeds the efficiency of a hand-tuned decay schedule, while not having to hand-tune its regularization weights. We hypothesize this improvement is primarily due to the better stability of adaptive weighing as seen in the significantly smaller standard deviation on the Meta-world tasks.

5.3 How does ROT compare to standard reward-based RL?

We compare the performance of ROT against DrQ-v2 [10], a state-of-the-art algorithm for image-based RL. As opposed to the reward-free setting ROT operates in, DrQ-v2 has access to environments rewards. We also compare against a demo-assisted variant of DrQ-v2 agent using the same pretraining and regularization scheme as ROT (which we refer to as Demo-DrQ-v2). The results in Fig. 6 show that ROT method handily outperforms DrQ-v2. This clearly demonstrates the usefulness of imitation learning in domains where expert demonstrations are available over reward-based RL. Interestingly, we also find that our soft Q-filtering based regularization can accelerate learning of RL with task rewards, which can be seen in the high performance of Demo-DrQ-v2.

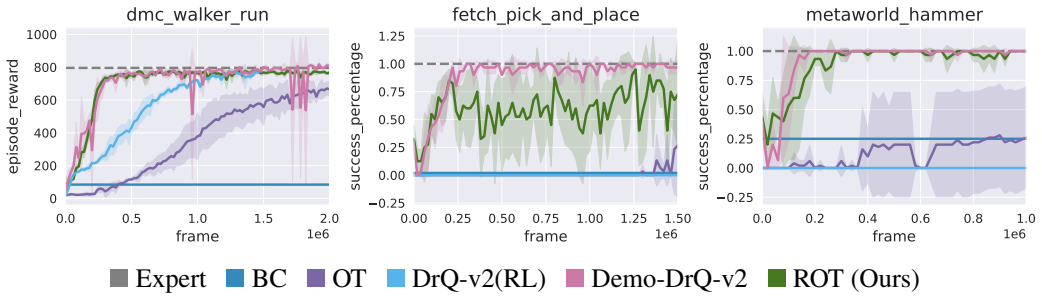

Figure 6: Comparison of ROT against DrQ-v2, a reward-based RL method. Here we see that ROT can outperform plain RL that requires explicit task-reward. However, we also observe that this RL method combined with our regularization scheme provides strong results.

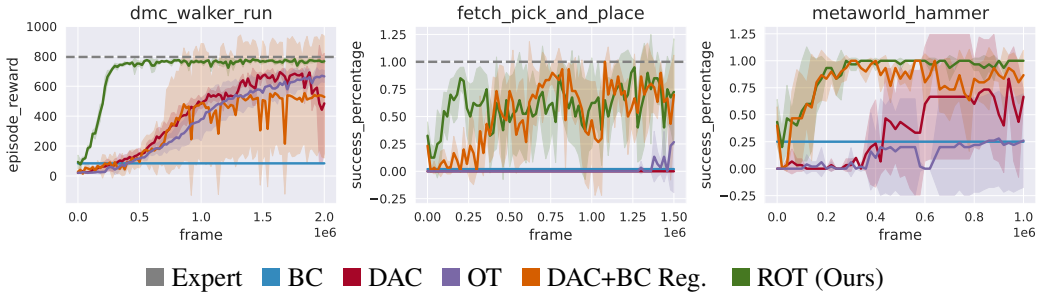

Figure 8: Ablation analysis on the choice of base IRL method. We find that although adversarial methods benefit from regularized BC, the gains seen are smaller compared to ROT.

5.4 How important are the design choices in ROT?

Importance of pretraining and regularizing the IRL policy Fig. 7 compares the following variants of ROT on the DeepMind Control Walker Run task: (a) Training the IRL policy from scratch (OT); (b) Finetuning a pretrained BC policy without BC regularization (BC+OT); (c) Training the IRL policy from scratch with BC regularization (OT+BC Reg.). We observe that pretraining the IRL policy (BC+OT) does not provide a significant difference without regularization. This can be attributed to the 'forgetting behavior' of pre-trained policies, studied in Nair et al. [11]. Interestingly, we see that even without BC pretraining, keeping the policy close to a behavior distribution (OT+BC Reg.) can yield improvements in efficiency over vanilla training from scratch. Our key takeaway from these experiments is that both pretraining and BC regularization are required to obtain sample-efficient imitation learning.

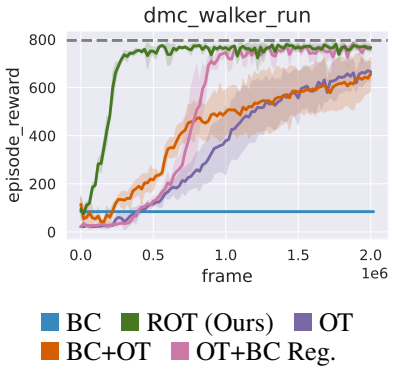

Figure 7: Comparison between ablated versions of ROT on the Walker Run task.

Choice of IRL method In ROT, we build on OT-based IRL instead of adversarial IRL. This is because adversarial IRL methods require iterative reward learning, which produces a highly non-stationary reward function for policy optimization. In Fig. 8, we compare ROT with adversarial IRL methods that use our pretraining and adaptive BC regularization technique (DAC+BC Reg.). We find that our soft Q-filtering method does improve prior state-of-the-art adversarial IRL (DAC+BC Reg. vs. DAC in Fig. 8). However, our OT-based approach (ROT) is more stable and on average leads to more efficient learning.

6 Related Work

Imitation Learning (IL) IL refers to the setting where agents learn from demonstrations without access to environment rewards. IL can be broadly categorized into Behavior Cloning (BC) [1, 23] and Inverse Reinforcement Learning (IRL) [6, 26]. BC solely learns from offline demonstrations but suffers on out-of-distributions samples [7] whereas IRL focuses on learning a robust reward function through online interactions but suffers from sample inefficiency [9]. Deep IRL methods can be further divided into two categories: (1) adversarial learning [38] based methods, and (2) state-matching [39, 40] based methods. GAIL [8] is an adversarial learning based formulation inspired by maximum entropy IRL [41] and GANs [38]. There has been a significant body of work built up on GAIL proposing alternative losses [36, 42, 35], and enhancing its sample efficiency by porting it to an off-policy setting [9]. There have also been visual extensions of these adversarial learning approaches [43–45, 13]. In this work, we find that although adversarial methods produce competent policies, they are inefficient due to the non-stationarity associated with iterative reward inference.

