Quantifying Differences in Reward Functions

Adam Gleave1, Michael Dennis1, Shane Legg2, Stuart Russell1, Jan Leike2
1UC Berkeley  2DeepMind
gleave@berkeley.edu

Abstract

For many tasks, the reward function is too complex to be specified procedurally, and must instead be learned from user data. Prior work has evaluated learned reward functions by examining rollouts from a policy optimized for the learned reward. However, this method cannot distinguish between the learned reward function failing to reflect user preferences, and the reinforcement learning algorithm failing to optimize the learned reward. Moreover, the rollout method is highly sensitive to details of the environment the learned reward is evaluated in, which often differ in the deployment environment. To address these problems, we introduce the Equivalent-Policy Invariant Comparison (EPIC) distance to quantify the difference between two reward functions directly, without training a policy. We prove EPIC is invariant on an equivalence class of reward functions that always induce the same optimal policy. Furthermore, we find EPIC can be precisely approximated and is more robust than baselines to the choice of visitation distribution. Finally, we find that the EPIC distance of learned reward functions to the ground-truth reward is predictive of the success of training a policy, even in different transition dynamics.

1 Introduction

Reinforcement learning (RL) has reached or surpassed human performance in many domains with clearly-defined reward functions, such as games [26][19][28] and narrowly-scoped robotic manipulation tasks [20]. Unfortunately, the reward functions for most real-world tasks are difficult or impossible to procedurally specify. Even a task as simple as peg insertion from pixels has a non-trivial reward function that must usually be learned [27][IV.A]. Most real-world tasks have far more complex reward functions than this. In particular, tasks involving human interaction depend on complex and user-dependent preferences. These challenges have inspired work on learning a reward function, whether from demonstrations [17][23][31][9][4], preferences [1][30][7][24][32] or both [13][5].

Prior work usually evaluates the learned reward function $\hat{R}$ using the “rollout method”: training a policy $\pi_{\hat{R}}$ to optimize $\hat{R}$ and then examining rollouts from $\pi_{\hat{R}}$. Unfortunately, this method is computationally expensive because it requires us to solve an RL problem. Furthermore, the rollout method produces false negatives when the reward $\hat{R}$ matches user preferences, but the RL algorithm fails to maximize $\hat{R}$. The rollout method also produces false positives: of the many reward functions inducing the desired rollout in a given environment, only a small subset align with the user’s preferences. If the initial state distribution or transition dynamics change, misaligned rewards may induce undesirable policies.

For example, suppose a user likes apricots, tolerates plums and abhors durians. A reward function that prefers apricots to durians to plums induces the correct apricot-buying behavior at training time. But if the robot is shopping during an apricot shortage, it would buy a fruit the user hates: durians. A careful evaluation of the learned reward function before deployment should catch this error.

*Work partially conducted during an internship at DeepMind.

Preprint. Under review.
Table 1: Summary of the desiderata satisfied by each reward function distance. **Key** – the distance is: a **pseudometric** (section 3); **invariant** to potential shaping [18] and positive rescaling (section 3); a **computationally efficient** approximation achieving low error (section 6.1); **predictive** of the similarity of the trained policies (section 6.2); and **robust** to the choice of visitation distribution (section 6.3).

| Distance | Pseudometric | Invariant | Efficient | Predictive | Robust |
|----------|--------------|-----------|-----------|------------|--------|
| EPIC     | ✓            | ✓         | ✓         | ✓          | ✓      |
| NPEC     | X            | ✓         | X         | ✓          | X      |
| ERC      | ✓            | X         | ✓         | ✓          |        |

Reinforcement learning is founded on the observation that it is usually easier and more robust to specify a reward function, rather than a policy maximizing that reward function. Applying this insight to reward function analysis, we develop methods to compare reward functions **directly**, without training a policy. We summarize our desiderata for reward function distances in Table 1.

We introduce the **Equivalent-Policy Invariant Comparison (EPIC)** pseudometric that meets all five desiderata. EPIC (section 4) canonicalizes the reward functions’ potential-based shaping, then computes the correlation between the canonical rewards over a visitation distribution $D$ of transitions. For comparison, we also propose two baselines (section 5), **Episode Return Correlation (ERC)** and **Nearest Point in Equivalence Class (NPEC)**, which partially satisfy the desiderata.

EPIC works best when $D$ has support on all realistic transitions. In our experiments, we achieve this by using uninformative priors, such as a uniform distribution over transitions. Moreover, we find EPIC is robust to the exact choice of distribution $D$, producing similar results across a range of distributions, whereas ERC and especially NPEC are highly sensitive to the choice of $D$ (section 6.3).

Reward learning algorithms are typically benchmarked on tasks with a known ground-truth reward function $R$. When using the rollout method, it is common to report the **regret**: how much less true reward $R$ is obtained by a policy $\pi_\hat{R}$ optimized for the learned reward $\hat{R}$ versus a policy $\pi_R$ optimized for $R$. We find learned reward functions with low EPIC distance to the true reward $R$ induce policies with low regret in both the training and an unseen test environment (section 6.2). Reward functions with high EPIC distance fail in the test environment but sometimes work in the training environment. EPIC therefore has a lower false positive rate than the rollout method, making it particularly attractive in safety-critical applications where reliability is a key design goal.

2 Related work

There exists a variety of methods to learn reward functions. One prominent family is inverse reinforcement learning (IRL; [17]), which infers a reward function from demonstrations. The IRL problem is inherently underconstrained: many different reward functions can lead to the same demonstrations. Bayesian IRL [23] handles this ambiguity by inferring a posterior over reward functions. By contrast, Maximum Entropy IRL [31] selects the highest entropy reward function consistent with the demonstrations; this method has scaled to high-dimensional environments [8; 9].

An alternative approach is to learn from **preference comparisons** between two trajectories [1, 30, 7, 24]. T-REX [5] is a hybrid approach, learning from a ranked set of demonstrations. More directly, Cabi et al. [6] learn from “sketches” of cumulative reward over an episode.

To the best of our knowledge, there is no prior work that focuses on evaluating reward functions directly. The most closely related work is Ng et al. [18], identifying reward transformations guaranteed not to change the optimal policy. However, a variety of ad-hoc methods have been developed to evaluate reward functions. The rollout method – evaluating rollouts of a policy trained on the learned reward – is evident in the earliest work on IRL [17]. Fu et al. [9] refined the rollout method by testing on a transfer environment, inspiring our experiment in section 6.2. Recent work has compared reward functions by scatterplotting returns [15; 5], inspiring our ERC baseline (section 5.1).
3 Background

This section introduces material needed for the distances defined in subsequent sections. We start by defining a distance metric, then introduce the Markov Decision Process (MDP) formalism, and finally describe when reward functions induce the same optimal policy in any compatible MDP.

**Definition 3.1.** Let \(X\) be a set and \(d : X \times X \to [0, \infty)\) a function. \(d\) is a premetric if \(d(x, x) = 0\) for all \(x \in X\). \(d\) is a pseudometric if, furthermore, for all \(x, y, z \in X\), \(d(x, y) = d(y, x)\) and \(d(x, z) \leq d(x, y) + d(y, z)\). \(d\) is a metric if, furthermore, for all \(x, y \in X\), \(d(x, y) = 0 \iff x = y\).

We wish for \(d(R_A, R_B) = 0\) when reward functions \(R_A\) and \(R_B\) are in the same equivalence class, even if \(R_A \neq R_B\). This is forbidden in a metric but permitted in a pseudometric, while retaining other guarantees such as symmetry and triangle inequality that a metric provides. Accordingly, a pseudometric is usually the best choice for a distance \(d\) over reward functions.

**Definition 3.2.** A Markov Decision Process (MDP) \(M = (S, A, \gamma, \mu, T, R)\) consists of a set of states \(S\) and a set of actions \(A\); a discount factor \(\gamma \in [0, 1]\); an initial state distribution \(\mu(s)\); a transition distribution \(T(s' \mid s, a)\) specifying the probability of transitioning to \(s'\) from \(s\) after taking action \(a\); and a reward function \(R(s, a, s')\) specifying the reward upon taking action \(a\) in state \(s\) and transitioning to state \(s'\).

A trajectory \(\tau\) consists of a sequence of states and actions, \(\tau = (s_0, a_0, s_1, a_1, \cdots)\), where each \(s_i \in S\) and \(a_i \in A\). The return on a trajectory is defined as the sum of discounted rewards, \(g(\tau; R) = \sum_{t=0}^{\left| \tau \right|} \gamma^t R(s_t, a_t, s_{t+1})\), where the length of the trajectory \(|\tau|\) may be infinite.

In the following, we assume a discounted (\(\gamma < 1\)) infinite-horizon MDP. The results can be generalized to undiscounted (\(\gamma = 1\)) MDPs subject to regularity conditions needed for convergence.

A stochastic policy \(\pi(a \mid s)\) assigns probabilities to taking action \(a \in A\) in state \(s \in S\). The objective of an MDP is to find a policy \(\pi\) that maximizes the expected return, \(G(\pi) = E_{\tau(\pi)}[g(\tau; R)]\), where \(\tau(\pi)\) is a trajectory generated by sampling the initial state \(s_0\) from \(\mu\), each action \(a_t\) from the policy \(\pi(a_t \mid s_t)\) and successor states \(s_{t+1}\) from the transition distribution \(T(s_{t+1} \mid s_t, a_t)\). An MDP \(M\) has a set of optimal policies \(\pi^*(M)\) that maximize the expected return, \(\pi^*(M) = \arg \max_{\pi} G(\pi)\).

In this paper, we consider the setting where we only have access to an MDPR, \(M^- = (S, A, \gamma, \mu, T)\). The unknown reward function \(R\) must be learned from human data. Typically, only the state space \(S\), action space \(A\) and discount \(\gamma\) are known exactly, with the initial state \(\mu\) and transition dynamics \(T\) only observable from interacting with the environment \(M^-\). In the following, we describe an equivalence class whose members are guaranteed to have the same set of optimal policies in any MDPR \(M^-\) with fixed \(S, A, \gamma\) (allowing the unknown \(T\) and \(\mu\) to take arbitrary values).

**Definition 3.3.** A potential shaping reward is defined as \(R(s, a, s') = \gamma \Phi(s') - \Phi(s)\), given a potential \(\Phi : S \to \mathbb{R}\) and where \(\gamma\) is the MDP discount rate.

**Definition 3.4 (Reward Equivalence).** We define two bounded reward functions \(R_A\) and \(R_B\) to be equivalent, \(R_A \equiv R_B\), for a fixed \((S, A, \gamma)\) if and only if there exists a constant \(\lambda > 0\) and a bounded potential function \(\Phi : S \to \mathbb{R}\) such that for all \(s, s' \in S\) and \(a \in A\): \(R_B(s, a, s') = \lambda R_A(s, a, s') + \gamma \Phi(s') - \Phi(s)\).

Note \(R_A - R_B \equiv \text{Zero}\) (where Zero is the all-zero reward) if and only if \(R_A \equiv R_B\) with \(\lambda = 1\).

