BRM: BETTER REWARD MODELING VIA CRITIC FEATURE TRANSFER IN CONTINUOUS CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Preference-based Reinforcement Learning (PbRL) replaces hand-crafted reward functions by learning a reward model from pairwise trajectory preferences, achieving good performance in several continuous control tasks. However, the generalization of the reward model quickly becomes a bottleneck of the pipeline: it has to predict rewards for the millions of visited state-action pairs from a few hundred preference trajectories. This data asymmetry results in the reward model overfitting to the labeled data, ultimately degrading final performance. In this work, we address this issue by turning this data asymmetry into an opportunity. We propose BRM (**B**etter **R**eward **M**odeling), which copies the features of the critic, trained on the full replay buffer, and uses them as an input to the reward model, trained on only a few hundred labeled preferences. Across twelve tasks from the DeepMind Control Suite and MetaWorld and against seven PbRL baselines, BRM reaches a task-normalized Interquartile Mean (IQM) of 0.85 versus 0.55 for the strongest of them. Our empirical analysis indicates that BRM's gains are consistent with better reward generalization, most clearly on trajectories held out from the preference distribution. Our results suggest that the critic the agent already trains is an underused starting point for the reward model in existing PbRL methods.

1 INTRODUCTION

Reinforcement learning relies on a reward signal, and reward design can often be the hardest part of the problem since hand-crafting a scalar return that captures the intended task is challenging (Ng et al., 1999; Amodei et al., 2016). Even with a carefully designed reward for a given objective, an agent can end up hacking it, meaning that it finds a way to maximize it while behaving in a different unintended way (Skalse et al., 2022).

Preference-based reinforcement learning (PbRL) avoids the design problem and tries to mitigate the reward hacking issue by asking a user for a higher-level signal: pairwise comparisons of an agent's trajectories (Christiano et al., 2017; Wirth et al., 2017; Lee et al., 2021b;a). The pipeline is simple: (1) a policy collects trajectories, starting from random exploration (2) the user is asked for feedback on a sample of these trajectories (3) a reward model, usually a neural network, is trained on the resulting comparisons (4) this reward model guides policy learning, replacing the engineered return with one inferred from the preferences and (5) the loop repeats from step 2, drawing new trajectories from the improved policy.

Avoiding explicit reward specification comes with a cost: user labeling time is expensive, which limits how much feedback we can collect. This creates what we call the *data asymmetry*: the learning agent is training a value function (the critic, typically Soft Actor-Critic (SAC) (Haarnoja et al., 2018) for continuous control) on every environment transition, while the reward model is trained on only a few preferences. The reward model must therefore generalize from a few hundred short trajectory comparisons to millions of state-action pairs the policy visits during training, making it the natural bottleneck of the pipeline. Trained on scarce supervision, the reward network fits features that correlate with preferences in distribution but fail once the policy moves elsewhere (Tien et al., 2023).

Previous PbRL methods have generally addressed this generalization problem by improving the supervision, whether through pseudo-labels and temporal cropping (Park et al., 2022; Hu et al., 2024), query selection (Liang et al., 2022; Zhu et al., 2025), or regularization of the reward and policy

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.

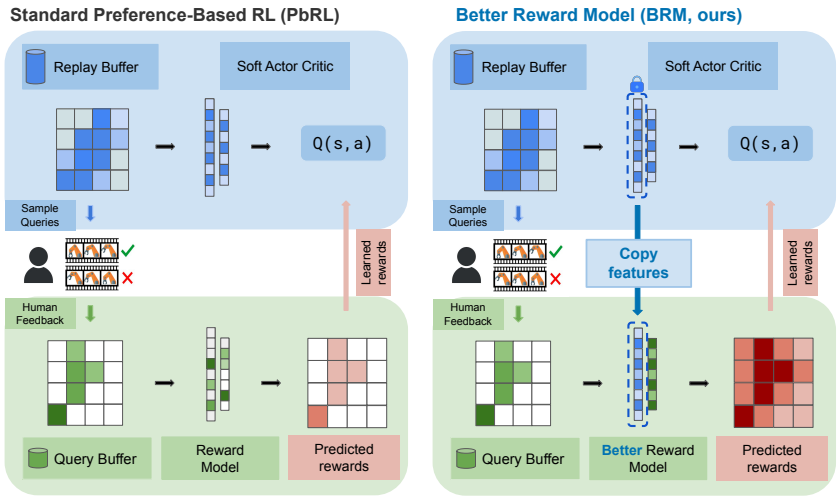

Figure 1: **BRM reuses frozen critic features for reward modeling.** The SAC critic trains on an order of magnitude more data than the reward model. **Left:** standard PbRL: reward model as an independent MLP that overfits on scarce preferences. **Right:** BRM copies the critic's frozen first-layer features $\varphi(\mathbf{s}, \mathbf{a})$ into the reward model, yielding more accurate rewards away from the labeled segments.

losses (Bai et al., 2026). These methods change what the reward model is trained on, but leave its input representation unchanged.

