On The Fragility of Learned Reward Functions

Anonymous Author(s)
Affiliation
Address
email

Abstract

Reward functions are notoriously difficult to specify, especially for tasks with complex goals. Reward learning approaches attempt to infer reward functions from human feedback and preferences. Prior works on reward learning mainly focus on achieving high final performance for agents trained alongside the reward function. However, many of these works fail to investigate whether the resulting learned reward model accurately captures the intended behavior. In this work, we focus on the *relearning* failures of learned reward models. We demonstrate that when they are reused to train randomly initialized policies by designing experiments on both tabular and continuous control environments. We found that the severity of relearning failure might be sensitive to changes in reward model design and the trajectory dataset. Finally, we discussed the potential limitations of our methods and emphases the need for more retraining-based evaluations in the literature.

1 Introduction

Reward functions for most real-world tasks are difficult or impossible to specify procedurally. Indeed, hand-designed reward functions frequently misspecify the task [17]. The field of reward learning attempts to overcome this challenge by designing algorithms to infer reward functions from data. These learned reward functions can be a succinct representation of desired behaviors [21], drastically reduce the amount of human feedback required to learn a task [8] and possibly even allow us to generalize behaviors to new environments [10].

One of the most promising approaches is to learn reward functions from binary human preferences over trajectory segments [8, 34]. This form of preference based reward learning is already being used to train large language models to summarize [30, 39], follow instructions [22], search the internet [20], and generally become more helpful and harmless [3].

In order to start gathering human preferences, any reward learning algorithm must first collect trajectories to elicit feedback on. These trajectory segments can be gleaned from offline data-sets collected from exploration policies or human demonstrators [14]. Alternatively, they can be sourced *online* from a policy trained with reinforcement learning to maximize the learned reward estimate [8]; We call this reinforcement learning agent the *sampler agent*. We focus on the online case in this paper as online data can enable higher performance by ensuring the agent performing the desired task has adequate coverage in the dataset [39].

Prior work has typically focused on the performance of the *sampler agent* trained jointly with the reward function. Unfortunately, the sampler agent performing the correct behavior does not guarantee that a proper reward function has been uncovered. In particular, when using RL to train a randomly initialized *relearner* policy on the learned reward, the reward may fail to induce the correct behavior despite the sampler agent behaving well [14]. If we only require a policy that works passably well in the exact training environment, this may not be an issue – we can use the sampler agent and throw away the learned reward. However, we argue that such a method cannot be accurately described

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

as learning a *reward* function – at most, it is a preference-based policy learning technique, using reward functions to give a helpful inductive bias during training. Moreover, it is desirable for many applications to truly uncover a reward function. For example, we might wish to train a new policy using a learned reward function with a more powerful RL algorithm or different agent architecture than used during reward learning.

Our paper empirically examines the relearner performance for learned reward functions in different settings. Since we have access to a ground truth reward in our synthetic experiments, we define poor relearner performance as achieving relatively low ground-truth returns. Our results show that relearners often learn very different policies than the generator, frequently achieving low ground-truth returns. Thus, we argue that current preference-based reward learning methods may produce reward functions that are not reliable as signals for policy relearning.

Our paper makes three key contributions:

1. We demonstrate that state-of-the-art reward learning algorithms can produce reward models that fail to train new agents from scratch in tabular and continuous control settings;
2. We demonstrate that the severity seems to increase as the trajectory dataset concentrates on high reward regions;
3. Finally, we show how these relearning failures can be sensitive to changes in reward model design; as an example, we demonstrate this effect for reward ensembles.

2 Related Work

Preference-based reward learning Our paper's primary focus is on methods that learn from *preference comparisons* between two trajectories [2, 33, 27, 8]. Preference comparison is one of the most scalable reward learning methods, successfully applied to fine-tune large transformer language models [22, 30, 20, 3] to enhance their performance at certain tasks. Note that trajectory comparison methods contain more information about the reward than demonstrations, so they tend to produce better results when available [28]. However, note that these methods may still fare poorly when the human preference feedback does not match their model of human rationality [19].

Other reward learning approaches Many other methods have been developed to learn reward functions from human data [15]. One of the most popular is *Inverse reinforcement learning* (IRL) [21] methods that infer a reward function from demonstrations [1, 25, 37, 36, 38, 9, 10]. T-REX [6] is a hybrid approach, learning from a *ranked* set of demonstrations. An alternative approach learns from "sketches" of cumulative reward over an episodeCabi et al. [7].