**Proposition 3.5.** The binary relation \(\equiv\) is an equivalence relation. Let \(R_A, R_B, R_C : S \times A \times S \to \mathbb{R}\) be bounded reward functions. Then \(\equiv\) is reflexive, \(R_A \equiv R_A\); symmetric, \(R_A \equiv R_B\) implies \(R_B \equiv R_A\); and transitive, \((R_A \equiv R_B) \land (R_B \equiv R_C)\) implies \(R_A \equiv R_C\).

**Proof.** See section [A.3.1] in supplementary material.

The expected return of potential shaping \(\gamma \Phi(s') - \Phi(s)\) on a trajectory segment \((s_0, \cdots, s_T)\) is \(\gamma^T \Phi(s_T) - \Phi(s_0)\). The first term \(\gamma^T \Phi(s_T) \to 0\) as \(T \to \infty\), while the second term \(\Phi(s_0)\) only depends on the initial state, and so potential shaping does not change the set of optimal policies \([13]\).

Scaling a reward function by a positive factor \(\lambda > 0\) scales the expected return of all trajectories by \(\lambda\), leaving the set of optimal policies unchanged. The set of optimal policies is also invariant to a constant shift \(c \in \mathbb{R}\) of the reward, however this can already be obtained by shifting \(\Phi\) by \(\frac{c}{\gamma}\).

---

\(^1\) Note constant shifts in the reward of an undiscounted MDP would cause the value function to diverge. Fortunately, the shaping \(\gamma \Phi(s') - \Phi(s)\) is unchanged by constant shifts to \(\Phi\) when \(\gamma = 1\).
We can now define EPIC in terms of the Pearson distance between canonically shaped rewards. Viewing we would like for small changes in the input reward to produce small changes in the canonical \( \Phi \). The Equivalent-Policy Invariant Comparison distance is a pseudometric. Theorem 4.7.

This novel distance canonicalizes the reward functions’ potential-based shaping, then compares their canonically shaped versions over distance between reward functions \( \rho \). In other words, \( R_A \) and \( R_B \) induce the same optimal policies for all initial state distributions \( \mu \) and transition dynamics \( T \).

4 Comparing reward functions with EPIC

In this section we introduce the Equivalent-Policy Invariant Comparison (EPIC) pseudometric. This novel distance canonicalizes the reward functions’ potential-based shaping, then compares the canonical representatives using Pearson distance, which is invariant to scale. Together, this construction makes EPIC invariant on reward equivalence classes. See section 3.2 for proofs.

We define the canonically shaped reward \( C_{D_S,D_A} (R) \) as an expectation over some arbitrary distributions \( D_S \) and \( D_A \) over states \( S \) and actions \( A \) respectively. This construction means \( C_{D_S,D_A} (R) \) only depends on \( (S,A,\gamma) \) and not on the initial state distribution \( \mu \) or transition dynamics \( T \). In particular, no environment interaction is required to compute \( C_{D_S,D_A} (R) \).

**Definition 4.1** (Canonically Shaped Reward). Let \( R : S \times A \times S \rightarrow \mathbb{R} \) be a reward function. Given distributions \( D_S \) and \( D_A \) over states \( S \) and actions \( A \) respectively, let \( S \) and \( S' \) be random variables independently sampled from \( D_S \) and \( A \) sampled from \( D_A \). We define the canonically shaped \( R \) to be:

\[
C_{D_S,D_A} (R) (s, a, s') = R(s, a, s') + \mathbb{E} [\gamma R(s', A, S') - R(s, A, S') - \gamma R(S, A, S')].
\]

Informally, if \( R' \) is shaped by \( \Phi \), then increasing \( \Phi(s') \) increases \( R'(s, a, s') \) by \( \gamma \Phi(s') \) but decreases \( \mathbb{E} [\gamma R'(s', A, S')] \) by \( \gamma \Phi(s') \), canceling. Similarly increasing \( \Phi(s) \) decreases \( R'(s, a, s') \) but increases \( \mathbb{E} [\gamma R(s', A, S')] \). Finally, \( \mathbb{E} [\gamma R(S, A, S')] \) centers the reward, canceling constant shift.

**Proposition 4.2** (The Canonically Shaped Reward is Invariant to Shaping). Let \( R : S \times A \times S \rightarrow \mathbb{R} \) be a reward function and \( \Phi : S \rightarrow \mathbb{R} \) a potential function. Let \( \gamma \in [0,1] \) be a discount rate, and \( D_S \) and \( D_A \) be distributions over states \( S \) and actions \( A \) respectively. Let \( R' \) denote \( R \) shaped by \( \Phi: R'(s, a, s') = R(s, a, s') + \gamma \Phi(s') - \Phi(s) \). Then the canonically shaped \( R' \) and \( R \) are equal:

\[
C_{D_S,D_A} (R') = C_{D_S,D_A} (R).
\]

Proposition 4.2 holds for arbitrary distributions \( D_S \) and \( D_A \). However, distributions with broad support over realistic states and actions produce more stable canonical transformations. Specifically, we would like for small changes in the input reward to produce small changes in the canonical representative. That is, letting \( \epsilon \) be a small noise term, \( C_{D_S,D_A} (R + \epsilon) \approx C_{D_S,D_A} (R) \).

Viewing \( R \) as a real vector, \( C_{D_S,D_A} (R) \) is a linear transformation with coefficients given by the joint distribution for \( S \times A \times S' \). This transformation is most stable when the coefficients are uniform, so we favor distributions with broad support in our experiments. However, sometimes it is appropriate to place less weight on certain states and actions, e.g. if they’re known to be physically unreachable.

So far, we have removed any dependence on potential shaping. We must still normalize the scale of rewards, and then compare the normalized rewards. The Pearson distance does this in a single step.

**Definition 4.3.** The Pearson distance between random variables \( X \) and \( Y \) is defined by the expression

\[
D_\rho(X,Y) = \frac{1}{\sqrt{2}} \sqrt{1 - \rho(X,Y)}, \text{ where } \rho(X,Y) \text{ is the Pearson correlation between } X \text{ and } Y.
\]

**Lemma 4.4.** The Pearson distance \( D_\rho \) is a pseudometric.

**Lemma 4.5.** Let \( a, b \in (0, \infty), c, d \in \mathbb{R} \) and \( X, Y \) be random variables. Then it follows that

\[
0 \leq D_\rho(aX + c, bY + d) = D_\rho(X,Y) \leq 1.
\]

We can now define EPIC in terms of the Pearson distance between canonically shaped rewards.

**Definition 4.6** (Equivalent-Policy Invariant Comparison (EPIC) pseudometric). Let \( D \) be some visitation distribution over transitions \( s \xrightarrow{a} s' \). Let \( S, A, S' \) be random variables jointly sampled from \( D \). Let \( D_S \) and \( D_A \) be some distributions over states \( S \) and actions \( A \) respectively. The Equivalent-Policy Invariant Comparison (EPIC) distance between reward functions \( R_A \) and \( R_B \) is the Pearson distance between their canonically shaped versions over \( D \):

\[
D_{\text{EPIC}}(R_A, R_B) = D_\rho( C_{D_S,D_A} (R_A) (S, A, S'), C_{D_S,D_A} (R_B) (S, A, S')).
\]

**Theorem 4.7.** The Equivalent-Policy Invariant Comparison distance is a pseudometric.
We choose state and action distributions without training a policy. Given the lack of established methods, we develop two alternatives as baselines: Episode Return Correlation (ERC) and Nearest Point in Equivalence Class (NPEC).

5 Baseline approaches for comparing reward functions

To the best of our knowledge, EPIC is the first method to quantitatively evaluate reward functions without training a policy. Given the lack of established methods, we develop two alternatives as baselines: Episode Return Correlation (ERC) and Nearest Point in Equivalence Class (NPEC).

5.1 Episode Return Correlation (ERC)

The goal of an MDP is to maximize expected episode return, so it is natural to compare reward functions by the returns they induce. If the return of a reward function $R_A$ is a positive affine transformation of another reward $R_B$, then $R_A$ and $R_B$ have the same set of optimal policies. This suggests using Pearson distance, which is invariant to positive affine transformations.

**Definition 5.1** (Episode Return Correlation (ERC) pseudometric). Let $\mathcal{D}$ be some distribution over trajectories. Let $E$ be a random variable sampled from $\mathcal{D}$. The Episode Return Correlation distance between reward functions $R_A$ and $R_B$ is the Pearson distance between their episode returns on $\mathcal{D}$,

$$D_{\text{ERC}}(R_A, R_B) = D_\rho(g(E; R_A), g(E; R_B)).$$

Prior work has scatterplot the return of $R_A$ against $R_B$ over episodes [5] figure 3] and fixed-length segments [15] section D]. ERC is the Pearson distance of such plots, so is a natural baseline. We approximate ERC by the correlation of episode returns on a finite collection of rollouts.

Under special conditions, ERC is invariant to shaping. Let $R$ be a reward function and $\Phi$ a potential function, and define the shaped reward $R'(s, a, s') = R(s, a, s') + \gamma \Phi(s') - \Phi(s)$. The return under the shaped reward on a trajectory $\tau = (s_0, a_0, \cdots, s_T)$ is $g(\tau; R') = g(\tau; R) + \gamma^T \Phi(s_T) - \Phi(s_0)$.
We evaluate EPIC and the baselines ERC and NPEC in a variety of continuous control tasks. First, we compute the distance between hand-designed reward functions, finding EPIC to be the most reliable distance. Although NPEC produces qualitatively similar results, it has a high degree of approximation error. Moreover, ERC sometimes suffers from pathological failures, such as assigning a high distance to equivalent rewards. Second, we find the distance of learned reward functions to a ground-truth reward predicts the return obtained by policy training, even in an unseen test environment. Finally, we show EPIC is robust to the exact choice of visitation distribution $D$, whereas ERC and especially NPEC are highly sensitive to the choice of $D$.

5.2 Nearest Point in Equivalence Class (NPEC)

NPEC takes the minimum $L^p$ distance between equivalence classes. See section [A.3.3] for proofs.

**Definition 5.2 ($L^p$ distance).** Let $D$ be a visitation distribution over transitions $s \xrightarrow{a} s'$ and let $p \geq 1$ be a power. The $L^p$ distance between reward functions $R_A$ and $R_B$ is the $L^p$ norm of their difference:

$$D_{L^p}(R_A, R_B) = \left( \mathbb{E}_{s,a,s' \sim D} \left[ |R_A(s,a,s') - R_B(s,a,s')|^p \right] \right)^{1/p}.$$

**Proposition 5.3.** (1) $D_{L^p}$ is a metric in $L^p$ space, where functions $f$ and $g$ are identified if $f = g$ almost everywhere on $D$. (2) It is a pseudometric when $f$ and $g$ are identified if $f = g$ at all points.

The $L^p$ distance is sensitive to shaping and positive rescaling that do not change the optimal policy. A natural solution is to take the distance from the nearest point in the equivalence class. The Unnormalized Nearest Point in Equivalence Class distance is: $D_{\text{UNPEC}}^U(R_A, R_B) = \inf_{R'_A \equiv R_A} D_{L^p}(R'_A, R_B)$.