A different direction, underexplored on the reward-modeling side of PbRL, is transfer learning. In supervised learning, a standard way to generalize from limited labels is to reuse features from a network trained on a related task with more data (Yosinski et al., 2014; Kornblith et al., 2019).

Where would such a network come from in PbRL? The critic is a natural candidate. First, the tasks are related, since the value is the discounted sum of rewards. Second, the critic is trained on the entire replay buffer, an order of magnitude more data than the reward model uses. And this does not require additional training, since the critic is trained as part of the RL algorithm.

BRM (**B**etter **R**eward **M**odeling) does this. We freeze the first-layer features of the agent's critic and use them as the input features to the reward model (Figure 1). We hypothesize that these features, trained on all state-action pairs the policy visits, should help the reward model generalize beyond the few states observed in the preferences it is trained on.

Our contributions are as follows:

1. We propose BRM (Section 4), which reuses the first-layer features of the agent's critic as input to the reward model, with no auxiliary loss, no additional environment interactions, and no change to the policy or critic.

2. Across twelve DeepMind Control Suite and MetaWorld tasks, BRM improves over seven PbRL baselines in aggregate, and the same modification improves three different PbRL algorithms when added to them (Section 5).

3. We ablate the source of the gain, showing that it comes from the critic features themselves rather than from added capacity or dimensionality, and that BRM's reward-prediction accuracy improves most on trajectories held out from the preference distribution (Section 5).

Beyond the immediate gains, our results suggest that transfer learning between RL components is a practical approach for PbRL, applicable to off-policy actor-critic pipelines that already train a critic alongside the reward model.

2 Related Work

Preference-based reinforcement learning. Christiano et al. (2017) brought PbRL (Wirth et al., 2017) to the deep-network setting: learn a scalar reward from pairwise trajectory comparisons under a Bradley-Terry loss, with the reward model as a standalone multilayer perceptron (MLP). Most subsequent methods keep this architecture and improve the supervision: PEBBLE pretrains the agent and relabels stale buffers (Lee et al., 2021b), SURF adds pseudo-labels (confident reward-model predictions on unlabeled pairs) and temporal cropping (training on random sub-segments of labeled pairs) (Park et al., 2022), RUNE uses reward-uncertainty exploration (Liang et al., 2022), MRN meta-learns reward parameters through implicit differentiation, with the outer loss measuring Q-function accuracy on preferences (Liu et al., 2022), QPA selects preference queries close to the current policy's visitation distribution (Hu et al., 2024), RIME filters out noisy preference labels with a divergence-based criterion (Cheng et al., 2024), and PPE shapes the preference buffer for coverage (Zhu et al., 2025). Preference Transformer (Kim et al., 2023) departs from the MLP and models segment preferences as a learned weighted sum of non-Markovian rewards via a transformer. These vary what the reward model sees, how it is queried, or how preferences are aggregated; most leave the reward model's input representation untouched, the axis on which BRM acts. The closest exception is REED, which learns the reward model's state-action representation with a self-supervised temporal-consistency objective that encodes environment dynamics (Metcalf et al., 2023); BRM instead reuses the critic's existing features directly, with no auxiliary loss. We verify this orthogonality by composing BRM with QPA, PEBBLE, and RUNE (Section 5.3).

Regularization in preference-based reward learning. Supervised learning of a neural network on a small dataset is prone to overfitting (Zhang et al., 2017; Goodfellow et al., 2016). Standard remedies include weight decay (Krogh & Hertz, 1991), dropout (Srivastava et al., 2014), and data augmentation (Hernández-García & König, 2018; Shorten & Khoshgoftaar, 2019). With only a few hundred preferences, the reward model in the PbRL setup is no exception to it. Explicit regularization has been tried in this context too: STAR combines a bounded margin on the preference loss with a conservative Q-estimate that constrains policy updates, adding regularization to both the preference learning and the policy learning objectives (Bai et al., 2026). STAR also uses the Q-function, but to constrain the policy, while BRM uses the critic's features as input to the reward model. On the data-augmentation side, SURF and QPA crop trajectory segments to enlarge the preference set (Park et al., 2022; Hu et al., 2024).