Reward hacking Pan et al. provides the first systematic empirical study of *reward hacking*: RL agents exploiting misspecified reward functions [23]. Notably, they find that increasing agent capabilities, such as by increasing the RL policy's model size, can sometimes lead to *worse* performance on the ground truth reward, while performance on the misspecified *proxy* reward increases. In contrast to our work, Pan et al. only study reward hacking in hand-designed rewards designed to illustrate the phenomenon, whereas we investigate this phenomenon in learned rewards.

Reward hacking has also been studied from a theoretical perspective by Skalse et al., who propose a formal definition of reward hacking [29]. Concretely, they define a proxy reward as being *hackable* relative to a true reward if there exists a policy π' favored over policy π in the proxy reward, but where π' is actively worse than π under the true reward. Note that this definition is extremely demanding—indeed, if the policy is unconstrained, then a proxy reward is unhackable only if it induces an identical policy ordering to the ground truth. Although this is an exciting ideal to strive towards, in our paper, we focus on more practically attainable measures, such as relearner performance being close to sampler performance.

The most closely related work is by [14], which evaluates their learned reward functions by freezing them and training a new policy, analogous to our *relearner* evaluation [14, Section 3.2]. However, this was only a small, half-page section of their paper, and they did not examine factors that may increase or decrease the chances/severity of relearning failures.

Another important related work is by Reddy et al., who observes that rewards can fail to generalize due to a lack of informative trajectories in their training data [26]. They attempt to ameliorate this by

querying humans on diverse hypothetical trajectories generated from a model. However, their method requires a world model and primarily focuses on taking advantage of this model to improve reward quality. They do not examine the effects of reward model architecture. In addition, they assume the user provides feedback through quantitative reward labels, whereas we focus on the more realistic and widely used preference comparison setting.

Finally, past work has found that language models trained on a preference-based reward model can learn to exploit their reward model [30, Section 4.3]. In a similar vein, Bai et al. found that the ability of their reward model to correctly model responses degraded when it had high learned reward [3, Section 4.2]. As expected, in both works reward models did appear to generalize better as they increased in size. We hope that future papers will continue this trend and include detailed analysis of how effective their reward models are at supervising new agents.

Robustness of learned rewards from demonstrations There has also been work to improve the robustness of learned reward models in the learning from demonstrations domain. Xu and Denil [35] propose addressing the reward generalization problem via *positive-unlabeled* learning [16], with human feedback (e.g. demonstrations) being "positive" data and rollouts from the agent being unlabeled data. They demonstrate this improves robustness in a semi-supervised RL and imitation learning setting. Meanwhile, Fu et al. [10] propose an IRL method to learn state-only reward functions disentangled from transition dynamics via a state-only reward. By contrast, our work focuses on the more scalable preference-based reward learning setting.

3 Background

3.1 Deep RL from Human Preferences

We follow the framework of learning a preference model $\hat{r}_\phi$ from trajectory comparisons. Our method is the closest to deep reinforcement learning from human preferences [8]. It consists of three phases performed repeatedly.

3.1.1 Preference Elicitation

The algorithm samples trajectory segments σ at random as in [14], which consist of a sequence of states and actions $(s_0, a_0, s_1, a_1, \cdots, s_n)$ without any reward labels. Then it elicits preferences $y \in \{\succ, \prec, \equiv\}$ for each pair of segments from a human. The preferences are stored into a preference dataset $\mathcal{D}$ as a triple (σ^0, σ^1, y). The algorithm assumes that these preferences are sampled from the Bradly-Terry model [4],

$$P(\sigma^0 \succ \sigma^1) = \frac{\exp\left(\sum_{s,a,s' \in \tau_1} r(s,a,s')\right)}{\exp\left(\sum_{s,a,s' \in \tau_1} r(s,a,s')\right) + \exp\left(\sum_{s,a,s' \in \tau_2} r(s,a,s')\right)}. \tag{1}$$

Although an imperfect fit for actual human data, the Bradley-Terry model is a widely used approximation in preference learning [8, 14, 18].