Note the infimum over both $R'_A \equiv R_A$ and $R'_B \equiv R_B$ would always be zero, since all equivalence classes come arbitrarily close to the origin in $L^p$ space. However, taking the infimum over only $R'_A \equiv R_A$ causes $D_{\text{NPEC}}^U(R_A, R_B)$ to be sensitive to the scale of $R_B$. We fix this by normalizing.

**Definition 5.4.** The Nearest Point in Equivalence Class (NPEC) premetric is defined by:

$$D_{\text{NPEC}}(R_A, R_B) = \frac{D_{\text{NPEC}}^U(R_A, R_B)}{D_{\text{NPEC}}^U(\text{Zero}, R_B)} \text{ when } D_{\text{NPEC}}^U(\text{Zero}, R_B) \neq 0 \text{ and } 0 \text{ otherwise.}$$

Note if $D_{\text{NPEC}}^U(\text{Zero}, R_B) = 0$ then $D_{\text{NPEC}}^U(R_A, R_B) = 0$ since $R_A$ can be scaled arbitrarily close to Zero. Since all policies are optimal for $R \equiv \text{Zero}$, we choose $D_{\text{NPEC}}(R_A, R_B) = 0$ in this case.

**Proposition 5.5.** $D_{\text{NPEC}}$ is a premetric.

Note that $D_{\text{NPEC}}$ is not in general a pseudometric: see proposition [A.1] for a counterexample. It is, however, bounded and invariant to shaping, similar to EPIC.

**Theorem 5.6.** Let $R_A, R'_A, R_B, R'_B : S \times A \times S \rightarrow \mathbb{R}$ be reward functions such that $R_A \equiv R'_A$ and $R_B \equiv R'_B$. Then $0 \leq D_{\text{NPEC}}(R'_A, R'_B) = D_{\text{NPEC}}(R_A, R_B) \leq 1$.

The infimum in $D_{\text{NPEC}}^U$ can be computed exactly in a tabular setting, but in general we must approximate it using gradient descent. This gives an upper bound for $D_{\text{NPEC}}^U$, but the quotient of upper bounds $D_{\text{NPEC}}$ may be too low or too high. See section [A.1.2] for details of the approximation.

6 Experiments

We evaluate EPIC and the baselines ERC and NPEC in a variety of continuous control tasks. First, we compute the distance between hand-designed reward functions, finding EPIC to be the most reliable distance. Although NPEC produces qualitatively similar results, it has a high degree of approximation error. Moreover, ERC sometimes suffers from pathological failures, such as assigning a high distance to equivalent rewards. Second, we find the distance of learned reward functions to a ground-truth reward predicts the return obtained by policy training, even in an unseen test environment. Finally, we show EPIC is robust to the exact choice of visitation distribution $D$, whereas ERC and especially NPEC are highly sensitive to the choice of $D$. 

6 Experiments

We evaluate EPIC and the baselines ERC and NPEC in a variety of continuous control tasks. First, we compute the distance between hand-designed reward functions, finding EPIC to be the most reliable distance. Although NPEC produces qualitatively similar results, it has a high degree of approximation error. Moreover, ERC sometimes suffers from pathological failures, such as assigning a high distance to equivalent rewards. Second, we find the distance of learned reward functions to a ground-truth reward predicts the return obtained by policy training, even in an unseen test environment. Finally, we show EPIC is robust to the exact choice of visitation distribution $D$, whereas ERC and especially NPEC are highly sensitive to the choice of $D$. 

6 Experiments

We evaluate EPIC and the baselines ERC and NPEC in a variety of continuous control tasks. First, we compute the distance between hand-designed reward functions, finding EPIC to be the most reliable distance. Although NPEC produces qualitatively similar results, it has a high degree of approximation error. Moreover, ERC sometimes suffers from pathological failures, such as assigning a high distance to equivalent rewards. Second, we find the distance of learned reward functions to a ground-truth reward predicts the return obtained by policy training, even in an unseen test environment. Finally, we show EPIC is robust to the exact choice of visitation distribution $D$, whereas ERC and especially NPEC are highly sensitive to the choice of $D$. 

6 Experiments
6.1 Comparing hand-designed reward functions

We compare procedurally specified reward functions in four tasks. Figure 2 presents results in the proof-of-concept PointMaze task. The results for Gridworld, HalfCheetah and Hopper, in section A.2.4, are qualitatively similar. In PointMaze the agent can accelerate left or right on a line. The reward functions include (○) or exclude (□) a quadratic penalty \( \|a\|^2 \). The sparse reward (S) gives a reward of 1 in the region around the goal state. The dense reward (D) is a shaped version of the sparse reward. The magnitude reward (M) is the negative distance of the agent from the goal.

We find that EPIC correctly identifies the equivalent reward pairs (S ○ - D ○) and (S ○ - D ○) with estimated distance < 1 \times 10^{-3}. By contrast, NPEC suffers from considerable approximation error: \( D_{NPEC}(D, S) = 0.58 \). Similarly, ERC’s erroneous handling of shaping when the initial state is stochastic produces \( D_{ERC}(D, S) = 0.56 \). In the next section, we compare learned rewards.

6.2 Predicting policy performance from reward distance

We train reward models on the PointMaze task from Fu et al. [9], and evaluate the ground-truth (GT) return of a policy optimized for the learned reward. Table 2 shows that rewards with low distance from GT achieve high returns. Rewards with high distance sometimes work but are context sensitive.

PointMaze is a MuJoCo environment where a point mass agent must navigate around a wall to reach a goal. The train and test variants differ only in the position of the wall. We evaluate four reward learning algorithms: Regression onto reward labels [target method from [7], section 3.3], Preference comparisons on trajectories [7], and adversarial IRL with a state-only (AIRL SO) and state-action (AIRL SA) reward model [9]. All models are trained using synthetic data from an oracle with access to the ground-truth; see section A.2.2 for details.

Both Regression and Pref achieve very low distances, producing near-expert policy performance in both the train and test variants. The AIRL SO and AIRL SA models have distances an order of magnitude greater. The more expressive AIRL SA achieves near-expert performance in train but fails to transfer to test. The less-expressive AIRL SO has poor performance in both variants, although the generator policy (trained simultaneously with the reward) performs reasonably in train.

Due to reward ambiguity, rewards such as AIRL * that are distant from the ground-truth GT can still produce a good policy. For example, a “memorized” reward function that assigns reward only to states visited by an expert will induce the expert policy in the train variant. Nonetheless, it will have a large distance from GT, even if the visitation distribution \( D \) only contains transitions from \( train \). This is appropriate since in test the “expert” policy produced by this reward runs straight into the wall.

6.3 Sensitivity of reward distance to visitation state distribution

We would like the reward distances to be robust to the exact choice of visitation distribution \( D \). In Table 1 we report distances calculated under distributions induced by rollouts from three different
We believe the EPIC distance will be a highly informative addition to the evaluation toolbox, and we find EPIC is robust to varying $D$. This is important since it is common for the initial state distribution or transition dynamics to change between the training environment where the reward function was learned, and the test environment where the system is deployed \cite{22,2,21}. For example, one might learn a reward and policy in simulation, and then fine-tune the policy in the real world with the learned reward.

Nonetheless, even with EPIC some care must be taken when choosing $D$. Typically, $D$ is collected via rollouts of some exploration policy in an environment. This works well when the deployment environment has a similar set of reachable states to the rollout environment, even if some details of the dynamics – such as the position of the wall in PointMaze – differ. However, when the deployment environment allows a transition $(s, a, s')$ that is not physically attainable in the rollout environment, then $D$ will place no support on this transition and the reward $R(s, a, s')$ can take arbitrary values without affecting the distance. In general, any black-box method for assessing reward models – including the rollout method – only has predictive power on transitions that it visits during testing.

7 Discussion

Our novel EPIC distance compares reward functions directly, without training a policy. We have proved it satisfies the axioms of a pseudometric, and moreover is bounded and invariant to equivalent rewards. Empirically, we find the EPIC distance between procedurally specified reward functions is more reliable than the NPEC and ERC baselines.

Furthermore, we find the distance of learned reward functions to the ground-truth reward predicts the return of policies optimized for the learned reward, in both the train and unseen test environments. This is important since it is common for the initial state distribution or transition dynamics to change between the training environment where the reward function was learned, and the test environment where the system is deployed \cite{22,2,21}. For example, one might learn a reward and policy in simulation, and then fine-tune the policy in the real world with the learned reward.

An important direction for future work is to evaluate reward models trained on noisy and biased human data. Such models will have a higher EPIC distance from the ground-truth than models trained on synthetic data, however some algorithms may be more robust to imperfect feedback than others.

Benchmarks are an important driver of progress in machine learning. Unfortunately, traditional policy-based metrics do not provide any guarantees as to the fidelity of the learned reward function. We believe the EPIC distance will be a highly informative addition to the evaluation toolbox, and would encourage researchers to report EPIC distance in addition to policy-based metrics.

### Table 2: Distances of reward models from ground-truth (GT), and the mean GT return of policies optimized from-scratch for the reward model in the train and test variants of PointMaze. We also report returns for AIRL’s generator policy, jointly trained with the reward. Distances (1000× scale) use visitation distribution $D$ from rollouts in the train environment of: a uniform random policy $\pi_{\text{unif}}$, an expert $\pi^*$ and a mixture of these policies. $D_S$ and $D_A$ are computed by marginalizing $D$. 95% confidence intervals (see Table A.6) are tighter than ±1% for EPIC and ERC but are as large as ±50% for NPEC due to high variance across seeds.

| Reward Model | $1000 \times D_{\text{EPIC}}$ | $1000 \times D_{\text{NPEC}}$ | $1000 \times D_{\text{ERC}}$ | Episode Return |
|--------------|----------------------|----------------------|----------------------|-----------------|
| Model | $\pi_{\text{unif}}$ | $\pi^*$ | Mix | $\pi_{\text{unif}}$ | $\pi^*$ | Mix | $\pi_{\text{unif}}$ | $\pi^*$ | Mix | Gen. | Train | Test |
| GT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | — | — | — |
| Regress | 41.9 | 36.5 | 25.9 | 0.519 | 14.9 | 0.140 | 4.78 | 40.9 | 1.39 | — | — | — |
| Pref | 50.5 | 54.4 | 32.9 | 2.99 | 204 | 1.78 | 15.0 | 180 | 8.15 | — | — | — |
| AIRL SA | 488 | 600 | 395 | 684 | 3550 | 426 | 448 | 382 | 234 | —9.44 | —28.5 | —11.7 |
| AIRL SO | 548 | 614 | 390 | 823 | 3030 | 376 | 506 | 467 | 208 | —6.69 | —6.91 | —28.5 |
Broader Impact

Reinforcement learning (RL) is used to choose push notifications to send to billions of users [10], and in the coming years we are likely to see more large-scale deployment of RL systems in interactive applications. This gives renewed urgency to specifying an appropriate objective for these systems to optimize. Even with the best of intentions, designers cannot possibly anticipate the desires of all their users, let alone procedurally specify this. In the absence of better methods, engineers often resort to proxies such as user engagement. While such metrics are often correlated with user satisfaction, Goodhart’s law predicts this correlation will break down when used as an optimization objective [12]. This can lead to negative unintended consequences, such as addiction to online platforms.