Transfer learning and frozen pretrained features. A long line of work in supervised learning shows that reusing features from a well-trained related task is one of the most reliable ways to generalize from limited labels: Yosinski et al. (2014) quantify per-layer transferability in CNNs, DeCAF (Donahue et al., 2014) reuses mid-level ImageNet activations as off-the-shelf features for downstream recognition, and Razavian et al. (2014) show frozen CNN features rival task-specific systems on recognition benchmarks. The pattern scales to foundation models: CLIP (Radford et al., 2021) learns visual encoders from image-text pairs that transfer zero-shot to diverse downstream tasks. In RL, the same idea drives work that freezes externally pretrained vision encoders for downstream policy learning: PVR (Parisi et al., 2022) shows ImageNet-pretrained encoders match state-based input for control, and R3M (Nair et al., 2022) pretrains a manipulation-oriented encoder on egocentric human video. BRM brings this transfer logic to the reward side of PbRL, freezing the agent's own SAC critic rather than an externally pretrained encoder.

Representation learning in reinforcement learning. A separate line of work learns auxiliary representations for the policy or value function. These methods attach contrastive (Laskin et al., 2020), temporal-coherence (Stooke et al., 2021), self-predictive (Schwarzer et al., 2021), or prototype-based (Yarats et al., 2021) objectives to observation encoders; pixel-domain agents lean on data augmentation (Yarats et al., 2022). Bisimulation encoders (Zhang et al., 2021) respect reward-aware metrics; auxiliary-task agents share representations across heads via self-supervised objectives (Jaderberg et al., 2017); successor features decompose value into reusable bases across tasks via an explicit basis-and-weights factorization (Barreto et al., 2017), in contrast to BRM's implicit reuse of features the critic grows by default. BRM differs in three ways: it reuses *state-action* features, not observation-only encoders; it adds *no auxiliary loss*, since the features are already learned by the SAC critic; and it targets the *reward model*, not the policy or value function.

3 PRELIMINARIES

In this section we provide background on Preference-based RL, focusing on two main components of BRM, the reward function and the critic, before concluding with prior results relating the two.

Markov decision process. We work in the standard MDP setting (Puterman, 1994) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, T, \gamma)$ with continuous state and action spaces $\mathcal{S} \subseteq \mathbb{R}^{d_s}$, $\mathcal{A} \subseteq \mathbb{R}^{d_a}$, reward r, transition T, and discount $\gamma \in (0, 1)$; the goal is to find a policy π maximizing expected discounted return $\mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)\right]$, where the expectation is w.r.t. random variables $\mathbf{s}_t$ and $\mathbf{a}_t$ following the state-action distribution induced by π. To do so, it is often useful to compute the Q-function

$$Q^\pi(\mathbf{s}, \mathbf{a}) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t) \middle| \mathbf{s}_0 = \mathbf{s}, \mathbf{a}_0 = \mathbf{a} \right], \tag{1}$$

that provides the return for a given choice of policy and initial state-action pair.

Preference-based RL. In PbRL the reward r is not observed. Instead, a user provides binary preferences $y \in \{0, 1\}$ over pairs of trajectory segments (σ^0, σ^1), where each segment $\sigma = \{(\mathbf{s}_t, \mathbf{a}_t)\}_{t=1}^{H}$ is a short sequence of state-action pairs. A reward model $\hat{r}_\psi(\mathbf{s}, \mathbf{a})$ is fit to be consistent with the preferences through the Bradley-Terry model (Bradley & Terry, 1952):

$$P_\psi[\sigma^1 \succ \sigma^0] = \frac{\exp \sum_t \hat{r}_\psi(\mathbf{s}_t^1, \mathbf{a}_t^1)}{\sum_{i \in \{0,1\}} \exp \sum_t \hat{r}_\psi(\mathbf{s}_t^i, \mathbf{a}_t^i)}. \tag{2}$$

The parameters ψ (e.g. the weights of a neural network) are obtained by minimizing the cross-entropy loss over the preference dataset $\mathcal{D}_{\text{pref}}$:

$$\mathcal{L}_{\text{pref}}(\psi) = -\mathbb{E}_{(\sigma^0, \sigma^1, y) \sim \mathcal{D}_{\text{pref}}} \left[y \log P_\psi[\sigma^1 \succ \sigma^0] + (1 - y) \log P_\psi[\sigma^0 \succ \sigma^1] \right]. \tag{3}$$

In standard PbRL, $\hat{r}_\psi$ is a small multilayer perceptron, often combined with an ensemble of K heads for uncertainty estimation (Lee et al., 2021b). The reward model takes the place of the unobserved r inside an off-policy actor-critic loop. Training proceeds over M feedback rounds: at round $m \in \{1, \ldots, M\}$, N new segment pairs are selected from a replay buffer $\mathcal{D}$ of the agent's past transitions for the user to label, and the reward model is refit; we write $\hat{r}_m$ for the reward model in use at round m.

