Task-Relevant Adversarial Imitation Learning

Konrad Żołna[1,2,*] **Scott Reed**[2,*] **Alexander Novikov**[2] **Sergio Gómez Colmenarejo**[2]

David Budden[2] **Serkan Cabi**[2] **Misha Denil**[2] **Nando de Freitas**[2] **Ziyu Wang**[2]

[1]Jagiellonian University [2]DeepMind

Abstract: We show that a critical vulnerability in adversarial imitation is the tendency of discriminator networks to learn spurious associations between visual features and expert labels. When the discriminator focuses on task-irrelevant features, it does not provide an informative reward signal, leading to poor task performance. We analyze this problem in detail and propose a solution that outperforms standard Generative Adversarial Imitation Learning (GAIL). Our proposed method, Task-Relevant Adversarial Imitation Learning (TRAIL), uses constrained discriminator optimization to learn informative rewards. In comprehensive experiments, we show that TRAIL can solve challenging robotic manipulation tasks from pixels by imitating human operators without access to any task rewards, and clearly outperforms comparable baseline imitation agents, including those trained via behaviour cloning and conventional GAIL.

Keywords: Adversarial Imitation, Robot Manipulation

1 Introduction

Generative Adversarial Networks (GANs) have been remarkably successful in image generation [1, 2], and have inspired similar approaches to imitate behaviour. In Generative Adversarial Imitation Learning (GAIL) [3], a discriminator network is trained to distinguish agent and expert behaviour through its observations, and is then used as a reward function. GAIL agents can overcome the exploration challenge by taking advantage of expert demonstrations, while also potentially achieving high asymptotic performance. Despite these attractive properties, GAIL has not had the same impact as GANs. In particular, robust adversarial imitation from pixels for control applications such as robotic manipulation remains an open challenge.

One important reason for this challenge is causal confusion in the GAIL discriminator. Given limited expert data and high-dimensional observations, the discriminator tends to exploit spurious associations between task-irrelevant features and expert labels.

To illustrate the problem, consider a simplified case where a discriminator has two feature sets. One is task-relevant, genuinely indicative of performance. The other features will be referred to as spurious, since they result in spurious associations in the discriminator. See Figure 1 for a visualization.

At the start of training, the discriminator can use either task-relevant or spurious features to accurately distinguish the agent from expert observations. However, as the agent improves, the task-relevant features become less predictive, which forces the discriminator to rely more on spurious

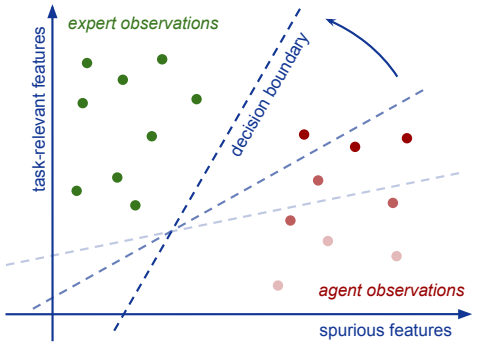
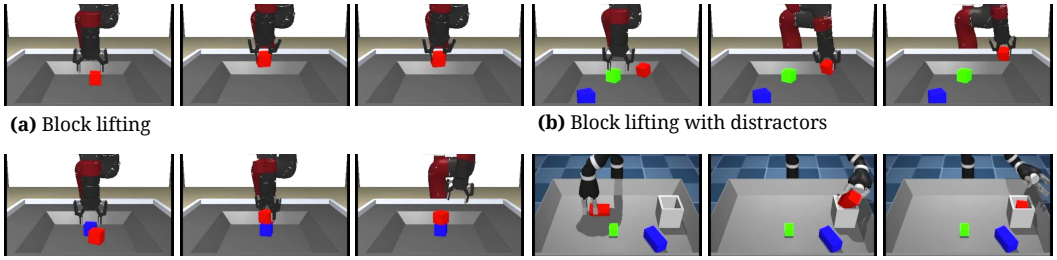


Figure 1: The decision boundary generated by a GAIL discriminator based on both task-relevant and spurious features. As the agent improves (more intense red dots), it produces observations closer to the expert w.r.t. task-relevant features. As a result, the discriminator decision boundary must increasingly rely on spurious features.

*Equal contribution. Work done at DeepMind. Corresponding authors: {kondiz,reedscot}@google.com.

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



(a) Block lifting **(b)** Block lifting with distractors

(c) Block stacking **(d)** Block insertion with distractors

Figure 2: TRAIL agents solving a variety of manipulation tasks, including with distractor objects (b,d). A comparable GAIL agent can solve lifting (a) but fails when distractor objects are added (b-d), as additional objects trigger the formation of spurious associations in the discriminator. A video showing TRAIL and GAIL agents performing these tasks can be watched at `https://youtu.be/46rSpBY5p4E`.

features. This causes the rewards to become uninformative, and the agent performance therefore degrades. The problem is especially challenging when some of the spurious features are beyond agent control (e.g. initial conditions or overall brightness). Note that GAN generators fully control the discriminator input and hence, the problem of spurious associations may be more severe in the motor control setting as compared to image GAN training.

Even in low-dimensional settings, care must be taken to account for spurious associations. For example, when using GAIL to learn simulated humanoid controllers from motion capture [4], there is a distributional shift that the discriminator could mistakenly associate with expert behavior. Learning from pixels and adding props further compounds the problem, making robotic manipulation a particularly challenging domain for adversarial imitation.

Can the problem of spurious association in discriminator networks be overcome using generic regularization or data augmentation alone? Research in the field of causality [5, 6] suggests not, in the general case. If a model is not identifiable because of confounding, additional assumptions are needed in order to estimate causal effects [7, 8].

We propose to introduce such assumptions in a flexible way. Specifically, we regularize the discriminator to be unable to distinguish between *constraining sets*, which consists of expert or agent observations, such that its elements can *only* be identified as belonging to one or the other using spurious features. This data-driven approach discourages the discriminator from forming spurious associations and, importantly, does not require enumerating all possible spurious features.

We do not attempt to automate the choice of constraining sets for all possible tasks. Rather, we show that our proposed approach can drastically improve agent performance in several challenging simulated robotic manipulation tasks. Other problem settings might require constructing different constraint sets, but our proposed approach would still apply.

This paper makes the following contributions:

1. Highlights a fundamental problem in adversarial imitation, showing that GAIL discriminators do in practice exploit spurious associations, decreasing task performance.
2. Introduces *Task-Relevant Adversarial Imitation Learning* (TRAIL), which effectively constrains the discriminator to focus on task-relevant patterns.
3. Improve performance on several vision-based robotic manipulation tasks (see Figure 2).

2 Related work

The use of demonstrations to help agent training has been studied extensively in robotics [9, 10, 11] with approaches ranging from Q-learning [12] to behavioral cloning (BC) [13].

Behavioral Cloning BC is effective in solving many control problems [13, 14, 15, 16]. It has also been successfully applied to initialize reinforcement learning (RL) training [17]. However, BC is vulnerable to cascading errors [18]. This often necessitates a large number of demonstrations for satisfactory performance. Furthermore, BC agents typically cannot exceed demonstrator performance.

Inverse RL Inverse reinforcement learning (IRL) as a way of learning reward functions from demonstrations [19, 20, 21]. RL can then be used to optimize that learned reward. Recently, Finn et al.

[22] approached continuous robotic control problems with success by applying Maximum Entropy IRL algorithms which are very closely related to GAIL [23] and have similar drawbacks.

