HOW USEFUL IS QUANTILIZATION FOR MITIGATING SPECIFICATION-GAMING?

Ryan Carey
Oxford University
ryan.carey@philosophy.ox.ac.uk

ABSTRACT

For some tasks, there exists a goal that perfectly describes what the designer wants the AI system to achieve. For many tasks, however, the best available proxy objective is only a rough approximation of the designer's intentions. When given such a goal, a system that optimizes the proxy objective tends to select degenerate solutions where the proxy reward is very different from the designer's true reward function. One way to counteract the tendency toward specification-gaming is quantilization, a method that interpolates between imitating demonstrations, and optimizing the proxy objective. If the demonstrations are of adequate quality, and the proxy reward overestimates performance, then quantilization has better guaranteed performance than other strategies. However, if the proxy reward underestimates performance, then either imitation or optimization will offer the best guarantee. This work introduces three new gym environments: Mountain Car-RR, Hopper-RR, and Video Pinball-RR, and shows that quantilization outperforms baselines on these tasks.

1 INTRODUCTION

Machine learning algorithms have often been most successful when there exists a known reward function that coincides with the designer's preferences. For many practical problems, it is hard for the designer to write down a function that includes all aspects of what they want the machine learning system to achieve, and instead much of their intent is left implicit. In such situations, maximizing the best available proxy reward function can lead to unusual behaviour:

- A deep reinforcement learner, trained to play the game Coastrunners, maximized its score by traveling in a small, fixed circle, in order to hit a single boost, rather than completing the race (Amodei & Clark, 2016).
- Agents evolved to walk, managed to maximize their velocity by growing large legs, and then falling over (Ha, 2018; Sims, 1994; Lehman et al., 2018).
- Systems trained to swim or to locomote in simulated physical environments performed these tasks by obtaining free energy, by exploiting errors in the mechanics of collisions, and imprecision in the methods used for numerical integration (Sims, 1994; Lehman et al., 2018).

Sometimes, a designer values this kind of creativity, and can use it to patch errors in the reward function, or the physics model in which learning is occurring. In other cases, however, it is simply unhelpful. Where an agent scores highly on the explicit reward but poorly with respect to the designer's desires, we say that the agent is gaming the designer's specifications.

One way to avoid specification-gaming is to transmit our implicit desires to the agent by providing demonstrations of the task. One algorithm designed to mitigate specification-gaming in this setting is quantilization (Taylor, 2016). In quantilization, the agent orders the demonstrations according to their attainment of the proxy reward, and then imitates just those demonstrations that fall in some top q-quantile. Quantilization can surpass the average performance of the demonstrations, yet it only diverges from the demonstrations to a limited degree, and so the extent to which it will game the specifications is bounded.

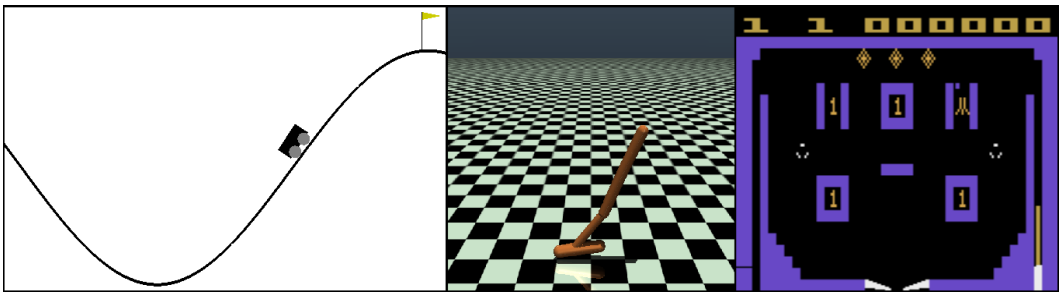

Figure 1: Three examples of specification-gaming: A SARSA agent trained to increase rightward displacement stays on the right hill rather than going to the finish-line; a PPO agent trained on average ankle angle, leans forward and falls over rather than walking; a Rainbow agent traps the ball between the upper-right bumpers, where it will bounce indefinitely, rather than hitting the ball around the table.

The aim of this piece is to investigate when quantilization avoids or does not avoid specification-gaming, and to understand the reasons underlying its success or failure. This includes a mixture of theoretical and empirical investigation.

The theoretical analysis in Section 2 begins by introducing a new framework for describing scenarios where the explicit reward differs from the performance function: the Robust Reward Problem. Then, the performance guarantees for quantilizers are compared with those obtained by optimization and imitation.

Section 3 presents three concrete Robust Reward Tasks, and describes how quantilization empirically performs in these tasks relative to baselines. Section 4 reviews related work, then 5 concludes.

2 Quantilization and its Theoretical Analysis

2.1 Setting and Goal

In order for an AI system to be robust to reward mis-specification, it must perform well not just for the explicit reward function that it has been given, but for some class of plausible performance measures that might correspond to the designer's intent (Hadfield-Menell et al., 2017).