Off-policy actor-critic RL. This class of RL algorithms maintains a parametric Q-function $Q_\theta(\mathbf{s}, \mathbf{a})$ and a parametric policy $\pi_\phi(\mathbf{a} \mid \mathbf{s})$, optimized by alternating policy evaluation and policy improvement on transitions $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ stored in the replay buffer $\mathcal{D}$. Policy evaluation enforces Q_θ to be consistent with the Bellman operator,

$$\mathcal{L}_Q(\theta) = \mathbb{E}_{(\mathbf{s}, \mathbf{a}, \mathbf{s}') \sim \mathcal{D}} \left[\left(Q_\theta(\mathbf{s}, \mathbf{a}) - r(\mathbf{s}, \mathbf{a}) - \gamma \mathbb{E}_{\mathbf{a}' \sim \pi_\phi} \left[Q_{\bar{\theta}}(\mathbf{s}', \mathbf{a}') \right] \right)^2 \right], \tag{4}$$

where $Q_{\bar{\theta}}$ is a target network, while policy improvement updates π_ϕ to maximize Q_θ on states sampled from $\mathcal{D}$. We instantiate this loop with Soft Actor-Critic (SAC) (Haarnoja et al., 2018; 2019), which adds a maximum-entropy term to the policy objective and trains two critics $Q_{\theta_1}, Q_{\theta_2}$ to reduce overestimation, and use it as our backbone RL solver in the PbRL loop, replacing r with $\hat{r}_\psi$.

Reward-predictive features are good Q-function features. Several deep RL methods augment the Q-function learning objective (Eq. 4) with auxiliary losses capturing the transition and reward functions of the MDP (Gelada et al., 2019; Schwarzer et al., 2021; Zhang et al., 2021). The rationale behind this line of work is that a representation enabling reward and transition approximation provably upper bounds the approximation error of the Q-function. For example, in the DeepMDP work of Gelada et al. (2019), it is shown (Lemma 3) that given a continuous feature map $\varphi : \mathcal{S} \mapsto \mathbb{R}^d$ that induces approximation errors L_φ^R and L_φ^T for the reward and transition functions respectively, the approximation error between the true Q-function Q^π and an approximate Q-function $\tilde{Q}_\varphi^\pi$ will be upper bounded by

$$\mathbb{E}_\pi |Q^\pi(\mathbf{s}, \mathbf{a}) - \tilde{Q}_\varphi^\pi(\mathbf{s}, \mathbf{a})| \leq \frac{(L_\varphi^R + \gamma L_{\tilde{Q}} L_\varphi^T)}{1 - \gamma}, \tag{5}$$

where $L_{\tilde{Q}}$ is the Lipschitz constant of $\tilde{Q}^{\pi}_{\varphi}$ induced by the choice of φ. Similar results were proved in the approximate state abstraction setting (see e.g. Sec. 5.2 of Abel et al. (2016)), relating reward and transition error to Q-function approximation error. In short, a representation that predicts reward and transitions well is enough to approximate the Q-function: features that serve the reward also serve the value function.

4 Better Reward Modeling via Critic Feature Transfer

What should the reward model reuse from the critic, and how? We consider this question in three steps: why sharing features is natural yet challenging in PbRL (Sec. 4.1), the resulting reward architecture (Sec. 4.2), and the training protocol (Sec. 4.3).

4.1 Value and Reward Feature Sharing

On the challenges of feature sharing in PbRL. Given the results of Eq. 5, a natural yet under-explored idea in PbRL would be to share the reward and Q-function features. Indeed, in PbRL a reward model is learned by default, and sharing its features with the Q-function might improve the Q-function's approximation error. The PbRL data asymmetry also opens a reverse direction the bound does not cover: since the dataset size for training the Q-function is an order of magnitude larger than the preference data available for learning the reward model, Q-function features might benefit the reward model. BRM pursues this reverse direction, motivated by the data asymmetry.