A natural alternative is to instead learn the reward function from user feedback. This democratizes AI systems: rather than the designer picking an optimization objective, the users choose how they wish the AI system to interact with them. The system could even learn a reward function for each user, and optimize that objective on an individual basis (subject to some global fairness constraints).

However, current reward learning algorithms have considerable limitations. We believe quantifying differences in reward functions will help improve benchmarking of reward learning algorithms, spurring algorithmic improvements. Additionally, we anticipate it to be useful as a verification method prior to deployment of a learned reward model.

The main potential downside we see from our work is if people place too much trust in the reward distances proposed in this paper. While we are confident the distance between reward functions will be a highly informative addition to the set of available evaluation metrics, it is intended as a complement and not a replacement for other forms of testing. In particular, it will continue to be important to test the trained policy prior to deployment, especially in safety-critical systems. Even if the learned reward function is correct, there is no guarantee that the policy behaves as desired – for example, policies often suffer from adversarial examples [14, 11].

References

[1] Riad Akrour, Marc Schoenauer, and Michele Sebag. Preference-based policy learning. In Machine Learning and Knowledge Discovery in Databases, 2011.
[2] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. arXiv: 1606.06565v2 [cs.AI], 2016.
[3] Dario Amodei, Paul Christiano, and Alex Ray. Learning from human preferences, June 2017. URL https://openai.com/blog/deep-reinforcement-learning-from-human-preferences/.
[4] Dzmitry Bahdanau, Felix Hill, Jan Leike, Edward Hughes, Arian Hosseini, Pushmeet Kohli, and Edward Grefenstette. Learning to understand goal specifications by modelling reward. In ICLR, 2019.
[5] Daniel S. Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In ICML, 2019.
[6] Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, Oleg Sushkov, David Barker, Jonathan Scholz, Misha Denil, Nando de Freitas, and Ziyu Wang. Scaling data-driven robotics with reward sketching and batch reinforcement learning. arXiv: 1909.12200v2 [cs.RO], 2019.
[7] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In NIPS, pages 4299–4307, 2017.
[8] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In ICML, 2016.
[9] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adverserial inverse reinforcement learning. In ICLR, 2018.
[10] Jason Gauci, Edoardo Conti, Yitao Liang, Kittipat Virochsiri, Yuchen He, Zachary Kaden, Vivek Narayanan, Xiaohui Ye, Zhengxing Chen, and Scott Fujimoto. Horizon: Facebook’s open source applied reinforcement learning platform. arXiv: 1811.00260 [cs.LG], 2018.

[11] Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. Adversarial policies: Attacking deep reinforcement learning. In ICLR, 2020.

[12] C. A. E. Goodhart. Problems of monetary management: the UK experience. In Monetary Theory and Practice, pages 91–121. Springer, 1984.

[13] Ashley Hill, Antonin Raffin, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, Rene Traore, Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. Stable Baselines. [https://github.com/hill-a/stable-baselines], 2018.

[14] Sandy H. Huang, Nicolas Papernot, Ian J. Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. arXiv:1702.02284v1 [cs.LG], 2017.

[15] Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in Atari. In NeurIPS, pages 8011–8023, 2018.

[16] Charles L. Lawson and Richard J. Hanson. Solving Least Squares Problems. SIAM, 1995.

[17] Andrew Y. Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In ICML, 2000.

[18] Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: theory and application to reward shaping. In NIPS, 1999.

[19] OpenAI. OpenAI Five. [https://blog.openai.com/openai-five/] 2018.

[20] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolaos Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving Rubik’s Cube with a robot hand. arXiv: 1910.07113v1 [cs.LG], 2019.

[21] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In ICRA, pages 3803–3810, 2018.

[22] Joaquin Quiñonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence, editors. Dataset Shift in Machine Learning. MIT Press, 2008.

[23] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In IJCAI, 2007.

[24] Dorsa Sadigh, Anca D. Dragan, S. Shankar Sastry, and Sanjit A. Seshia. Active preference-based learning of reward functions. In RSS, July 2017.

[25] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv:1707.06347v2 [cs.LG], 2017.

[26] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. Nature, 529(7587):484–489, 2016.

[27] Mel Vecerik, Oleg Sushkov, David Barker, Thomas Rothörl, Todd Hester, and Jon Scholz. A practical approach to insertion with variable socket position using deep reinforcement learning. In ICRA, 2019.
[28] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[29] Steven Wang, Adam Gleave, and Sam Toyer. imitation: implementations of inverse reinforcement learning and imitation learning algorithms. [https://github.com/humancompatibleai/imitation](https://github.com/humancompatibleai/imitation), 2020.

[30] Aaron Wilson, Alan Fern, and Prasad Tadepalli. A Bayesian approach for policy learning from trajectory preference queries. In *NIPS*, 2012.

[31] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008.

[32] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. arXiv: 1909.08593v2 [cs.CL], 2019.
A Supplementary material

A.1 Approximation Procedures

A.1.1 Sample-based approximation for EPIC distance

We approximate EPIC distance (definition 4.6) using a sample-based approach. Specifically, we sample a batch $B_V$ of $N_V$ samples from the visitation distribution $D$, and a batch $B_M$ of $N_M$ samples from the joint state and action distributions $D_S \times D_A$. We approximate the canonically shaped rewards (definition 4.1) by taking the mean over $B_M$:

$$C_{D_S,D_A}(R)(s, a, s') = R(s, a, s') + \mathbb{E}[\gamma R(s', A, S') - R(s, A, S') - \gamma R(S, A, S')]$$

$$\approx R(s, a, s') + \frac{1}{N_M} \sum_{(x,u) \in B_M} R(s', u, x) - \frac{1}{N_M} \sum_{(x,u) \in N_M} R(s, u, x) - \frac{1}{N_M} \sum_{(x',u) \in B_M} R(x, u, x').$$

We then compute the Pearson distance between the approximate canonically shaped rewards on the batch of samples $B_V$.

A.1.2 Optimization-based approximation for NPEC distance

$D_{NPEC}(R_A, R_B)$ (section 5.2) is defined as the infimum of $L^p$ distance over an infinite set of equivalent reward functions $R \equiv R_A$. We approximate this using gradient descent on the reward model:

$$R_{\nu,c,w}(s, a, s') = \exp(\nu)R_A(s, a, s') + c + \gamma \Phi_w(s') - \Phi_w(s),$$

where $\nu, c \in \mathbb{R}$ are scalar weights and $w$ is a vector of weights parameterizing a deep neural network $\Phi_w$. The constant $c \in \mathbb{R}$ is unnecessary if $\Phi_w$ has a bias term, but its inclusion simplifies the optimization problem.

We optimize $\nu, c, w$ to minimize the mean of the cost $J(\nu, c, w) = D(R_{\nu,c,w}(s, a, s'), R_B(s, a, s'))$ on samples $(s, a, s')$ from a visitation distribution $D$. Note the mean cost upper bounds the true NPEC distance since $R_{\nu,c,w} \equiv R_A$.

We found empirically that $\nu$ and $c$ need to be initialized close to their optimal values for gradient descent to reliably converge. To resolve this problem, we initialize the affine parameters to $\nu \leftarrow \log \lambda$ and $c$ found by:

$$\arg\min_{\lambda > 0, c \in \mathbb{R}} \mathbb{E}_{s, a, s', \sim D} (\lambda R_A(s, a, s') + c - R_B(s, a, s'))^2.$$

We use the active set method of Lawson and Hanson [16] to solve this constrained least-squares problem. These initial affine parameters minimize the $L^p$ distance $D_{Lp}(R_{\nu,c,0}(s, a, s'), R_B(s, a, s'))$ under the metric $\ell(x,y) = (x-y)^2$ with the potential fixed at $\Phi_0(s) = 0$.

A.1.3 Confidence Intervals

We report confidence intervals to help measure the degree of error introduced by the approximation. Since approximate distances may not be normally distributed, we use bootstrapping to produce a distribution-free confidence interval. For EPIC and NPEC, we compute independent approximate distances over different seeds, and then compute a bootstrapped confidence interval on the distances for each seed. We use 30 seeds for EPIC but only 3 seeds for NPEC due to its greater computational requirements. In ERC, computing the distance is very cheap, so we instead apply bootstrapping to the collected episodes, computing the ERC distance for each bootstrapped episode sample.

A.2 Experiments

A.2.1 Hyperparameters for Approximate Distances

Table A.1 summarizes the hyperparameters and distributions used to compute the distances between reward functions. Most parameters are the same across all environments. We use a visitation distribution of uniform random transitions $D_{\text{unif}}$ in the simple GridWorld environment with known deterministic dynamics. In other environments, the visitation distribution is sampled from rollouts.
Table A.1: Summary of hyperparameters and distributions used in experiments. The uniform random 
visitation distribution $D_{\text{unif}}$ samples states and actions uniformly at random, and samples the next 
state from the transition dynamics. Random policy $\pi_{\text{unif}}$ takes uniform random actions. The synthetic 
expert policy $\pi^*$ was trained with PPO on the ground-truth reward. Mixture samples actions from 
either $\pi_{\text{unif}}$ or $\pi^*$, switching between them at each timestep with probability 0.05. Warmstart Size is 
the size of the dataset used to compute initialization parameters described in section A.1.2.

| Parameter                | Value                     | In experiment                  |
|--------------------------|---------------------------|--------------------------------|
| Visitation Distribution  | Random transitions $D_{\text{unif}}$ | GridWorld                     |
|                          | Rollouts from $\pi_{\text{unif}}$ | PointMass, HalfCheetah, Hopper |
|                          | $\pi_{\text{unif}}$, $\pi^*$ and Mixture | PointMaze                     |
| Bootstrap Samples        | 10 000                    | All                            |
| Discount $\gamma$        | 0.99                      | All                            |
| EPIC                     |                           |                                |
| State Distribution $D_S$ | $N(0, 1)$ standard Gaussian | PointMass, HalfCheetah, Hopper |
|                          | Marginalized from $D$     | PointMaze                      |
| Action Distribution $D_A$| $U[-1, 1]$ continuous uniform | PointMass, HalfCheetah, Hopper |
|                          | Marginalized from $D$     | PointMaze                      |
| Seeds                    | 30                        | All                            |
| Samples $N_V$            | 32 768                    | All                            |
| Mean Samples $N_M$       | 32 768                    | All                            |
| NPEC                     |                           |                                |
| Seeds                    | 3                         | All                            |
| Total Timesteps          | $1 \times 10^6$          | All                            |
| Optimizer                | Adam                      | All                            |
| Learning Rate            | $1 \times 10^{-2}$        | All                            |
| Batch Size               | 4096                      | All                            |
| Warmstart Size           | 16 386                    | All                            |
| Loss $\ell$             | $\ell(x, y) = (x - y)^2$ | All                            |
| ERC                      |                           |                                |
| Episodes                 | 131 072                   | All                            |

of a policy. We use a random policy $\pi_{\text{unif}}$ for PointMass, HalfCheetah and Hopper in the hand- 
designed reward experiments (section 6.1). In PointMaze, we compare three visitation distributions 
(section 6.3) induced by rollouts of $\pi_{\text{unif}}$, an expert policy $\pi^*$ and a Mixture of the two policies, 
sampling actions from either $\pi_{\text{unif}}$ or $\pi^*$ and switching between them with probability 0.05 per 
timestep.