3.1.2 Reward Model Fitting

Using Eq. 1 as a likelihood function for data $\mathcal{D}$ one can perform maximum likelihood estimation to infer r. Eq. 1 is simply a logistic regression model where the rewards summed over the trajectory segments are the logits. We can then treat the human preference feedback y as labels to a binary classification problem. If we model the learned reward function as a neural network with parameters ϕ, we can update it by minimizing the cross-entropy loss:

$$\mathcal{L} = -\mathbb{E}\big[y(0) \log P_\phi[\sigma^0 \succ \sigma^1] + y(1) \log P_\phi[\sigma^1 \succ \sigma^0]\big]$$

3.1.3 Policy optimization

After fitting the preference model, we can apply several off-the-shelf reinforcement learning algorithms to maximize the expected return given by the learned reward.

4 Training and Evaluation Procedure

Reward Learning We train reward models with **synthetic data** that is sampled from the Bradley-Terry model of Eq. 1 with r set to the ground-truth reward. We train the sample policy using soft Q-Learning [12]. We use soft actor-critic (SAC) [13] implementations of Stable-Baselines3 [24] in the locomotion control tasks. Both algorithms are off-policy and use a replay buffer, which ensures their high sample efficiency compared to on-policy RL algorithms. Note that the learned reward function $\hat{r}_\phi$ changes during training, so we relabel the transitions in the replay buffer after each iteration, similar to PEBBLE [18]. The main difference between our algorithm and PEBBLE is that we omit the unsupervised pre-training stage used in PEBBLE. We used the implementations from Imitation Learning Baseline Implementations [32] to generate the experiments.

We compute a normalized version of the learned reward using an Exponential Moving Average to normalize the reward to mean zero and unit standard deviation. We used this normalized reward for policy optimization. Note that normalizing the reward does not change the optimal policy, which is invariant to positive affine transformations. However, it does simplify the optimization problem. In particular, a normalized reward is a more stable objective for the critic to learn over time. Additionally, RL hyperparameters can depend on the reward scale (for example, learning rate should be set inversely proportional to reward scale) – normalizing the learned reward therefore allows us to use a consistent set of hyperparameters.

In the tabular setting, all reward networks only take the current state as a one-hot vector as input to improve interoperability. They consist of a multi-layer perceptron with two hidden layers of size 256 and ReLU activations, similar to those used in PEBBLE [18].

Reward Ensembles Christiano et al. and Lee et al. use reward ensembles to maintain an uncertainty estimate over the learned reward [8, 18]. In this paper, we explore how they may have an additional benefit by reducing the variance of off-distribution transitions. As in these prior works, we train each ensemble member on bootstrapped data sets, normalize them separately, and use their mean as the reward.

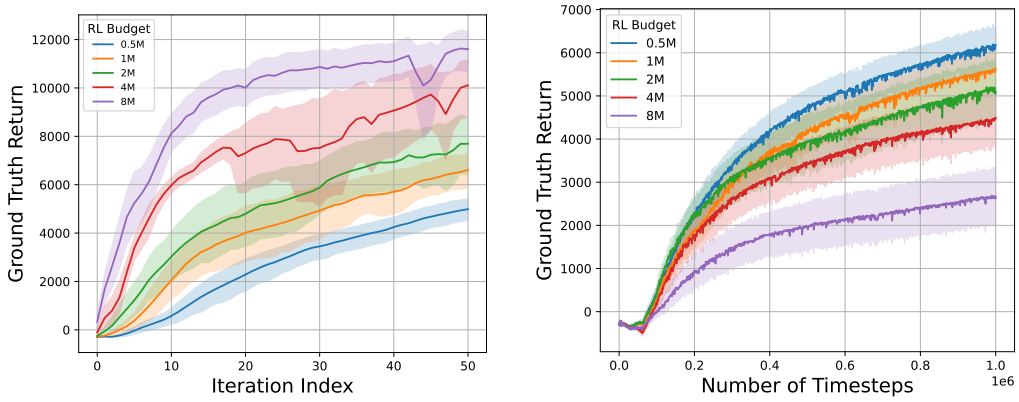
Relearning We freeze the learned reward and train a new, randomly initialized, *relearner* policy to evaluate our reward functions. Finally, we evaluate this policy under the ground truth reward. This is similar to the method employed by Ibarz et al. [14, Section 3.2] to study reward hacking. The continuous control settings consist of training a new agent from the current reward function using the same RL algorithm as the sampler, then evaluating it under the ground truth reward. In the tabular setting, we take this a step further by solving for the soft-optimal policy [12] under the learned reward function with temperature 0.1 and discount factor 0.99. Again we evaluate the relearners' performance under the ground truth reward.