In standard RL, φ is trained against a single reward r through the Bellman target, and r is fixed for the whole run. PbRL proceeds over feedback rounds though, where each round adds N new preference queries, after which the reward model is refit before the next round. We write φ_m for the critic features extracted while Q_θ is trained under $\hat{r}_m$. Feature sharing in PbRL would reuse φ_m, trained only to support $\hat{r}_m$, as input once round $m{+}1$ arrives, to help fit $\hat{r}_{m+1} \neq \hat{r}_m$. Unlike standard RL, φ_m is not targeting the same fixed r anymore. A natural choice would be to share the last layer, since it is the most specialized to predict $\hat{r}_m$. In PbRL, however, this might not be the case, since features shaped tightly around $\hat{r}_m$ may be too specialized to transfer to $\hat{r}_{m+1}$, so naive last-layer sharing might degrade performance.

Early layers transfer better. In the supervised learning community, it was shown that generality of representations negatively correlates with depth: early layers transfer across tasks better than late ones (Yosinski et al., 2014), as the latter specialize toward whatever targets the head fits (here, the current Q and reward models).

We observe a similar effect in PbRL. Overall, first-layer critic features make a better input to the reward model than second-layer ones (Section 5.4), even though the latter are often more linearly correlated with the true r (Appendix C.2). We therefore share the first layer of the Q-function and not the last one.

When critic features are not reward-informative. Early in training the critic has been trained on little data, so its frozen features contain little reward signal, and a reward model built on them alone might have nothing to learn from. We therefore keep a direct path from the raw state-action pair to the reward head, alongside the shared features. The reward model falls back on this path while the features are weak and uses them once they help. We detail this skip connection in Section 4.2.

4.2 Reward Model Architecture

Frozen critic features. We extract the first hidden layer of the SAC critic Q_{θ_1} and define the *frozen critic features*

$$\varphi(\mathbf{s}, \mathbf{a}) = \mathrm{ReLU}(W_Q[\mathbf{s}; \mathbf{a}] + b_Q) \in \mathbb{R}^{d_h}, \tag{6}$$

with $W_Q \in \mathbb{R}^{d_h \times (d_s + d_a)}$, $b_Q \in \mathbb{R}^{d_h}$, with the width d_h inherited from the SAC critic's. These parameters are not updated by the preference loss $\mathcal{L}_{\text{pref}}$ and evolve only through the SAC critic updates of Eq. 4. The frozen features are updated upon receiving new feedback from the user and before fitting a new reward model.

Skip connection. Early in training, before the reward model and the critic have converged upon receiving sufficient preference information, φ may contain little reward-relevant information. We therefore concatenate it with the raw observation-action pair before the reward head, a concatenative skip connection:

$$\mathbf{z}(\mathbf{s}, \mathbf{a}) = [\varphi(\mathbf{s}, \mathbf{a}) \; ; \; \mathbf{s} \; ; \; \mathbf{a}] \in \mathbb{R}^{d_h + d_s + d_a}. \tag{7}$$

The head can collapse the φ-block to zero and recover the from-scratch input a vanilla MLP reward model would have seen, so it is at least as expressive as an MLP on $(\mathbf{s}, \mathbf{a})$ alone; whenever φ is informative, the head can use it. The critic is trained on the reward model's output, and the reward model takes the critic's features as input. Appendix B.2 discusses why this circular dependency is harmless, and a dual-critic ablation (Appendix D.2) finds no significant effect of decoupling the two.

Ensemble reward head. On top of $\mathbf{z}$ we train an ensemble of $K = 3$ heads, as in PEBBLE (Lee et al., 2021b) and RUNE (Liang et al., 2022). Each head $\hat{r}_{\psi_k}$ is a two-layer MLP,

$$\hat{r}_{\psi_k}(\mathbf{s}, \mathbf{a}) = \tanh\Big(W_k^{(2)} \, \mathrm{ReLU}\Big(W_k^{(1)} \mathbf{z}(\mathbf{s}, \mathbf{a}) + b_k^{(1)}\Big) + b_k^{(2)}\Big), \tag{8}$$

with $W_k^{(1)} \in \mathbb{R}^{256 \times (d_h + d_s + d_a)}$ and $W_k^{(2)} \in \mathbb{R}^{1 \times 256}$. The ensemble mean is the reward signal sent to SAC. The variance can be used as an uncertainty estimate for exploration (Liang et al., 2022).

4.3 BRM Training Integration

BRM integrates into an off-policy actor-critic PbRL loop with a single change to the reward model (two lines in Algorithm 1). The critic features φ are not part of the reward optimizer; they evolve through SAC's standard critic updates and the reward model inherits the improvement at no additional cost. Each time new preference labels are queried, we re-extract φ from the current Q_{θ_1}, reinitialize all K ensemble heads from scratch, and retrain them on the full preference buffer. The reset is cheap (we cap reward training at the same gradient-step budget as the baselines, see Appendix A.1) and helps since heads trained against φ_m would receive a different basis once φ_{m+1} replaces it. Algorithm 1 records the surrounding loop.