Learning from Demonstrations Hester et al. [24] developed Deep Q-Learning from Demonstration (DQfD), in which expert trajectories are added to experience replay and jointly used to train agents along with their own experiences. This was later extended to better handle sparse-reward problems [25, 26]. Despite their efficiency, these methods still require access to rewards for learning.

Confounders and Causality The study of spurious associations and causality is vast, so here we refer to the most relevant recent work. In imitation learning, interventional approaches have been recently considered [27]. In non-interventional settings, instrumental variables and proxy variables are two popular approaches for adding extra information into models to increase their identifiability. Instrumental variables have played an important role in offline reinforcement learning [28, 29] and proxy variables have gained recent momentum in machine learning [30, 31, 32]. In this paper, we propose an alternative data-driven solution, which is both simple and effective.

GAIL Following the success of GANs in image generation, GAIL [3] applies adversarial learning to the problem of imitation. Although many variants are introduced in the literature [33, 34, 4, 35, 36, 37], making GAIL work for high-dimensional input spaces, particularly raw pixels, remains a challenge.

The problem of forming spurious associations may resemble the problem of the discriminator overfitting which has been addressed before [38, 39, 40]. Unstructured regularization, however, cannot stop the discriminator from fitting to visual features that are systematically different between the agent and the demonstrator. Often it may be easier for a discriminator to exploit spurious features compared to task-relevant features, in which case strong regularization would intensify the problem.

Stadie et al. [41] extend GAIL to the setting of third person imitation, in which the demonstrator and agent observations come from different views. To prevent the discriminator from exploiting viewpoint, gradient flipping from an auxiliary classifier is used to learn domain-invariant features. However, in our robotic manipulation setting, we found that the proposed method of extracting domain-invariant discriminator features did not adequately prevent the formation of spurious associations. Unlike in Stadie et al. [41], our goal is not third-person or cross-domain imitation, but rather a visual imitation method that is robust to spurious associations that could form even within a *single* domain.

Several recent works have focused on improving the sample efficiency of GAIL [40, 42]. Common to these approaches and to this work, is the use of off-policy actor critic agents and experience replay, to improve the utilization of available experience.

3 RL and Adversarial Imitation Background

Following the notation of Sutton and Barto [43], a Markov Decision Process (MDP) is a tuple $(\mathcal{S}, \mathcal{A}, R, P, \gamma)$ with states $\mathcal{S}$, actions $\mathcal{A}$, reward function $R(s, a)$, transition distribution $P(s'|s, a)$, and discount γ. An agent in state $s \in S$ takes action $a \in A$ according to its policy π and moves to state $s' \in \mathcal{S}$ according to the transition distribution. The goal is to find a policy that maximizes the expected sum of discounted rewards, represented by the action value function $Q^\pi(s, a) = \mathbb{E}^\pi[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$, where $\mathbb{E}^\pi$ is an expectation over trajectories starting from $s_0 = s$ and taking action $a_0 = a$ and thereafter running the policy π.

To apply RL, it is essential to have access to the reward function which is often hard to design and evaluate [44]. In addition, sparse rewards can cause exploration difficulties that pose great challenges to RL algorithms. We therefore look to adversarial imitation learning to derive a reward function from expert demonstrations.

In adversarial imitation, a reward function is learned by training a discriminator network D to distinguish between agent and expert states, or optionally state-action pairs. In the state only case, the discriminator objective is:

$$\max_{\psi} \; \mathbb{E}_{s\sim\pi_E}[\log D_\psi(s)] + \mathbb{E}_{s\sim\pi_A}[\log(1 - D_\psi(s))] \tag{1}$$

where π_A is the agent and π_E the expert policy. The discriminator can be used to obtain a reward function, which in our case is simply $R(s) = -\log(1 - D_\psi(s))$. The policy π_A is trained to maximize this reward function. The original GAIL [3] was trained on-policy with an additional entropy regularizer, but has since been adapted to the off-policy actor-critic setting [45, 39].

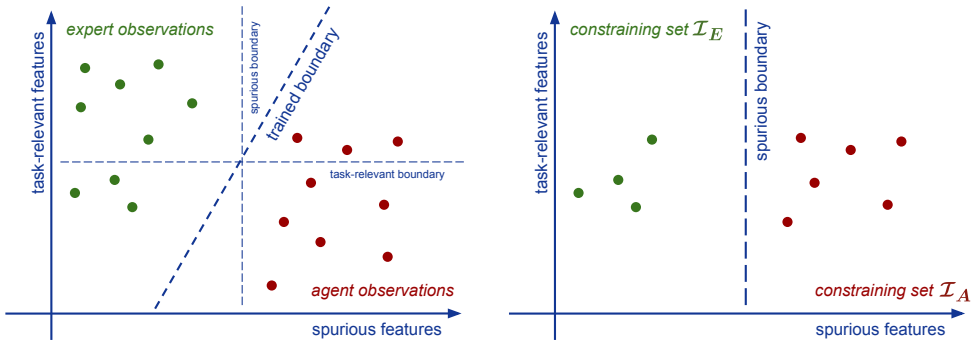


Figure 3: On the left, we illustrate a trained boundary separated into spurious and task-relevant components. Ideally, to provide informative rewards, our discriminator should only consist of the task-relevant component. On the right, constraining sets $\mathcal{I}_E$ and $\mathcal{I}_A$ are constructed such that only spurious features can be used to discriminate between them, which isolates the spurious decision boundary. Intuitively, our method works by *unlearning* this spurious boundary, so that the discriminator better captures the task-relevant boundary.

4 Task-Relevant Adversarial Imitation Learning (TRAIL)

TRAIL is designed to prevent the discriminator from forming spurious associations. To this end, we regularize the discriminator to be unable to distinguish between *constraining sets* $\mathcal{I}_E$ and $\mathcal{I}_A$, which consist of expert and agent observations, respectively, such that their elements can *only* be identified as belonging to one or the other set using spurious features. See Figure 3 for an illustration.

The TRAIL discriminator uses a cross-entropy objective as in GAIL (Eq. 1), but with an accuracy constraint applied to observations from $\mathcal{I}_E \cup \mathcal{I}_A$. For these observations, the cross-entropy objective is reversed if $accuracy(\mathcal{I}_E, \mathcal{I}_A) \geq \frac{1}{2}$ and zero otherwise, where $accuracy(\mathcal{I}_E, \mathcal{I}_A)$ is defined as the average discriminator accuracy on a balanced set of observations from the constraining sets, i.e.

$$\frac{1}{2}\mathbb{E}_{s\in\mathcal{I}_E}\left[\mathbf{1}_{\mathrm{D}_\psi(\mathrm{s})\geq\frac{1}{2}}\right] + \frac{1}{2}\mathbb{E}_{s\in\mathcal{I}_A}\left[\mathbf{1}_{\mathrm{D}_\psi(\mathrm{s})<\frac{1}{2}}\right]. \tag{2}$$

Intuitively, we discourage the discriminator from identifying and exploiting spurious patterns, and force the discriminator to forget them if they are in use.

Concretely given a batch of N examples $s_E \sim \pi_E$, $s_A \sim \pi_A$ from the expert and agent, and constraining observations $\hat{s}_E \subset \mathcal{I}_E$ and $\hat{s}_A \subset \mathcal{I}_A$, the TRAIL discriminator maximizes

$$\mathcal{L}_\psi(s_E, s_A, \hat{s}_E, \hat{s}_A) = \quad G_\psi(s_E, s_A) - \mathbf{1}_{accuracy(\hat{s}_E,\hat{s}_A)\geq\frac{1}{2}} G_\psi(\hat{s}_E, \hat{s}_A),$$

where $G_\psi(s_E, s_A)$ is finite sample estimate of the GAIL discriminator loss for D_ψ, which equals

$$G_\psi(s_E, s_A) = \sum_{i=1}^{N} \log D_\psi(s_E^{(i)}) + \log[1 - D_\psi(s_A^{(i)})], \tag{3}$$

and $\mathbf{1}_{accuracy(\hat{s}_E,\hat{s}_A)\geq\frac{1}{2}}$ indicates whether the constraint is violated.