As a framework for analyzing these kinds of problems, it is useful to suppose that the agent's performance U^* can be decomposed into an explicit reward U and an implicit (and unknown) reward I. In order to constrain the possible values of I, the agent is also given some demonstrations. Formally, we can define a robust reward problem as:

Definition 1 (Robust reward problem)**.** *A robust reward problem is described by a tuple* $P = \langle \mathcal{A}, U, \mathcal{I}, D \rangle$*:*

> $\mathcal{A}$*, the action space*
> $U : \mathcal{A} \to \mathbb{R}$*, the explicit reward function*
> $\mathcal{I} \subseteq \mathbb{R}^{\mathcal{A}}$*, the space of implicit reward functions*
> $D \in \Delta\mathcal{A}$*, the distribution over actions that comes from human demonstrations, where* $\Delta\cdot$ *denotes the probability simplex (i.e.* $\Delta A := \{a \in \mathbb{R}^{|\mathcal{A}|} : a_i \geq 0, \sum_i a_i = 1\}$*)*

In a robust reward problem, the agent chooses a strategy $S \in \Delta\mathcal{A}$*. Then, an action* a *is sampled from* S*, and the performance is the sum of explicit reward and implicit reward:* $U^*(a) := U(a) + I(a)$*.*

If a robust reward problem has an unlimited set of implicit rewards then no strategy can assure good performance. So this paper focuses solely on implicit reward functions that meet the following condition:

Definition 2 (k-Adequate demonstration)**.** *A distribution* $D \in \Delta\mathcal{A}$ *demonstrates an implicit reward function* I *k-adequately for* $k \in \mathbb{R}^+$*, iff:*

$$\mathbb{E}_{a\sim D}[I(a)] \geq -k$$

The logic behind k-adequacy is that a good teacher may make leave some information out of their utility function (or their explicit instructions). But any required behavior that is not encoded in the utility function must instead properly demonstrated.

This paper analyzes which strategies achieve the best guaranteed performance across all k-adequately demonstrated functions I in a robust reward problem, i.e.:

$$\max_{S \in \Delta A} \min_{I \in \mathcal{I}: \mathbb{E}_{a \sim D}[I(a)] \geq -k} \mathbb{E}_{a \sim S}[U(a) + I(a)] \tag{1}$$

2.2 Quantilization and performance guarantees

Definition 3 (q-quantilizer)**.** *Let A be the random variable defined by the distribution D and let $(a_1, \ldots, a_n)$ be a re-ordering[1] of the actions $\mathcal{A}$ such that $U(a_1) \leq \cdots \leq U(a_n)$.*

Let $q \in (0, 1]$ be the leniency. Let the threshold be $M_q = \min\{M \in \mathbb{R} : P(U(A) > M) \leq q\}$ if $q < 1$ and $U(a_1)$ if $q = 1$, and define i_q such that $U(a_{i_q}) = M_q$. Then, a Q_q quantilizer is the agent that uniformly samples actions from $Q_q : \mathcal{A} \to [0, 1]$ where:

$$Q_q(a_i) = \begin{cases} \frac{D(a_i)}{q} & \text{if } i > i_q \\ 1 - \sum_{j=i_q+1}^{n} \frac{D(a_j)}{q} & \text{if } i = i_q \\ 0 & \text{otherwise} \end{cases}$$

Intuitively, a 0.1-quantilizer selects a random action from the top 10% of the demonstrations, ranked according to the explicit utility function U. Quantilization has optimization and imitation as its extreme cases: a 1-quantilizer equals the demonstration distribution D, whereas in the limit of decreasing leniency ($q \to 0$), a quantilizer chooses the optimal action in the support of D. (Taylor, 2016).

A quantilizer only chooses actions at-most $\frac{1}{q}$ times more frequently than the base distribution.

Lemma 4 (Quantilizer property)**.** *A q-quantilizer selects any action a with probability upper-bounded by: $Q_q(a) \leq \frac{D(a)}{q}$.*

It can now be demonstrated how well a quantilizer performs for robust reward problems, assuming k-adequacy, for various different implicit reward spaces. Firstly, Taylor (2016) gives a guarantee for robust reward problems with $\mathcal{I} \subseteq (-\infty, 0]^{\mathcal{A}}$. This is extended slightly to cover Robust Reward Problems with $\mathcal{I} \subseteq (-\infty, c]$, as shown by a proof in Appendix A:

Theorem 5 (Quantilizer guarantee (Taylor, 2016))**.** *For any k-adequately demonstrated reward function I in a robust reward problem P with $\mathcal{I} \subseteq (-\infty, c]^{\mathcal{A}}$, $c \in \mathbb{R}^+$, each q-quantilizer strategy Q_q has performance lower-bounded by $\mathbb{E}_{a \sim Q_q}[U^*(a)] \geq \mathbb{E}_{a \sim Q_q}[U(a)] - \frac{k}{q} + \frac{q-1}{q}c$.*

