WHEN SHOULD AN AGENT TRUST A REWARD? TRAJECTORY-SUPPORTED LEARNING UNDER CORRUPTED REWARD OBSERVATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning treats observed rewards as supervision, yet reward observations can be corrupted by sensing, transmission, or learned evaluation. Without structural assumptions, trajectory data cannot distinguish corruption from rare but valid rewards. We study sparse or transient channel corruption whose perturbations are weakly context-coupled relative to genuine reward-bearing events. RECONRL estimates per-sample reward trust from *conditional reward–trajectory association rather than reward abnormality*. A separated temporal view and a local transition view perform critic-output-independent joint-versus-product discrimination, and a monotone gate converts their evidence into a weight on the backbone-native critic loss without reconstructing the reward. Matched diagnostics hold reward magnitude and frequency fixed while changing trajectory support. Paired continuous and discrete backbones, corruption-severity sweeps, mechanism OOD, and held-out-task transfer then test whether this sample-level trust signal preserves genuine supervision and improves control.

Keywords: reinforcement learning; reward corruption; robust reinforcement learning; trajectory consistency; reliable learning.

1 INTRODUCTION

Reinforcement learning (RL) turns scalar feedback into behavioral supervision, so one incorrect reward observation can influence many later value updates. Simulator rewards are usually read directly from program state (Sutton and Barto, 2018); other pipelines can mediate feedback through sensors, perception modules, human-preference models, or learned evaluators (Everitt et al., 2017; Christiano et al., 2017; Chen et al., 2025; Nguyen et al., 2024). Robust reward-modeling work shows that learned evaluators can also encode spurious or unstable signals (Liu et al., 2025a; Srivastava et al., 2026). The resulting sample-level question is: how strongly should a particular observed reward influence learning when the reward channel can fail?

Consider two observations of the same rare $+R$ reward. In one trajectory, the agent reaches a distinctive goal state and separated context is consistent with success; in the other, the same reward value appears on an ordinary transition because the channel is corrupted. Magnitude and marginal frequency are identical, but the trajectory evidence is not. This yields the central distinction: *rarity is not unreliability*. Reward trust should depend on conditional trajectory support rather than unusualness alone.

Existing work attacks corrupted feedback from several directions. Corrupt-reward MDPs formalize a gap between task reward and observed reward (Everitt et al., 2017; 2019); robust Q-learning and robust policy-gradient methods tolerate adversarial contamination using aggregate robust estimation or filtering (Maity and Mitra, 2024; Zhang et al., 2021); reward-poisoning defenses reason about adaptive attackers (Nika et al., 2023); and recent offline or preference-based methods address contaminated feedback datasets (Mandal et al., 2025; Nika et al., 2026; Tran et al., 2026). Their primary objects are robust estimators, policies, or uncertainty sets; RECONRL instead assigns a trajectory-supported trust weight to each observed reward before that sample shapes value learning.

A second line studies the reliability of learned rewards. Calibration, robust training, proxy structure, and causal objectives reduce systematic evaluator error (Huang et al., 2025; Hong et al., 2025; Laidlaw et al., 2025; Srivastava et al., 2026). RewardBench-style evaluation and reward-model audits reveal failures that coarse aggregate scores can miss (Lambert et al., 2025; Liu et al., 2025b; Malik et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2026). Existing approaches typically decide which estimator, policy, or reward model should be trusted. Our setting asks a finer-grained question: given one observed reward already stored in replay, how much should that transition contribute to value learning?

We estimate that contribution through conditional association rather than reward abnormality. A temporal path removes a guard interval around the candidate reward and tests whether separated past and future context supports the value; a transition path tests compatibility with (s_t, a_t, s_{t+1}). Each path uses joint-versus-product discrimination, and a monotone fusion rule converts the resulting evidence into a trust weight w_t. The observed reward stays in the backbone's Bellman target; only the per-transition critic loss is reweighted. Future context is used after it has been observed and stored in replay, while data collection remains online.

We first test whether the trust score separates equally rare valid and corrupted rewards, then test whether the same weighting rule improves control when inserted into different RL backbones. Severity sweeps vary the corruption fraction, mechanism OOD freezes support learned from impulse and short bursts, and held-out-task transfer retrains support parameters under a fixed method design. Clean-transition trust is reported together with corruption rejection and return.

Our contributions are threefold.

1. **We formulate reward reliability as conditional trajectory support under explicit non-identifiability.** We distinguish unrestricted corruption, which trajectories cannot identify, from sparse or transient perturbations that are weakly coupled to the evidence views used for trust estimation. This makes the target of inference a support weight, not an unobservable reconstructed reward.
2. **We estimate reward trust through conditional association rather than reward abnormality.** A separated temporal view and a local transition view implement this test through critic-output-independent joint-versus-product discrimination, then combine their evidence with a monotone fusion rule.
3. **We design matched diagnostics and paired-backbone experiments for the rarity-versus-support hypothesis.** The evaluation separately measures rare-valid preservation, corruption rejection, clean-transition trust, severity robustness, OOD transfer, and downstream control, rather than relying on average return alone.

2 Related Work

2.1 Corrupted Rewards and Robust RL

Corrupted reward channels differ from ordinary stochastic rewards because the observation mechanism itself can be wrong (Everitt et al., 2017; 2019). Robust MDP methods instead optimize against uncertainty sets over rewards or dynamics (Nilim and El Ghaoui, 2005; Wiesemann et al., 2013). More direct corruption work includes robust Q-learning (Maity and Mitra, 2024), filtered policy-gradient methods under strong data corruption (Zhang et al., 2021), online reward-poisoning defenses (Nika et al., 2023), and recent offline or preference-based robustification (Mandal et al., 2025; Nika et al., 2026; Tran et al., 2026). ReConRL intervenes at the reward-observation level inside deep replay: each sample receives a trajectory-supported trust weight learned from corrupted replay without clean rewards or corruption labels.

2.2 Reward Reliability and Evaluation

Reward-learning and reward-optimization work studies robust training (Liu et al., 2025a), correlated proxies (Laidlaw et al., 2025), calibration (Huang et al., 2025), causal reward structure (Srivastava et al., 2026), and pessimistic or partial-identifiability objectives (Yu et al., 2026; Lazzati and Metelli, 2026). Reward-model benchmarks increasingly emphasize structured diagnostics rather than a single average score (Lambert et al., 2025; Malik et al., 2026; Frick et al., 2025). Information-theoretic debiasing and inference-time reward diagnostics identify failures that are not reducible to score magnitude (Li et al., 2026a;b); causal preference learning and reward-hacking monitors test whether high reward is supported by the intended evidence (Hwang et al., 2026; Wang et al., 2026). In our setting the scalar environment reward is already present in replay. The decision is therefore made at transition level: how much that observed reward should contribute to value learning when its marginal rarity is held fixed.

3 PROBLEM FORMULATION

Let the environment generate a latent task reward r_t and the learner observe

$$\tilde{r}_t = r_t + c_t, \qquad z_t = \mathbf{1}[c_t \neq 0], \qquad \Pr(z_t = 1) \leq \epsilon, \tag{1}$$

where c_t is a sparse or transient channel perturbation and z_t is unavailable to training. The learner observes $(s_t, a_t, s_{t+1}, \tilde{r}_t)$ and replay context, but never r_t or z_t.

Proposition 1 (Unrestricted corruption is not identifiable)**.** *Without structural restrictions on c_t, there exist clean-reward and corruption processes that induce the same joint distribution over observed trajectories and rewards. No trajectory-only rule can distinguish them.*

Proof sketch. Fix any observed process over $(s_t, a_t, s_{t+1}, \tilde{r}_t)$. One compatible latent decomposition sets $r_t^{(1)} = \tilde{r}_t$ and $c_t^{(1)} = 0$. Choose any nonzero latent perturbation process Δ_t satisfying $\Pr(\Delta_t \neq 0) \leq \epsilon$. A second decomposition sets $r_t^{(2)} = \tilde{r}_t - \Delta_t$ and $c_t^{(2)} = \Delta_t$. Both satisfy $\tilde{r}_t = r_t^{(i)} + c_t^{(i)}$ and the sparsity bound in Eq. (1), and they induce exactly the same law over all observable trajectories, while their latent clean rewards and corruption indicators differ whenever $\Delta_t \neq 0$. Any estimator that depends only on observed trajectories must therefore return the same output under both decompositions and cannot identify which latent process generated the data. A formal version is given in Appendix A.1.