Optimal Transport (OT) OT [39, 40] is a tool for comparing probability measures while including the geometry of the space. In the context of IL, OT computes an alignment between a set of agent and expert observations using distance metrics such as Sinkhorn [46], Gromov-Wasserstein [47], GDTW [48], CO-OT [49] and Soft-DTW [50]. For many of these distance measures, there is an associated IL algorithm, with SIL [14] using Sinkhorn, PWIL [15] using greedy Wasserstein, GDTW-IL [48] using GDTW, and GWIL [51] using Gromov-Wasserstein. Recent work from Cohen et al. [13] demonstrates that the Sinkhorn distance [14] produces the most efficient learning among the discussed metrics. They further show that SIL is compatible with high-dimensional visual observations and encoded representations. Inspired by this, ROT adopts the Sinkhorn metric for its OT reward computation, and improves upon SIL through adaptive behavior regularization.

Behavior Regularized Control Behavior regularization is a widely used technique in offline RL [52] where explicit constraints are added to the policy improvement update to avoid bootstrapping on out-of-distribution actions [53–58]. In an online setting with access to environment rewards, prior work [18, 12] has shown that behavior regularization can be used to boost sample efficiency by finetuning a pretrained policy via online interactions. For instance, Jena et al. [19] demonstrates the effectiveness of behavior regularization to enhance sample efficiency in the context of adversarial IL. ROT builds upon this idea by extending to visual observations, OT-based IL, and adaptive regularization, which leads to improved performance (see Section 5.4). We also note that the idea of using adaptive regularization has been previously explored in RL [30]. However, ROT uses a soft, continuous adaptive scheme, which on initial experiments provided significantly faster learning compared to hard assignments.

7 Conclusion and Limitations

In this work, we propose Regularized Optimal Transport (ROT), a new imitation learning algorithm that alleviates the challenge of exploration and significantly improves sample efficiency by using a pretrained policy in conjunction with an adaptive regularization scheme for online finetuning. Although we demonstrate superior performance compared to prior work on a varied set of simulated environments, there are a few limitations in this work: (a) Since our OT-based approach aligns agents with demonstrations without task-specific rewards, it relies on the demonstrator being an 'expert'. Extending ROT to suboptimal demonstrations would be an exciting future direction. (b) Performing BC pretraining and BC-based regularization requires access to expert actions, which may not be present in real-world demonstrations from humans. Recent work on using inverse models to infer actions given observational data could alleviate this challenge [59]. (c) While we show substantially better performance on simulated control tasks, the true test for ROT will be its applicability to real-world control problems. Given our results on simulated robotic problems, where ROT can learn policies with arbitrary environment initialization from visual observations, we look forward to future work that extends ROT to real versions of these simulated problems.

References

[1] D Pomerleau. An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems*, 1, 1998. 1, 9

[2] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[3] Petter N Kolm and Gordon Ritter. Modern perspectives on reinforcement learning in finance. *Modern Perspectives on Reinforcement Learning in Finance (September 6, 2019). The Journal of Machine Learning in Finance*, 1(1), 2020. 1

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016. 1

[5] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8. IEEE, 2018. 1

[6] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000. 1, 3, 9

[7] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011. 1, 3, 4, 5, 9

[8] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016. 1, 2, 3, 6, 9, 18

[9] Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. *arXiv preprint arXiv:1809.02925*, 2018. 1, 2, 3, 6, 9, 14, 18

[10] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021. 1, 2, 3, 4, 5, 7, 14, 15, 16, 17, 18

[11] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020. 1, 2, 3, 5, 8, 14

[12] Ikechukwu Uchendu, Ted Xiao, Yao Lu, Banghua Zhu, Mengyuan Yan, Joséphine Simon, Matthew Bennice, Chuyuan Fu, Cong Ma, Jiantao Jiao, et al. Jump-start reinforcement learning. *arXiv preprint arXiv:2204.02372*, 2022. 1, 9, 14

[13] Samuel Cohen, Brandon Amos, Marc Peter Deisenroth, Mikael Henaff, Eugene Vinitsky, and Denis Yarats. Imitation learning from pixel observations for continuous control, 2022. URL https://openreview.net/forum?id=JLbXkHkLCG6. 1, 2, 3, 4, 5, 9, 14, 16

[14] Georgios Papagiannis and Yunpeng Li. Imitation learning with sinkhorn distances. *arXiv preprint arXiv:2008.09167*, 2020. 2, 3, 6, 9, 15, 18

[15] Robert Dadashi, Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal wasserstein imitation learning. *arXiv preprint arXiv:2006.04678*, 2020. 2, 3, 6, 9, 18

[16] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2

[17] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014. 2, 14, 15

[18] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017. 2, 4, 5, 7, 9, 14

[19] Rohit Jena, Changliu Liu, and Katia Sycara. Augmenting gail with bc for sample efficient imitation learning. *arXiv preprint arXiv:2001.07798*, 2020. 2, 5, 9, 14

[20] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018. 2, 5, 17

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

[22] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020. 2

[23] Faraz Torabi, Garrett Warnell, and Peter Stone. Recent advances in imitation learning from observation. *arXiv preprint arXiv:1905.13566*, 2019. 2, 9

[24] Albert Zhan, Philip Zhao, Lerrel Pinto, Pieter Abbeel, and Michael Laskin. A framework for efficient robotic manipulation. *arXiv preprint arXiv:2012.07975*, 2020. 2

[25] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. *arXiv preprint arXiv:2008.04899*, 2020. 2

[26] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004. 3, 9

[27] Philip A Knight. The sinkhorn–knopp algorithm: convergence and applications. *SIAM Journal on Matrix Analysis and Applications*, 30(1):261–275, 2008. 3, 15

[28] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015. 3, 4, 14

[29] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992. 3

[30] Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 6292–6299. IEEE, 2018. 5, 9

[31] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012. 5, 17

[32] Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018. 5, 17

[33] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey 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. 5, 6, 17

[34] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017. 5, 17

[35] Faraz Torabi, Garrett Warnell, and Peter Stone. Generative adversarial imitation from observation. *arXiv preprint arXiv:1807.06158*, 2018. 6, 9, 18