A.2.2 Training Learned Reward Models

For the experiments on learned reward functions (sections 6.2 and 6.3), we trained reward models 
using adversarial inverse reinforcement learning (AIRL; 9), preference comparison (7) and by 
regression onto the ground-truth reward [target method from 7, section 3.3]. For AIRL, we use 
an existing open-source implementation [29]. We developed new implementations for preference 
comparison and regression, available at — double blind: supplementary material —. We also 
use the RL algorithm proximal policy optimization (PPO, 25) on the ground-truth reward to train 
expert policies to provide demonstrations for AIRL, and on learned reward models to evaluate their 
performance.

For PPO and AIRL we used the default hyperparameters in tables A.2 and A.3, finding them 
adequate and so performing no further tuning. For preference comparison we performed a sweep 
over batch size, trajectory length and learning rate to decide on the hyperparameters in table A.4. 
Total timesteps was selected once diminishing returns were observed in loss curves. The exact value 
of the regularization weight was found to be unimportant, largely controlling the scale of the output 
at convergence. Finally, for regression we performed a sweep over batch size, learning rate and total 
timesteps to decide on the hyperparameters in table A.5. We found batch size and learning rate to
Table A.2: Hyperparameters for proximal policy optimisation (PPO) \cite{schulman2017proximal}. We used the implementation and default hyperparameters from Hill et al. \cite{hill2018retro}. PPO was used to train expert policies on ground-truth reward and to optimize learned reward functions for evaluation.

| Parameter                      | Value          | In environment |
|--------------------------------|----------------|----------------|
| Total Timesteps                | $1 \times 10^6$| All            |
| Batch Size                     | 16384          | PointMaze      |
| Discount $\gamma$              | 0.99           |                |
| Entropy Coefficient            | 0.01           |                |
| Learning Rate                  | $2.5 \times 10^{-4}$ |            |
| Value Function Coefficient     | 0.5            |                |
| Gradient Clipping Threshold    | 0.5            |                |
| Ratio Clipping Threshold       | 0.2            |                |
| Lambda (GAE)                   | 0.95           |                |
| Minibatches                    | 4              |                |
| Optimization Epochs            | 4              |                |
| Parallel Environments          | 8              |                |

Table A.3: Hyperparameters for adversarial inverse reinforcement learning (AIRL) used in Wang et al. \cite{wang2018adversarial}.

| Parameter                      | Value          |
|--------------------------------|----------------|
| RL Algorithm                   | PPO \cite{schulman2017proximal} |
| Total Timesteps                | 102 400        |
| Discount $\gamma$              | 0.99           |
| Demonstration Timesteps        | 100 000        |
| Generator Batch Size           | 2048           |
| Discriminator Batch Size       | 50             |
| Entropy Weight                 | 1.0            |
| Reward Function Architecture   | MLP, two 32-unit hidden layers |
| Potential Function Architecture| MLP, two 32-unit hidden layers |

Table A.4: Hyperparameters for preference comparison used in our implementation of Christiano et al. \cite{christiano2017deep}.

| Parameter                      | Value          | Range Tested       |
|--------------------------------|----------------|--------------------|
| Total Timesteps                | $5 \times 10^6$| $[1, 10 \times 10^6]$ |
| Batch Size                     | 10 000         | $[500, 250 000]$   |
| Trajectory Length              | 5              | $[1, 100]$         |
| Learning Rate                  | $1 \times 10^{-2}$ | $[1 \times 10^{-4}, 1 \times 10^{-1}]$ |
| Discount $\gamma$              | 0.99           |                    |
| Reward Function Architecture   | MLP, two 32-unit hidden layers | |
| Output L2 Regularization Weight| $1 \times 10^{-4}$ |        |

Table A.5: Hyperparameters for regression used in our implementation of Christiano et al. \cite{christiano2017deep} target method from section 3.3.

| Parameter                      | Value          | Range Tested       |
|--------------------------------|----------------|--------------------|
| Total Timesteps                | $10 \times 10^6$| $[1, 20 \times 10^6]$ |
| Batch Size                     | 4096           | $[256, 16384]$     |
| Learning Rate                  | $2 \times 10^{-2}$ | $[1 \times 10^{-3}, 1 \times 10^{-1}]$ |
| Discount $\gamma$              | 0.99           |                    |
| Reward Function Architecture   | MLP, two 32-unit hidden layers | |
be relatively unimportant with many combinations performing well, but regression was found to converge slowly but steadily requiring a relatively large $10 \times 10^6$ timesteps for good performance in our environments.

All algorithms are trained on synthetic data generated from the ground-truth reward function. AIRL is provided with a large demonstration dataset of 100 000 timesteps from an expert policy trained on the ground-truth reward, similar in size to the total number of timesteps AIRL is trained for (see table A.3). In preference comparison and regression, each batch is sampled afresh from the visitation distribution specified in table A.1 and labeled according to the ground-truth reward.

### A.2.3 Computing infrastructure

Experiments were conducted on a small number of n1-standard-96 Google Cloud Platform VM instances, with 48 CPU cores on an Intel Skylake processor and 360 GB of RAM. It takes less than a week of compute on a single n1-standard-96 instance to run all the experiments described in this paper.

### A.2.4 Comparing hand-designed reward functions

We compute distances between hand-designed reward functions in four environments: GridWorld, PointMass, HalfCheetah and Hopper. The reward functions for GridWorld are described in Figure A.1, and the distances are reported in Figure A.2. We report the approximate distances and confidence intervals between reward functions in the other environments in Figures A.3, A.4 and A.5.

We find the (approximate) EPIC distance closely matches our intuitions for similarity between the reward functions. NPEC often produces similar results to EPIC, but unfortunately is dogged by optimization error. This is particularly notable in higher-dimensional environments like HalfCheetah and Hopper, where the NPEC distance often exceeds the theoretical upper bound of 1.0 and the confidence interval width is frequently larger than 0.2.

By contrast, ERC distance generally has a tight confidence interval, but systematically fails in the presence of shaping. For example, it confidently assigns large distances between equivalent reward pairs in PointMass such as $S_D - D$. However, ERC produces reasonable results in HalfCheetah and Hopper where rewards are all similarly shaped. In fact, ERC picks up on a detail in Hopper that EPIC misses: whereas EPIC assigns a distance of around 0.71 between all rewards of different types (running vs backflipping), ERC assigns lower distances when the rewards are in the same direction (forward or backward). Given this, ERC may be attractive in some circumstances, especially given the ease of implementation. However, we would caution against using it in isolation due to the likelihood of misleading results in the presence of shaping.

### A.2.5 Comparing learned reward functions

Previously, we reported the mean approximate distance from a ground-truth reward of four learned reward models in PointMaze (Table 2). Since these distances are approximate, we report 95% lower and upper bounds computed via bootstrapping in Table A.6. We also include the relative difference of the upper and lower bounds from the mean, finding the relative difference to be fairly consistent across reward models for a given algorithm and visitation distribution pair. The relative difference is less than 1% for all EPIC and ERC distances. However, NPEC confidence intervals can be as wide as 50%: this is due to the method’s high variance, and the small number of seeds we were able to run because of the method’s computational expense.
Figure A.1: Heatmaps of reward functions $R(s, a, s')$ for a 3×3 deterministic gridworld. $R(s, \text{stay}, s)$ is given by the central circle in cell $s$. $R(s, a, s')$ is given by the triangular wedge in cell $s$ adjacent to cell $s'$ in direction $a$. Optimal action(s) (for infinite horizon, discount $\gamma = 0.99$) have bold labels against a hatched background. See figure A.2 for the distance between all reward pairs.
Figure A.2: Distances between hand-designed reward functions for the $3 \times 3$ deterministic Gridworld environment. See figure [A.1] for definitions of each reward. Distances are computed using tabular algorithms. We do not report confidence intervals since these algorithms are deterministic and exact up to floating point error.
Figure A.3: Approximate distances between hand-designed reward functions in PointMass. The visitation distribution $D$ is sampled from rollouts of a policy $\pi_{\text{unif}}$ taking actions uniformly at random. Key: $\circ$ quadratic control penalty, $\otimes$ no control penalty. $S$ is $\text{Sparse}(x) = \mathbb{1}_{|x| < 0.05}$, $D$ is shaped $\text{Dense}(x, x') = \text{Sparse}(x) + |x' - x|$, while $M$ is $\text{Magnitude}(x) = -|x|$. Confidence Interval (CI): 95% CI computed by bootstrapping over 10,000 samples.

Figure A.4: Approximate distances between hand-designed reward functions in HalfCheetah. The visitation distribution $D$ is sampled from rollouts of a policy $\pi_{\text{unif}}$ taking actions uniformly at random. Key: a is a reward proportional to the change in center of mass and moving forward is rewarded when a to the right, and moving backward is rewarded when a to the left. $\circ$ quadratic control penalty, $\otimes$ no control penalty. Confidence Interval (CI): 95% CI computed by bootstrapping over 10,000 samples.
Figure A.5: Approximate distances between hand-designed reward functions in Hopper. The visitation distribution $D$ is sampled from rollouts of a policy $\pi_{\text{unif}}$ taking actions uniformly at random. Key: $\leftarrow$ is a reward proportional to the change in center of mass and $\rightarrow$ is the backflip reward defined in Amodei et al. [3, footnote]. Moving forward is rewarded when $\leftarrow$ or $\rightarrow$ is to the right, and moving backward is rewarded when $\leftarrow$ or $\rightarrow$ is to the left. Quadratic control penalty, $\Rightarrow$ no control penalty. **Confidence Interval (CI):** 95% CI computed by bootstrapping over 10 000 samples.
Table A.6: Approximate distances of learned reward models from the ground-truth (GT). We report the 95% bootstrapped lower and upper bounds, the mean, and a 95% bound on the relative error from the mean. Distances (1000× scale) use visitation distribution $D$ from rollouts in the *train* environment of: a uniform random policy $\pi_{\text{unif}}$, an expert $\pi^*$ and a mixture of these policies. $D_S$ and $D_A$ are computed by marginalizing $D$.