Theorem 6 (Quantilizer optimality (Taylor, 2016))**.**

- *a For any strategy S in a robust reward problem P with $\mathcal{I} = (-\infty, c]^{\mathcal{A}}$, $c \in \mathbb{R}^+$, if S is a distribution over the support of the demonstrations D, then there exists a k-adequately demonstrated implicit reward $I \in \mathcal{I}$ that upper-bounds performance as: $\mathbb{E}_{a \sim S}[U^*(a)] \leq \mathbb{E}_{a \sim Q_q}[U(a)] - \frac{k}{q} + \frac{(q-1)}{q}c$ where $q := \min_{a \in supp(S)} \frac{D(a)}{S(a)}$.*
- *b For any strategy S for which some action a has $S(a) > 0$, $D(a) = 0$, some implicit reward k-adequately demonstrated by D gives performance unbounded below.*

Taken together, these theorems imply that if $\mathcal{I} = (-\infty, c]^{\mathcal{A}}$, then the strategy that guarantees best performance for all k-adequately demonstrated reward functions is the q-quantilizer.

Importantly, some algorithms that combine imitation and optimization may have performance unbounded below as in Theorem 6b. For example, as the number of actions approaches a continuum, all

[1] In cases where the quantile is not uniquely determined because different actions have the same utility, they may be ordered arbitrarily, as per Taylor (2016)

actions have $D(a) = 0$, so any pure strategy in a continuum setting has reward unbounded below, including all algorithms that choose one action to maximize a weighted sum of optimization and imitation objectives.

The next results will extend Taylor (2016) by considering whether quantilization is optimal in two further cases. First, the implicit reward may be bounded below by some $-c$. Second, it may be neither bounded above, nor below.

Theorem 7 (Optimizer/imitator guarantee)**.** *For any k-adequately demonstrated reward function I in a robust reward problem P with $\mathcal{I} \subseteq [-c, \infty)^{\mathcal{A}}$, $c \in \mathbb{R}^+$, the U-optimal action has performance lower-bounded by $U^*(a) \geq \max_{a \in \mathcal{A}} U(a) - c$ and the demonstration distribution D has performance lower-bounded by $\mathbb{E}_{a \sim D}[U^*(a)] \geq \mathbb{E}_{a \sim D}[U(a)] - k$.*

Theorem 8 (Optimizer/imitator optimality)**.** *For any strategy S in a robust reward problem P with $\mathcal{I} = (-c, \infty]^{\mathcal{A}}$, $c \in \mathbb{R}^+$, there exists a k-adequately demonstrated implicit reward $I \in \mathcal{I}$ that upper-bounds performance as: $\mathbb{E}_{a \sim S}[U^*(a)] \leq \mathbb{E}_{a \sim D}[U(a)] - k$ and $\mathbb{E}_{a \sim S}[U^*(a)] \leq \max_{a \in \mathcal{A}} U(a) - c$*

These results imply that if $\mathcal{I} = (-c, \infty]^{\mathcal{A}}$, the strategy that gets best performance will be the U-optimal action, or else will be the demonstration distribution D, and (and not q-quantilization with $q \in (0, 1)$).

Theorem 9 (Imitator guarantee)**.** *For any k-adequately demonstrated reward function I in a robust reward problem P with $\mathcal{I} \subseteq \mathbb{R}^{\mathcal{A}}$, the demonstration distribution D has performance lower-bounded by $\mathbb{E}_{a \sim D}[U(a)] - k$.*

Theorem 10 (Imitator optimality)**.** *For any strategy S that does not equal the demonstration distribution D in a robust reward problem P with $\mathcal{I} = \mathbb{R}^{\mathcal{A}}$, $c \in \mathbb{R}^+$, there exists a k-adequately demonstrated implicit reward $I \in \mathcal{I}$ that gives performance $\mathbb{E}_{a \sim S}[U^*(a)] = -\infty$*

Proofs are given in the appendix. Altogether, although quantilization gives the best guarantee in the upper-bounded implicit reward setting, this does not carry over to the unbounded implicit reward setting, where it lacks any performance guarantee, nor to the lower-bounded implicit reward setting, where its guarantee will be inferior to that of either imitation or optimization.

3 Experiments

3.1 Environments

Three robust-reward (RR) environments have been developed. In order of increasing complexity, they are: Mountain Car-RR, Hopper-RR, and Video Pinball-RR. Each of these operates the same as a corresponding environment from OpenAI Gym (Brockman et al., 2016), except that the reward is changed. These environments (and the rest of the project) are available on Github[2].

Mountain Car-RR is based on the usual Mountain Car gym environment, involving a car situated between two hills. The performance is -1 per timestep, which is the reward in the original gym environment. What the agent is given, however, as its explicit (proxy) reward, is its rightward displacement at each timestep.

In Hopper-RR, the performance is a $1 +$ *horizontal movement* per timestep, as in the original gym environment. That is, the performance for a whole episode is a sum of episode duration and horizontal displacement during that episode. The explicit reward is the average forward lean across the episode, which is given at the end of each episode.