Within the task and replay-phase stratum c used for negative sampling, define two evidence views $v_t^{\text{temp}} = h_t^{\text{sep}}$ and $v_t^{\text{dyn}} = (s_t, a_t, s_{t+1})$. The modeled regime requires genuine reward-bearing events to exhibit positive conditional lift in at least one view, $p(r_t \mid v, c)/p(r_t \mid c) > 1$, while sparse or transient corruptions remain closer to context-mismatched draws. Here p denotes a density or probability mass function as appropriate. Corruptors that reproduce the clean reward–context joint are observationally indistinguishable under this formulation.

For an evidence view v and reward r, joint-versus-product discrimination within the same stratum has Bayes-optimal logit

$$\operatorname{logit} D^*(v, r \mid c) = \log p(r \mid v, c) - \log p(r \mid c). \tag{2}$$

We suppress c in later notation when it is unambiguous. This logit measures conditional lift relative to the replay-stratified reward marginal. Product-of-marginals pairs have expected likelihood ratio one and non-positive expected log-ratio. Because matched observations contain sparse corruption, trimmed positive losses estimate this lift while limiting contaminated positive examples.

4 METHOD

4.1 OVERVIEW

Figure 1 provides a conceptual view of RECONRL. Two separately parameterized, critic-output-independent paths score temporal and transition support. The transition panel illustrates local transition cues that may underlie reward compatibility; the implemented estimator directly encodes (s_t, a_t, s_{t+1}) through q_ϕ, without requiring a separate dynamics or uncertainty model. A monotone gate maps the two support scores to $w_t \in [w_{\min}, 1] \subseteq [0, 1]$, and w_t reweights only the backbone-native critic loss. The observed reward, Bellman target, and actor objective remain native to the backbone.

4.2 TEMPORALLY SEPARATED RETURN EVIDENCE

A TD-error filter reuses a critic trained by the reward it is asked to assess. The temporal path instead excludes a guard interval around the candidate and tests whether separated trajectory context supports the same reward value.

For guard size g, context horizon $H > g$, and discount factor $\gamma \in (0, 1)$, define

$$G_t^- = \sum_{j=g}^{H} \gamma^{j-g} \tilde{r}_{t-j}, \qquad G_t^+ = \sum_{j=g}^{H} \gamma^{j-g} \tilde{r}_{t+j}, \tag{3}$$

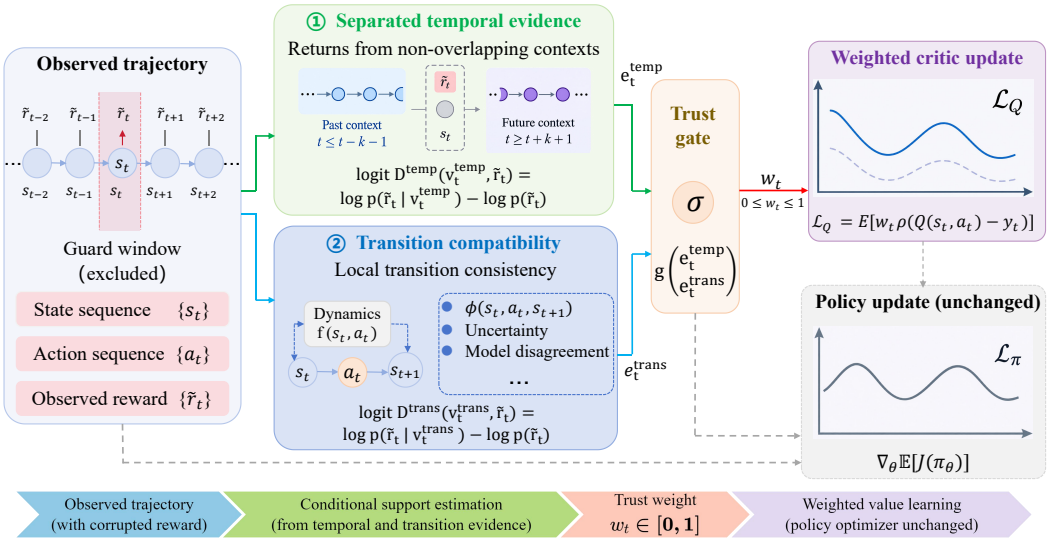


Figure 1: Conceptual overview of ReConRL. The temporal view uses separated trajectory context, while the transition view evaluates local reward–transition compatibility. Dynamics, uncertainty, and disagreement in the transition panel illustrate possible manifestations of local compatibility rather than additional learned modules; the implementation directly encodes (s_t, a_t, s_{t+1}) with q_ϕ. The two scores are fused monotonically into $w_t \in [w_{\min}, 1] \subseteq [0, 1]$, which reweights the backbone-native critic loss while leaving the policy update unchanged. The figure label "trans" corresponds to the superscript "dyn" used in Eqs. (5)–(6).

with out-of-range terms masked. The binary validity masks m_t^- and m_t^+ are explicit encoder inputs, so padding is distinguishable from a genuine zero reward. A context encoder f_ψ receives state-action features from the separated windows, (G_t^-, G_t^+), and (m_t^-, m_t^+), producing h_t^{sep}. Rewards in $[t-g+1, t+g-1]$ are not provided to this path. In replay-based training, G_t^+ is computed only after that future context has already been collected and stored. Support-network architectures are specified in Appendix A.2.

We train $D_\psi^{\text{temp}}(h, r)$ to distinguish matched pairs $(h_t^{\text{sep}}, \tilde{r}_t)$ from rewards independently permuted within the same task and replay phase. With K negatives,

$$\mathcal{L}_{\text{temp}} = -\mathbb{E}\left[\log D_\psi^{\text{temp}}(h_t^{\text{sep}}, \tilde{r}_t) + \frac{1}{K}\sum_{k=1}^{K} \log(1 - D_\psi^{\text{temp}}(h_t^{\text{sep}}, \tilde{r}_{t,k}^-))\right]. \tag{4}$$

Matched positives include corrupted samples when $z_t = 1$; the largest positive losses are trimmed at the fixed $(1-\hat{\epsilon})$ quantile, where $\hat{\epsilon}$ is the prespecified contamination upper bound. The output $e_t^{\text{temp}} = D_\psi^{\text{temp}}(h_t^{\text{sep}}, \tilde{r}_t)$ estimates conditional support without using the policy critic.

4.3 Transition Compatibility and Monotone Fusion

Some rewards are supported immediately rather than by delayed return structure. Terminal success, collision, or contact rewards can coincide with distinctive local transition structure even when future rewards are uninformative. Figure 1 depicts such structure schematically through dynamics-related compatibility cues; the implementation does not instantiate these cues as separate models. Instead, a single transition encoder $q_\phi(s_t, a_t, s_{t+1})$ learns the relevant representation directly from the observed transition.

Let $e_\phi(\tilde{r}_t)$ encode the scalar reward. Their compatibility score is

$$e_t^{\text{dyn}} = D_\phi^{\text{dyn}}(s_t, a_t, s_{t+1}, \tilde{r}_t) = \sigma(g_\phi([q_\phi(s_t, a_t, s_{t+1}); e_\phi(\tilde{r}_t)])), \tag{5}$$

trained with an independent within-task reward permutation. Its density-ratio interpretation matches Eq. (2) with $v = v_t^{\text{dyn}}$.

The two views are complementary: either may support a genuine event, while a single overconfident view should not dominate trust. The fusion model acts directly on their probabilities. Let

$$\begin{aligned} x_t &= [e_t^{\text{temp}}, e_t^{\text{dyn}}, e_t^{\text{temp}} e_t^{\text{dyn}}], \qquad \boldsymbol{\lambda} = \text{softplus}(\boldsymbol{\zeta}) \succeq 0, \\ u_t &= \sigma(b + \boldsymbol{\lambda}^\top x_t), \qquad w_t = w_{\min} + (1 - w_{\min}) u_t^\beta, \\ &\beta > 0, \qquad 0 \le w_{\min} < 1. \end{aligned} \tag{6}$$

Because $e_t^{\text{temp}}, e_t^{\text{dyn}} \in [0, 1]$, the fusion logit is coordinate-wise monotone: its partial derivatives are $\lambda_1 + \lambda_3 e_t^{\text{dyn}}$ and $\lambda_2 + \lambda_3 e_t^{\text{temp}}$, both nonnegative. Thus stronger evidence from either branch cannot lower trust, and the monotonicity is preserved by the increasing map from u_t to w_t.