Algorithm 1 BRM integrated into a PbRL training loop. Colored lines indicate the only modifications to the base PbRL algorithm.

Initialize π_ϕ, Q_{θ_1}, Q_{θ_2}, heads $\{\hat{r}_{\psi_k}\}_{k=1}^K$, replay $\mathcal{D}$, preferences $\mathcal{D}_{\text{pref}} \leftarrow \emptyset$
Explore with an intrinsic reward $\hat{r}_1$ (Lee et al., 2021b)
$m \leftarrow 1$
for each training iteration **do**
  Step $\pi_\phi, Q_{\theta_1}, Q_{\theta_2}$ via SAC using $\hat{r}_m$ as reward; append transition to $\mathcal{D}$
  if time to query round m, and $m \leq M$ **then**
    Select N segment pairs, obtain preference labels, add to $\mathcal{D}_{\text{pref}}$
    Refresh frozen features $\varphi_m(\cdot) \leftarrow \mathrm{ReLU}(W_Q[\cdot] + b_Q)$ from Q_{θ_1}
    Reinitialize heads $\{\hat{r}_{\psi_k}\}_{k=1}^K$, then train on $\mathcal{D}_{\text{pref}}$ with input $\mathbf{z} = [\varphi_m(\mathbf{s}, \mathbf{a}); \mathbf{s}; \mathbf{a}]$ via Eq. 3, giving $\hat{r}_{m+1}$
    $m \leftarrow m + 1$
  end if
end for

5 Experiments

We evaluate BRM on twelve control tasks against seven PbRL baselines. Our experiments address three questions:

1. Does BRM improve over prior PbRL methods? (Section 5.2)
2. Does it improve other PbRL algorithms when added to them? (Section 5.3)
3. Is the gain due to the feature source, and does the reward model generalize better away from the labeled segments? (Section 5.4)

5.1 Experimental Setup

Environments. We evaluate on six DeepMind Control Suite (DMC) tasks (Tassa et al., 2018) (walker-walk, cheetah-run, walker-run, quadruped-walk, quadruped-run, humanoid-stand) and six MetaWorld (MW) manipulation tasks (Yu et al., 2020) (door-unlock, door-open, drawer-open, button-press, window-open, sweep-into). DMC reports episode return; MW reports success rate.

Baselines and protocol. We compare against seven PbRL baselines grouped by what they change. QPA (Hu et al., 2024), PPE (Zhu et al., 2025), and RUNE (Liang et al., 2022) act on query selection or exploration. SURF (Park et al., 2022) adds pseudo-labels and temporal cropping. RIME (Cheng et al., 2024) filters noisy preference labels. STAR (Bai et al., 2026) regularizes both the reward loss and the policy updates. PEBBLE (Lee et al., 2021b) is the common backbone. All baselines use their published 3-layer MLP reward model with 256 hidden units. PEBBLE, RUNE, SURF, PPE, RIME, and STAR use a $K{=}3$ ensemble, while QPA uses $K{=}1$ with $20\times$ temporal-cropping augmentation. BRM replaces the reward model's input with $d_h{=}1024$ frozen critic features and uses a $K{=}3$ ensemble of two-layer heads. All methods share the same SAC backbone and a scripted preference oracle (Lee et al., 2021b;a), and we re-run all of them under this shared protocol. We report mean $\pm$ standard error (SE) over 10 seeds (5 for ablations) and use Welch's t-test ($p < 0.05$) for significance. Aggregate scores are task-normalized IQM (Agarwal et al., 2021), with each DMC return divided by 1000 and each MetaWorld success rate divided by 100, and a 95% stratified-bootstrap confidence interval. Baselines use their published hyperparameters; BRM introduces no additional ones. Hyperparameter details are in Appendix A.1.

5.2 Does BRM Improve over Prior PbRL Methods?

Figure 2 shows the learning curves and the aggregate IQM across all twelve tasks. BRM achieves a task-normalized IQM (Agarwal et al., 2021) of 0.85 versus 0.55 for QPA, with a 95% confidence interval that does not overlap with any other method's. We provide per-baseline significance tests in Appendix A.3.