[36] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017. 6, 9, 18

[37] Seyed Kamyar Seyed Ghasemipour, Richard Zemel, and Shixiang Gu. A divergence minimization perspective on imitation learning methods. In *Conference on Robot Learning*, pages 1259–1277. PMLR, 2020. 6, 18

[38] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 9

[39] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009. 9

[40] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019. 9

[41] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008. 9

[42] Huang Xiao, Michael Herman, Joerg Wagner, Sebastian Ziesche, Jalal Etesami, and Thai Hong Linh. Wasserstein adversarial imitation learning. *arXiv preprint arXiv:1906.08113*, 2019. 9

[43] Edoardo Cetin and Oya Celiktutan. Domain-robust visual imitation learning with mutual information constraints. *arXiv preprint arXiv:2103.05079*, 2021. 9

[44] Sam Toyer, Rohin Shah, Andrew Critch, and Stuart Russell. The magical benchmark for robust imitation. *Advances in Neural Information Processing Systems*, 33:18284–18295, 2020.

[45] Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Visual adversarial imitation learning using variational models. *Advances in Neural Information Processing Systems*, 34, 2021. 9

[46] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013. 9

[47] Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-wasserstein averaging of kernel and distance matrices. In *International Conference on Machine Learning*, pages 2664–2672. PMLR, 2016. 9

[48] Samuel Cohen, Giulia Luise, Alexander Terenin, Brandon Amos, and Marc Deisenroth. Aligning time series on incomparable spaces. In *International Conference on Artificial Intelligence and Statistics*, pages 1036–1044. PMLR, 2021. 9

[49] Ievgen Redko, Titouan Vayer, Rémi Flamary, and Nicolas Courty. Co-optimal transport. *arXiv preprint arXiv:2002.03731*, 2020. 9

[50] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International conference on machine learning*, pages 894–903. PMLR, 2017. 9

[51] Arnaud Fickinger, Samuel Cohen, Stuart Russell, and Brandon Amos. Cross-domain imitation learning via optimal transport. *arXiv preprint arXiv:2110.03684*, 2021. 9

[52] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020. 9

[53] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021. 9

[54] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[55] Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. {OPAL}: Offline primitive discovery for accelerating offline reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=V69LGwJ0lIN.

[56] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

[57] Noah Y Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396*, 2020.

[58] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019. 9

[59] Ilija Radosavovic, Xiaolong Wang, Lerrel Pinto, and Jitendra Malik. State-only imitation learning for dexterous manipulation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7865–7871. IEEE, 2020. 9

[60] Richard Bellman. A markovian decision process. *Journal of mathematics and mechanics*, pages 679–684, 1957. 14

[61] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 14

[62] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. 14

A Background

A.1 Reinforcement Learning (RL)

We study RL as a discounted infinite-horizon Markov Decision Process (MDP) [60, 61]. For pixel observations, the agent's state is approximated as a stack of consecutive RGB frames [62]. The MDP is of the form $(\mathcal{O}, \mathcal{A}, P, R, \gamma, d_0)$ where $\mathcal{O}$ is the observation space, $\mathcal{A}$ is the action space, $P : \mathcal{O} \times \mathcal{A} \to \Delta(\mathcal{O})$ is the transition function that defines the probability distribution over the next state given the current state and action, $R : \mathcal{O} \times \mathcal{A} \to \mathbb{R}$ is the reward function, γ is the discount factor and d_0 is the initial state distribution. The goal is to find a policy $\pi : \mathcal{O} \to \Delta(\mathcal{A})$ that maximizes the expected discount sum of rewards $\mathbb{E}_\pi[\Sigma_{t=0}^{\infty} \gamma^t R(\boldsymbol{o}_t, \boldsymbol{a}_t)]$, where $\boldsymbol{o}_0 \sim d_0$, $a_t \sim \pi(\boldsymbol{o}_t)$ and $\boldsymbol{o}_{t+1} \sim P(.|\boldsymbol{o}_t, \boldsymbol{a}_t)$.

B Issue with Fine-tuning Actor-Critic Frameworks

In this paper, we use n-step DDPG proposed by Yarats et al. [10] as our RL optimizer for actor-critic based reward maximization. DDPG [28] concurrently learns a deterministic policy π_ϕ using deterministic policy gradients (DPG) [17] and a Q-function Q_θ by minimizing a n-step Bellman residual (for n-step DDPG). For a parameterized actor network $\pi_\phi(s)$ and a critic function $Q_\theta(s, a)$, the deterministic policy gradients (DPG) for updating the actor weights is given by

$$\begin{aligned} \nabla_\phi J &\approx \mathbb{E}_{s_t \sim \rho_\beta} \left[\nabla_\phi \, Q_\theta(s, a)|_{s=s_t, a=\pi_\phi(s_t)} \right] \\ &= \mathbb{E}_{s_t \sim \rho_\beta} \left[\nabla_a \, Q_\theta(s, a)|_{s=s_t, a=\pi_\phi(s_t)} \, \nabla_\phi \, \pi_\phi(s)|_{s=s_t} \right] \end{aligned} \tag{6}$$

Here, ρ_β refers to the state visitation distribution of the data present in the replay buffer at time t. From Eq. 6, it is clear that the policy gradients in this framework depend on the gradients with respect to the critic value. Hence, as mentioned in [11, 12], naively initializing the actor with a pretrained policy while using a randomly initialized critic results in the untrained critic providing an exceedingly poor signal to the actor network during training. As a result, the actor performance drops immediately and the good behavior of the informed initialization of the policy gets forgotten. In this paper, we propose an adaptive regularization scheme that permits finetuning a pretrained actor policy in an actor-critic framework. As opposed to Rajeswaran et al. [18], Jena et al. [19] which employ on-policy learning, our method is off-policy and aims to leverage the sample efficient characteristic of off-policy learning as compared to on-policy learning [9].