Constraining sets $\mathcal{I}_E$ and $\mathcal{I}_A$ are drawn from demonstrations and agent episodes, respectively, such that an observation can *only* be identified as belonging to one or the other using spurious features.

In general, tasks with non-stationary distributional shifts could be devised, which would make it difficult or impossible to construct such sets. However, in many situations of interest, including our robotic manipulation setup, it is easy to propose effective and general constraining sets.

For example, a straightforward way to collect observations suitable for $\mathcal{I}_E$ and $\mathcal{I}_A$ is to execute a random policy in both expert and agent settings. This yields observations of purposeless behaviour, with no task information, varying only in task-irrelevant features. Another way to construct $\mathcal{I}_E$ and $\mathcal{I}_A$ is to use *early frames* from expert and agent episodes. Since in early frames little or no task behavior is present, this strategy turns out to be effective and no extra data has to be collected. This strategy also improves robustness with respect to variation in the initial conditions of the task; see for example block insertion in Figure 2(d).

Importantly, if the sets $\mathcal{I}_E$ and $\mathcal{I}_A$ capture some type of irrelevance but not all types, their inclusion will still result in improvements in performance. In this regard, TRAIL improves over GAIL whenever the designer has any prior knowledge of which aspects of the data are task-irrelevant.

5 Experimental setup

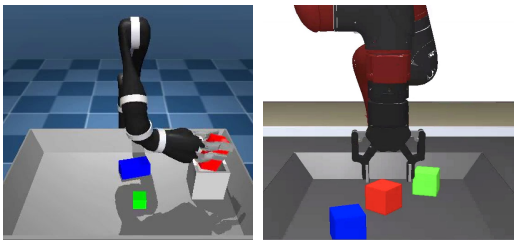


Figure 4: Two work spaces, *Jaco* (left) which uses the Jaco arm and is 20 $\times$ 20 cm, and *Sawyer* (right) which uses the Sawyer arm and more closely resembles a real robot cage and is 35 $\times$ 35 cm.

Environments We focus on solving simulated robot manipulation tasks. The environment implements two work-spaces: one with a Kinova Jaco arm (*Jaco*), and the other with a Sawyer arm (*Sawyer*). See Figure 4 for visualization and supplementary material A for a detailed description. Environment task rewards (which are not required by GAIL or by TRAIL) are sparse and equal to +1 for each step when a given task is solved and 0 otherwise. The maximum reward for the episode is 200, because this is the length of a single evaluation episode. For each task we collect 100 human demonstrations.

Data augmentation We find that data augmentation is necessary to prevent discriminator overfitting and therefore, it is used in all discriminator-based methods in this work (including all baselines). See supplementary material B for details and relevant ablations.

Actor early stopping When the agent starts to act as desired, it generates data that is indistinguishable from the expert in its behaviour. This forces the discriminator to rely on spurious features. To avoid this, we propose to stop and restart each actor episode after a certain number of steps such that successful behavior is rarely represented in agent data. This enables the discriminator to recognize the goal condition as representative of expert behavior, because it now appears much more frequently in expert observations than in agent ones. The optimal stopping step number can be found using grid search but to avoid hand-tuning, we found that the discriminator score can be used to derive an adaptive stopping criterion. Concretely, we restart an episode if the discriminator score at the current step exceeds the median score of the episode so far for $T_{patience}$ consecutive steps (in practice we set $T_{patience} = 10$). In all experiment we use the adaptive method unless clearly stated in the text. See ablations in Section 6.5 for comparison.

TRAIL Our method proposes to modify the discriminator objective. This is orthogonal to the rest of the choices that constitute GAIL algorithms (for example on-policy or off-policy). We keep these choices the same to guarantee fair comparison across methods. We used early frames to construct the constraining sets except as indicated in the relevant ablations (see Section 6.4).

Baselines The most fundamental baselines are behavior cloning, which can be trained from demonstrations only (without any interactions with the environment), and conventional GAIL. The baseline GAIL in our experiments uses Eq. 1 to train its discriminator. It is comparable to our proposed TRAIL, only differing in the discriminator objective function (due to the use of constraining sets) and actor early stopping. We also implement GAIL+AES which is the baseline GAIL enhanced with actor early stopping. Therefore, the only difference between GAIL+AES and TRAIL, is that the latter uses constraining sets. This baseline helps us to evaluate the significance of all components.

Our RL agents are based on the off-policy actor-critic D4PG algorithm [46] because of its stability and data-efficiency (see supplementary material G). Following Vecerik et al. [25], we add expert demonstrations into the agents' experience replay, and refer to the resulting RL algorithm as D4PG from Demonstrations (D4PGfD). The GAIL-based models in this paper differ from D4PGfD only by the use of the discriminator-based rewards instead of ground truth environment rewards.

We used single frames as observations but sequences could also be used, and we do not require expert actions. Not requiring actions enables learning in very off-policy settings, where the action dimensions and distributions of the demonstrator (another robot or human) could be different from those of the agent [47]. Finally, to make our comparisons as fair as possible we used a shared implementation, and the same agent configurations (e.g. number of actors) and network architecture.

6 Results

6.1 Evaluation on diverse manipulation tasks

We first focus on four tasks which do not have any purposely introduced spurious features; *lift* and *stack* with the *Sawyer* robot, and *stack banana* and *insertion* with the *Jaco* robot.

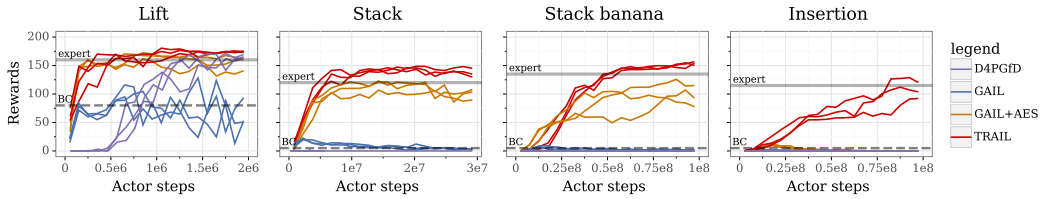


Figure 5: Results comparing TRAIL, GAIL+AES and baselines for diverse manipulation tasks.

The results are shown in Figure 5. The tasks are very challenging as evidenced by the performance of BC agents which only partially solve one task – *lift*. The baseline GAIL fails, even though the experiments are conducted in fully controlled environments (seemingly free of spurious features), indicating that GAIL discriminators identify less evident spurious features like positions of distracting objects in expert demonstrations. TRAIL reaches expert-level performance on all four tasks. GAIL+AES always performs worse than TRAIL and completely fails for the *insertion* task, stressing the importance of the constraining sets which directly address the problem of spurious associations.

D4PGfD, which uses sparse ground truth rewards, is able to solve only *lift*, which suggests that dense discriminator-based rewards aid exploration and are critical to solve more challenging tasks.

6.2 Expert with different appearance

In this section, we intentionally introduce a visual spurious feature by changing the expert appearance, increasing the difficulty of the *lift* task. Figure 6 shows the difference in appearance, which is to vary the gripper color from light to dark. This is meant to simulate the type of distributional shifts that commonly occur when dealing with real robots, where the appearance can change after initial data collection due to scratches and wear, for example. The results are presented in Figure 6.