5 Experiments

First, we investigate the occurrence of relearning failures in the continuous control domain. Here we find that increasing the number of training timesteps the sampler takes between sampling trajectories for labeling increases the severity of relearning failures. Next, we focus on the effects of reward model design and observe that reward ensembles may reduce reward hacking in tabular environments by reducing the variance of off-distribution transitions.

5.1 Environment Setups

Locomotion Control Task We ran reward learning and relearning on a MuJoCo locomotion task [31] – the `HalfCheetah` environment from the *seals* benchmark suite [11], a modification of `gym/HalfCheetah-v3` which adds the `x`-coordinate of the robot's center of mass (COM) to the first dimension of the observation space. The ground-truth reward function of the `HalfCheetah` environment is the linear combination of the `x`-velocity of the robot's COM and a control cost dependent on the L_2 norm of the action vector. Consequently, the reward function in *seals* `HalfCheetah` is a function of the observations, which is not strictly true in the original *gym* [5] environment, avoiding a potential confounder.



(a) *Reward learning* learning curve. (b) *Relearning* learning curve.

Figure 1: Anti-correlated sampeler and relearner learning curves in `HalfCheetah` (a) x-axis represents the number of iterations of each run. Each iteration consists of three phases of preference elicitation, reward model fitting and policy optimization. RL budget represents the total number of RL timesteps completed in policy optimization. (b) x-axis represents the number of timesteps during *relearning*. In both plots, solid lines and shaded lines represent the mean and 90% confidence interval of multiple runs under each setting.

Tabular Environment We constructed the **stay inside environment**, which consists of a 20x20 closed grid of cells. The top "outside" and bottom "inside" halves of the environment are separated by a wall with a narrow two cell gap in the middle. The reward for each state is shown in Figure 3 (a), with reward values ranging from +10 to -1.

5.2 Preference Trajectory Dataset Imbalance and Relearning Failure

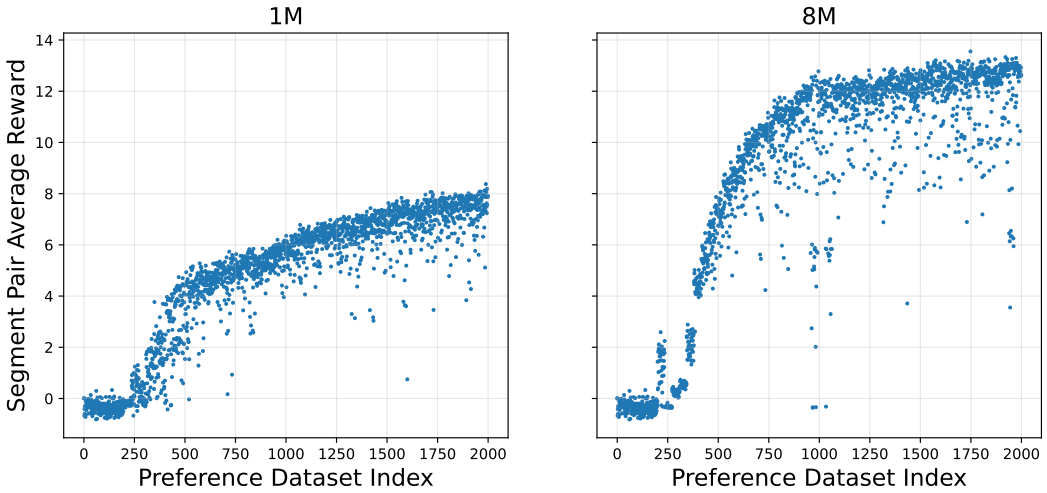
The reward model is effectively a function of the preference dataset over collected trajectories. Thus, it is a good place to start looking for factors that affect the severity of relearning failures. One of the most important ways to change the preference dataset is to vary the number of timesteps T spent training the sampler between collecting samples. Here we call the total number of interactions between the sampler and the environment the *RL budget*. Note this does not mean changing the number of comparisons we collect but simply means we spend more time training the sampler between collecting them.