C Optimal Transport (OT) Formulation

The policy π_ϕ encompasses a feature preprocessor f_ϕ which transforms observations into informative state representations. Some examples of a preprocessor function f_ϕ are an identity function, a mean-variance scaling function and a parametric neural network. In this work, we use a parametric neural network as f_ϕ. Given a cost function $c : \mathcal{O} \times \mathcal{O} \to \mathbb{R}$ defined in the preprocessor's output space and an OT objective g, the optimal alignment between an expert trajectory $\mathbf{o}^e$ and a behavior trajectory $\mathbf{o}^b$ can be computed as

$$\mu^* \in \underset{\mu \in \mathcal{M}}{\arg\min} \, g(\mu, f_\phi(\mathbf{o}^b), f_\phi(\mathbf{o}^e), c) \tag{7}$$

where $M = \{\mu \in \mathbb{R}^{T \times T} : \mu \mathbf{1} = \mu^T \mathbf{1} = \frac{1}{T} \mathbf{1}\}$ is the set of coupling matrices and the cost c can be the Euclidean or Cosine distance. In this work, inspired by [13], we use the entropic Wasserstein distance with cosine cost as our OT metric, which is given by the equation

$$\begin{aligned} g(\mu, f_\phi(\mathbf{o}^b), f_\phi(\mathbf{o}^e), c) &= \mathcal{W}^2(f_\phi(\mathbf{o}^b), f_\phi(\mathbf{o}^e)) \\ &= \sum_{t,t'=1}^{T} C_{t,t'} \, \mu_{t,t'} \end{aligned} \tag{8}$$

where the cost matrix $C_{t,t'} = c(f_\phi(\mathbf{o}^b), f_\phi(\mathbf{o}^e))$. Using Eq. 8 and the optimal alignment μ^* obtained by optimizing Eq. 7, a reward signal can be computed for each observation using the equation

$$r^{OT}(o_t^{\boldsymbol{b}}) = -\sum_{t'=1}^{T} C_{t,t'}\mu^*_{t,t'} \tag{9}$$

Intuitively, maximizing this reward encourages the imitating agent to produce trajectories that closely match demonstrated trajectories. Since solving Eq. 7 is computationally expensive, approximate solutions such as the Sinkhorn algorithm [27, 14] are used instead.

D Algorithmic Details

Algorithm 1 ROT: Regularized Optimal Transport

Require:
Expert Demonstrations $\mathcal{T}^e \equiv \{(o_t, a_t)_{t=0}^T\}_{n=0}^N$
Pretrained policy π^{BC}
Replay buffer $\mathcal{D}$, Training steps T, Episode Length L
Task environment env
Parametric networks for RL backbone (e.g., the encoder, policy and critic function for DrQ-v2)
A discriminator D for adversarial baselines

Algorithm:
$\pi^{ROT} \leftarrow \pi^{BC}$ ▷ Initialize with pretrained policy
for each timestep t = 1...T **do**
  if done **then**
    $r_{1:L}$ = rewarder$_{OT}$(episode) ▷ OT-based reward computation
    Update episode with $r_{1:L}$ and add $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1}, r_t)$ to $\mathcal{D}$
    $\mathbf{o}_t = env.reset(),$ done = False, episode = [ ]
  end if
  $\mathbf{a}_t = \pi^{ROT}(\mathbf{o}_t)$
  $\mathbf{o}_{t+1},$ done $= env.step(\mathbf{a}_t)$
  episode.append($[\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1}]$)
  Update backbone-specific networks and reward-specific networks using $\mathcal{D}$
end for

D.1 Implementation

Algorithm 1 describes our proposed algorithm, Regularized Optimal Transport (ROT), for sample efficient imitation learning for continuous control tasks. Further implementation details are as follows:

Actor-critic based reward maximization We use a recent n-step DDPG proposed by Yarats et al. [10] as our RL backbone. The deterministic actor is trained using deterministic policy gradients (DPG) [17] given by Eq. 6. The critic is trained using clipped double Q-learning similar to Yarats et al. [10] in order to reduce the overestimation bias in the target value. This is done using two Q-functions, $Q_{\theta 1}$ and $Q_{\theta 2}$. The critic loss for each critic is given by the equation

$$\mathcal{L}_{\theta_k} = \mathbb{E}_{(s,a)\sim D_\beta}\left[(Q_{\theta_k}(s,a) - y)^2\right] \forall\, k \in \{1,2\} \tag{10}$$

where $\mathcal{D}_\beta$ is the replay buffer for online rollouts and y is the target value for n-step DDPG given by

$$y = \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n \min_{k=1,2} Q_{\bar{\theta}_k}(s_{t+n}, a_{t+n}) \tag{11}$$

Here, γ is the discount factor, r is the reward obtained using OT-based reward computation and $\bar{\theta}_1$, $\bar{\theta}_2$ are the slow moving weights of target Q-networks.

Target feature processor to stabilize OT rewards The OT rewards are computed on the output of the feature processor f_ϕ which is initialized with a parametric neural network. Hence, as the weights of f_ϕ change during training, the rewards become non-stationary resulting in unstable training. In order to increase the stability of training, the OT rewards are computed using a target feature processor $f_{\phi'}$ [13] which is updated with the weights of f_ϕ every T_{update} environment steps. For state-based observations, f_ϕ corresponds to a 'trunk' network which is a single layer neural network. For pixel-based observations, f_ϕ includes DrQ-v2's encoder followed by the 'trunk' network.

D.2 Hyperparameters

The complete list of hyperparameters is provided in Table 1. Similar to Yarats et al. [10], there is a slight deviation from the given setting for the Walker Stand/Walk/Run task from the DeepMind Control suite where we use a mini-batch size of 512 and a n-step return of 1.

Method	Parameter	Value
Common	Replay buffer size	150000
	Learning rate	$1e^{-4}$
	Discount γ	0.99
	n-step returns	3
	Action repeat	2
	Seed frames	12000
	Mini-batch size	256
	Agent update frequency	2
	Critic soft-update rate	0.01
	Feature dim	50
	Hidden dim	1024
	Optimizer	Adam
ROT	Exploration steps	0
	DDPG exploration schedule	0.1
	Target feature processor update frequency(steps)	20000
	Reward scale factor	10
	Fixed weight λ_0	0.03
	Linear decay schedule for $\lambda_1(i)$	linear(1,0.1,20000)
OT	Exploration steps	2000
	DDPG exploration schedule	linear(1,0.1,500000)
	Target feature processor update frequency(steps)	20000
	Reward scale factor	10
DAC	Exploration steps	2000
	DDPG exploration schedule	linear(1,0.1,500000)
	Gradient penalty coefficient	10

Table 1: List of hyperparameters.