(a) 95% lower bound $D_{\text{LOW}}$ of approximate distance.

| Reward | $1000 \times D_{\text{LOW}}^{\text{EPIC}}$ | $1000 \times D_{\text{LOW}}^{\text{NPEC}}$ | $1000 \times D_{\text{LOW}}^{\text{ERC}}$ |
|--------|---------------------------------|---------------------------------|---------------------------------|
| Model  | $\pi_{\text{unif}}$ $\pi^*$ Mix | $\pi_{\text{unif}}$ $\pi^*$ Mix | $\pi_{\text{unif}}$ $\pi^*$ Mix |
| Regress| 41.7 36.4 25.9                    | 0.506 13.3 0.126                | 4.75 40.7 1.38                 |
| Pref   | 50.2 54.3 32.7                    | 2.80 159 1.76                   | 150 179 8.11                   |
| ARL SO | 484 599 393                       | 673 2640 417                    | 446 380 232                    |
| ARL SA | 544 614 388                       | 804 1630 370                    | 505 465 206                    |

(b) Mean approximate distance $D$. Results are the same as Table 2.

| Reward | $1000 \times D^{\text{EPIC}}$ | $1000 \times D^{\text{NPEC}}$ | $1000 \times D^{\text{ERC}}$ |
|--------|--------------------------------|--------------------------------|--------------------------------|
| Model  | $\pi_{\text{unif}}$ $\pi^*$ Mix | $\pi_{\text{unif}}$ $\pi^*$ Mix | $\pi_{\text{unif}}$ $\pi^*$ Mix |
| Regress| 41.9 36.5 25.9                    | 0.519 14.9 0.140                | 4.78 40.9 1.39                 |
| Pref   | 50.5 54.4 32.9                    | 2.99 204 1.78                   | 150 180 8.15                   |
| ARL SO | 488 600 395                       | 684 3550 426                    | 448 382 234                    |
| ARL SA | 548 614 390                       | 823 3030 376                    | 506 467 208                    |

(c) 95% upper bound $D_{\text{UP}}$ of approximate distance.

| Reward | $1000 \times D_{\text{UP}}^{\text{EPIC}}$ | $1000 \times D_{\text{UP}}^{\text{NPEC}}$ | $1000 \times D_{\text{UP}}^{\text{ERC}}$ |
|--------|---------------------------------|---------------------------------|---------------------------------|
| Model  | $\pi_{\text{unif}}$ $\pi^*$ Mix | $\pi_{\text{unif}}$ $\pi^*$ Mix | $\pi_{\text{unif}}$ $\pi^*$ Mix |
| Regress| 42.2 36.5 26.0                    | 0.535 16.8 0.162                | 4.80 41.1 1.40                 |
| Pref   | 50.9 54.5 33                      | 3.16 240 1.80                   | 15.1 181 8.19                  |
| ARL SO | 492 601 397                       | 694 4420 436                    | 450 384 235                    |
| ARL SA | 552 614 392                       | 848 4660 385                    | 508 469 209                    |

(d) Relative 95% confidence interval $D_{\text{REL}} = \max \left( \frac{\text{Upper Mean} - \text{Lower Mean}}{\text{Mean}} - 1, 1 - \frac{\text{Lower Mean}}{\text{Mean}} \right)$ in percent. The population mean is contained within $\pm D_{\text{REL}}\%$ of the sample mean in Table A.6b with 95% probability.

| Reward | $D_{\text{REL}}^{\text{EPIC}}\%$ | $D_{\text{REL}}^{\text{NPEC}}\%$ | $D_{\text{REL}}^{\text{ERC}}\%$ |
|--------|---------------------------------|---------------------------------|---------------------------------|
| Model  | $\pi_{\text{unif}}$ $\pi^*$ Mix | $\pi_{\text{unif}}$ $\pi^*$ Mix | $\pi_{\text{unif}}$ $\pi^*$ Mix |
| Regress| 0.662 0.0950 0.352               | 3.04 12.9 16.0                 | 0.589 0.544 0.620               |
| Pref   | 0.683 0.158 0.411                | 6.31 21.8 1.41                 | 0.499 0.538 0.481               |
| ARL SO | 0.875 0.115 0.522                | 1.60 25.8 2.37                 | 0.449 0.504 0.621               |
| ARL SA | 0.654 0.0331 0.397               | 3.15 53.9 2.34                 | 0.382 0.445 0.540               |
A.3 Proofs

A.3.1 Background

**Proposition 3.5.** The binary relation $\equiv$ is an equivalence relation. Let $R_A, R_B, R_C : S \times A \times S \to \mathbb{R}$ be bounded reward functions. Then $\equiv$ is reflexive, $R_A \equiv R_A$; symmetric, $R_A \equiv R_B$ implies $R_B \equiv R_A$; and transitive, $(R_A \equiv R_B) \wedge (R_B \equiv R_C)$ implies $R_A \equiv R_C$.

**Proof.** $R_A \equiv R_A$ since $R_A(s, a, s') = \lambda R_A(s, a, s') + \gamma \Phi(s') - \Phi(s)$ for all $s, s' \in S$ and $a \in A$ for $\lambda = 1 > 0$ and $\Phi(s) = 0$, a bounded potential function.

Suppose $R_A \equiv R_B$. Then there exists some $\lambda > 0$ and a bounded potential function $\Phi : S \to \mathbb{R}$ such that $R_B(s, a, s') = \lambda R_A(s, a, s') + \gamma \Phi(s') - \Phi(s)$ for all $s, s' \in S$ and $a \in A$. Rearranging, we have:

$$R_A(s, a, s') = \frac{1}{\lambda} R_B(s, a, s') + \gamma \left( \frac{-1}{\lambda} \Phi(s') \right) - \left( \frac{-1}{\lambda} \Phi(s) \right).$$

Since $\frac{1}{\lambda} > 0$ and $\Phi'(s) = \frac{-1}{\lambda} \Phi(s)$ is a bounded potential function, it follows that $R_B \equiv R_A$.

Finally, suppose $R_A \equiv R_B$ and $R_B \equiv R_C$. Then there exists some $\lambda_1, \lambda_2 > 0$ and bounded potential functions $\Phi_1, \Phi_2 : S \to \mathbb{R}$ such that for all $s, s' \in S$ and $a \in A$:

$$R_B(s, a, s') = \lambda_1 R_A(s, a, s') + \gamma \Phi_1(s') - \Phi_1(s)$$
$$R_C(s, a, s') = \lambda_2 R_B(s, a, s') + \gamma \Phi_2(s') - \Phi_2(s)$$

Substituting the expression for $R_B$ into the expression for $R_C$:

$$R_C(s, a, s') = \lambda_2 (\lambda_1 R_A(s, a, s') + \gamma \Phi_1(s') - \Phi_1(s)) + \gamma \Phi_2(s') - \Phi_2(s)$$
$$= \lambda_1 \lambda_2 R_A(s, a, s') + \gamma (\lambda_2 \Phi_1(s') + \Phi_2(s')) - (\lambda_2 \Phi_1(s) + \Phi_2(s))$$
$$= \lambda R_A(s, a, s') + \gamma \Phi(s') - \Phi(s),$$

where $\lambda = \lambda_1 \lambda_2 > 0$ and $\Phi(s) = \lambda_2 \Phi_1(s) + \Phi_2(s)$ is bounded. Thus $R_A \equiv R_C$.  

\[\square\]

A.3.2 Equivalent-Policy Invariant Comparison (EPIC) pseudometric

**Proposition 4.2** (The Canonically Shaped Reward is Invariant to Shaping). Let $R : S \times A \times S \to \mathbb{R}$ be a reward function and $\Phi : S \to \mathbb{R}$ a potential function. Let $\gamma \in [0, 1]$ be a discount rate, and $D_S$ and $D_A$ be distributions over states $S$ and $A$ respectively. Let $R'$ denote $R$ shaped by $\Phi$: $R'(s, a, s') = R(s, a, s') + \gamma \Phi(s') - \Phi(s)$. Then the canonically shaped $R'$ and $R$ are equal: $C_{D_S, D_A}(R') = C_{D_S, D_A}(R)$.

**Proof.** Let $s, a, s' \in S \times A \times S$. Then by substituting in the definition of $R'$ and using linearity of expectations:

$$C_{D_S, D_A}(R') (s, a, s') \triangleq R'(s, a, s') + \mathbb{E} [\gamma R'(s', a, S') - R'(s, a, S') - \gamma R(S, A, S')]$$
$$= (R(s, a, s') + \gamma \Phi(s') - \Phi(s)) + \mathbb{E} [\gamma R(s', a, S') + \gamma^2 \Phi(S') - R(s, a, S')]$$
$$- \mathbb{E} [R(s, A, S') + \gamma \Phi(S') - \Phi(s)] - \mathbb{E} [\gamma R(S, A, S') + \gamma^2 \Phi(S') - \gamma \Phi(S)]$$
$$= R(s, a, s') + \mathbb{E} [\gamma R(s', a, S') - R(s, a, S') - \gamma R(S, A, S')]$$
$$+ (\gamma \Phi(s') - \Phi(s)) + \mathbb{E} [\Phi(s) - \gamma \Phi(s')]$$
$$+ \mathbb{E} [\gamma^2 \Phi(S') - \gamma \Phi(S')] - \mathbb{E} [\gamma^2 \Phi(S) - \gamma \Phi(S)]$$
$$= R(s, a, s') + \mathbb{E} [\gamma R(s', a, S') - R(s, A, S') - \gamma R(S, A, S')]$$
$$\triangleq C_{D_S, D_A}(R) (s, a, s'),$$

where the penultimate step uses $\mathbb{E} [\Phi(S')] = \mathbb{E} [\Phi(S)]$ since $S$ and $S'$ are identically distributed. \[\square\]

**Lemma 4.5.** Let $a, b \in (0, \infty)$, $c, d \in \mathbb{R}$ and $X, Y$ be random variables. Then it follows that $0 \leq D_p(aX + c, bY + d) = D_p(X, Y) \leq 1$. 

21
Proof. $D_\rho(aX + c, bY + d) = D_\rho(X, Y)$ immediate from $\rho(X, Y)$ invariant to positive affine transformations. Have $-1 \leq \rho(X, Y) \leq 1$, so $0 \leq 1 - \rho(X, Y) \leq 2$ thus $0 \leq D_\rho(X, Y) \leq 1$. □

Lemma 4.4. The Pearson distance $D_\rho$ is a pseudometric.

Proof. For a random variable $V$, define a standardized (zero mean and variance) version:

$$\hat{V} = \frac{V - \mathbb{E}[V]}{\sqrt{\mathbb{E}[(V - \mathbb{E}[V])^2]}}.$$ 

The Pearson correlation coefficient on random variables $X$ and $Y$ is equal to the expected product of these standardized random variables:

$$\rho(X, Y) = \mathbb{E}[\hat{X}\hat{Y}].$$

Let $X$, $Y$ and $Z$ be random variables.

Identity. Have $\rho(X, X) = 1$, so $D_\rho(X, X) = 0$.

Symmetry. Have $\rho(X, Y) = \rho(Y, X)$ by commutativity of multiplication, so $D_\rho(X, Y) = D_\rho(Y, X)$.