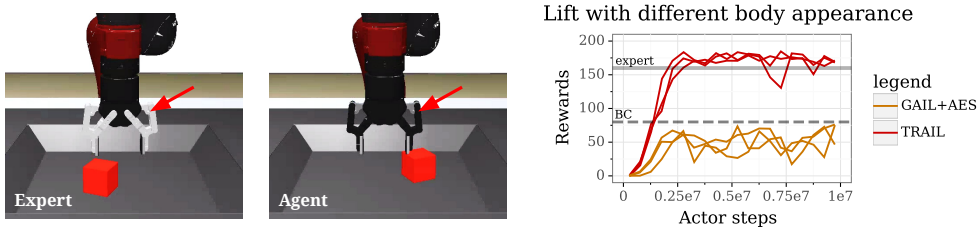


Figure 6: When the appearance differs (as highlighted by the arrow), TRAIL clearly outperforms baselines.

With the modified expert appearance, GAIL+AES performance degrades in block lifting, despite the use of data augmentation and the addition of 100-step actor early stopping (AES). TRAIL solves the task and achieves performance better than the expert thus being robust to spurious features.

6.3 Block lifting with distractors

The previous section showed that a consistent difference in expert appearance degrades the performance of GAIL, while TRAIL remains unaffected. In this section, *instead* of changing the expert appearance, we alter the positions of props in the environment. We consider another variant of the lift task, *lift distracted*, when two extra blocks are added (blue and green, see Figure 2(a,b)). Here, there are *no* differences in expert appearance.

In this section, on top of the previously defined baselines, we consider additional GAIL-based approaches proposed by Reed et al. [39]; using either a randomly initialized convolutional network, or a convolutional critic network, to provide fixed vision features on top of which a tiny discriminator network is trained. We call these two baselines GAIL+random and GAIL+critic respectively.

As shown in Figure 7, all methods, including new baselines, perform satisfactorily on *lift*, but GAIL+AES and TRAIL clearly do best. As expected, the performance of BC on *lift distracted* is similar to its performance on *lift*, despite the two additional blocks. The two additional blocks, however, affect all GAIL-based baselines, despite being irrelevant to the task, while TRAIL succeeds.

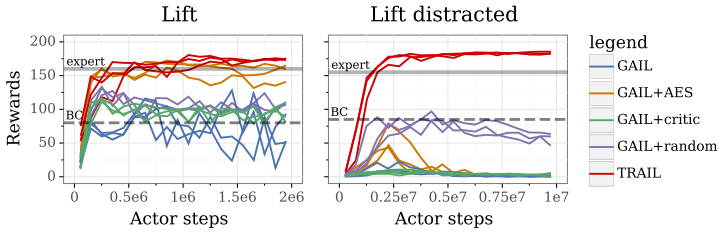


Figure 7: Results for *lift* and *lift distracted*. When distractors are introduced only TRAIL succeeds.

Even for this relatively easy task in the simulated environment without evident task-irrelevant features, the baseline discriminators find spurious associations to obtain perfect accuracy. It is achieved by simply memorizing all 100 initial block positions from the demonstration set. Note that this can not be prevented using unstructured regularization, as a discriminator has to be able to infer blocks' positions from the image to provide informative reward. However, the positions of task-irrelevant objects should be ignored. TRAIL is able to handle the variety of initial cube positions, achieving better than expert performance on *lift distracted*.

We conducted an additional experiment termed *lift distracted seeded*, in which the initial block positions are randomly drawn from the expert demonstrations. Therefore, it is impossible to reliably discriminate between expert and agent episodes using distractor positions. GAIL+AES performed much better on *lift distracted seeded* than on *lift distracted*, indicating that the discriminator does form a spurious association between distractor positions and expert labeling. We provide more detailed analysis of *lift distracted* (and the *lift distracted seeded* variant) in supplementary material C.

6.4 Constructing the constraining sets $\mathcal{I}_E$ and $\mathcal{I}_A$

In the previous sections, early frames were used to construct the constraining sets. Here, we compare the performance of this approach with another previously mentioned strategy: the use of a random policy to construct $\mathcal{I}_E$ and $\mathcal{I}_A$. To tease out the differences between these two variants, we consider a harder task by combining two previous tasks: *lift distracted* and *different body appearance*. The results are presented in Figure 8 which also shows the difference in robot appearance and distractors.

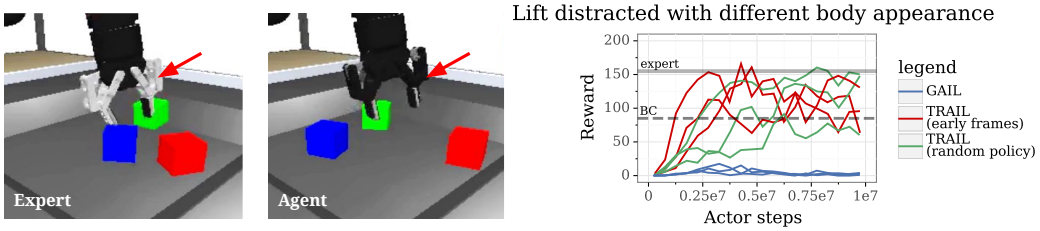


Figure 8: The hardest variant of *lift* (red cube) task; not only robot appearance changes but also distractors are added. There is no clear difference between TRAIL variants but both of them decisively outperform GAIL.

The new task is indeed harder and it takes longer for TRAIL methods to achieve performance better than the BC baseline, which is not affected by the different body appearance. GAIL is clearly outperformed and does not take off.

The performance of TRAIL-early and TRAIL-random is similar. Hence, by default we simply use early frames. This choice is pragmatic as it does not require that we collect any additional data, and hence the comparison with GAIL and other baselines is fair. It is also simple to apply in practice, even if one no longer has access to the expert setup. Finally, it is general enough to be successfully used across all robotic manipulation tasks considered in this work.

To decide how many initial frames should be used to construct $\mathcal{I}_E$ and $\mathcal{I}_A$, we conducted an ablation study and found that the method is not very sensitive to this choice (see supplementary material D). Hence, we chose 10 initial frames, and intentionally used the same number for all tasks to further emphasize generality of this choice.

6.5 Ablation studies

Actor early stopping We analyze the importance of actor early stopping on 3 tasks in the *Jaco* work-space: lift red cube (*lift*), put red cube in box (*box*), and stack red cube on blue cube (*stack*). We consider three variants of GAIL+AES with the following termination policies: a) fixed step (50), b) adaptive (as described in Section 5), and c) based on ground truth task rewards (as adaptive but uses ground truth rewards instead of discriminator estimates; we also use $T_{patience} = 10$).

Results are presented in Figure 9. Termination based on task reward is clearly superior; although unrealistic in practice, it defines the performance upper-bound and clearly shows that actor early stopping is beneficial. The adaptive approach is robust and reaches human performance on all tasks. A fixed termination policy, when tuned, can be very effective. The same fixed termination step, however, does not work for all tasks (see supplementary material E for further details).

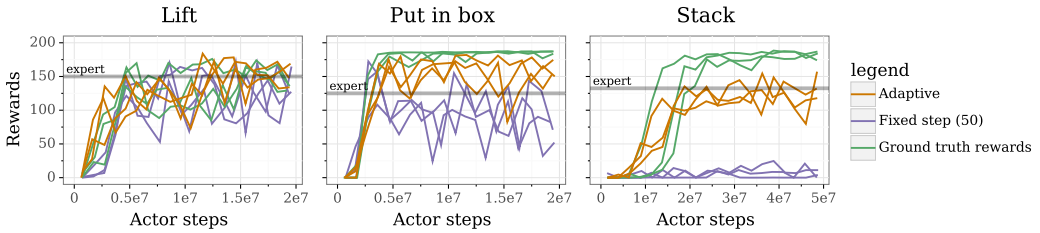


Figure 9: Results for *lift*, *box*, and *stack* on *Jaco* environment for GAIL+AES with different variants of AES.