Video Pinball-RR is introduced because it is an Atari game where it is known that reward can be achieved unconventionally, by trapping the ball between the upper-right bumpers, and nudging the table repeatedly to keep the ball there indefinitely. Although this is a clever solution, suppose that the designer wants the AI system to achieve a high-score without carrying out this behaviour. Then, let the explicit reward U be the in-game score, and let the implicit reward, I be $-\lambda$ each timestep if the ball is between the upper right bumpers, and 0 otherwise. Trajectories were taken from the Atari Grand Challenge dataset (Kurin et al., 2017). Further details on these experiments are available in Appendix B.

[2] www.github.com/mtrazzi/quantilizers

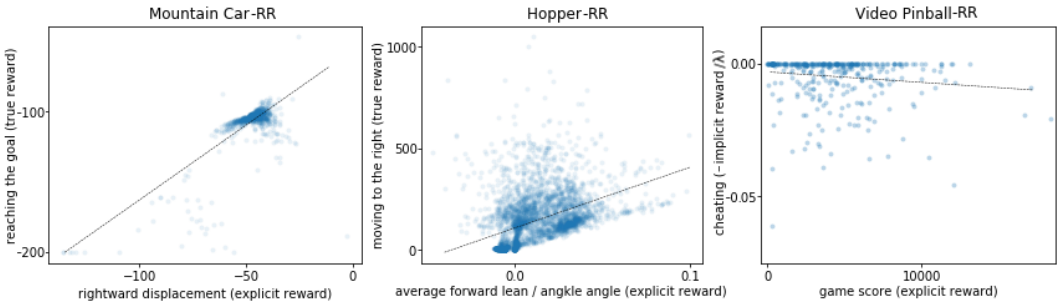

Figure 2: The demonstration distributions for each of the three robust reward tasks. For Mountain Car-RR and Hopper-RR, the implicit reward (difference between performance and explicit reward) is unbounded. Implicit reward in Video Pinball-RR is upper-bounded at 0.

Task	**Trajectories**	**Explicit reward** (U)	**True reward** (U^*)
Mountain Car-RR	901	Rightward displacement	In-built score
Hopper-RR	3835	Forward lean	In-built score
Video Pinball-RR	380	In-built score	Score - $\lambda\times$ proportion of time between bumpers

Table 1: Characteristics of the three datasets

To describe these multi-step problems as Robust Reward Problems, $\mathcal{A}$ must be understood as the set of agent policies. The implicit reward space may be set to $\mathcal{I} = \mathbb{R}^{\mathcal{A}}$, and the explicit and implicit reward functions U and I are the returns across a whole episode (these reward functions are summarized in 3.1).

3.2 Methods

For each task, quantilizers with $0.125 \leq q \leq 0.5$ are compared to an imitator (i.e. a 1-quantilizer), and to a reinforcement learning agent trained to maximize the proxy reward U.

For Mountain Car-RR and Hopper-RR, the imitator and quantilizers were trained with behavioral cloning, using a two-layer 20-node MLP. In Hopper-RR, since the action consisted of controlling three joints, three networks were used to prediction actions in $\{-1, 0, 1\}$, and for rollouts, actions were selected by taking the expected value of the predicted movement at each joint. The reinforcement learning agent for Mountain Car-RR was a SARSA agent trained with a linear function approximator with a radial basis function kernel. For Hopper, the reinforcement learning agent was proximal policy optimization (PPO) (Schulman et al., 2017). For MountainCar-RR, 901 trajectories were used, which were by the experimenter manually playing Mountain Car. For Hopper-RR, 3835 trajectories were used, 1462 of which were produced by the author, and 2273 of them were produced by a student volunteer, over a total time of around four hours.

For Video Pinball-RR, the imitator and quantilizers were trained with behavioral cloning using a deep convolutional architecture, whereas the reinforcement learning agent was trained with Rainbow (Hessel et al., 2018). In each of these, the convolutional architecture was the same, based on the default settings from Hessel et al. (2018). For Video Pinball-RR, 380 trajectories were taken from the second batch of the Atari Grand Challenge dataset (Kurin et al., 2017).

3.3 Results and Discussion

In experiments, there are (at least) two sources of additional error that may compromise the theoretical guarantees. Firstly, the quantilizers (like the imitation algorithm) only have access to trajectories that are sampled from the base distribution, but lack access to the base distribution itself. Second, the quantilizer only approximates the distribution Q_q, and so its performance may be worse than the guarantee, which applies to Q_q itself. So it is useful to see how these algorithms perform in practice, and whether this coincides with the cases where they have theoretical guarantees.