Triangle Inequality. For any random variables $A, B$:

$$\mathbb{E}[(\hat{A} - \hat{B})^2] = \mathbb{E}[(\hat{A}^2 - 2\hat{A}\hat{B} + \hat{B}^2) = \mathbb{E}[\hat{A}^2] - 2\mathbb{E}[\hat{A}\hat{B}] + \mathbb{E}[\hat{B}^2] = 2 - 2\mathbb{E}[\hat{A}\hat{B}] = 2 (1 - \rho(A, B)) = 4D_\rho(A, B)^2.$$ So:

$$4D_\rho(X, Z)^2 = \mathbb{E}[(\hat{X} - \hat{Z})^2] = \mathbb{E}[\hat{X} - \hat{Y} + \hat{Y} - \hat{Z})^2] = \mathbb{E}[\hat{X} - \hat{Y})^2] + \mathbb{E}[\hat{Y} - \hat{Z})^2] + 2\mathbb{E}[(\hat{X} - \hat{Y}) (\hat{Y} - \hat{Z})] = 4D_\rho(X, Y)^2 + 4D_\rho(Y, Z)^2 + 8\mathbb{E}[(\hat{X} - \hat{Y}) (\hat{Y} - \hat{Z})].$$

Since $\langle A, B \rangle = \mathbb{E}[AB]$ is an inner product over $\mathbb{R}$, it follows by the Cauchy-Schwarz inequality that $\mathbb{E}[AB] \leq \sqrt{\mathbb{E}[A^2]\mathbb{E}[B^2]}$. So:

$$D_\rho(X, Z)^2 \leq D_\rho(X, Y)^2 + D_\rho(Y, Z)^2 + 2D_\rho(X, Y)D_\rho(Y, Z) = (D_\rho(X, Y) + D_\rho(Y, Z))^2.$$ Taking the square root of both sides:

$$D_\rho(X, Z) \leq D_\rho(X, Y) + D_\rho(Y, Z),$$ as required. □

Thorem 4.8. Let $R_A, R'_A, R_B, R'_B : S \times A \times S \to \mathbb{R}$ be reward functions such that $R'_A \equiv R_A$ and $R'_B \equiv R_B$. Then $0 \leq D_{\text{EPI}_C}(R'_A, R'_B) = D_{\text{EPI}_C}(R_A, R_B) \leq 1$. 

22
Proof. The result follows from $D_{\rho}$ being a pseudometric. Let $R_A$, $R_B$ and $R_C$ be reward functions mapping from transitions $S \times A \times S$ to real numbers $\mathbb{R}$.

Identity. Have:
\[
D_{\text{EPIC}}(R_A,R_A) = D_{\rho}(C_{D_S,D_A}(R_A)(S,A,S'), C_{D_S,D_A}(R_A)(S,A,S'))
= 0,
\]
since $D_{\rho}(X,X) = 0$.

Symmetry. Have:
\[
D_{\text{EPIC}}(R_A,R_B) = D_{\rho}(C_{D_S,D_A}(R_A)(S,A,S'), C_{D_S,D_A}(R_B)(S,A,S'))
= D_{\rho}(C_{D_S,D_A}(R_B)(S,A,S'), C_{D_S,D_A}(R_A)(S,A,S'))
= D_{\text{EPIC}}(R_B,R_A),
\]
since $D_{\rho}(X,Y) = D_{\rho}(Y,X)$.

Triangle Inequality. Have:
\[
D_{\text{EPIC}}(R_A,R_C) = D_{\rho}(C_{D_S,D_A}(R_A)(S,A,S'), C_{D_S,D_A}(R_C)(S,A,S'))
\leq D_{\rho}(C_{D_S,D_A}(R_A)(S,A,S'), C_{D_S,D_A}(R_B)(S,A,S'))
+ D_{\rho}(C_{D_S,D_A}(R_B)(S,A,S'), C_{D_S,D_A}(R_C)(S,A,S'))
= D_{\text{EPIC}}(R_A,R_B) + D_{\text{EPIC}}(R_B,R_C),
\]
since $D_{\rho}(X,Z) \leq D_{\rho}(X,Y) + D_{\rho}(Y,Z)$.

A.3.3 Nearest Point in Equivalence Class (NPEC) premetric

Proposition 5.3. (1) $D_{L^p}$ is a metric in $L^p$ space, where functions $f$ and $g$ are identified if $f = g$ almost everywhere on $\mathcal{D}$. (2) It is a pseudometric when $f$ and $g$ are identified if $f = g$ at all points.

Proof. (1) $D_{L^p}$ is a metric in the $L^p$ space since $L^p$ is a norm in the $L^p$ space, and $d(x,y) = \|x - y\|$ is always a metric. (2) As $f = g$ at all points implies $f = g$ almost everywhere, certainly $D_{L^p}(R,R) = 0$. Symmetry and triangle inequality do not depend on identity so still hold.

Proposition 5.5. $D_{\text{NPEC}}$ is a premetric.

Proof. Let $R_A, R_B$ be bounded reward functions on $S \times A \times S \to \mathbb{R}$.

Respects identity: $D_{\text{NPEC}}(R_A,R_A) = 0$.

If $D^U_{\text{NPEC}}(\text{Zero}, R_A) = 0$ then $D_{\text{NPEC}}(R_A, R_A) = 0$ as required. Suppose from now on that $D_{\text{NPEC}}(R_A, R_A) \neq 0$. It follows from prop 5.3 that $D_{L^p}(R_A, R_A) = 0$. Since $X \equiv X$, 0 is an upper bound for $D^U_{\text{NPEC}}(R_A, R_A)$. By prop 5.3, $D_{L^p}$ is non-negative, so this is also a lower bound for $D^U_{\text{NPEC}}(R_A, R_A).$ So $D^U_{\text{NPEC}}(R_A, R_A) = 0$ and:
\[
D_{\text{NPEC}}(R_A, R_A) = \frac{D^U_{\text{NPEC}}(R_A, R_A)}{D^U_{\text{NPEC}}(\text{Zero}, R_A)} = \frac{0}{D^U_{\text{NPEC}}(\text{Zero}, R_A)} = 0.
\]

Well-defined: $D_{\text{NPEC}}(R_A, R_B) \geq 0$

By prop 5.3, it follows that $D_{L^p}(R, R_B) \geq 0$ for all reward functions $R : S \times A \times S$. Thus 0 is a lower bound for $\{D_{L^p}(R, R_B) : R : S \times A \times S\}$, and thus certainly a lower bound for $\{D_{L^p}(R, Y) : R \equiv X\}$ for any reward function $X$. Since the infimum is the largest lower bound, it follows that for any reward function $X$:
\[
D^U_{\text{NPEC}}(X, R_B) \triangleq \inf_{R \equiv X} D_{L^p}(R, R_B) \geq 0.
\]

In the case that $D^U_{\text{NPEC}}(\text{Zero}, R_B) = 0$, then $D_{\text{NPEC}}(R_A, R_B) = 0$ which is non-negative. From now on, suppose that $D^U_{\text{NPEC}}(\text{Zero}, R_B) \neq 0$. The quotient of a non-negative value with a positive value is non-negative, so:
\[
D_{\text{NPEC}}(R_A, R_B) = \frac{D^U_{\text{NPEC}}(R_A, R_B)}{D^U_{\text{NPEC}}(\text{Zero}, R_B)} \geq 0.
\]
Note when $D_{L^p}$ is a metric, then $D_{NPEC}(X, Y) = 0$ if and only if $X = Y$.

**Proposition A.1.** $D_{NPEC}$ is not symmetric in the undiscounted case.

**Proof.** We will provide a counterexample showing that $D_{NPEC}$ is not symmetric.

Choose the state space $S$ to be binary $\{0, 1\}$ and the actions $A$ to be the singleton $\{0\}$. Choose the visitation distribution $D$ to be uniform on $s \to s$ for $s \in S$. Take $\gamma = 1$, i.e. undiscounted. Note that as the successor state is always the same as the start state, potential shaping has no effect on $D_{direct}$, so WLOG we will assume potential shaping is always zero.

Now, take $R_A(s) = 2s$ and $R_B(s) = 1$. Take $p = 1$ for the $L^p$ distance. Observe that $D_{L^p}(Zero, R_A) = \frac{1}{2} (|0| + |2|) = 1$ and $D_{L^p}(Zero, R_B) = \frac{1}{2} (|1| + |1|) = 1$. Since potential shaping has no effect, $D_{UNPEC}(Zero, R_A) = D_{L^p}(Zero, R_A)$ and so $D(Zero, R_A) = 1$ and $D(Zero, R_B) = 1$.

Now:

$$D_{UNPEC}(R_A, R_B) = \inf_{\lambda > 0} D_{L^p}(\lambda R_A, R_B)$$

$$= \inf_{\lambda > 0} \frac{1}{2} (|1| + |2\lambda - 1|)$$

$$= \frac{1}{2}$$

with the infimum attained at $\lambda = \frac{1}{2}$. But:

$$D_{UNPEC}(R_B, R_A) = \inf_{\lambda > 0} D_{L^p} (\lambda R_B, R_A)$$

$$= \inf_{\lambda > 0} \frac{1}{2} f(\lambda)$$

$$= \frac{1}{2} \inf_{\lambda > 0} f(\lambda),$$

where:

$$f(\lambda) = |\lambda| + |2 - \lambda|, \quad \lambda > 0.$$ 

Note that:

$$f(\lambda) = \begin{cases} 
2 & \lambda \in (0, 2], \\
2\lambda - 2 & \lambda \in (2, \infty). 
\end{cases}$$

So $f(\lambda) \geq 2$ on all of its domain, thus:

$$D_{UNPEC}(R_B, R_A) = 1.$$ 

Consequently:

$$D_{NPEC}(R_A, R_B) = \frac{1}{2} \neq 1 = D_{NPEC}(R_B, R_A).$$

**Proposition A.2** (Properties of $D_{NPEC}^U$). Let $R_A, R_B : S \times A \times S \to \mathbb{R}$ be bounded reward functions, and $\lambda \geq 0$. Then $D_{NPEC}^U$:

- **Is invariant under $\equiv$ in source:**
  $$D_{NPEC}^U(R_A, R_B) = D_{NPEC}^U(R_B, R_B)$$ if $R_A \equiv R_B$.

- **Invariant under scale-preserving $\equiv$ in target:**
  $$D_{NPEC}^U(R_A, R_A) = D_{NPEC}^U(R_A, R_B)$$ if $R_A - R_B \equiv Zero$.

- **Scalable in target:**
  $$D_{NPEC}^U(R_A, \lambda R_B) = \lambda D_{NPEC}^U(R_A, R_B).$$

- **Bounded:**
  $$D_{NPEC}^U(R, R_B) \leq D_{NPEC}^U(Zero, R_B).$$
Proof. We will show each case in turn.

Invariance under \( \equiv \) in source

If \( R_A \equiv R_B \), then:

\[
D_{NPEC}^U(R_A, R_B) \equiv \inf_{R \equiv R_A} D_{Lp}(R, R_B)
\]

\[
= \inf_{R \equiv R_B} D_{Lp}(R, R_B)
\]

\[
\equiv D_{NPEC}^U(R_B, R_B),
\]

since \( R \equiv R_A \) if and only if \( R \equiv R_B \) as \( \equiv \) is an equivalence relation.