The fusion layer has its own contrastive objective. For an observed tuple, x_t^+ uses the matched reward $\tilde{r}_t$; x_t^- is recomputed after replacing that reward by a marginal-preserving within-task shuffle or a short local temporal replacement. After a branch warm-up of N_{warm} support updates, we optimize

$$\mathcal{L}_{\text{fuse}} = \text{Trim}_{1-\hat{\epsilon}}\big[-\log u(x_t^+)\big] + \lambda_f \mathbb{E}\big[-\log\big(1 - u(x_t^-)\big)\big], \tag{7}$$

Here $\text{Trim}_{1-\hat{\epsilon}}$ removes the largest $\hat{\epsilon}$ fraction of weak-positive losses, and $\lambda_f > 0$ weights the synthetic-negative term. Branch scores are stop-gradient inputs to Eq. (7), so this loss updates only b and $\boldsymbol{\lambda}$; the two evidence networks continue to train with their own objectives. During the branch warm-up the learner uses $w_t = 1$, so the backbone trains exactly as usual; trust weighting is activated only after both branches complete N_{warm} updates. After warm-up, each learner update is followed by one update to each support branch and one fusion update. Fusion negatives use marginal-preserving shuffles and local temporal replacements. The default model does not use magnitude-perturbed negatives; scale perturbations are used only in stress evaluation and in the dedicated shortcut ablation of Appendix Table 13.

The weighted critic update shown in Figure 1 is schematic; Eq. (8) gives the backbone-agnostic form used in all experiments. ReConRL acts as a loss-reweighting layer. Let $\ell_{Q,t}^{\text{base}}$ denote the native per-transition critic loss of the chosen learner, including its own Bellman target, target-network mechanism, distributional parameterization, and loss shape. We replace only its sample weight:

$$\ell_{Q,t}^{\text{ReConRL}} = \text{sg}(w_t)\, \ell_{Q,t}^{\text{base}}, \qquad \mathcal{L}_Q^{\text{ReConRL}} = \mathbb{E}_{t \sim \mathcal{B}}\big[\text{sg}(w_t) \ell_{Q,t}^{\text{base}}\big]. \tag{8}$$

Here sg denotes stop-gradient and $\mathcal{B}$ denotes the replay sampling distribution. Thus SAC retains its entropy-regularized twin-critic target, CrossQ and SimBa retain their native critic objectives, and DQN/QR-DQN retain their value or distributional Bellman losses. The target construction, critic architecture, actor objective, entropy-temperature update, and optimizer are otherwise unchanged; gradients from Eq. (8) do not enter the support networks. ReConRL itself uses no clean rewards, corruption labels, or clean reference trajectories for support learning or reward normalization.

5 Experiments

The experiments first test whether the trust score separates rare-valid from equally rare corrupted rewards, then test whether that separation improves control under paired backbones. Additional experiments vary corruption severity, freeze support under unseen corruption mechanisms, transfer the fixed method design to held-out tasks, and remove individual evidence and fusion components.

5.1 Setup and Metrics

Benchmarks. We use three controlled diagnostic MDPs, six development continuous-control tasks, two held-out continuous tasks for design transfer, and four Atari environments for discrete control. GoalEvent-v0 matches a rare valid reward to an equally rare injected reward of the same magnitude; ContextSwap-v0 holds event rarity fixed while changing which event is reward-bearing; FrequencyShift-v0 fixes the unsupported event and reward magnitude while varying only its marginal occurrence rate. The diagnostic suite uses ten paired seeds; the large-scale benchmarks use at least five. Exact state counts, horizons, task identities, and budgets are fixed in Appendix A.

Corruptions. The online core benchmark uses impulse, short burst, and marginal replacement at realized corrupted-transition fractions of 5%, 15%, and 30%. The mechanism-OOD experiment uses a separate two-stage protocol. In Stage 1, each episode independently selects impulse or short-burst corruption with probability 0.5, always at a 15% realized corrupted-transition fraction; the

Table 1: Paired control performance at 15% core corruption. Task cells are task-level IQMs over the three core mechanisms and paired seeds; aggregate bootstrap uncertainty is summarized in Appendix Table 18.

(a) Continuous-control development tasks: return retention.

Method	HC	Hop	Walk	Ant	Cheetah	Quad	Dev IQM	$Q_{0.10}$
SAC	0.721	0.642	0.779	0.688	0.821	0.738	0.732	0.556
SAC + ReConRL	0.823	0.736	0.846	0.826	0.906	0.789	0.818	0.688
CrossQ	0.772	0.703	0.824	0.806	0.858	0.787	0.797	0.613
CrossQ + ReConRL	0.848	0.778	0.890	0.858	0.851	0.835	0.846	0.728
SimBa	0.816	0.768	0.861	0.794	0.893	0.832	0.826	0.672
SimBa + ReConRL	0.886	0.762	0.923	0.906	0.968	0.883	0.889	0.786

(b) Discrete-control benchmark: return retention.

Method	Pong	Breakout	Seaquest	Q*bert	IQM
DQN	0.708	0.541	0.632	0.579	0.606
DQN + ReConRL	0.821	0.690	0.712	0.755	0.742
QR-DQN	0.751	0.612	0.681	0.645	0.663
QR-DQN + ReConRL	0.744	0.758	0.806	0.789	0.776

resulting mixed stream trains the support branches and fusion on the same task. Stage 2 freezes all support/fusion parameters, discards the Stage 1 actor, critic, and replay buffer, initializes a fresh learner, and trains from scratch under sign flip, marginal replacement, long burst, or unseen-scale corruption, each at 15%. Held-out-task transfer reinitializes and trains all support and fusion parameters from the new task's unlabeled corrupted replay while keeping architecture, hyperparameters, threshold-selection rule, and training protocol fixed. Appendix A gives the exact corruption equations and reference-buffer construction.

Comparisons. The paired backbone comparison is SAC versus SAC+ReConRL, CrossQ versus CrossQ+ReConRL, and SimBa versus SimBa+ReConRL (Haarnoja et al., 2018; Bhatt et al., 2024; Lee et al., 2025a). DQN and QR-DQN provide the discrete-control pairs (Mnih et al., 2015; Dabney et al., 2018). TD-MPC2, DreamerV3, SimbaV2, FoG, Annealed-Q, SPO, and FPG provide additional reference results in Appendix Table 10 (Hansen et al., 2024; Hafner et al., 2025; Lee et al., 2025b; Kang et al., 2025; Omura et al., 2025; Xie et al., 2025; Zhang et al., 2021). Paired online runs match environment seeds, corruption RNG streams, initialization seeds, interaction budgets, update-to-data ratios, and evaluation schedules. Trajectories and replay contents remain algorithm-dependent.

All controlled diagnostics use the same DQN learner. Table 2 compares reward-trust rules under a common learner. The rarity and TD-error heuristics use prespecified scalar weighting rules whose average trust budget is matched on development diagnostics; their exact formulas and frozen tuning grids are given in Appendix A.3.

Metrics. Clean evaluation reward is never exposed to training. For each task/backbone, let J_{clean} be the return of the backbone trained on uncorrupted rewards and J_{random} the random-policy return. We report un-clipped return retention

$$\mathrm{Ret} = \frac{J_{\mathrm{corr}} - J_{\mathrm{random}}}{J_{\mathrm{clean}} - J_{\mathrm{random}}}, \tag{9}$$

so values above one and below zero remain visible. Table 1 reports the 15% core condition; each task entry is the IQM over the three core mechanisms and paired seeds. $Q_{0.10}$ is the 10th percentile over the corresponding task×mechanism outcomes, not over seeds.