E Environments

Table 2 lists the different tasks that we experiment with from the DeepMind Control suite [20, 31], OpenAI Robotics suite [32] and the Meta-world suite [33] along with the number of training steps and the number of demonstrations used. For the tasks in the OpenAI Robotics suite, we fix the goal while keeping the initial state randomized. No modifications are made in case of the DeepMind Control suite and the Meta-world suite. The episode length for all tasks in DeepMind Control is 1000 steps, for OpenAI Robotics is 50 steps and Meta-world is 125 steps (except bin picking which runs for 175 steps).

Suite	Tasks	Allowed Steps	# Demonstrations
DeepMind Control	Acrobot Swingup	2×10^6	10
	Cartpole Swingup		
	Cheetah Run		
	Finger Spin		
	Hopper Stand		
	Hopper Hop		
	Quadruped Run		
	Walker Stand		
	Walker Walk		
	Walker Run		
OpenAI Robotics	Fetch Reach	1.5×10^6	50
	Fetch Push		
	Fetch Pick and Place		
Meta-World	Hammer	1×10^6	1
	Drawer Close		
	Door Open		
	Bin Picking		
	Button Press Topdown		
	Door Unlock.		

Table 2: List of tasks used for evaluation.

F Demonstrations

For DeepMind Control tasks, we train expert policies using pixel-based DrQ-v2 [10] and collect 10 demonstrations for each task using this expert policy. The expert policy is trained using a stack of 3 consecutive RGB frames of size 84×84 with random crop augmentation. Each action in the environment is repeated 2 times. For OpenAI Robotics tasks, we train a state-based DrQ-v2 with hindsight experience replay [34] and collect 50 demonstrations for each task. The state representation comprises the observation from the environment appended with the desired goal location. For this, we did not do frame stacking and action repeat was set to 2. For Meta-World tasks, we use a single expert demonstration obtained using the task-specific hard-coded policies provided in their open-source implementation [33].

G Baselines

Throughout the paper, we compare ROT with several prominent imitation learning and reinforcement learning methods. Here, we give a brief description of each of the baseline models that have been used.

(a) **Expert:** For each task, the expert refers to the expert policy used to generate the demonstrations for the task (described in Appendix F).

(b) **Behavior Cloning (BC):** This refers to the behavior cloned policy trained on expert demonstrations.

(c) **Adversarial IRL (DAC):** Discriminator Actor Critic [9] is a state-of-the-art adversarial imitation learning method [8, 35, 9]. Since DAC outperforms prior work such as GAIL[8] and AIRL[36], it serves as our primary adversarial imitation baseline.

(d) **State-matching IRL (OT):** Sinkhorn Imitation Learning [14, 15] is a state-of-the-art state-matching imitation learning method [37] that approximates OT matching through the Sinkhorn Knopp algorithm. Since ROT is derived from similar OT-based foundations, we use SIL as our primary state-matching imitation baseline.

(e) **Finetune with fixed weight:** This is similar to ROT where instead of using a time-varying adaptive weight $\lambda(i)$, only the fixed weight λ_0 is used. λ_0 is set to a fixed value of 0.03.

(f) **Finetune with fixed schedule:** This is similar to ROT that uses both the fixed weight λ_0 and the time-varying adaptive weight $\lambda_1(i)$. However, instead of using Soft Q-filtering to compute $\lambda_1(i)$, a hand-coded linear decay schedule is used.

(g) **DrQ-v2 (RL):** DrQ-v2 [10] is a state-of-the-art algorithm for pixel-based RL. DrQ-v2 is assumed to have access to environment rewards as opposed to ROT which computes the reward using OT-based techniques.

(h) **Demo-DrQ-v2:** This refers to DrQ-v2 but with access to both environment rewards and expert demonstrations. The model is initialized with a pretrained BC policy followed by RL finetuning with an adaptive regularization scheme like ROT. During RL finetuning, this baseline has access to environment rewards.

(i) **BC+OT:** This is the same as the OT baseline but the policy is initialized with a pretrained BC policy. No adaptive regularization scheme is used while finetuning the pretrained policy.

(j) **OT+BC Reg.:** This is the same as the OT baseline with randomly initialized networks but during training, the adaptive regularization scheme is added to the objective function.

(k) **DAC+BC Reg.:** This is the same as ROT, but instead of using state-matching IRL (OT), adversarial IRL (DAC) is used.

H Training curve

H.1 Additional Experimental Results for ROT

Fig. 9 and Fig. 10 show the performance of ROT for pixel-based imitation on 10 tasks from the DeepMind Control suite, 3 tasks from the OpenAI Robotics suite and 6 tasks from the Meta-world suite. On all but one task, ROT is significantly more sample efficient than prior work. Fig. 11 provides additional results exhibiting similar improvements on state-based observations.

H.2 Importance of pretraining and regularizing the IRL policy

Extending the results shown in Fig. 7, we provide training curves from representative tasks in each suite in Fig. 13, thus demonstrating the need for pretraining and BC regularization in tandem for sample-efficient imitation learning.