Figure 1 shows the learning curves of the sampler and relearner experiments in `HalfCheetah`. For each RL budget setting, we perform ten runs of reward learning, and for each of these, we run five relearning evaluations for a total of 50 relearning runs. On these runs, we report the mean and 90% bootstrapped confidence interval(CI) for it. We find that despite higher RL budget leading to higher sampler returns during reward learning, the relearners' performance has the reverse trend, with increasing sampler time steps and decreasing relearner ground truth return.

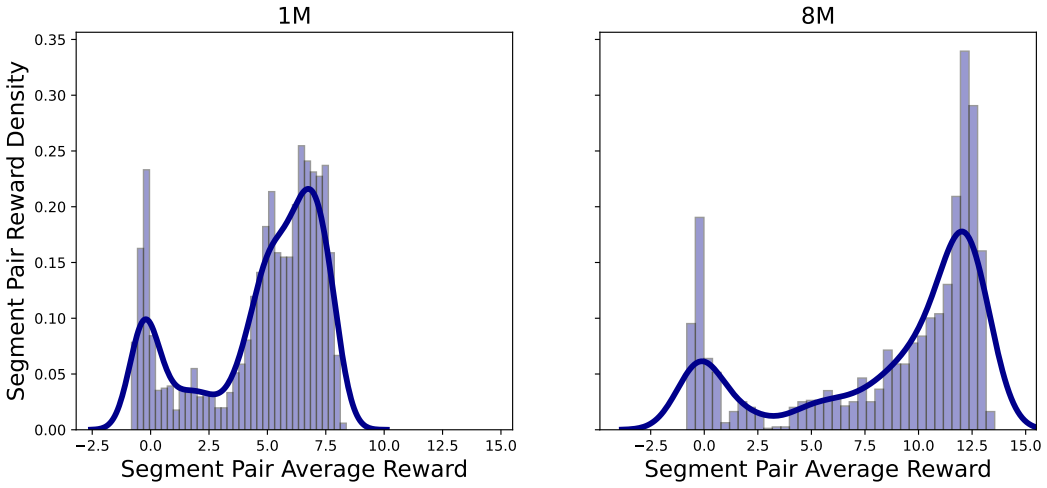
We can gain more insight into what is going on by exploring preference datasets shown in Figure 2. First let's consider the preference datset produced by one of the run with the highest-budget (8M timesteps). We find that the trajectory segments contained in this datset are very concentrated in high ground truth reward regions. On the other hand, when we consider the low-budget dataset (1M timesteps). The trajectory segments appear to provide a better coverage across all the ground truth reward scales within the support.

This concentration forms because a higher RL budget effectively speeds up RL training. This speed up reduces the number of trajectory segments that the sampler collects during the critical window when the sampler is still exploring the environment. Thus the vast majority of trajectories will be collected after the sampler is already preforming well, concentrating the sampled trajectories in this region.

We hypothesize that having an overwhelming proportion of high-reward trajectory segments in the preference dataset — and little preference data on trajectories in the transition from high to low reward



(a)



(b)

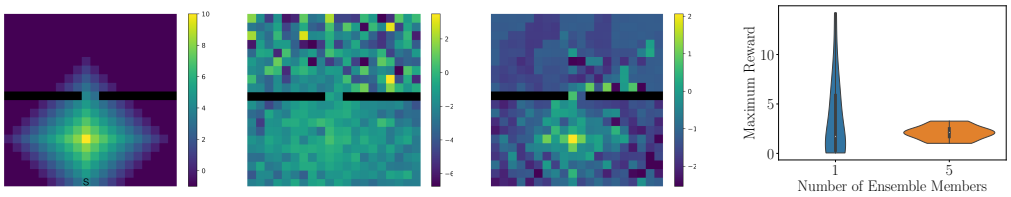
Figure 2: (a) Scatterplots of average reward of each segment pairs in example preference datasets with 1M and 8M RL budgets. (b) Density of segment pair average ground truth rewards in 2(a). 1M RL budget is roughly equivalent to 20K RL timesteps during policy optimization in each iteration. 8M RL budget is equivalent to 160K timesteps between consecutive iterations.