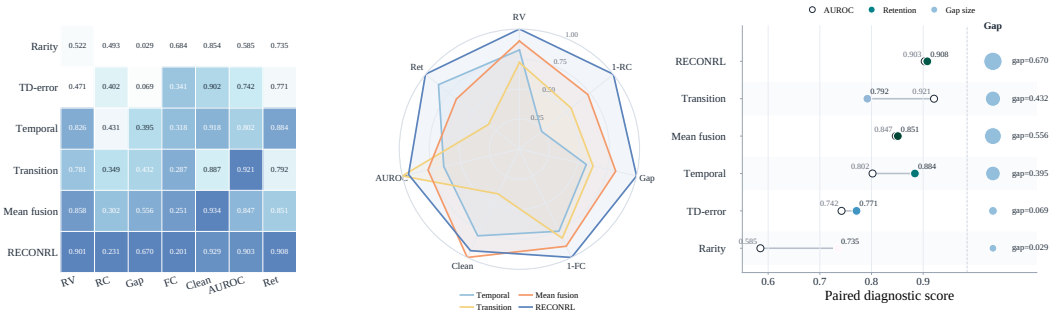
Soft trust statistics are corrupted mean trust $\mathrm{CMT} = \mathbb{E}[w_t \mid z_t = 1]$, genuine-event mean distrust $\mathrm{GMD} = \mathbb{E}[1 - w_t \mid z_t = 0, |r_t| > \tau_r]$, and clean mean trust $\mathrm{CleanTrust} = \mathbb{E}[w_t \mid z_t = 0]$. We fix τ_r before evaluation as the 90th percentile of $|r|$ in the fixed task-level clean reference buffer defined in Appendix A.3; diagnostic MDPs use their event labels instead. Thresholded rates

$$\mathrm{CRA}(\eta) = \Pr(w_t \geq \eta \mid z_t = 1), \qquad \mathrm{GRR}(\eta) = \Pr(w_t < \eta \mid z_t = 0,\ |r_t| > \tau_r) \tag{10}$$

are reported as operating curves over $\eta \in [w_{\min}, 1]$; if a scalar operating point is needed, $\eta = 0.5$ is fixed a priori. Because w_t is a *clean/trust* score while $z_t = 1$ denotes corruption, AUROC and

Table 2: Matched rarity–support diagnostics on the controlled DQN suite. AUROC uses corruption score $1 - w_t$ against $z_t = 1$; Ret denotes diagnostic return retention.

Method	RV Trust ↑	RC Trust ↓	RV−RC Gap ↑	FC Trust ↓	CleanTrust ↑	AUROC ↑	Ret ↑
Rarity filter	0.522	0.493	0.029	0.684	0.854	0.585	0.735
TD-error filter	0.471	0.402	0.069	0.341	0.902	0.742	0.771
Temporal only	0.826	0.431	0.395	0.318	0.918	0.802	0.884
Transition only	0.781	0.349	0.432	0.287	0.887	0.921	0.792
Mean fusion	0.858	0.302	0.556	0.251	0.934	0.847	0.851
Full ReConRL	0.901	0.231	0.670	0.201	0.929	0.903	0.908



(a) Metric profile **(b) Support-behavior signature** **(c) Ranked paired diagnostic view**

Figure 2: Matched rarity–support diagnostics. (a) Per-method trust, ranking, and retention metrics. (b) Support–behavior profiles contrast rare-valid preservation with rare/frequent corruption rejection and clean-transition trust; RC and FC trust are plotted as $1 - \mathrm{RC}$ and $1 - \mathrm{FC}$ so every radar axis is high-is-better. (c) Paired diagnostic view relates corruption-ranking AUROC and return retention to the RV−RC trust gap.

AUPRC are computed as $\mathrm{AUROC}(1 - w_t, z_t)$ and $\mathrm{AUPRC}(1 - w_t, z_t)$, respectively. ECE measures empirical trust–label agreement: in equal-mass bins B_m, mean w_t is compared with the empirical clean fraction $\Pr(z_t = 0)$. Aggregate control statistics use IQM, 95% stratified bootstrap intervals, performance profiles, and paired probability of improvement (Agarwal et al., 2021; Patterson et al., 2024). Appendix A reports a compact uncertainty summary and trust operating-point diagnostics.

5.2 Control and Matched Diagnostics

Table 1 consolidates continuous and discrete paired evaluations while keeping incompatible action spaces in separate panels. Development and held-out continuous tasks are not mixed: held-out transfer appears only in Table 3.

Across the paired backbones in Table 1, ReConRL raises development IQM from 0.732 to 0.818 with SAC, 0.797 to 0.846 with CrossQ, and 0.826 to 0.889 with SimBa. Discrete-control IQM rises from 0.606 to 0.742 with DQN and from 0.663 to 0.776 with QR-DQN.

Table 2 compares trust rules on matched-support diagnostics. Rare-valid (RV) and rare-corrupt (RC) events have matched reward magnitude and empirical frequency; frequent-corrupt (FC) changes only corruption frequency. The RV−RC trust gap measures support while reward rarity is held fixed.

Full ReConRL produces an RV−RC trust gap of 0.670 while reducing frequent-corrupt trust to 0.201. The rarity heuristic instead assigns higher trust to frequent corruption (0.684) than to rare corruption (0.493), although both lack trajectory support. Transition-only scoring attains the highest AUROC (0.921) but only 0.792 diagnostic retention, while the fused model reaches 0.908 retention. Figure 2 shows the corresponding trust and control profiles.

5.3 Robustness and Mechanism Analysis

Table 3 contains three robustness regimes. Panel A measures severity on the development suite; Panel B freezes support/fusion and trains fresh SAC learners under unseen corruption mechanisms; Panel C relearns support parameters on held-out tasks while keeping the method design fixed.

Table 3: Robustness to severity, corruption shift, and task transfer.

(a) Core severity: IQM return retention across development tasks.

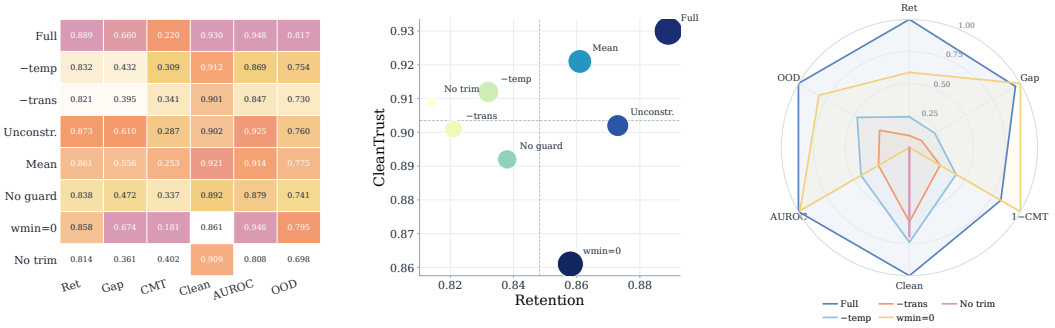
Method	I5	I15	I30	B5	B15	B30	R5	R15	R30	IQM
SAC	0.914	0.761	0.514	0.879	0.681	0.402	0.932	0.776	0.566	0.733
SAC + ReConRL	0.944	0.871	0.614	0.959	0.811	0.562	0.941	0.816	0.656	0.792
CrossQ	0.927	0.798	0.581	0.904	0.736	0.472	0.949	0.817	0.631	0.777
CrossQ + ReConRL	0.977	0.828	0.701	0.944	0.876	0.622	0.942	0.897	0.681	0.832
SimBa	0.947	0.842	0.648	0.925	0.779	0.563	0.958	0.858	0.691	0.819
SimBa + ReConRL	0.967	0.942	0.718	0.995	0.829	0.703	0.953	0.878	0.801	0.867

(b) Same-task mechanism OOD: frozen support/fusion, fresh SAC learner.
Seen Ret is the mean of fresh Stage-2 runs under 15% impulse and 15% short burst using the same frozen Stage-1 support checkpoint.

Method	Seen Ret	Flip Ret	Repl Ret	LongB Ret	Scale Ret	OOD IQM	OOD CMT
SAC	0.721	0.651	0.761	0.547	0.503	0.603	1.000
SAC + TD filter	0.758	0.706	0.779	0.603	0.568	0.655	0.487
SAC + rarity filter	0.747	0.674	0.804	0.582	0.612	0.643	0.541
SAC + temporal only	0.818	0.732	0.811	0.728	0.691	0.730	0.341
SAC + transition only	0.806	0.794	0.872	0.618	0.714	0.754	0.309
SAC + ReConRL	0.861	0.842	0.859	0.792	0.773	0.817	0.219

(c) Held-out-task design transfer: fresh support/fusion parameters. Transfer IQM is computed over pooled task×seed outcomes, not by applying IQM to the two displayed task aggregates.