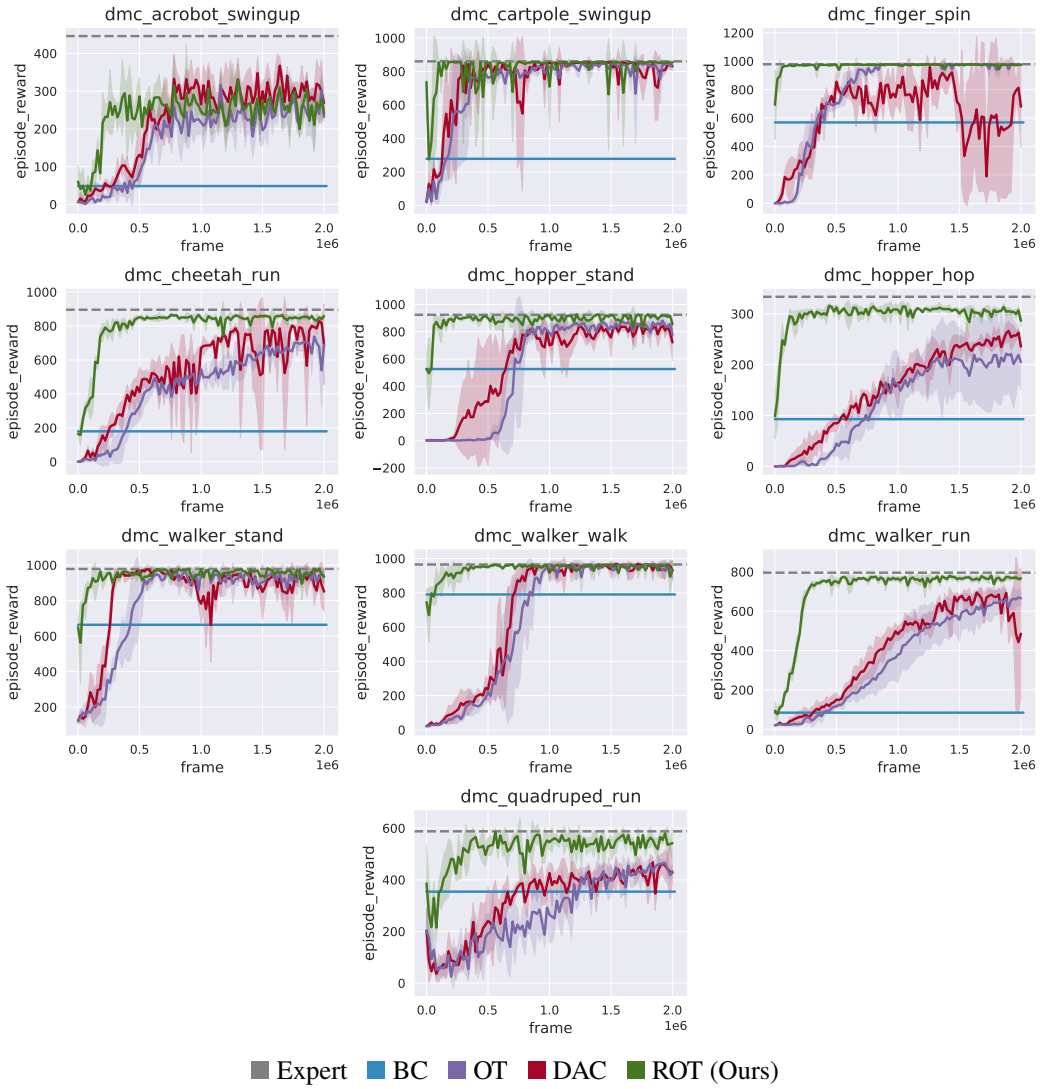

Figure 9: Pixel-based continuous control learning on 10 DMC environments. Shaded region represents ± 1 standard deviation across 5 seeds. We notice that ROT is significantly more sample efficient compared to prior work.

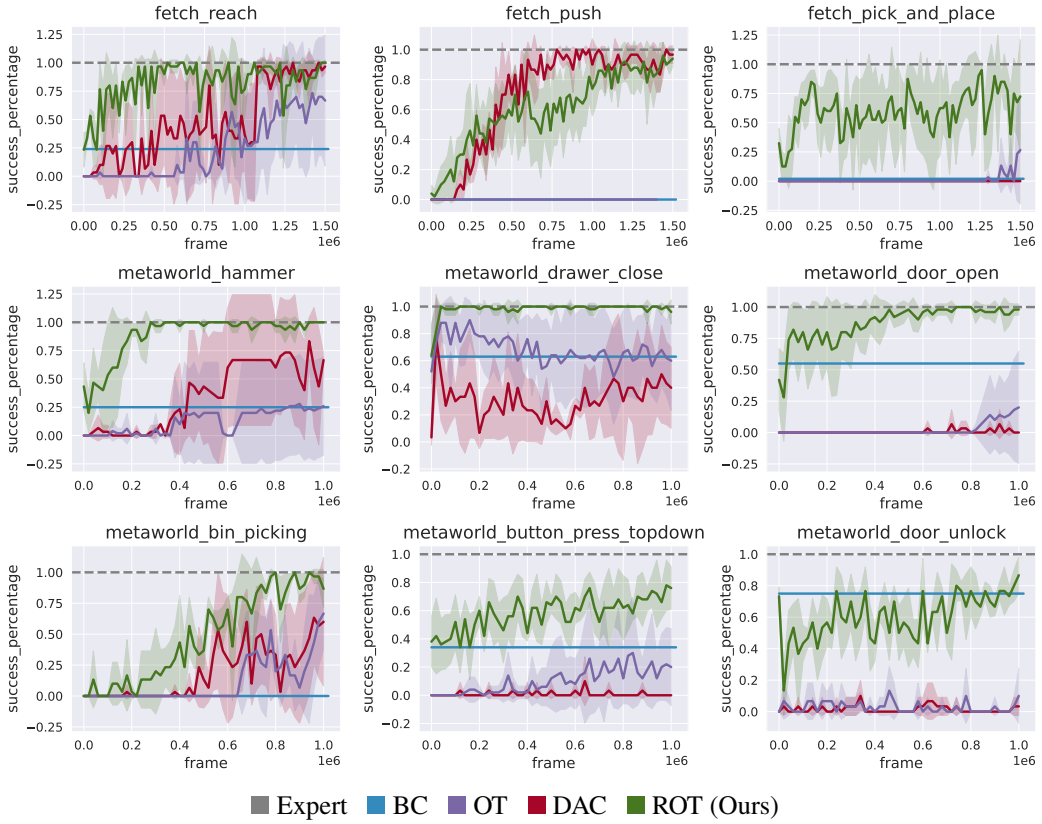

Figure 10: Pixel-based continuous control learning on 3 OpenAI Gym Robotics and 6 Meta-World tasks. Shaded region represents ± 1 standard deviation across 5 seeds. We notice that ROT is significantly more sample efficient compared to prior work.