The experiments show that successful agent episodes can trigger formation of spurious associations and this can be prevented directly with actor early stopping. However, the method is not sufficient in more difficult cases, as showed in the previous experiments on more challenging tasks.

TRAIL generalization abilities We analyze how the use of constraining sets affects discriminator generalization. We compare TRAIL with GAIL+AES, as the only difference between them is the use of constraining sets. We use the *lift distracted* task. To measure the discriminator's generalization capacity, we collected 25 extra holdout demonstrations. We visualize and compare discriminator predictions on training and hold-out demonstrations in Figure 10.

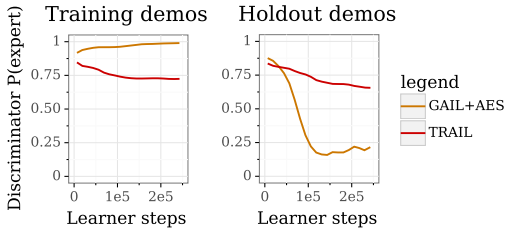


Figure 10: TRAIL generalizes better than GAIL+AES.

The TRAIL discriminator generalizes well, as it assigns similar values to training and hold-out demos throughout the training (red curves are aligned to each other). The GAIL+AES discriminator quickly forms spurious associations which do not generalize to holdout demos. Its predictions for training demos approach 1 while holdout demos are recognized as agent trajectories (predictions below 0.5).

Learning with an oracle discriminator To further understand the contribution of learned reward functions, we train agents with rewards from an oracle discriminator, which always assigns a reward of 1 for expert frames and 0 for agent frames. This simulates a discriminator that perfectly exploits a spurious association between task-irrelevant visual features and expert labels. On the *lift* task, agents using this fixed oracle reward achieve roughly half the reward of TRAIL asymptotically, and on *lift distracted* they do not take off. See supplementary material F for learning curves.

7 Conclusions

We have shown that a critical vulnerability in GAIL is the tendency of discriminators to form spurious associations between visual features and expert labels, which results in uninformative rewards. The problem can be triggered by many factors which can be addressed directly to improve agent performance, as done by GAIL+AES, but it does not scale to harder problems. TRAIL, however, does not require any knowledge about potential spurious associations as they are neutralized automatically. Without access to expert actions nor environment rewards, TRAIL achieved expert level on a diverse set of manipulation tasks, outperforming comparable GAIL, behavior cloning, and D4PGfD agents.

Acknowledgments

We would like to thank reviewers for their useful comments. Konrad Żołna is supported by the National Science Center, Poland (2017/27/N/ST6/00828, 2018/28/T/ST6/00211).

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[2] A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

[3] J. Ho and S. Ermon. Generative adversarial imitation learning. In *NeurIPS*, 2016.

[4] J. Merel, Y. Tassa, S. Srinivasan, J. Lemmon, Z. Wang, G. Wayne, and N. Heess. Learning human behaviors from motion capture by adversarial imitation. *Preprint arXiv:1707.02201*, 2017.

[5] H. A. Simon. Spurious correlation: A causal interpretation. *Journal of the American Statistical Association*, 1954.

[6] J. Pearl. An introduction to causal inference. *The international journal of biostatistics*, 2010.

[7] J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2009.

[8] J. Peters, D. Janzing, and B. Schölkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. MIT Press, 2017.

[9] P. Bakker and Y. Kuniyoshi. Robot see, robot do: An overview of robot imitation. In *AISB96 Workshop on Learning in Robots and Animals*, 1996.

[10] M. Kawato, F. Gandolfo, H. Gomi, and Y. Wada. Teaching by showing in kendama based on optimization principle, 1994.

[11] H. Miyamoto, S. Schaal, F. Gandolfo, H. Gomi, Y. Koike, R. Osu, E. Nakano, Y. Wada, and M. Kawato. A kendama learning robot based on bi-directional theory. *Neural networks*, 1996.

[12] S. Schaal. Learning from demonstration. In *NeurIPS*, 1997.

[13] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *NeurIPS*, 1989.

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

[15] Y. Duan, M. Andrychowicz, B. Stadie, O. J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning. In *NeurIPS*, 2017.

[16] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In *ICRA*, 2018.

[17] 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. *Preprint arXiv:1709.10087*, 2017.

[18] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011.

[19] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008.

[20] A. Y. Ng, S. J. Russell, et al. Algorithms for inverse reinforcement learning. In *ICML*, 2000.

[21] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

[22] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *ICML*, 2016.

[23] C. Finn, P. Christiano, P. Abbeel, and S. Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *Preprint arXiv:1611.03852*, 2016.

[24] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, G. Dulac-Arnold, J. Agapiou, J. Z. Leibo, and A. Gruslys. Deep q-learning from demonstrations. In *AAAI*, 2018.

[25] M. Vecerik, 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. *Preprint arXiv:1704.03732*, 2017.

[26] T. Pohlen, B. Piot, T. Hester, M. G. Azar, D. Horgan, D. Budden, G. Barth-Maron, H. van Hasselt, J. Quan, M. Večerík, et al. Observe and look further: Achieving consistent performance on atari. *Preprint arXiv:1805.11593*, 2018.

[27] P. de Haan, D. Jayaraman, and S. Levine. Causal confusion in imitation learning. In *NeurIPS*, 2019.

[28] S. Bradtke and A. Barto. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 1996.

[29] M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 2003.

[30] C. Louizos, U. Shalit, J. M. Mooij, D. A. Sontag, R. S. Zemel, and M. Welling. Causal effect inference with deep latent-variable models. In *NeurIPS*, 2017.

[31] C. Lu, B. Schölkopf, and J. M. Hernández-Lobato. Deconfounding reinforcement learning in observational settings. *Preprint arXiv:1812.10576*, 2018.

[32] G. Tennenholtz, S. Mannor, and U. Shalit. Off-policy evaluation in partially observable environments. *Preprint arXiv:1907.04543*, 2019.

[33] Y. Li, J. Song, and S. Ermon. Infogail: Interpretable imitation learning from visual demonstrations. In *NeurIPS*, 2017.

[34] J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning. In *ICLR*, 2018.

[35] Y. Zhu, Z. Wang, J. Merel, A. A. Rusu, T. Erez, S. Cabi, S. Tunyasuvunakool, J. Kramár, R. Hadsell, N. de Freitas, and N. Heess. Reinforcement and imitation learning for diverse visuomotor skills. In *RSS*, 2018.

[36] N. Baram, O. Anschel, I. Caspi, and S. Mannor. End-to-end differentiable adversarial imitation learning. In *ICML*, 2017.

[37] F. Behbahani, K. Shiarlis, X. Chen, V. Kurin, S. Kasewa, C. Stirbu, J. Gomes, S. Paul, F. A. Oliehoek, J. V. Messias, and S. Whiteson. Learning from demonstration in the wild. In *ICRA*, 2019.

[38] X. B. Peng, A. Kanazawa, S. Toyer, P. Abbeel, and S. Levine. Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow. *Preprint arXiv:1810.00821*, 2018.