Method	Hum Ret	Hum CMT	Manip Ret	Manip CMT	Transfer IQM
SAC	0.603	1.000	0.721	1.000	0.662
SAC + ReConRL	0.734	0.271	0.826	0.236	0.779
CrossQ	0.651	1.000	0.758	1.000	0.705
CrossQ + ReConRL	0.642	0.258	0.861	0.219	0.756
SimBa	0.704	1.000	0.803	1.000	0.754
SimBa + ReConRL	0.806	0.223	0.892	0.197	0.848



(a) Ablation profile **(b) Ablation frontier** **(c) Failure mode signature**

Figure 3: Mechanism ablations. (a) Metric profiles; (b) retention–CleanTrust frontier; (c) selected failure signatures. CMT is shown as $1 - \mathrm{CMT}$, so all radar axes are high-is-better.

Core-severity IQM increases for all three backbones, with gains remaining at 30% corruption. Under frozen-support mechanism OOD, ReConRL reaches 0.817 OOD IQM and 0.219 CMT; held-out-task transfer raises pooled IQM from 0.662 to 0.779 for SAC, 0.705 to 0.756 for CrossQ, and 0.754 to 0.848 for SimBa.

Table 4 removes the two evidence paths, monotonicity constraint, guard window, support floor, and positive trimming one at a time.

Full ReConRL has the highest retention (0.889), AUROC (0.948), and OOD retention (0.817). Removing either evidence path lowers both the RV−RC gap and control retention. Unconstrained fusion reduces the gap from 0.660 to 0.610 and OOD retention from 0.817 to 0.760. The zero-floor variant drives corrupted trust lower than the full model but also suppresses substantially more clean supervision.

Table 4: Mechanism ablations on the development subset; OOD Ret denotes Stage-2 unseen-mechanism retention.

Variant	Ret ↑	RV−RC Gap ↑	CMT ↓	CleanTrust ↑	AUROC ↑	OOD Ret ↑
Full ReConRL	0.889	0.660	0.220	0.930	0.948	0.817
− temporal branch	0.832	0.432	0.309	0.912	0.869	0.754
− transition branch	0.821	0.395	0.341	0.901	0.847	0.730
Unconstrained fusion	0.873	0.610	0.287	0.902	0.925	0.760
Mean fusion	0.861	0.556	0.253	0.921	0.914	0.775
No guard window	0.838	0.472	0.337	0.892	0.879	0.741
$w_{\min} = 0$	0.858	0.674	0.181	0.861	0.946	0.795
No positive trimming	0.814	0.361	0.402	0.909	0.808	0.698

Detection quality and learning utility. High corruption-ranking accuracy does not necessarily maximize control: transition-only attains the highest AUROC (0.921) but lower retention (0.792) than temporal-only (0.884) and full ReConRL (0.908). Likewise, $w_{\min} = 0$ lowers CMT from 0.220 to 0.181 but reduces CleanTrust from 0.930 to 0.861 and retention from 0.889 to 0.858. Useful trust must therefore reject unsupported rewards while preserving genuine supervision (Figure 3).

5.4 Statistical Protocol

Continuous-control conditions use five paired seeds and diagnostics use ten; Appendix Table 18 reports 95% stratified bootstrap intervals and paired probability of improvement. Hyperparameters are selected only on development tasks and training corruptions. We fix $\hat{\epsilon} = 0.30$ throughout (0.70 trimming; Appendix Table 12), freeze Stage 1 support for mechanism OOD, and reinitialize support/fusion on held-out tasks while preserving architecture and protocol.

Prespecified success requires a positive matched RV−RC gap without sacrificing clean or genuine-event trust, together with retention gains across multiple backbones and a held-out corruption family.

Scope of trajectory support. ReConRL assumes that corruption fails to reproduce genuine reward–context association in at least one evidence view. Temporal evidence is retrospective, with a past-only variant in Appendix Table 11; corruptors matching the clean conditional law remain unidentifiable by Proposition 1; and valid rewards may be supported by either temporal or local transition evidence. Cases outside these conditions require information beyond the observed trajectory.

6 Conclusion

Reward rarity alone does not determine reliability. Matched diagnostics show that conditional reward–trajectory association separates equally rare valid and corrupted feedback, while control results show that corruption ranking alone is insufficient when genuine supervision is discarded. ReConRL implements this principle with two critic-output-independent evidence views and a monotone critic-loss weight. Corruption that reproduces the clean reward–context joint remains unidentifiable from trajectories alone.

AI Use Disclosure

Generative AI tools were used solely for language editing and polishing.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C. Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and Jan Peters. CrossQ: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. In *International Conference on Learning Representations*, 2024.

Zhenfang Chen, Delin Chen, Rui Sun, Wenjun Liu, and Chuang Gan. Scaling autonomous agents via automatic reward modeling and planning. In *International Conference on Learning Representations*, 2025.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Will Dabney, Mark Rowland, Marc G. Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11791.

Tom Everitt, Victoria Krakovna, Laurent Orseau, and Shane Legg. Reinforcement learning with a corrupted reward channel. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4705–4713, 2017. doi: 10.24963/ijcai.2017/656.

Tom Everitt, Marcus Hutter, Ramana Kumar, and Victoria Krakovna. Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective. *arXiv preprint arXiv:1908.04734*, 2019.

Evan Frick, Tianle Li, Connor Chen, Wei-Lin Chiang, Anastasios Angelopoulos, Jiantao Jiao, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. How to evaluate reward models for RLHF. In *International Conference on Learning Representations*, 2025.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870, 2018.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025. doi: 10.1038/s41586-025-08744-2.

Nick Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Jiwoo Hong, Noah Lee, Eunki Kim, Guijin Son, Woojin Chung, Aman Gupta, Shao Tang, and James Thorne. On the robustness of reward models for language model alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 23682–23699, 2025.

Zeyu Huang, Zihan Qiu, Zili Wang, Edoardo M. Ponti, and Ivan Titov. Post-hoc reward calibration: A case study on length bias. In *International Conference on Learning Representations*, 2025.

Minjune Hwang, Yigit Korkmaz, Daniel Seita, and Erdem Biyik. Causally robust reward learning from reason-augmented preference feedback. In *International Conference on Learning Representations*, 2026.

Zilin Kang, Chenyuan Hu, Yu Luo, Zhecheng Yuan, Ruijie Zheng, and Huazhe Xu. A forget-and-grow strategy for deep reinforcement learning scaling in continuous control. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 28921–28942, 2025.

Cassidy Laidlaw, Shivam Singhal, and Anca Dragan. Correlated proxies: A new definition and improved mitigation for reward hacking. In *International Conference on Learning Representations*, 2025.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. RewardBench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1755–1797, 2025. doi: 10.18653/v1/2025.findings-naacl.96.

Filippo Lazzati and Alberto Maria Metelli. Robustness in the face of partial identifiability in reward learning. In *International Conference on Learning Representations*, 2026.

Hojoon Lee, Dongyoon Hwang, Donghu Kim, Hyunseung Kim, Jun Jet Tai, Kaushik Subramanian, Peter R. Wurman, Jaegul Choo, Peter Stone, and Takuma Seno. SimBa: Simplicity bias for scaling up parameters in deep reinforcement learning. In *International Conference on Learning Representations*, 2025a.

Hojoon Lee, Youngdo Lee, Takuma Seno, Donghu Kim, Peter Stone, and Jaegul Choo. Hyperspherical normalization for scalable deep reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 33352–33403, 2025b.

Jiachun Li, Pengfei Cao, Zhuoran Jin, Yubo Chen, Jiexin Xu, Huaijun Li, Xiaojian Jiang, Kang Liu, and Jun Zhao. Fixing the broken compass: Diagnosing and improving inference-time reward modeling. In *International Conference on Learning Representations*, 2026a.

Zhuo Li, Pengyu Cheng, Zhechao Yu, Feifei Tong, Anningzhe Gao, Tsung-Hui Chang, Xiang Wan, Erchao Zec, Xiaoxi Jiang, and Guanjun Jiang. Eliminating inductive bias in reward models with information-theoretic guidance. In *International Conference on Learning Representations*, 2026b.