— may cause the reward model to effectively over-fit to the high-reward region. This overfitting leads to poor supervision over randomly initialized policies. Overall, we believe this could be what is produce the observed relearning failures.

It's important to note, that we did not see a significant increase in relearning failure when increasing the RL budget in the tabular setting, see Appendix **??**. This may be because in a tabular setting the sampler either: finds the optimal policy induced by the learned reward function every iteration, so the sampler and relearner have equal performance, or insufficiently explores the environment and reward learning completely fails. This dichotomy leaves little room for the subtle degradation in relearner performance we see in Figure 1.

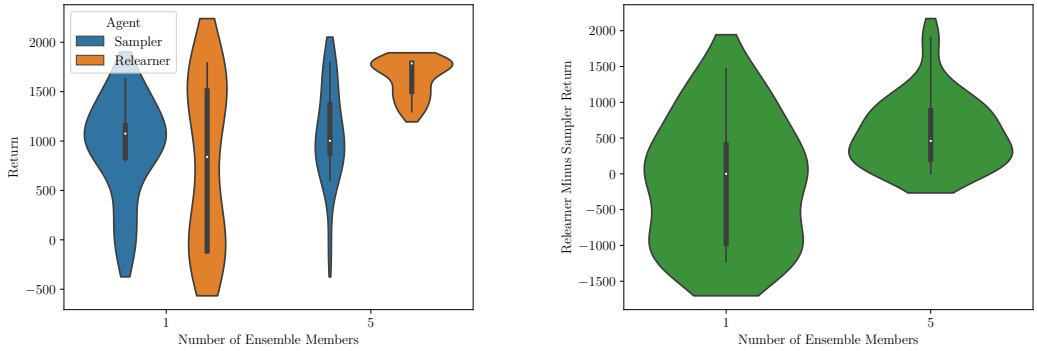
5.3 Reward Ensembles

Our tabular experiments provide a concrete, interpretable example of how relearning failures can emerge. Here we observe that certain reward model implementation details may have large effects on



(a) (b) (c) (d)

Figure 3: Comparing learned rewards with different ensemble sizes
(a) depicts the ground-truth reward of of the stay inside environment. Note that it is state only. (b) shows an example of a learned reward function found when using a single reward network. (c) is an example of a reward produced when using an ensemble of five reward nets. Finally, (d) shows the distribution of maximum reward across all transitions when using an ensemble of size 1 and 5.



(a) (b)

Figure 4: High variance reward functions lead to relearning failures in the stay inside environment
The difference in returns between the sampler agent and the relearner agent when evaluated on the ground truth reward function. These figures where generated with 20 seeds.

relearners but leave the sampler's performance unchanged. In particular, we focus on the effects of the size of the reward ensemble used in training.

When using a reward ensemble of size five all relearners preform at least as well as their respective samplers, as can be seen in Figure 4. However, if we use only a single reward, the relearners behaviour is no longer so consistent; some relearners do substantially better then their respective *samplers*, but almost as many do substantially worse, getting near zero return. Thus, while adding an ensemble has a minimal effect on the sampler it seems to radically change the behaviour of our relearners.

To understand why this happens we must consider the off-distribution behaviour of our reward models. In the stay inside environment, the samplers typically stay in the inside half of the environment. Thus there is often insufficient coverage of the outside half of the environment in our trajectory dataset. Thus, the reward off-distribution is completely unconstrained by the data. This means that small changes in the off-distribution behavior of our reward network can become critically important. Reward models based on neural networks can produce high rewards off distribution, see Figure 3. When these *reward delusion* are more rewarding than any of the in-distribution transitions, reward hacking can occur and cause relearning failures. Reward ensembles tends to have lower variance off distribution than an individual reward network. Thus, any reward delusions tend to have a lower reward (according to the reward model). This can be directly seen in Figure 3 (a-d). This effect reduces the chance that the optimal policy will be attracted to one of these spurious rewards during relearning, which is what we see in Figure 4.

6 Discussion and Limitations

First, our continuous control experiments show that relearning failures can worsen as the sampler is given more optimization power. We hypothesize that an imbalanced trajectory comparison dataset is responsible for these failures. Thus, when learning reward functions online using a sampling policy, it may be possible to over-concentrate the trajectory dataset and thus create reward models that are unsuitable for supervising new policies. We encourage future works to consider this failure carefully when designing trajectory sampling schemes.