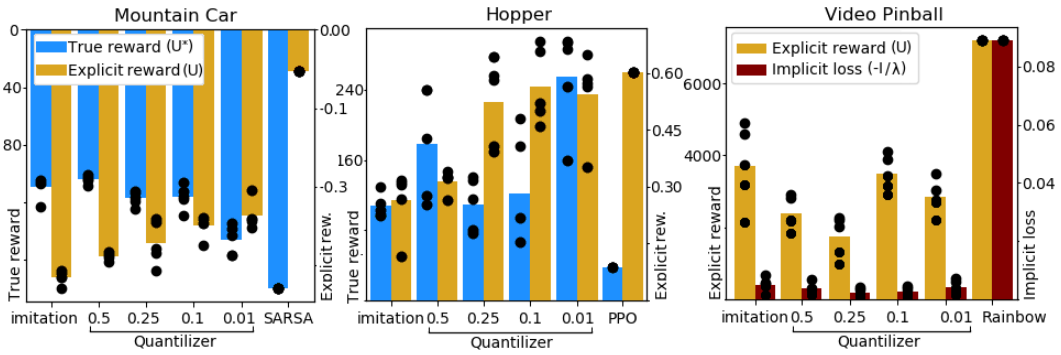

Figure 3: Performance of imitation, $0.5-$, $0.25-$, $0.1-$ and $0.01-$quantilizer (n=5 seeds), and reinforcement learning agents (respectively SARSA, PPO and Rainbow), on each of the three robust reward tasks. Higher score is better for true reward and explicit reward but worse for implicit loss. For Mountain Car and Hopper, the explicit reward increases when q decreases. For Video Pinball, no clear trend appeared.

Although the quantilizers did not have any guarantee of outperforming the optimizers, they did. In all three environments, the optimizer was able to obtain high explicit reward, but gamed the specifications, such that it obtained low U^*.

The quantilizers were less successful when compared to the imitation baseline. In Hopper-RR, the quantilizer with $q = 0.01$ substantially outperformed the imitation agent. This was expected because the quantilizers were imitating a higher-performing set of trajectories, as shown in Figure 3.1. The quantilizers with larger q did not substantially improve upon imitation. In Mountain Car-RR and Video Pinball-RR, the quantilizers performed somewhat worse than imitation.

In theory, quantilization should be expected to perform best in Video Pinball-RR, where there actually exists a guarantee, because implicit loss is upper-bounded at zero usage of the bumper trick (Figure 3.1). However, in practice, quantilization performed best in Hopper-RR, and only somewhat well in Mountain Car-RR and Video Pinball-RR.

4 Related work

Our work relates to the literature on specification-gaming, and mixtures of optimization and imitation. Specification-gaming is a well-known phenomenon in evolutionary algorithms and in robotics (Lehman et al., 2018), but the first formal proposal for addressing it was quantilization (Taylor, 2016). In Everitt et al. (2017), this was analyzed in the context of ergodic multi-step decision problems. The present work goes further by considering a more general problem setting, by extending our experiments to more complex problems, and by introducing human demonstrations. Interestingly, quantilization was independently rediscovered as an approach to model-based planning, where the proxy reward corresponds to the performance that a policy is estimated to obtain in a learned world-model (Mishra et al., 2017), although they do not analyze in much detail why this approach succeeds.

Quantilization is situated in a similar problem setting to broader efforts to combine optimization (including reinforcement learning) with imitation. To the author's knowledge, published algorithms are all variations on using a weighted sum of reinforcement learning and imitation rewards, for example Hester et al. (2018), which adds an initial pre-initialization step of pure imitation, and Nair et al. (2018), which ignores imitation samples where performance is worse than that expected by the AI system. These solutions, however, are not designed to deal with errors in the reward function, and, as shown in Section 2, they can incur arbitrarily severe losses if they are used to do so.

5 CONCLUSION

To sum up, quantilization has a wide potential range of applicability for reducing specification-gaming. In theory, it only provides useful guarantees when the possible implicit reward is upper-bounded. In practice, when the true reward is closer to the proxy reward, or local minima exist in the explicit reward, it may be useful for a much wider range of cases.

ACKNOWLEDGMENTS

Thanks to Michaël Trazzi for feedback and codebase improvements. Thank you anonymous reviewers, for suggested improvements to the text. Thanks to Marek Sklenka and Sam Clarke for help with data collection.

REFERENCES

Dario Amodei and Jack Clark. Faulty reward functions in the wild, 2016.

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

Tom Everitt, Victoria Krakovna, Laurent Orseau, Marcus Hutter, and Shane Legg. Reinforcement learning with a corrupted reward channel. *arXiv preprint arXiv:1705.08417*, 2017.

David Ha. Reinforcement learning for improving agent design. *arXiv preprint arXiv:1810.03779*, 2018.

Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart J Russell, and Anca Dragan. Inverse reward design. In *Advances in Neural Information Processing Systems*, pp. 6765–6774, 2017.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Łukasz Kidziński, Sharada Prasanna Mohanty, Carmichael Ong, Zhewei Huang, Shuchang Zhou, Anton Pechenko, Adam Stelmaszczyk, Piotr Jarosik, Mikhail Pavlov, Sergey Kolesnikov, et al. Learning to run challenge solutions: Adapting reinforcement learning methods for neuromusculoskeletal environments. *arXiv preprint arXiv:1804.00361*, 2018.