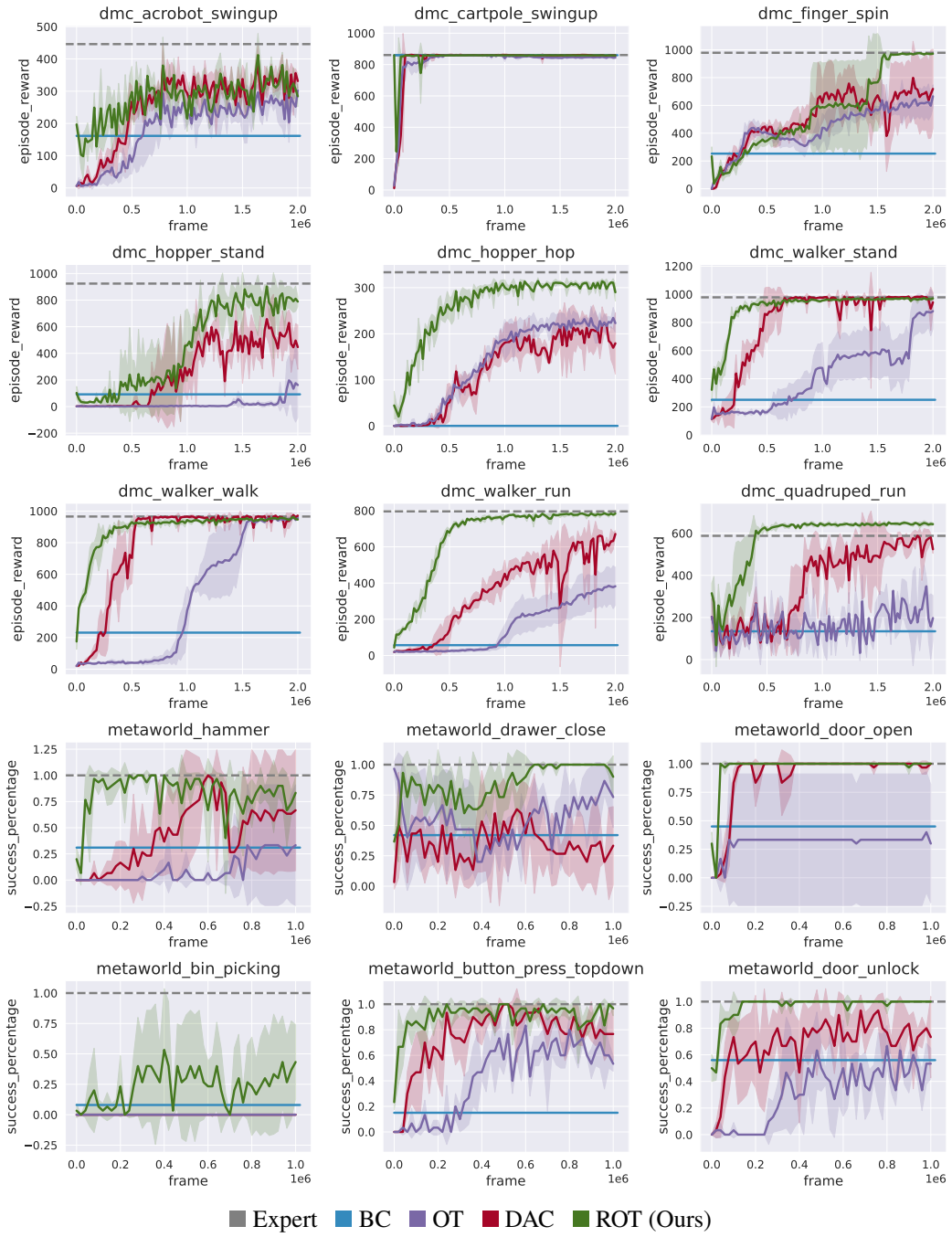

Figure 11: State-based continuous control learning on DMC and Meta-World tasks. We notice that ROT is significantly more sample efficient compared to prior work.

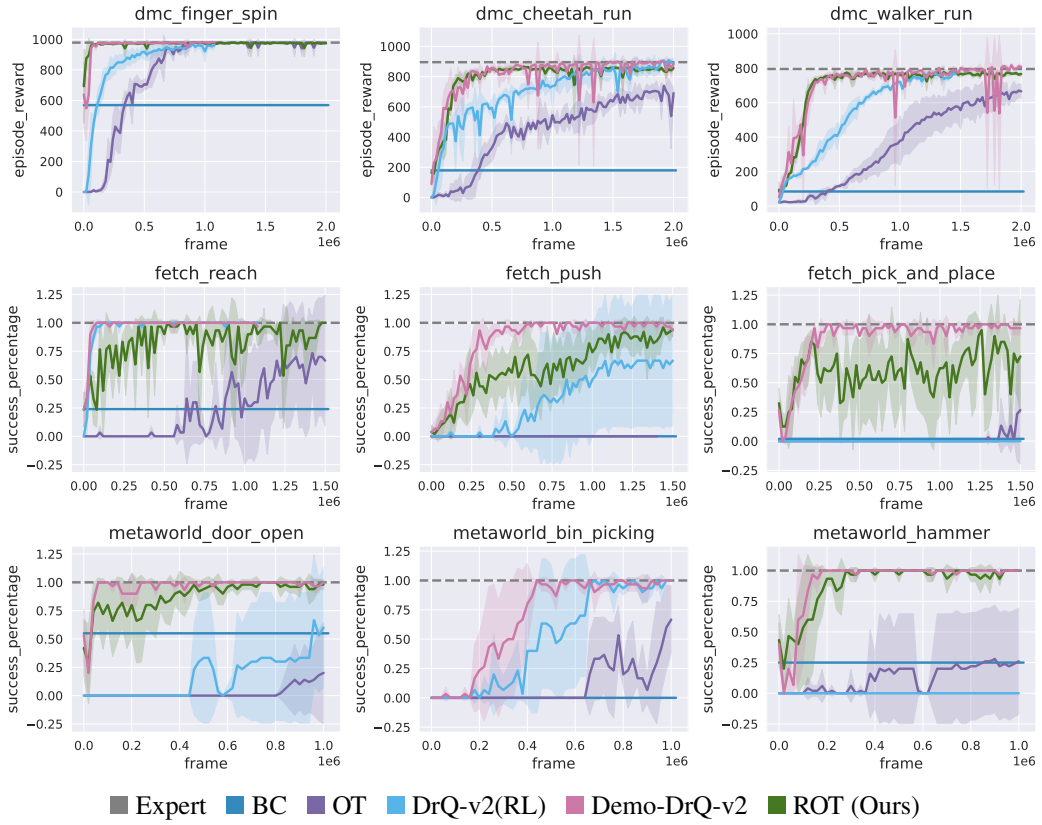

Figure 12: Pixel-based ablation analysis on the performance comparison of ROT against DrQ-v2, a reward-based RL method. Here we see that ROT can outperform plain RL that requires explicit task-reward. However, we also observe that this RL method combined with our regularization scheme provides strong results.