In our reward ensemble experiments, we demonstrate that reward ensembles may reduce the variance of off-distribution transitions. Indeed, anything that smooths the reward model beyond the region covered by the sampled comparison dataset may assist in addressing relearning failures, caused by reward hacking.

Our experiments have a few important limitations. First, our experiments may be limited to simple ground truth reward functions and environments. For example, in `Half-Cheetah-v3` [5, 11], the reward function is essentially a linear function of the observation, action and next observation. While these relearning failures also appear in more complex tasks [14, Section 3.2], it is unclear if it is precisely the same phenomenon that causes them. Further, the design decisions that seem to improve retraining performance in small-scale experiments, in our case, reward ensembles and less sampler training, may not be the same as those that address the problem at a larger scale. We leave such explorations to future work.

We have demonstrated that evaluations of relearning performance can differ substantially from the results of simply evaluating the sampling policy trained alongside the reward model. Thus, we hope to see future works include relearning evaluation as they appear to hold fruitful insights into the quality of the learned reward functions.

7 Author Contributions

To be completed after double-blinded review.

8 Acknowledgement

To be completed after double-blinded review.

References

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

[2] R. Akrour, M. Schoenauer, and M. Sebag. Preference-based policy learning. In D. Gunopulos, T. Hofmann, D. Malerba, and M. Vazirgiannis, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proceedings, Part I*, volume 6911 of *Lecture Notes in Computer Science*, pages 12–27. Springer, 2011. doi: 10.1007/978-3-642-23780-5_11. URL `https://doi.org/10.1007/978-3-642-23780-5_11`.

[3] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, N. Joseph, S. Kadavath, J. Kernion, T. Conerly, S. El-Showk, N. Elhage, Z. Hatfield-Dodds, D. Hernandez, T. Hume, S. Johnston, S. Kravec, L. Lovitt, N. Nanda, C. Olsson, D. Amodei, T. Brown, J. Clark, S. McCandlish, C. Olah, B. Mann, and J. Kaplan. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback, Apr. 2022.

[4] R. A. Bradley and M. E. Terry. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, 39(3/4):324–345, 1952. ISSN 0006-3444. doi: 10.2307/2334029.

[5] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

[6] D. S. Brown, W. Goo, P. Nagarajan, and S. Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 783–792. PMLR, 2019. URL `http://proceedings.mlr.press/v97/brown19a.html`.

[7] S. Cabi, S. Gómez Colmenarejo, A. Novikov, K. Konyushova, S. Reed, R. Jeong, K. Zolna, Y. Aytar, D. Budden, M. Vecerik, O. Sushkov, D. Barker, J. Scholz, M. Denil, N. de Freitas, and Z. Wang. Scaling data-driven robotics with reward sketching and batch reinforcement learning. In *Robotics: Science and Systems XVI*. Robotics: Science and Systems Foundation, July 2020. ISBN 978-0-9923747-6-1. doi: 10.15607/RSS.2020.XVI.076.

[8] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4299–4307, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/d5e2c0adad503c91f91df240d0cd4e49-Abstract.html`.

[9] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In M. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 49–58. JMLR.org, 2016. URL `http://proceedings.mlr.press/v48/finn16.html`.

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

[11] A. Gleave, P. Freire, S. Wang, and S. Toyer. seals: Suite of environments for algorithms that learn specifications. `https://github.com/HumanCompatibleAI/seals`, 2020.

[12] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine. Reinforcement learning with deep energy-based policies. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1352–1361. PMLR, 2017. URL `http://proceedings.mlr.press/v70/haarnoja17a.html`.

[13] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine. Soft actor-critic algorithms and applications. Technical report, 2018.

[14] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei. Reward learning from human preferences and demonstrations in Atari. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[15] H. J. Jeon, S. Milli, and A. D. Dragan. Reward-rational (implicit) choice: A unifying formalism for reward learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/2f10c1578a0706e06b6d7db6f0b4a6af-Abstract.html`.

[16] R. Kiryo, G. Niu, M. C. du Plessis, and M. Sugiyama. Positive-unlabeled learning with non-negative risk estimator. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1675–1685, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/7cce53cf90577442771720a370c3c723-Abstract.html`.