[39] S. Reed, Y. Aytar, Z. Wang, T. Paine, A. van den Oord, T. Pfaff, S. Gomez, A. Novikov, D. Budden, and O. Vinyals. Visual imitation with a minimal adversary, 2018.

[40] L. Blondé and A. Kalousis. Sample-efficient imitation learning via generative adversarial nets. *Preprint arXiv:1809.02064*, 2018.

[41] B. C. Stadie, P. Abbeel, and I. Sutskever. Third-person imitation learning. *Preprint arXiv:1703.01703*, 2017.

[42] F. Sasaki, T. Yohira, and A. Kawaguchi. Sample efficient imitation learning for continuous control. In *ICLR*, 2019.

[43] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[44] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine. End-to-end robotic reinforcement learning without reward engineering. *Preprint arXiv:1904.07854*, 2019.

[45] I. Kostrikov, K. K. Agrawal, D. Dwibedi, S. Levine, and J. Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *ICLR*, 2019.

[46] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, A. Muldal, N. Heess, and T. Lillicrap. Distributed distributional deterministic policy gradients. *Preprint arXiv:1804.08617*, 2018.

[47] K. Zolna, N. Rostamzadeh, Y. Bengio, S. Ahn, and P. O. Pinheiro. Reinforced imitation in heterogeneous action space. *Preprint arXiv:1904.03438*, 2019.

Supplementary material

A Detailed description of environment

All our simulations are conducted using MuJoCo[2] [1]. We test our proposed algorithms in a variety of different envrionments using simulated Kinova Jaco[3], and Sawyer robot arms[4] (see Figure 11). We use the Robotiq 2F85 gripper[5] in conjuction with the Sawyer arm.

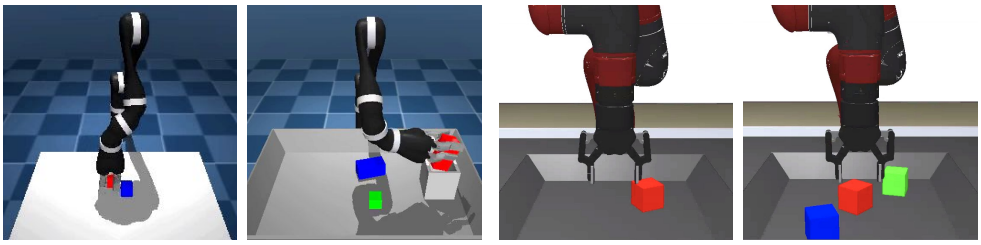


Figure 11: Two work spaces, *Jaco* (top) which uses the Jaco arm and is 20×20 cm, and *Sawyer* (bottom) which uses the Sawyer arm and more closely resembles a robot cage and is 35×35 cm.

To provide demonstrations, we use the SpaceNavigator 3D motion controller[6] to set Cartesion velocities of the robot arm. The gripper actions are implemented via the buttons on the controller. All demonstrations in our experiments are provided via human teleoperation and we collected 100 demonstrations for each experiment.

Jaco: When using the Jaco arm, we use joint velocity control (9DOF) where we control all 6 joints of arm and all 3 joints of the hand. The simulation is run with a numerical time step of 10 milliseconds, integrating 5 steps, to get a control frequency of 20HZ. The agent uses a frontal camera of size 64×64 (see Figure 12). For a full list of observations the agent sees, please refer to Table 1(a).

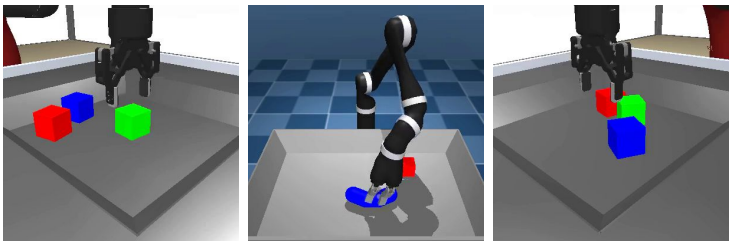


Figure 12: Illustration of the pixels inputs to the agent: front left camera (left), frontal camera (middle) and front right camera (right).

Sawyer: When using the Sawyer arm, we use Cartesian velocity control (6DOF) for the robot arm and add one additional action for the gripper resulting in 7 degrees of freedom. The simulation is run with a numerical time step of 10 milliseconds, integrating 10 steps, to get a control frequency of 10HZ. The agent uses two frontal cameras of size 64×64 situated on the left and right side of the robot cage respectively (see Figure 12). For a full list of observations the agent sees, please refer to Table 1(b).

For all environments considered in this paper, we provide sparse rewards (*i.e.*. if task is accomplished, the reward is 1 and 0 otherwise). In experiments regards our proposed methods, rewards are only used for evaluation purposes and not for training the agent.

[2] www.mujoco.org

[3] https://www.kinovarobotics.com/en/products/assistive-technologies/kinova-jaco-assistive-robotic-arm

[4] https://www.rethinkrobotics.com/sawyer/

[5] https://robotiq.com/products/2f85-140-adaptive-robot-gripper

[6] https://www.3dconnexion.com/spacemouse_compact/en/

Table 1: Observation and dimensions.

a) Jaco

Feature Name	Dimensions
frontal camera	$64 \times 64 \times 3$
base force and torque sensors	6
arm joints position	6
arm joints velocity	6
wrist force and torque sensors	6
hand finger joints position	3
hand finger joints velocity	3
hand fingertip sensors	3
grip site position	3
pinch site position	3

b) Sawyer

Feature Name	Dimensions
front left camera	$64 \times 64 \times 3$
front right camera	$64 \times 64 \times 3$
arm joint position	7
arm joint velocity	7
wrist force sensor	3
wrist torque sensor	3
hand grasp sensor	1
hand joint position	1
tool center point cartesian orientation	9
tool center point cartesian position	3
hand joint velocity	1

B Data augmentation

Traditional data augmentation has proved beneficial in imitation learning [2]. Surprisingly, this has received little attention in prior publications on GAIL. However, we find that data augmentation is a necessary procedure to prevent discriminator overfitting. Therefore, all discriminator-based methods in this work (including all baselines) use data augmentation with an exception of the experiments in this section. We distort images by randomly changing brightness, contrast and saturation, randomly cropping and rotating, and adding Gaussian noise. When multiple sensor inputs are available (e.g. multiple cameras), we also randomly drop some of the inputs, always leaving at least one active.

We also considered regularizing the GAIL discriminator with spectral normalization [3]. It performed slightly better than GAIL, but still failed in the presence of distractor objects, and we thus omit spectral normalization in the main experiments for simplicity.

In this section, we measure the importance of data augmentation. In Table 2, we report the best reward obtained in the first 12 hours of training (averaged for all seeds; see Figure 13 for full curves).

Table 2: Influence of data augmentation (evaluated on rewards) for *lift* and *lift distracted*.

Task	Regularization	Data augmentation Yes	Data augmentation No
lift	GAIL+AES	∼165	∼115
	TRAIL	∼155	∼165
lift distracted	GAIL+AES	∼30	∼5
	TRAIL	∼180	∼10

The results show that data augmentation is essential in *lift distracted*. The performance of TRAIL is not affected by the lack of data augmentation on the *lift* task, whereas the performance of GAIL+AES is.

We also checked the impact of data augmentation on the baseline GAIL. It was always beneficial. Since the baseline GAIL which uses data augmentation usually does not take off, the even worse version is not included in our plots.