Invariance under scale-preserving \( \equiv \) in target

If \( R_A-R_B \equiv \text{Zero} \), then we can write \( R_A(s, a, s') - R_B(s, a, s') = \gamma \Phi(s') - \Phi(s) \) for some potential function \( \Phi : S \to \mathbb{R} \). Then for any reward function \( R \), since \( D \) is induced by a norm:

\[
D_{Lp}(R, R_A) \equiv \mathbb{E}_{s, a, s' \sim \mathcal{D}} [D(R(s, a, s'), R_A(s, a, s'))]
\]

\[
= \mathbb{E}_{s, a, s' \sim \mathcal{D}} ||R(s, a, s') - R_A(s, a, s')||
\]

\[
= \mathbb{E}_{s, a, s' \sim \mathcal{D}} [(R(s, a, s') - \gamma \Phi(s') + \Phi(s)) - R_B(s, a, s')] ||
\]

\[
= \mathbb{E}_{s, a, s' \sim \mathcal{D}} [D(R(s, a, s') - \gamma \Phi(s') + \Phi(s), R_B(s, a, s'))]
\]

\[
\equiv D_{Lp}(f(R), R_B),
\]

where \( f(R)(s, a, s') = R(s, a, s') - \gamma \Phi(s') + \Phi(s) \). Crucially, note \( f(R) \) is a bijection on the equivalence class \( [R] \). Now, substituting this into the expression for NPEC premetric:

\[
D_{NPEC}^U(R_A, R_A) \equiv \inf_{R \equiv R_A} D_{Lp}(R, R_A)
\]

\[
= \inf_{R \equiv R_B} f_{R \equiv R_A}(R) = \inf_{R \equiv R_B} f_{R \equiv R_A}(R) = f_{R \equiv R_A}(R) = f_{R \equiv R_B}(R)
\]

\[
\equiv D_{NPEC}^U(R_A, R_B).
\]

Scalable in target First, note that \( D_{Lp} \) is absolutely scalable in both arguments:

\[
D_{Lp}(\lambda R_A, \lambda R_B) \equiv \mathbb{E}_{s, a, s' \sim \mathcal{D}} [D(\lambda R_A(s, a, s'), \lambda R_B(s, a, s'))]
\]

\[
= \mathbb{E}_{s, a, s' \sim \mathcal{D}} ||\lambda R_A(s, a, s') - \lambda R_B(s, a, s')||
\]

\[
= \mathbb{E}_{s, a, s' \sim \mathcal{D}} [||R_A(s, a, s') - R_B(s, a, s')||] \quad ||\text{absolutely scalable} \]

\[
= \lambda \mathbb{E}_{s, a, s' \sim \mathcal{D}} [||R_A(s, a, s') - R_B(s, a, s')||]
\]

\[
\equiv \lambda D_{Lp}(R_A, R_B).
\]

Now, for \( \lambda > 0 \), applying this to NPEC premetric:

\[
D_{NPEC}^U(R_A, \lambda R_B) \equiv \inf_{R \equiv R_A} D_{Lp}(R, \lambda R_B)
\]

\[
= \inf_{R \equiv R_A} \lambda D_{Lp}(R, R_B)
\]

\[
= \lambda \inf_{R \equiv R_A} D_{Lp}(R, R_B)
\]

\[
\equiv \lambda D_{NPEC}^U(R_A, R_B).
\]
In the case $\lambda = 0$, then:

$$D_{\text{NPEC}}^U(R_A, 0) \triangleq \inf_{R \equiv R_A} D_{L^p}(R, 0)$$

$$= \inf_{R \equiv R_A} \frac{1}{2} D_{L^p}(R, 0)$$

$$= \frac{1}{2} \inf_{R \equiv R_A} D_{L^p}(R, 0)$$

$$= \frac{1}{2} D_{\text{NPEC}}(R_A, 0).$$

Rearranging, we have:

$$D_{\text{NPEC}}^U(R_A, 0) = 0.$$  

**Boundedness**

Suppose $R_A$ is bounded by $B$: $|R_A(s, a, s')| \leq B$ for all $s, s' \in S$ and $a \in A$. Suppose the NPEC premetric $D_{\text{NPEC}}(0, R_B) = d$. Then for any $\epsilon > 0$, there exists some potential function $\Phi : S \rightarrow \mathbb{R}$ such that the $L^p$ of the potential shaping $R(s, a, s') \triangleq \gamma \Phi(s) - \Phi(s)$ from $R_B$ satisfies:

$$D_{L^p}(R, R_B) \leq d + \epsilon. \quad (7)$$

Let $\lambda \in [0, 1]$. Define:

$$R'_\lambda(s, a, s') \triangleq \lambda R_A(s, a, s') + R(s, a, s'),$$

and:

$$f_\lambda(s, a, s') = D(R'_\lambda(s, a, s'), R(s, a, s')).$$

Note that:

$$\lim_{\lambda \to 0^+} R'_\lambda = R_0' = R \text{ pointwise,}$$

and $R_0' = R$. Since $D$ is a metric it is continuous, and so:

$$\lim_{\lambda \to 0^+} f_\lambda = f_0 \text{ pointwise.}$$

Moreover, $f_0(s, a, s') = 0$ everywhere since $D(x, x) = 0$. Now:

$$|f_\lambda(s, a, s')| = D(R'_\lambda(s, a, s'), R(s, a, s'))$$

$$= ||R'_\lambda(s, a, s') - R(s, a, s')|| \\leq \lambda B.$$  

It follows by the bounded convergence theorem that:

$$\lim_{\lambda \to 0^+} D_{L^p}(R'_\lambda, R) = \lim_{\lambda \to 0^+} \mathbb{E}_{s, a, s' \sim D} [f_\lambda(s, a, s')]$$

$$= \mathbb{E}_{s, a, s' \sim D} \left[ \lim_{\lambda \to 0^+} f_\lambda(s, a, s') \right]$$

$$= \mathbb{E}_{s, a, s' \sim D} [f_0(s, a, s')]$$

$$= 0.$$  

So in particular, for any $\epsilon > 0$ there exists some $\lambda > 0$ such that:

$$D_{L^p}(R'_\lambda, R) \leq \epsilon. \quad (8)$$

Note that $R_A \equiv R'_\lambda$ for all $\lambda > 0$. So:

$$D_{\text{NPEC}}(R_A, R_B) \leq D_{L^p}(R'_\lambda, R_B) \\leq D_{L^p}(R'_\lambda, R) + D_{L^p}(R, R_B) \text{ prop. 5.3} \leq \epsilon + (d + \epsilon) \text{ eq. 7 and eq. 8} = d + 2\epsilon.$$
Then the normalized reward is:

\[ R^N(s, a, s') = \frac{R(s, a, s')}{\|R\|} \]

Note that \((\lambda R)^N = R^N\) for any \(\lambda > 0\) as norms are absolutely homogeneous.

We say a reward is **standardized** if it has been canonicalized and then normalized.

**Definition A.4** (Standardized Reward). Let \( R \) be a reward function mapping from transitions \( S \times A \times S \) to real numbers \( \mathbb{R} \). Then the standardized reward is:

\[ R^S = (C_{D_S, D_A}(R))^N. \]

Now, we can define a pseudometric based on the direct distance between the standardized rewards.

**Definition A.5** (Direct Distance Standardized Reward). Let \( D \) be some visitation distribution over transitions \( s \xrightarrow{a} s' \). Let \( D_S \) and \( D_A \) be some distributions over states \( S \) and \( A \) respectively. Let

\[ D_{NPEC}(R_A, R_B) \leq d. \]
\[ D_{\text{DDSR}}(R_A, R_B) = \frac{1}{2} D_{L^p} \left( R_A^S(S, A, S'), R_B^S(S, A, S') \right), \]

where the norm used for direct distance is the same norm used for normalization in \( R^N \).

For brevity, we omit the proof that \( D_{\text{DDSR}} \) is a pseudometric, but this follows from \( D_{L^p} \) being a pseudometric in a similar fashion to theorem 4.7. Note it additionally is invariant to equivalence classes, similarly to EPIC.

**Theorem A.6.** Let \( R_A, R_A', R_B, R_B' \) be reward functions mapping from transitions \( S \times A \times S \) to real numbers \( \mathbb{R} \) such that \( R_A \equiv R_A' \) and \( R_B \equiv R_B' \). Then:

\[ 0 \leq D_{\text{DDSR}}(R_A', R_B') = D_{\text{DDSR}}(R_A, R_B) \leq 1. \]

**Proof.** The invariance under the equivalence class follows from \( R^S \) being invariant to potential shaping and scale in \( R \). The non-negativity follows from \( D_{L^p} \) being a pseudometric. The upper bound follows from the rewards being normalized to norm 1 and the triangle inequality:

\[
D_{\text{DDSR}}(R_A, R_B) = \frac{1}{2} \| R_A^S - R_B^S \| \\
\leq \frac{1}{2} (\| R_A^S \| + \| R_B^S \|) \\
= \frac{1}{2} (1 + 1) \\
= 1. 
\]

Since both DDSR and EPIC are pseudometrics and invariant on equivalent rewards, it is interesting to consider the connection between them. In fact, under the \( L^2 \) norm with \( D \) chosen to be i.i.d. samples from the joint distribution \( D_S \times D_A \times D_S \), then DDSR recovers EPIC. First, we will show that canonical shaping centers the reward functions.

**Lemma A.7 (The Canonically Shaped Reward is Mean Zero).** Let \( R \) be a reward function mapping from transitions \( S \times A \times S \) to real numbers \( \mathbb{R} \). Then:

\[
E \left[ C_{D_S, D_A} (R) (S, A, S') \right] = 0. 
\]

**Proof.** Let \( X, U \) and \( X' \) be random variables that are independent of \( S, A \) and \( S' \) but identically distributed.

\[
E \left[ C_{D_S, D_A} (R) (S, A, S') \right] = E \left[ R(S, A, S') + \gamma R(S', U, X') - R(S, U, X') - \gamma R(X, U, X') \right] \\
= E \left[ R(S, A, S') \right] + \gamma E \left[ R(S', U, X') \right] - E \left[ R(S, U, X') \right] - \gamma E \left[ R(X, U, X') \right] \\
= E \left[ R(S, U, X') \right] + \gamma E \left[ R(X, U, X') \right] - E \left[ R(S, U, X') \right] - \gamma E \left[ R(X, U, X') \right] \\
= 0,
\]

where the penultimate step follows since \( A \) is identically distributed to \( U \), and \( S' \) is identically distributed to \( X' \) and therefore to \( X \).

Recall from the proof of lemma 4.4 that:

\[
D_p(U, V) = \frac{1}{2} \sqrt{E \left[ (\hat{U} - \hat{V})^2 \right]} \\
= \frac{1}{2} \| \hat{U} - \hat{V} \|_2,
\]

where \( \| \cdot \|_2 \) is the \( L^2 \) norm (treating the random variables as functions on a measure space) and \( \hat{U} \) is a centered (zero-mean) and rescaled (unit variance) random variable. By lemma A.7, the canonically
shaped reward functions are already centered under the joint distribution $\mathcal{D}_S \times \mathcal{D}_A \times \mathcal{D}_S$, and normalization by the $L^2$ norm also ensures they have unit variance. Consequently:

$$D_{\text{EPIC}}(R_A, R_B) = D_\rho \left( C_{\mathcal{D}_S, \mathcal{D}_A} (R_A) (S, A, S'), C_{\mathcal{D}_S, \mathcal{D}_A} (R_B) (S, A, S') \right)$$

$$= \frac{1}{2} \left\| C_{\mathcal{D}_S, \mathcal{D}_A} (R_A)(S, A, S') - C_{\mathcal{D}_S, \mathcal{D}_A} (R_B)(S, A, S') \right\|_2$$

$$= \frac{1}{2} \left\| R_A^S(S, A, S') - R_B^S(S, A, S') \right\|_2$$

$$= \frac{1}{2} D_{L^p} (R_A^S(S, A, S'), R_B^S(S, A, S'))$$

$$= D_{\text{DDSR}}(R_A, R_B).$$