Tianqi Liu, Wei Xiong, Jie Ren, Lichang Chen, Junru Wu, Rishabh Joshi, Yang Gao, Jiaming Shen, Zhen Qin, Tianhe Yu, Daniel Sohn, Anastasia Makarova, Jeremiah Zhe Liu, Yuan Liu, Bilal Piot, Abe Ittycheriah, Aviral Kumar, and Mohammad Saleh. RRM: Robust reward model training mitigates reward hacking. In *International Conference on Learning Representations*, 2025a.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. RM-Bench: Benchmarking reward models of language models with subtlety and style. In *International Conference on Learning Representations*, 2025b.

Sreejeet Maity and Aritra Mitra. Robust Q-learning under corrupted rewards. In *2024 IEEE 63rd Conference on Decision and Control*, pages 1181–1186, 2024. doi: 10.1109/CDC56724.2024.10885945.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A. Smith, Hannaneh Hajishirzi, and Nathan Lambert. RewardBench 2: Advancing reward model evaluation. In *International Conference on Learning Representations*, 2026.

Debmalya Mandal, Andi Nika, Parameswaran Kamalaruban, Adish Singla, and Goran Radanovic. Corruption robust offline reinforcement learning with human feedback. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pages 4429–4437, 2025.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015. doi: 10.1038/nature14236.

Manh Hung Nguyen, Tzu-Yin Chao, Ching-Chun Hsiao, Yung-Hui Li, and Ching-Chun Huang. Self-supervise reinforcement learning method for vacant parking space detection based on task consistency and corrupted rewards. *IEEE Transactions on Intelligent Transportation Systems*, 25 (2):1346–1363, 2024. doi: 10.1109/TITS.2023.3319531.

Andi Nika, Adish Singla, and Goran Radanovic. Online defense strategies for reinforcement learning against adaptive reward poisoning. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 335–358, 2023.

Andi Nika, Debmalya Mandal, Parameswaran Kamalaruban, Adish Singla, and Goran Radanovic. Corruption-robust offline multi-agent reinforcement learning from human feedback. In *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pages 1864–1872, 2026.

Arnab Nilim and Laurent El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005. doi: 10.1287/opre.1050.0216.

Motoki Omura, Kazuki Ota, Takayuki Osa, Yusuke Mukuta, and Tatsuya Harada. Gradual transition from Bellman optimality operator to Bellman operator in online reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 47176–47195, 2025.

Andrew Patterson, Samuel Neumann, Martha White, and Adam White. Empirical design in reinforcement learning. *Journal of Machine Learning Research*, 25(318):1–63, 2024.

Pragya Srivastava, Harman Singh, Rahul Madhavan, Gandharv Patil, Sravanti Addepalli, Arun Suggala, Rengarajan Aravamudhan, Soumya Sharma, Anirban Laha, Aravindan Raghuveer, Karthikeyan Shanmugam, and Doina Precup. Robust reward modeling via causal rubrics. In *International Conference on Learning Representations*, 2026.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

Nam Phuong Tran, Andi Nika, Goran Radanovic, Long Tran-Thanh, and Debmalya Mandal. Sparse offline reinforcement learning with corruption robustness. In *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pages 1738–1746, 2026.

Xinpeng Wang, Nitish Joshi, Barbara Plank, Rico Angell, and He He. Is It Thinking or Cheating? Detecting Implicit Reward Hacking by Measuring Reasoning Effort. In *International Conference on Learning Representations*, 2026.

Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. Robust Markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013. doi: 10.1287/moor.1120.0566.

Zhengpeng Xie, Qiang Zhang, Fan Yang, Marco Hutter, and Renjing Xu. Simple policy optimization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 68813–68824, 2025.

Zhuohao Yu, Steven Wu, and Adam Block. From curiosity to caution: Mitigating reward hacking for Best-of-N with pessimism. In *International Conference on Learning Representations*, 2026.

Xuezhou Zhang, Yiding Chen, Xiaojin Zhu, and Wen Sun. Robust policy gradient against strong data corruption. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12391–12401, 2021.

A Additional Experimental Specification

The appendix specifies task identities, support architectures, controlled diagnostics, corruption generators, optimization choices, and supplementary sensitivity analyses.

A.1 Proof of Proposition 1

Proof. Let $\mathcal{P}_{\mathrm{obs}}$ denote any probability law over the observable process $\{(s_t, a_t, s_{t+1}, \tilde{r}_t)\}_{t\geq 0}$. A first latent model sets $r_t^{(1)} = \tilde{r}_t$ and $c_t^{(1)} = 0$ almost surely. Choose any nonzero latent process Δ_t on the same probability space satisfying $\Pr(\Delta_t \neq 0) \leq \epsilon$; a second model sets $r_t^{(2)} = \tilde{r}_t - \Delta_t$ and $c_t^{(2)} = \Delta_t$. For every realization and every t, both models satisfy $r_t^{(i)} + c_t^{(i)} = \tilde{r}_t$ and the sparsity bound in Eq. (1). Hence both induce exactly $\mathcal{P}_{\mathrm{obs}}$ over all quantities available to a trajectory-only estimator, while their latent rewards and corruption indicators disagree whenever $\Delta_t \neq 0$. Any

measurable rule of the observed trajectory must therefore have the same distribution under both latent models and cannot identify which decomposition is correct. Structural restrictions are necessary for reward corruption to be distinguishable. □

Table 5: Task suite, evaluation roles, and interaction budgets.

Family	Task	Evaluation role	Obs.	Action	Budget
Diagnostic	GoalEvent-v0	diagnostic	state	disc.	100k
Diagnostic	ContextSwap-v0	diagnostic	state	disc.	100k
Diagnostic	FrequencyShift-v0	diagnostic	state	disc.	100k
MuJoCo	HalfCheetah-v4	development	state	cont.	1M
MuJoCo	Hopper-v4	development	state	cont.	1M
MuJoCo	Walker2d-v4	development	state	cont.	1M
MuJoCo	Ant-v4	development	state	cont.	1M
MuJoCo	Humanoid-v4	transfer	state	cont.	2M
DMControl	cheetah-run	development	state	cont.	1M
DMControl	quadruped-run	development	state	cont.	1M
DMControl	manipulator-bring-ball	transfer	state	cont.	1M
Atari	Pong	discrete benchmark	pixels	disc.	10M
Atari	Breakout	discrete benchmark	pixels	disc.	10M
Atari	Seaquest	discrete benchmark	pixels	disc.	10M
Atari	Q*bert	discrete benchmark	pixels	disc.	10M

A.2 Support Architecture and Implementation

Support Features and Networks. The same support architecture and optimizer settings are used in every run. Let o_t denote the state feature supplied to the support networks. For vector-state continuous control, o_t is the observation standardized with running mean and variance from the current learner's replay and clipped to $[-5, 5]$; actions remain in their native normalized range. For the diagnostic MDPs, o_t is a one-hot state vector and the two actions are one-hot encoded. For Atari, o_t is the 512-dimensional pre-output feature of the DQN/QR-DQN convolutional encoder passed through a stop-gradient connection. Observed scalar rewards are standardized for both support branches with running mean and variance from the current learner's replay, using variance floor 10^{-6}. Support losses never update the learner encoder, and no critic/Q output is supplied to either support branch.

For the temporal path, every valid context step $[o_j, a_j, \tilde{r}_j, m_j]$ is mapped by a shared two-layer MLP with widths $(128, 128)$ and SiLU activations. Masked mean pooling is applied separately to past and future windows. The two pooled vectors, (G_t^-, G_t^+), and the validity fractions are concatenated and passed through a two-layer $(256, 128)$ SiLU MLP to produce the 128-dimensional h_t^{sep}. The temporal discriminator concatenates h_t^{sep} with a 32-dimensional reward embedding and uses a $(128, 64, 1)$ MLP.

For the transition path, q_ϕ is a $(256, 256, 128)$ SiLU MLP over $[o_t, a_t, o_{t+1}, o_{t+1} - o_t]$. The standardized scalar reward is embedded by e_ϕ with a $(32, 32)$ SiLU MLP. The compatibility head g_ϕ uses widths $(128, 64, 1)$. All support MLPs use Xavier initialization and Adam with the support learning rate in Table 8; branch and fusion parameters are disjoint. Fusion itself contains only b and the three nonnegative coefficients in Eq. (6).