On DMC, BRM improves over QPA on all six tasks (Table 1). On MetaWorld, the largest improvement is on door-unlock, where BRM roughly doubles the success rate. STAR is above BRM on door-open and drawer-open, but neither gap is statistically significant (Appendix A.3). STAR performs below the other baselines on most DMC tasks; we note that it is reimplemented from its paper, which evaluates it mostly offline and under different feedback budgets (Appendix A.1). Drawer-open is a saturation case. PEBBLE already reaches 71%, and BRM matches but does not improve, suggesting the reward model is near the ceiling under the current query budget.

Table 1: **BRM is best on ten of twelve tasks** (mean $\pm$ SE, 10 seeds). BRM uses QPA as backbone. DMC: episode return. MW: success rate (%). Best result per task in **bold**. Last column: normalized gain over QPA, (BRM $-$ QPA)/QPA. Per-baseline significance tests in Appendix A.3.

	Task	PEBBLE	RUNE	SURF	QPA	PPE	RIME	STAR	BRM (ours)	vs. QPA
DMC (return)	walker-walk	656 ± 48	568 ± 46	526 ± 43	751 ± 50	645 ± 57	542 ± 51	314 ± 42	$\mathbf{929 \pm 18}$	+23.7%
	cheetah-run	512 ± 47	429 ± 70	366 ± 43	489 ± 67	600 ± 53	305 ± 47	457 ± 23	$\mathbf{704 \pm 22}$	+43.9%
	walker-run	556 ± 19	483 ± 26	259 ± 37	617 ± 47	536 ± 43	451 ± 48	213 ± 28	$\mathbf{684 \pm 62}$	+10.8%
	quadruped-walk	348 ± 50	381 ± 65	239 ± 30	523 ± 94	623 ± 72	363 ± 66	284 ± 40	$\mathbf{838 \pm 46}$	+60.1%
	quadruped-run	297 ± 35	352 ± 39	217 ± 43	409 ± 44	473 ± 43	324 ± 52	178 ± 19	$\mathbf{640 \pm 45}$	+56.6%
	humanoid-stand	111 ± 39	131 ± 40	144 ± 26	295 ± 60	517 ± 16	44 ± 32	13 ± 6	$\mathbf{726 \pm 37}$	+146.3%
MetaWorld (%)	door-unlock	82.9 ± 8.7	71.5 ± 10.3	79.1 ± 9.1	47.8 ± 10.5	53.5 ± 15.1	28.4 ± 12.6	65.5 ± 14.0	$\mathbf{94.4 \pm 5.4}$	+97.5%
	door-open	53.1 ± 14.7	66.2 ± 14.3	68.2 ± 14.9	54.5 ± 9.1	55.5 ± 14.5	31.0 ± 9.8	$\mathbf{76.0 \pm 12.8}$	69.7 ± 15.2	+27.9%
	drawer-open	$\mathbf{71.4 \pm 12.7}$	51.2 ± 14.1	34.4 ± 14.3	50.0 ± 16.5	55.0 ± 15.3	9.5 ± 9.4	70.7 ± 14.2	67.1 ± 14.9	+34.2%
	button-press	57.3 ± 5.7	42.2 ± 10.8	44.8 ± 5.5	72.6 ± 9.2	39.7 ± 6.8	31.1 ± 10.7	43.4 ± 7.8	$\mathbf{91.6 \pm 5.2}$	+26.2%
	window-open	61.4 ± 13.4	69.1 ± 12.8	69.8 ± 11.0	73.6 ± 9.0	10.9 ± 5.2	44.4 ± 11.7	71.0 ± 11.4	$\mathbf{87.8 \pm 9.8}$	+19.3%
	sweep-into	31.3 ± 11.7	5.6 ± 2.4	24.8 ± 10.1	35.0 ± 12.3	20.1 ± 9.2	39.3 ± 11.7	38.2 ± 14.7	$\mathbf{47.8 \pm 13.4}$	+36.6%

5.3 Does BRM Improve Other PbRL Algorithms?

BRM modifies the reward model's input representation and does not change the preference loss or the policy update. We can therefore apply the same frozen-feature substitution to other PbRL algorithms. We verify this on PEBBLE and RUNE across seven tasks (Table 2). PEBBLE+BRM improves over PEBBLE on six of seven tasks. RUNE+BRM improves over RUNE on every task. The one regression is PEBBLE+BRM on drawer-open, within standard error and again the saturation case of Section 5.2.