Vitaly Kurin, Sebastian Nowozin, Katja Hofmann, Lucas Beyer, and Bastian Leibe. The atari grand challenge dataset. *arXiv preprint arXiv:1705.10998*, 2017.

Joel Lehman, Jeff Clune, Dusan Misevic, Christoph Adami, Julie Beaulieu, Peter J Bentley, Samuel Bernard, Guillaume Belson, David M Bryson, Nick Cheney, et al. The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities. *arXiv preprint arXiv:1803.03453*, 2018.

Nikhil Mishra, Pieter Abbeel, and Igor Mordatch. Prediction and control with temporal segment models. In *International Conference on Machine Learning*, pp. 2459–2468, 2017.

Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6292–6299. IEEE, 2018.

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

Karl Sims. Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pp. 15–22. ACM, 1994.

Jessica Taylor. Quantilizers: A safer alternative to maximizers for limited optimization. In *AAAI Workshop: AI, Ethics, and Society*, 2016.

A PROOFS

Proof of Lemma 4. For actions $a \neq a_{i_q}$, the upper bound follows immediately from Definition 3.

For a_{i_q}, note that the probability of sampling actions from $\{a_1, ..., a_{i_q}\}$ is by construction higher than q. Therefore:

$$\begin{aligned} q &\leq \sum_{i \geq i_q} D(a_i) \\ &= D(a_{i_q}) + \sum_{i > i_q} D(a_i) \\ \therefore \frac{D(a_{i_q})}{q} &\geq 1 - \frac{\sum_{i > i_q} D(a_i)}{q} = Q_q(a_{i_q}) \end{aligned}$$

□

Theorem 5 (Quantilizer guarantee (Taylor, 2016))**.** *For any k-adequately demonstrated reward function I in a robust reward problem P with $\mathcal{I} \subseteq (-\infty, c]^{\mathcal{A}}$, $c \in \mathbb{R}^+$, each q-quantilizer strategy Q_q with $q \in (0, 1]$ has performance lower-bounded by $\mathbb{E}_{a \sim Q_q}[U^*(a)] \geq \mathbb{E}_{a \sim Q_q}[U(a)] - \frac{k}{q} + \frac{q-1}{q}c$.*

Proof of Theorem 5. Since I is k-adequately demonstrated, we have:

$$\begin{aligned} -k &\leq \mathbb{E}_{a \sim D}[I(a)] \\ &= \sum_a D(a)(I(a) - c) + c \\ &\leq \sum_a q Q_q(a)(I(a) - c) + c \qquad \text{(from Lemma 4 and } I(a) - c \leq 0) \\ &= q\mathbb{E}_{a \sim Q_q}[I(a)] - cq + c \\ \therefore \mathbb{E}_{a \sim Q_q}[I(a)] &\geq \frac{-k}{q} + \frac{(q-1)}{q}c \end{aligned}$$

□

Theorem 6 (Quantilizer optimality (Taylor, 2016))**.**

- *a For any strategy S in a robust reward problem P with $\mathcal{I} = (-\infty, c]^{\mathcal{A}}$, $c \in \mathbb{R}^+$, if S is a distribution over the support of the demonstrations D, then there exists a k-adequately demonstrated implicit reward $I \in \mathcal{I}$ that upper-bounds performance as: $\mathbb{E}_{a \sim S}[U^*(a)] \leq \mathbb{E}_{a \sim Q_q}[U(a)] - \frac{k}{q} + \frac{(q-1)}{q}c$ where $q := \min_{a \in supp(S)} \frac{D(a)}{S(a)}$.*
- *b For any strategy S for which some action a has $S(a) > 0$, $D(a) = 0$, some implicit reward function k-adequately demonstrated by D gives performance unbounded below.*

Proof of Theorem 6.
a) Let $q^* = \min_{a \in \text{supp}(S)} \frac{D(a)}{S(a)} \in (0, 1]$ and a^* be such that $\frac{D(a^*)}{S(a^*)} = q^*$. Then, the function I defined by:

$$I(a) = \begin{cases} \frac{-k-(1-D(a^*))c}{D(a^*)} & a \in a^* \\ c & a \notin a^* \end{cases}$$

has $\mathbb{E}_{a \sim S}[I(a)] \geq -k$ so is a k-adequately demonstrated function. Furthermore, if we instead take the expected value over S, we get the bound for the implicit reward of a q^*-quantilizer (per Theorem 5):

$$\mathbb{E}_{a \sim S}[I(a)] = S(a^*)\frac{-k - (1 - D(a^*))c}{D(a^*)} + (1 - S(a^*))c = -\frac{k}{D(a^*)/S(a^*)} + \frac{D(a^*)/S(a^*) - 1}{D(a^*)/S(a^*)}c$$