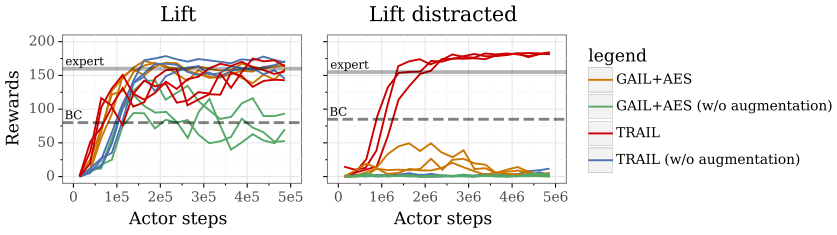


Figure 13: Results for *lift* and *lift distracted* in *Sawyer* work space. TRAIL and GAIL+AES are by default with data augmentation. Additional results without data augmentation are presented to show its importance.

C Block lifting with distractors

In Section 6.3, we consider a variant of the lift task, *lift distracted*, when two extra blocks are added. We also conducted an additional experiment termed *lift distracted seeded*, in which the initial block positions are randomly drawn from the expert demonstrations. Therefore, it prevents from discriminating between expert and actor episodes using distractor positions. Note this initialization procedure is not applied at the evaluation time, keeping the evaluation scores comparable between *lift distracted* and *lift distracted seeded*. The results for all lift variants are presented in Figure 14.

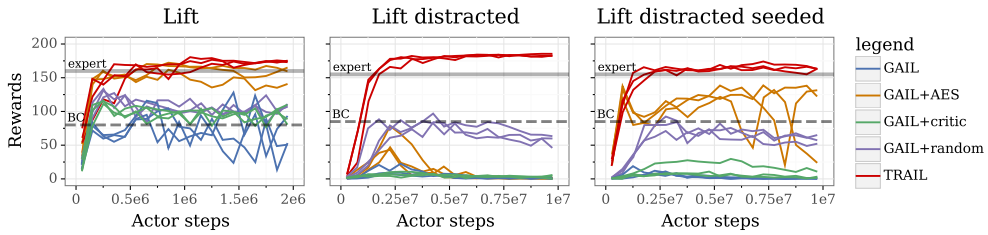


Figure 14: Results for *lift*, *lift distracted*, and *lift distracted seeded*. Only TRAIL excels.

All methods perform satisfactorily on *lift*, but the proposed methods, GAIL+AES and TRAIL, clearly outperform the other methods.

As expected, the performance of BC on *lift distracted* is similar to its performance on *lift*, despite the two additional blocks. The two additional blocks, however, affect all GAIL-based baselines, despite being irrelevant to the task. Only TRAIL succeeds.

It turns our that even for this relatively easy task in the simulated environment without evident task-irrelevant features, the baseline discriminators find spurious associations to obtain perfect accuracy. It is achieved by simply memorizing all 100 initial block positions from the demonstration set. Note that this can not be prevented using unstructured regularization, as a discriminator has to be able to infer blocks' positions from the image to provide informative reward. However, the positions of task-irrelevant objects should be ignored. TRAIL is the only method that is able to handle the variety of initial cube positions, achieving better than expert performance on *lift distracted*.

GAIL+AES performed much better on *lift distracted seeded* than on *lift distracted*, indicating that the discriminator do form a spurious association between distractor positions and expert labeling.

Interestingly, GAIL+random performs reasonably on both *lift distracted* and *lift distracted seeded*. Given GAIL+random's strong performance, we conducted additional experiments to evaluate its effectiveness when trained with adaptive early stopping. In Figure 15 we present GAIL+AES+random, which is GAIL-random trained with actor early stopping.

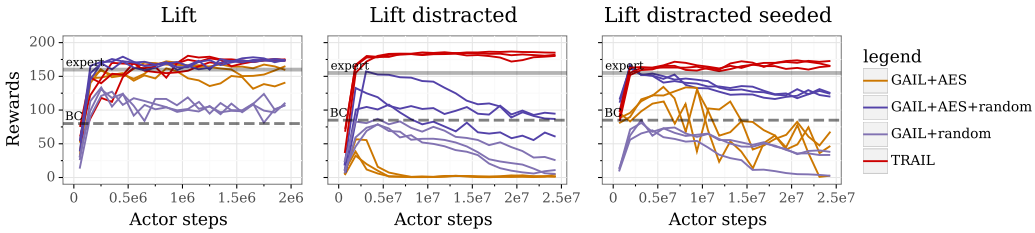


Figure 15: Results for *lift*, *lift distracted*, and *lift distracted seeded* including GAIL+AES+random baseline.

When random features are used it is harder to identify and exploit the spurious associations. Indeed, GAIL+AES+random and TRAIL are the only methods exceeding BC performance on *lift distracted*. However, TRAIL performance is clearly better (obtains higher rewards and never gets overfitted) which means that GAIL+AES+random is still prone to exploiting spurious associations.

D Early frames ablation study

Here, we vary the number of early frames of each episode used to form the constraining sets. We report that a range of values from 1 up to 20 works well across both tasks (Put in box and Stack banana), with performance gradually degrading as the value increased beyond 20 (Figure 16).

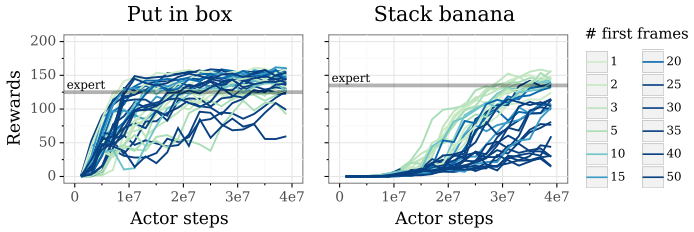


Figure 16: TRAIL performance, varying the number of first frames in each episode used to form $\mathcal{I}$.

E Fixed termination policy

As mentioned in the subsection 6.5, the most basic early termination policy – fixed step termination – may be very effective if tuned. Since the tuning may be expensive in practice, we recommend using adaptive early stopping (GAIL+AES). However, for the sake of completeness we provide results for fixed step termination policy depending on the hyperparameter tuned. The results for *stack* task are presented in Figure 17. As can be inferred from the figure, the performance is very sensitive to the fixed step hyperparameter. We refer to Section 6.5 for more comments on all methods.

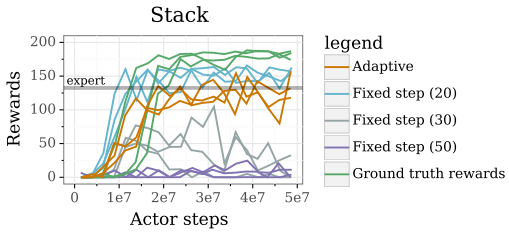


Figure 17: Results for *stack* in *Jaco* work space. Fixed step termination policy can be very effective but the final performance is very sensitive to the hyperparameter. GAIL+AES does not need tuning nor access to the environment reward.

F Learning with oracle discriminator

To further understand the contribution of learned reward functions, we train agents with rewards from an oracle discriminator, which always assigns a reward of 1 for expert frames and 0 for agent frames. This simulates a discriminator that perfectly exploits a spurious association between task-irrelevant visual features and expert labels. On the *lift* task, agents using this fixed oracle reward achieve roughly half the reward of TRAIL asymptotically, and on *lift distracted* they do not solve the task (average rewards are less than 5). The learning curves are presented in Figure 18.

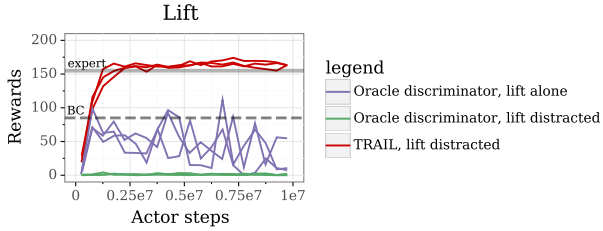


Figure 18: With fixed rewards, the agent is able to learn *lift* somewhat, but performs worse than TRAIL. When distractor blocks are added, the fixed reward agent fails to learn completely.