A.3 Diagnostic Learner and Heuristic Baselines

All three controlled diagnostics use the same DQN backbone: one-hot state input, two 64-unit ReLU hidden layers, Adam with learning rate 10^{-3}, discount 0.99, replay capacity 50k, learner batch 128, 1k-transition replay warm-up, and a hard target-network update every 250 learner steps. ϵ-greedy exploration decays linearly from 1.0 to 0.05 over the first 20k interactions and remains at 0.05 thereafter; every run uses the 100k interaction budget in Table 5.

Let d_t be the percentile rank of the observed-reward marginal density estimated from the current learner's replay, with larger d_t indicating a more common reward. A one-dimensional Gaussian KDE is refit every 5k interactions using only rewards already present in replay. The rarity heuristic uses

$$w_t^{\text{rar}} = w_{\min} + (1 - w_{\min}) d_t^{\beta_r}. \tag{11}$$

The scalar $\beta_r \in \{0.5, 1, 2, 4\}$ is selected once on the development diagnostics to match the mean trust of full ReConRL as closely as possible; ties are broken by diagnostic return and the chosen

value is frozen for all reported diagnostic, OOD, and stress comparisons. For the TD-error heuristic, the native one-step absolute TD residual $|\delta_t|$ is normalized by its running median m_δ and mapped to

$$w_t^{\mathrm{TD}} = w_{\min} + (1 - w_{\min}) \exp\left[-|\delta_t|/(\tau_{\mathrm{TD}} m_\delta + 10^{-8})\right]. \tag{12}$$

We choose $\tau_{\mathrm{TD}} \in \{0.5, 1, 2, 4\}$ once on the same development diagnostics by the same mean-trust matching rule. The resulting heuristic baselines use a matched mean-trust budget across the development diagnostics.

Controlled Diagnostic Suite. Each diagnostic is a two-action finite-horizon MDP with a fixed seed-generated transition graph. GoalEvent-v0 uses 21 states and horizon 25; a cue opens a rare goal transition with probability 0.05, which yields reward $+1$, while a matched $+1$ corruption is injected on an ordinary transition at the same empirical event frequency. ContextSwap-v0 uses 15 states and horizon 20 with two event transitions of equal probability 0.05; only one event is reward-bearing, and the intervention swaps which event carries $+1$ while preserving event rarity. FrequencyShift-v0 uses 15 states and horizon 20, fixes the same unsupported $+1$ corruption mechanism, and changes only its marginal occurrence rate over $\{1\%,5\%,20\%\}$. The reward magnitude is fixed at one in all three diagnostics. Table 6 summarizes the controlled constructions used by the matched-diagnostic experiments.

Table 6: Exact controlled-diagnostic specifications.

Environment	States	Horizon	Valid-event prob.	Reward	Corrupt injection	Controlled variable
GoalEvent-v0	21	25	0.05	$+1$	matched 0.05	support at fixed magnitude/frequency
ContextSwap-v0	15	20	0.05 / 0.05	$+1$	event swap	reward–context association
FrequencyShift-v0	15	20	0.05	$+1$	0.01 / 0.05 / 0.20	corruption frequency

Fixed Clean Reference Buffer. Each task has one benchmark-only clean reference buffer $\mathcal{D}_{\mathrm{ref}}^{\mathrm{task}}$ generated once with dedicated seed 314159 and reused unchanged by every method and experimental seed. Continuous-control buffers contain 100k transitions collected from the native stochastic actor of a clean SAC reference policy trained for the task's standard budget; Atari buffers contain 250k transitions from a clean DQN reference policy evaluated with $\epsilon = 0.05$; finite diagnostic MDPs use their exact clean reward distribution rather than an estimated buffer. The reference runs are disjoint from all paired seeds. They are used only to define benchmark corruptions and evaluation quantities: $\sigma_r = \max\{\mathrm{Std}_{\mathcal{D}_{\mathrm{ref}}}(r), 10^{-6}\}$, $\hat{p}_{\mathrm{ref}}(r)$, τ_r, marginal-replacement samples, and post-hoc rarity bins. The buffer never enters RECONRL, the heuristic baselines, learner updates, learner-side normalization, or hyperparameter selection.

Corruption Generators. Let $z_t \in \{0, 1\}$ denote the corruption mask, with generator parameters chosen so the *realized* corrupted-transition fraction matches the requested level within 0.5 percentage points. Core additive corruptions use $\kappa = 2$; stress sweeps use $\kappa \in \{1, 2, 3\}$; unseen-scale OOD uses $\kappa \in \{4, 5\}$.

$$\text{Impulse:} \quad \tilde{r}_t = r_t + z_t \xi_t \kappa \sigma_r, \qquad \xi_t \sim \mathrm{Unif}\{-1, +1\}, \tag{13}$$

$$\text{Replacement:} \quad \tilde{r}_t = (1 - z_t) r_t + z_t r_j^{\mathrm{ref}}, \qquad r_j^{\mathrm{ref}} \sim \hat{p}_{\mathrm{ref}}(r), \tag{14}$$

$$\text{Sign flip:} \quad \tilde{r}_t = (1 - 2z_t) r_t. \tag{15}$$

For short and long bursts, a burst-start process is tuned to the target realized fraction; length is sampled uniformly from {2,3,4} or {8,...,16}, respectively, and one sign ξ_b and magnitude $\kappa_b \sigma_r$ are held constant within each burst. Local temporal shift is a stress-only corruption: for selected timesteps, $\tilde{r}_t = r_{t+\Delta}$ with Δ sampled uniformly from $\{-4, -3, -2, 2, 3, 4\}$ subject to episode boundaries. Marginal replacement samples from the fixed benchmark-only reference buffer, so its reward source is identical across methods and never enters learner updates. Table 7 summarizes the corruption families and their roles in core, OOD, and stress evaluation.

Support defaults in Table 8 are fixed before mechanism-OOD and held-out-task evaluation and reused across paired backbones. Backbone-specific learner hyperparameters, including target-network updates and learner batch sizes, follow the corresponding original/default implementation and are held identical within each baseline versus baseline+RECONRL pair.

Table 9 reports per-task retention while keeping development and held-out transfer tasks explicit.

Table 7: Corruption families and fixed benchmark roles.

Mechanism	Realized fraction	Scale / source	Length	Role
Impulse	5/15/30% core; 15% Stage-1	$2\sigma_r$	1	core / Stage-1 train
Short burst	5/15/30% core; 15% Stage-1	$2\sigma_r$	2–4	core / Stage-1 train
Marginal replacement	5/15/30% core; 15% OOD	$\hat{p}_{\mathrm{ref}}(r)$	1	core + mechanism OOD
Sign flip	15% OOD	native	1	mechanism OOD
Long burst	15% OOD	$2\sigma_r$	8–16	mechanism OOD
Unseen scale	15% OOD	4–$5\sigma_r$	1	mechanism OOD
Local temporal shift	5/15/30%	temporal	1	stress only

Table 8: Default RECONRL support hyperparameters.

Parameter	Default	Sweep
Guard g	2	{1,2,4,8}
Context horizon H	16	{8,16,32}
Negatives K	3	{1,3,7}
Support floor $w_{\min}$	0.10	[0,0.30]
Sharpness β	2	[0.5,4]
Contamination bound $\hat{\epsilon}$	0.30	{0.10,0.20,0.30,0.40,0.50}
Trim quantile	0.70	fixed by $1-\hat{\epsilon}$
Support LR	3×10^{-4}	$\{1,3,10\}\times10^{-4}$
Fusion weight λ_f	1.0	{0.5,1,2}
Fusion warm-up N_{warm}	5k updates	{2k,5k,10k}
Support updates / learner update	1	{0.5,1,2}
Support batch size	256	{256,512}

Tables 13, 14, 15, 16, and 17 report synthetic-negative ablations, rarity-stratified trust, alternative fusion rules, compute and memory overhead, and additional discrete-control trust statistics, respectively.

Rarity Stratification. Reward rarity is defined by marginal density for continuous rewards and empirical frequency for discrete rewards. For continuous rewards we fit a one-dimensional Gaussian KDE once to the fixed task-level reference buffer $\mathcal{D}_{\mathrm{ref}}^{\mathrm{task}}$ and rank each observed reward by $\hat{p}_{\mathrm{ref}}(\tilde{r})$. Bins are defined once per task by reference-buffer density percentiles, reused for every method and seed, and used only for post-hoc evaluation, never by RECONRL.