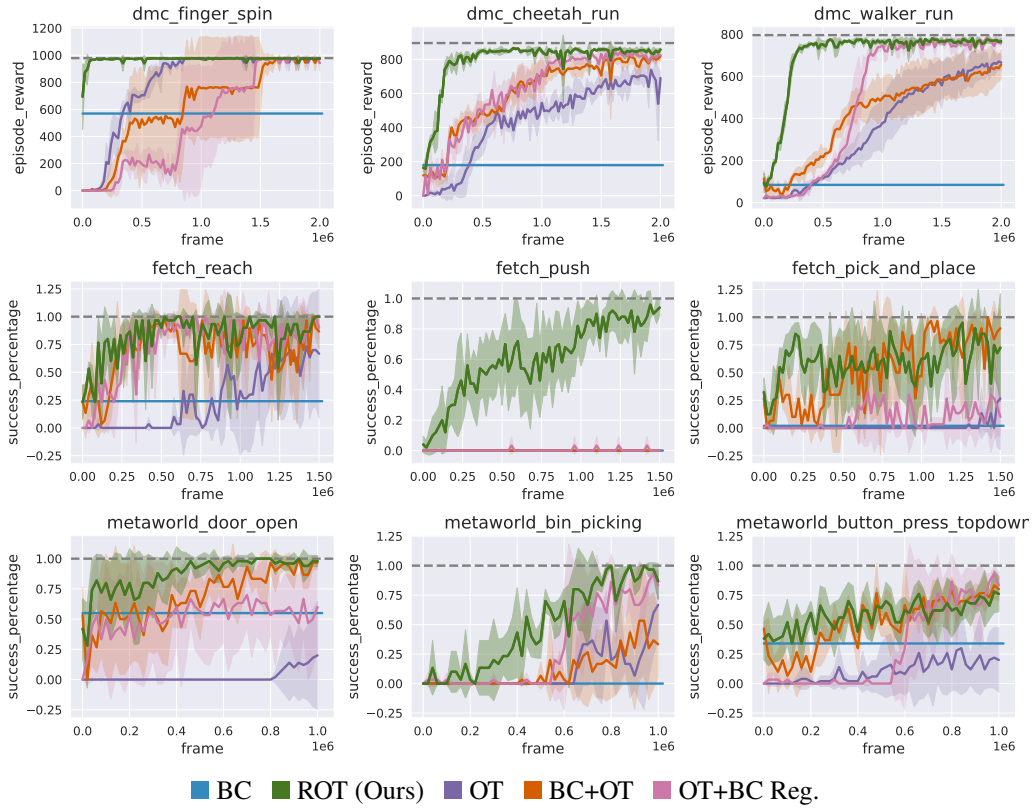

Figure 13: Pixel-based ablation analysis on the importance of pretraining and regularizing the IRL policy. The key takeaway from these experiments is that both pretraining and BC regularization are required to obtain sample-efficient imitation learning.

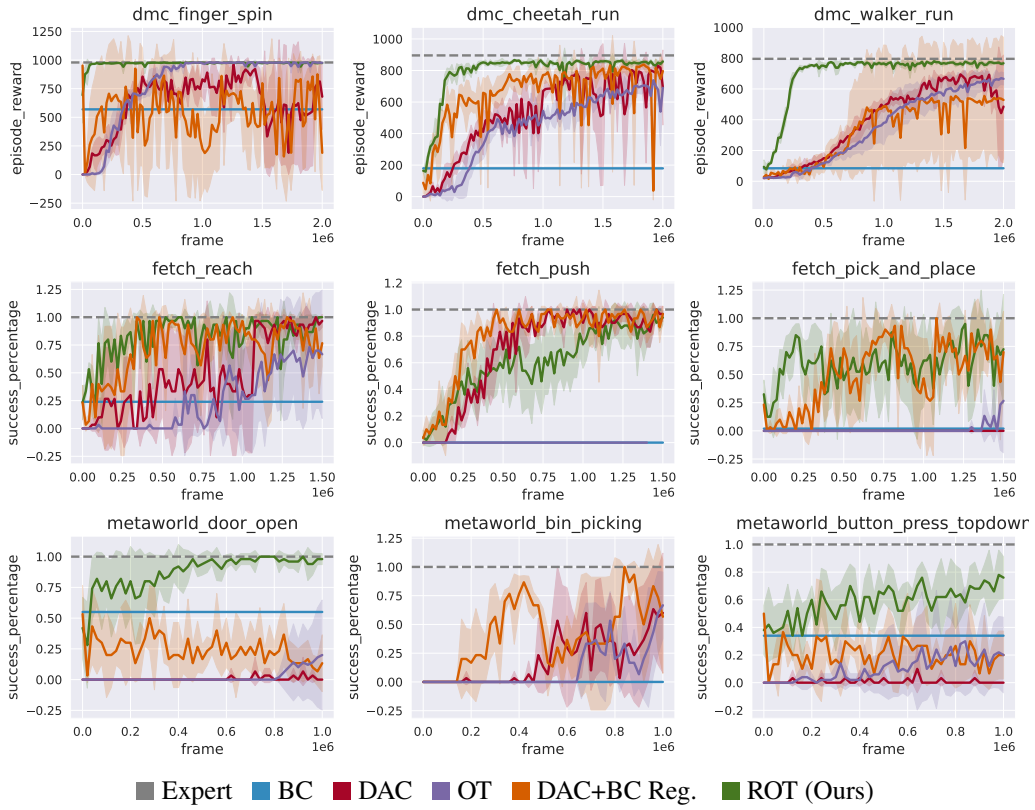

Figure 14: Pixel-based ablation analysis on the choice of base IRL method. We find that although adversarial methods benefit from regularized BC, the gains seen are smaller compared to ROT.