Since $q^* = \min_{a \in \text{supp}(S)} \frac{D(a)}{S(a)}$, the quantilizer chooses top actions at least as often as S: $\sum_{i>i_{q^*}} \frac{D(a_i)}{q^*} \geq \sum_{i>i_{q^*}} S(a_i)$. The better actions beat the median, and the worse actions are inferior:
$U(a_j) \leq M_{q^*} \leq U(a_i)$ for $j < i_{q^*} < i$. It follows that the explicit reward is upper-bounded by that of a q^*-quantilizer:

$$\begin{aligned}
\mathbb{E}_{Q_q}[U(a)] &= \left(1 - \sum_{i>i_{q^*}} \frac{D(a_i)}{q^*}\right) M_{q^*} + \sum_{i>i_{q^*}} \frac{D(a_i)}{q^*} U(a_i) \\
&\geq \left(1 - \sum_{i>i_q^*} \frac{D(a_i)}{q^*}\right) M_{q^*} + \left(\sum_{i>i_{q^*}} \frac{D(a_i)}{q^*} - \sum_{i>i_{q^*}} S(a^*)\right) M_{q^*} + \sum_{i>i_{q^*}} S(a_i)U(a_i) \\
&\geq \left(1 - \sum_{i>i_{q^*}} S(a^*)\right) M_{q^*} + \sum_{i>i_{q^*}} S(a_i)U(a_i) \\
&\geq \sum_i S(a_i)U(a_i)
\end{aligned}$$

Adding these two figures yields $\mathbb{E}_{a\sim S}[U^*(a)] \leq \mathbb{E}_{a\sim Q_{q^*}}[U(a)] - \frac{k}{q^*} + \frac{(q^*-1)}{q^*}c$, giving the result (and showing that performance is at least as bad as a q^*-quantilizer).

b) If a strategy S places any probability on any action a' not supported by D, then $I(a) = \begin{cases} -m & a \in a^* \\ 0 & \text{otherwise} \end{cases}$ (where $m \in \mathbb{R}^+$) is a k-adequately-demonstrated implicit reward, which will give implicit reward unbounded below for a particular value of q (for instance $q = \frac{1}{2}$) if m is sufficiently large, and hence the same for performance. □

Theorem 7 (Optimizer/imitator guarantee)**.** *For any k-adequately demonstrated reward function I in a robust reward problem P with $\mathcal{I} \subseteq [-c, \infty)^{\mathcal{A}}$, $c \in \mathbb{R}^+$, the U-optimal action has performance lower-bounded by $U^*(a) \geq \max_{a\in\mathcal{A}} U(a) - c$ and the demonstration distribution D has performance lower-bounded by $\mathbb{E}_{a\sim D}[U^*(a)] \geq \mathbb{E}_{a\sim D}[U(a)] - k$.*

Proof of Theorem 7. By assumption, $I(a) \in [-c, \infty)$, so an optimizer that chooses a U-optimal action a^* will guarantee performance of $U^*(a^*) = U(a^*) + I(a^*) \geq U(a^*) - c$ By k-adequacy, imitation of D will guarantee performance of $\mathbb{E}_{a\sim D}[U^*(a)] = \mathbb{E}_{a\sim D}[U(a)] + \mathbb{E}_{a\sim D}[I(a)] \geq \mathbb{E}_{a\sim D}[U(a)] - k$ □

Theorem 8 (Optimizer/imitator optimality)**.** *For any strategy S in a robust reward problem P with $\mathcal{I} = (-c, \infty]^{\mathcal{A}}$, $c \in \mathbb{R}^+$, there exists a k-adequately demonstrated implicit reward $I \in \mathcal{I}$ that upper-bounds performance as: $\mathbb{E}_{a\sim S}[U^*(a)] \leq \mathbb{E}_{a\sim D}[U(a)] - k$ and $\mathbb{E}_{a\sim S}[U^*(a)] \leq \max_{a\in\mathcal{A}} U(a) - c$*

Proof of Theorem 8. For any strategy S, let $A' = \arg\min_{a\in\text{supp}(D)} \frac{S(a)}{D(a)}$. Let $\lambda = \frac{S(A')}{D(A')} \in [0, 1]$. Then we can decompose S into a weighted sum of the demonstration distribution D, and the distribution S', where $S = \lambda D + (1 - \lambda)S'$, where the support of S' does not intersect A'.