[17] V. Krakovna, J. Uesato, V. Mikulik, M. Rahtz, T. Everitt, R. Kumar, Z. Kenton, J. Leike, and S. Legg. Specification gaming: the flip side of ai ingenuity, Apr 2020. URL `https://deepmindsafetyresearch.medium.com/specification-gaming-the-flip-side-of-ai-ingenuity-c85bdb0deeb4`.

[18] K. Lee, L. M. Smith, and P. Abbeel. PEBBLE: feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 6152–6163. PMLR, 2021. URL `http://proceedings.mlr.press/v139/lee21i.html`.

[19] K. Lee, L. M. Smith, A. D. Dragan, and P. Abbeel. B-pref: Benchmarking preference-based reinforcement learning. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL `https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/d82c8d1619ad8176d665453cfb2e55f0-Abstract-round1.html`.

[20] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders, X. Jiang, K. Cobbe, T. Eloundou, G. Krueger, K. Button, M. Knight, B. Chess, and J. Schulman. WebGPT: Browser-assisted question-answering with human feedback, June 2022.

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

[22] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe. Training language models to follow instructions with human feedback, Mar. 2022.

[23] A. Pan, K. Bhatia, and J. Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=JYtwGwIL7ye.

[24] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22 (268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html.

[25] D. Ramachandran and E. Amir. Bayesian inverse reinforcement learning. In M. M. Veloso, editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 2586–2591, 2007. URL http://ijcai.org/Proceedings/07/Papers/416.pdf.

[26] S. Reddy, A. D. Dragan, S. Levine, S. Legg, and J. Leike. Learning human objectives by evaluating hypothetical behavior, 2020. URL http://proceedings.mlr.press/v119/reddy20a.html.

[27] D. Sadigh, A. D. Dragan, S. Sastry, and S. A. Seshia. Active preference-based learning of reward functions. In N. M. Amato, S. S. Srinivasa, N. Ayanian, and S. Kuindersma, editors, *Robotics: Science and Systems XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16, 2017*, 2017. doi: 10.15607/RSS.2017.XIII.053. URL http://www.roboticsproceedings.org/rss13/p53.html.

[28] J. Skalse, M. Farrugia-Roberts, S. Russell, A. Abate, and A. Gleave. Invariance in policy optimisation and partial identifiability in reward learning, 2022.

[29] J. Skalse, N. H. R. Howe, D. Krasheninnikov, and D. Krueger. Defining and characterizing reward hacking. In *NeurIPS*, 2022.

[30] N. Stiennon, L. Ouyang, J. Wu, D. M. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. Christiano. Learning to summarize from human feedback, Feb. 2022.

[31] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, pages 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109. URL https://doi.org/10.1109/IROS.2012.6386109.

[32] S. Wang, S. Toyer, A. Gleave, and S. Emmons. The imitation library for imitation learning and inverse reinforcement learning. https://github.com/HumanCompatibleAI/imitation, 2020.

[33] A. Wilson, A. Fern, and P. Tadepalli. A Bayesian approach for policy learning from trajectory preference queries. In *NIPS*, 2012.

[34] C. Wirth, R. Akrour, G. Neumann, and J. Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017. URL http://jmlr.org/papers/v18/16-634.html.

[35] D. Xu and M. Denil. Positive-unlabeled reward learning. In J. Kober, F. Ramos, and C. J. Tomlin, editors, *4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA*, volume 155 of *Proceedings of Machine Learning Research*, pages 205–219. PMLR, 2020. URL https://proceedings.mlr.press/v155/xu21c.html.

[36] B. D. Ziebart. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. PhD thesis, CMU, 2010.

[37] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In D. Fox and C. P. Gomes, editors, *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*, pages 1433–1438. AAAI Press, 2008. URL http://www.aaai.org/Library/AAAI/2008/aaai08-227.php.

[38] B. D. Ziebart, J. A. Bagnell, and A. K. Dey. Modeling interaction via the principle of maximum causal entropy. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 1255–1262. Omnipress, 2010. URL https://icml.cc/Conferences/2010/papers/28.pdf.

[39] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. F. Christiano, and G. Irving. Fine-tuning language models from human preferences. *CoRR*, abs/1909.08593, 2019. URL http://arxiv.org/abs/1909.08593.