G D4PG

We use D4PG [4] as our main training algorithm. Briefly, D4PG is a distributed off-policy reinforcement learning algorithm for continuous control problems. In a nutshell, D4PG uses Q-learning for policy evaluation and Deterministic Policy Gradients (DPG) [5] for policy optimization. An important characteristic of D4PG is that it maintains a replay memory $\mathcal{M}$ (possibility prioritized [6]) that stores SARS tuples which allows for off-policy learning. D4PG also adopts target networks for increased training stability. In addition to these principles, D4PG utilized distributed training, distributional value functions, and multi-step returns to further increase efficiency and stability. In this section, we explain the different ingredients of D4PG.

D4PG maintains an online value network $Q(s, a|\theta)$ and an online policy network $\pi(s|\phi)$. The target networks are of the same structures as the value and policy network, but are parameterized by different parameters θ' and ϕ' which are periodically updated to the current parameters of the online networks.

Given the Q function, we can update the policy using DPG:

$$\mathcal{J}(\phi) = \mathbb{E}_{s_t \sim \mathcal{M}}\big[\nabla_\phi Q(s_t, \pi(s_t|\phi)|\theta)\big]. \tag{4}$$

Instead of using a scalar Q function, D4PG adopts a distributional value function such that $Q(s_t, a|\theta) = \mathbb{E}\big[Z(s_t, a|\theta)\big]$ where Z is a random variable such that $Z = z_i$ w.p. $p_i \propto \exp(\omega(s_t, a|\theta))$. The z_i's take on V_{bins} discrete values that ranges uniformly between V_{min} and V_{max} such that $z_i = V_{min} + i\frac{V_{max}-V_{min}}{V_{bins}}$ for $i \in \{0, \cdots, V_{bins} - 1\}$.

To construct a bootstrap target, D4PG uses N-step returns. Given a sampled tuple from the replay memory: $s_t, a_t, \{r_t, r_{t+1}, \cdots, r_{t+N-1}\}, s_{t+N}$, we construct a new random variable Z' such that $Z' = z_i + \sum_{n=0}^{N-1} \gamma^n r_{t+n}$ w.p. $p_i \propto \exp(\omega(s_{t+N}, \pi(s_{t+N}|\phi')|\theta'))$. Notice, Z' no longer has the same support. We therefore adopt the same projection Φ employed by [7]. The training loss for the value function

$$\mathcal{L}(\theta) = \mathbb{E}_{s_t, a_t, \{r_t, \cdots, r_{t+N-1}\}, s_{t+N} \sim \mathcal{M}}\big[H\big(\Phi(Z'), Z(s_t, a_t|\theta)\big)\big], \tag{5}$$

where H is the cross entropy.

D4PG is also distributed following [6]. Since all learning processes only rely on the replay memory, we can easily decouple the 'actors' from the 'learners'. D4PG therefore uses a large number of independent actor processes which act in the environment and write data to a central replay memory process. The learners could then draw samples from the replay memory for learning. The learner also serves as a parameter server to the actors which periodically update their policy parameters from the learner.

In our experiments, we always have access to expert demonstrations. We, therefore adopt the practice from DQfD and DDPGfD and put the demonstrations into our replay buffers. For more details see Algorithms 1 and 2.

Algorithm 1 Actor

Given: an experience replay memory $\mathcal{M}$
for $n_{episodes}$ **do**
  for $t = 1$ **to** T **do**
    Sample action from task policy: $a_t \leftarrow \pi(s_t)$
    Execute action a_t and observe new state s_{t+1}, and reward r_t.
    Store transition (s_t, a_t, r_t, s_{t+1}) in memory $\mathcal{M}$
  end for
end for

Algorithm 2 Learner

Given: an off-policy RL algorithm $\mathbb{A}$, a replay buffer $\mathcal{M}$, a replay buffer of expert demonstrations $\mathcal{M}_e$
Initialize $\mathbb{A}$
for $n_{updates}$ **do**
  Sample transitions (s_t, a_t, r_t, s_{t+1}) from $\mathcal{M}$ to make a minibatch B.
  Sample transitions (s_t, a_t, r_t, s_{t+1}) from $\mathcal{M}_e$ enlarge the minibatch B.
  Perform a actor update step with Eqn. (4).
  Perform a critic update step with Eqn. (5).
  Update the target actor/critic networks every k steps.
end for

H Network architecture and hyperparameters

Actor and critic share a residual pixel encoder network with eight convolutional layers (3x3 convolutions, three 2-layer blocks with 16, 32, 32 channels), instance normalization [8] and exponential linear units [9] between layers.

The policy is a 3-layer MLP with ReLU activations with hidden layer sizes (300, 200). The critic is a 3-layer MLP with ReLU activations with hidden layer sizes (400, 300). For a illustration of the network. Please see Figure 19. The discriminator network uses a pixel encoder of the same architecture as the actor critic, followed by a 3-layer MLP with ReLU activations and hidden layer sizes (32, 32). Table 3 shows all network hyper parameters.

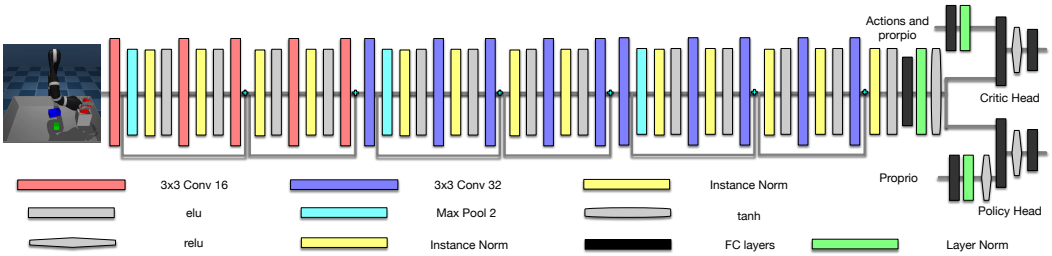


Figure 19: Network architecture for the policy and critic.

Table 3: Hyper parameters used in all experiments.

Parameters	**Values**
Actor/Critic Input Width and Height	64×64
Actor-Critic Parameters	
V_{min}	-50
V_{max}	150
V_{bins}	21
N step	1
Learning rate	10^{-4}
Optimizer	Adam [10]
Batch size	256
Target update period	100
Discount factor (γ)	0.99
Replay capacity	10^6
Number of actors	32 or 128
Imitation Parameters	
Discriminator learning rate	10^{-4}
Discriminator Input Width	48
Discriminator Input Height	48

References

[1] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *IROS*, 2012.

[2] G. Berseth and C. J. Pal. Visual imitation learning with recurrent siamese networks. *Preprint arXiv:1901.07186*, 2019.

[3] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

[4] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, A. Muldal, N. Heess, and T. Lillicrap. Distributed distributional deterministic policy gradients. *Preprint arXiv:1804.08617*, 2018.

[5] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

[6] D. Horgan, J. Quan, D. Budden, G. Barth-Maron, M. Hessel, H. Van Hasselt, and D. Silver. Distributed prioritized experience replay. *Preprint arXiv:1803.00933*, 2018.

[7] M. G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. *Preprint arXiv:1707.06887*, 2017.

[8] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *Preprint arXiv:1607.08022*, 2016.

[9] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *Preprint arXiv:1511.07289*, 2015.

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *Preprint arXiv:1412.6980*, 2014.