Then, there exists a k-adequately demonstrated implicit reward function

$$I = \begin{cases} -c & a \in \text{supp}(S') \\ \frac{-k + cp_D(a\in S')}{p_D(a\in A')} & \text{if } a \in A' \\ 0 & \text{otherwise} \end{cases}$$

giving performance of:

$$\begin{aligned}
\mathbb{E}_{a\sim S}[U^*(a)] &= \lambda\mathbb{E}_{a\sim D}[U(a) + I(a)] + (1 - \lambda)\mathbb{E}_{a\sim S'}[U(a) + I(a)] \\
&= \lambda(\mathbb{E}_{a\sim D}[U(a)] - k) + (1 - \lambda)\mathbb{E}_{a\sim S'}[U(a) - c] \\
&\leq \lambda(\mathbb{E}a \sim D[U(a) - k)(1 - \lambda)(\max_{a\in\mathcal{A}} U(a) - c)
\end{aligned}$$

Since this is a convex combination of $\mathbb{E}_{a\sim D}[U(a)] - k$ and $\max_{a\in\mathcal{A}} U(a) - c$, it cannot be greater than either. □

Theorem 9 (Imitator guarantee)**.** *For any k-adequately demonstrated reward function I in a robust reward problem P with $\mathcal{I} \subseteq \mathbb{R}^{\mathcal{A}}$, the demonstration distribution D has performance lower-bounded by $\mathbb{E}_{a\sim D}[U^*(a)] \geq \mathbb{E}_{a\sim D}[U(a)] - k$.*

Proof of Theorem 9. From k-adequacy, imitation has the guarantee that $\mathbb{E}_{a\sim D}[U(a) + I(a)] \geq \mathbb{E}_{a\sim D}[U(a)] - k$ □

Theorem 10 (Imitator optimality)**.** *For any strategy S that does not equal the demonstration distribution D in a robust reward problem P with $\mathcal{I} = \mathbb{R}^{\mathcal{A}}$, $c \in \mathbb{R}^+$, there exists a k-adequately demonstrated implicit reward $I \in \mathcal{I}$ that gives performance $\mathbb{E}_{a\sim S}[U^*(a)] = -\infty$*

Proof of Theorem 10. For any strategy S that does not equal the base distribution D, there can be unbounded expected implicit loss. This is because such a strategy S must contain at least one action a^+ that is taken more frequently in S than in D, and at least one action a^- that is taken less frequently. For any desired level of implicit reward $l \ll 0$, there exists a k-adequately demonstrated reward

$$I(a) = \begin{cases} \frac{-\alpha}{D(a^+)} & a = a^+ \\ \frac{\alpha}{D(a^-)} & a = a^- \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha = \frac{l}{S(a^-)/D(a^-) - S(a^+)/D(a^+)}$ such that $\mathbb{E}_{a\sim S}[I(a)] = \mathbb{E}_{a\sim S}[U(a)] + l$, so the overall performance $\mathbb{E}_{a\sim S}[U(a) + I(a)]$ is unbounded below. □

B Further experimental details

Here, for each task, we provide details regarding the proxy reward, collection of trajectories, hyperparameter values, and other information to assist with replication.

In Mountain Car-RR, for the proxy reward, we used the rightward position of the cart, which starts at $(-0.6, -0.4)$ and is 0.5 at the goal position (same as in OpenAI gym).

SARSA with a linear function approximator was used because if optimizer for U^*, it was able to achieve good performance, unlike a deep q learner, which reaches a local minimum even when given the true reward. SARSA was trained for 200 episodes, with learning rate 0.01 with four radial basis function kernels, with gamma values of 5, 2, 1, and 0.5, and 100 components each. The quantilizer and imitation systems were trained with a two layers, each 20 nodes wide for 200 iterations, which are the same hyperparameters used for quantilization and imitation in Hopper-RR.

In Hopper-RR, the proxy reward was the mean ankle angle (i.e. clockwise angle between the tibia and the right part of the foot in radians) as a reward at the end of an episode. This incentivized gaits that lean forward at the ankle, which is a desired property in reward shaping for gait Kidziński et al. (2018), and in fact is correlated with performance in humans (cf. Figure 2). For gathering human data, we discretized the original OpenAI gym action space (continuous in three dimensions) into 3^3 actions (the user had six control keys, two for each axis, and could apply either a positive $+1$, negative -1 or zero force to each one at each timestep). PPO was trained for 2048 steps with learning rate $3e-4$ with gamma 0.99, and lambda 0.95.

In Video Pinball-RR, we define the true score U^* as $U^* = U + I$ where U is the explicit reward (here the in-game score) and I is the implicit reward, $-\lambda$ times the proportion of time spent between the bumpers, measured using a hard-coded function that tells if the ball is between the bumpers or not). $\lambda \in [0, +\infty[$ is a weight denoting how strongly the designer disprefers the agent using the bumper trick. We used standard Deepmind wrappers that downsample to 84×84 frames, skip multiple frames at initialization and successive frames together in sets of four. In order to match the environment with that of the Atari Grand Challenge data, we did not have the end of a life trigger the end of an episode, nor did we clip rewards. The imitator and quantilizers were trained for $10,000$ steps, using a learning rate of $1e-3$, a batch size of 512. The rainbow agent was trained for 50M frames. The imitator, quantilizer and rainbow agent were all trained with the same deep convolutional architecture as in

Hessel et al. (2018), using out-sizes of $(32, 64, 64)$, kernel sizes of $(8, 4, 3)$, stride-sizes of $(4, 2, 1)$, followed by two linear layers of size 512.