B ADDITIONAL REPRODUCIBILITY NOTES

Bootstrap draws, per-seed returns, and performance-profile coordinates accompany the compact interval summary in Table 18; Table 19 reports the scalar trust diagnostics that accompany the full CRA/GRR operating curves. Implementations additionally log the latent corruption mask and clean reward to an evaluation-only stream that is never exposed to training; temporal negatives are sampled from the same task and replay phase with minimum separation $2H$, and transition negatives use an independent permutation. During the first N_{warm} support updates the backbone uses $w_t = 1$. After warm-up, each learner update is followed by one temporal-branch update, one transition-branch update, and one fusion-contrastive update; the fusion objective receives stop-gradient branch probabilities. For mechanism OOD, Stage 1 chooses impulse versus short burst independently per episode with probability 0.5 at fixed 15% realized corruption; Stage 2 seen-reference numbers are averaged over fresh learners trained separately under 15% impulse and 15% short burst. For held-out-task transfer, all learned parameters of the temporal branch, transition branch, and fusion layer are reinitialized and trained from unlabeled replay on the new task; only architecture, hyperparameters, threshold-selection rule, and training protocol transfer unchanged. All support checkpoints, corruption-generator seeds, environment seeds, and paired-backbone seeds are retained for audit and exact reruns.

Table 9: Per-task return retention for continuous-control development and held-out transfer tasks. Hum and Manip denote held-out transfer tasks; the remaining columns are development tasks.

Method	HC	Hop	Walk	Ant	Hum	Cheetah	Quad	Manip
SAC	0.721	0.642	0.779	0.688	0.603	0.821	0.738	0.721
SAC + ReConRL	0.823	0.736	0.846	0.826	0.734	0.906	0.789	0.826
CrossQ	0.772	0.703	0.824	0.806	0.651	0.858	0.787	0.758
CrossQ + ReConRL	0.848	0.778	0.890	0.858	0.642	0.851	0.835	0.861
SimBa	0.816	0.768	0.861	0.794	0.704	0.893	0.832	0.803
SimBa + ReConRL	0.886	0.762	0.923	0.906	0.806	0.968	0.883	0.892

Table 10: Reference algorithms under the common corruption protocol. Compute is relative training GPU cost under the common harness, normalized to SAC $= 1.0$.

Method	IQM	$Q_{0.10}$	Impulse	Burst	Replacement	Compute
TD-MPC2	0.795	0.632	0.821	0.744	0.802	1.56
DreamerV3	0.771	0.605	0.804	0.729	0.781	1.74
SimbaV2	0.826	0.671	0.851	0.793	0.829	1.22
FoG	0.842	0.688	0.864	0.806	0.847	1.39
Annealed-Q	0.811	0.684	0.823	0.767	0.856	0.98
SPO	0.832	0.701	0.839	0.789	0.872	1.11
FPG	0.806	0.655	0.844	0.781	0.789	1.31

Table 11: Past-only temporal branch for causal online deployment.

Variant	Return	CMT	GMD	CleanTrust	AUROC	OOD
Past + future	0.889	0.220	0.112	0.930	0.948	0.817
Past only	0.856	0.274	0.141	0.914	0.903	0.776
Transition only	0.832	0.309	0.128	0.912	0.869	0.754

Table 12: Sensitivity to the fixed contamination upper bound.

$\hat{\epsilon}$	0.10	0.20	**0.30**	0.40	0.50
Return retention	0.836	0.874	0.889	0.878	0.861
CMT	0.318	0.248	0.220	0.228	0.198
CleanTrust	0.939	0.934	0.930	0.916	0.889
RV−RC gap	0.515	0.604	0.660	0.647	0.655

Table 13: Synthetic-negative ablations. Matched gap is the RV−RC trust difference on the matched diagnostic.

Negatives	Matched gap	CMT	CleanTrust	Return	OOD
Marginal shuffle	0.571	0.273	0.927	0.863	0.744
+ local temporal replacement	0.660	0.220	0.930	0.889	0.817
+ scale perturbation	0.637	0.191	0.871	0.851	0.772

Table 14: Trust stratified by marginal-reward rarity.

Method	Rare C	Rare X	Mid C	Mid X	Common C	Common X	Trust gap
Full ReConRL	0.900	0.210	0.940	0.200	0.950	0.240	0.713
Temporal only	0.860	0.340	0.910	0.320	0.930	0.370	0.557
Transition only	0.880	0.310	0.920	0.280	0.930	0.330	0.603
Rarity filter	0.480	0.440	0.740	0.690	0.930	0.860	0.053

Rare = lowest-density 0–5%; mid = 5–50%; common = 50–100%; C/X = clean/corrupted; Trust gap = mean(C−X) across the three strata.

Table 15: Fusion-rule ablations beyond the main monotonicity test.

Fusion	Ret	RV−RC gap	CMT	CleanTrust	AUROC	OOD Ret
Monotone learned	0.889	0.660	0.220	0.930	0.948	0.817
Unconstrained linear	0.873	0.610	0.287	0.902	0.925	0.760
Mean	0.861	0.556	0.253	0.921	0.914	0.775
Max	0.849	0.492	0.334	0.945	0.876	0.724
Product	0.835	0.621	0.171	0.842	0.919	0.751

Table 16: Compute and memory overhead under matched interaction budgets.

Method	Params (M)	Updates/step	GPU h/task	FPS	Replay GB	IQM/GPUh
SAC	0.54	1.0	3.8	17800	1.8	0.193
SAC + ReConRL	1.21	4.0	5.4	12500	2.2	0.151
CrossQ	0.81	1.0	4.4	15900	1.9	0.181
CrossQ + ReConRL	1.48	4.0	6.0	11300	2.3	0.141
SimBa	2.43	2.0	6.1	12100	2.1	0.135
SimBa + ReConRL	3.10	5.0	7.8	9300	2.5	0.114

Table 17: Additional discrete-control trust metrics by backbone.

Method	Pong	Breakout	Seaquest	Q*bert	CMT	CleanTrust	AUROC	IQM
DQN	0.708	0.541	0.632	0.579	1.000	1.000	0.500	0.606
DQN + ReConRL	0.821	0.690	0.712	0.755	0.273	0.908	0.924	0.742
QR-DQN	0.751	0.612	0.681	0.645	1.000	1.000	0.500	0.663
QR-DQN + ReConRL	0.744	0.758	0.806	0.789	0.238	0.921	0.941	0.776

Table 18: Aggregate uncertainty for paired backbone comparisons. CIs are 95% stratified bootstrap intervals for the paired aggregate difference. $P(\text{improvement})$ denotes the paired probability-of-improvement statistic across matched outcomes and is distinct from the bootstrap sign probability of the aggregate IQM difference.

Comparison	Base IQM	+ReConRL IQM	Δ 95% CI	$P(\text{improvement})$
SAC, development	0.732	0.818	[0.049, 0.122]	0.94
CrossQ, development	0.797	0.846	[0.010, 0.086]	0.84
SimBa, development	0.826	0.889	[0.024, 0.103]	0.90
DQN, discrete	0.606	0.742	[0.071, 0.202]	0.96
QR-DQN, discrete	0.663	0.776	[0.041, 0.177]	0.91
SAC, held-out tasks	0.662	0.779	[0.049, 0.181]	0.92
CrossQ, held-out tasks	0.705	0.756	[-0.012, 0.113]	0.77
SimBa, held-out tasks	0.754	0.848	[0.032, 0.151]	0.90

Table 19: Matched-diagnostic operating statistics. AUROC/AUPRC use corruption score $1 - w_t$ with $z_t = 1$ as the positive label; CRA and GRR use the prespecified $\eta = 0.5$ operating point.

Method	AUROC ↑	AUPRC ↑	ECE ↓	CRA@.5 ↓	GRR@.5 ↓
Rarity filter	0.585	0.314	0.168	0.643	0.207
TD-error filter	0.742	0.481	0.121	0.421	0.156
Temporal only	0.802	0.697	0.093	0.267	0.104
Transition only	0.921	0.822	0.102	0.194	0.137
Mean fusion	0.847	0.761	0.071	0.226	0.081
Full ReConRL	0.903	0.836	0.058	0.142	0.067