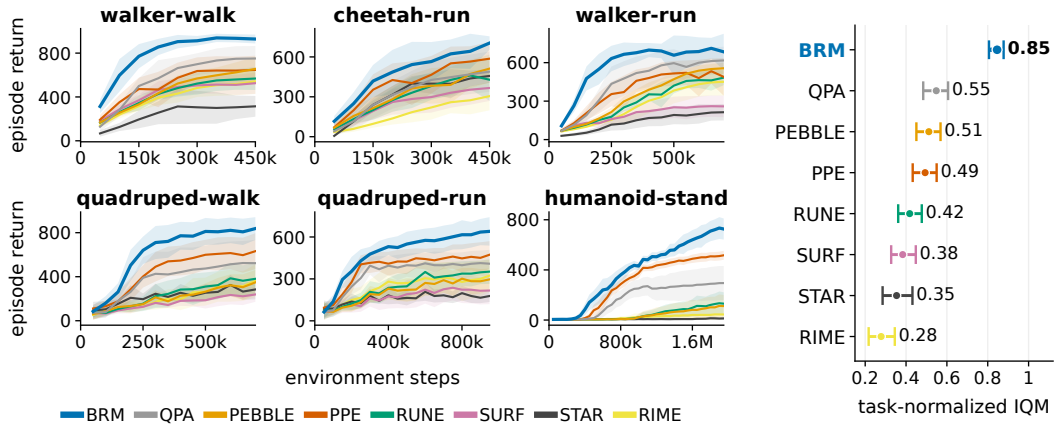

Figure 2: **BRM outperforms all baselines on DMC and in aggregate IQM** (10 seeds; shaded / error bars: 95% CI). *Left*: per-task curves on the six DMC tasks; BRM (blue) outperforms all baselines. *Right*: aggregate task-normalized IQM across all 12 tasks (Section 5.1; higher is better); BRM's 95% CI does not overlap with any other method's.

Table 2: **BRM improves three different PbRL algorithms** (mean ± SE, 10 seeds). DMC: episode return. MW: success rate (%). **+B** denotes the BRM variant. Best per algorithm pair in **bold**.

	DeepMind Control (return)				MetaWorld (success %)		
Method	walker-walk	cheetah-run	quadruped-walk	walker-run	door-unlock	door-open	drawer-open
QPA	751 ± 50	489 ± 67	523 ± 94	617 ± 47	47.8 ± 10.5	54.5 ± 9.1	50.0 ± 16.5
QPA+B	$\mathbf{929 \pm 18}$	$\mathbf{704 \pm 22}$	$\mathbf{838 \pm 46}$	$\mathbf{684 \pm 62}$	$\mathbf{94.4 \pm 5.4}$	$\mathbf{69.7 \pm 15.2}$	$\mathbf{67.1 \pm 14.9}$
PEBBLE	656 ± 48	512 ± 47	348 ± 50	556 ± 19	82.9 ± 8.7	53.1 ± 14.7	$\mathbf{71.4 \pm 12.7}$
PEBBLE+B	$\mathbf{851 \pm 40}$	$\mathbf{634 \pm 62}$	$\mathbf{485 \pm 85}$	$\mathbf{708 \pm 21}$	$\mathbf{94.8 \pm 3.9}$	$\mathbf{55.2 \pm 15.1}$	62.4 ± 14.5
RUNE	568 ± 46	429 ± 70	381 ± 65	483 ± 26	71.5 ± 10.3	66.2 ± 14.3	51.2 ± 14.1
RUNE+B	$\mathbf{842 \pm 36}$	$\mathbf{542 \pm 64}$	$\mathbf{531 \pm 78}$	$\mathbf{709 \pm 15}$	$\mathbf{99.9 \pm 0.1}$	$\mathbf{76.6 \pm 13.2}$	$\mathbf{70.2 \pm 13.4}$

5.4 What Drives the Gain?

Does the gain come from the feature source or from added capacity? We run three ablations on walker-walk, cheetah-run, door-unlock, and drawer-open (5 seeds) to isolate the feature source from raw model capacity. None matches BRM in aggregate (Figure 3), and all three fall below QPA on drawer-open (per-task results in Table 10). An *equal-parameter MLP* matched to BRM's head capacity without frozen features underperforms even the smaller QPA baseline on drawer-open, so capacity alone amplifies overfitting. Standard regularization (L_2 weight decay, dropout) also fails to close the gap (Appendix D.1). *Random features* are competitive on locomotion but unreliable on manipulation, so dimensionality alone is not sufficient. *Policy features* from the SAC actor's first layer collapse on drawer-open. The actor takes only $\mathbf{s}$ as input (Haarnoja et al., 2018), so these features cannot encode action dependence. The critic features, trained against the reward through the Bellman target and depending on both $\mathbf{s}$ and $\mathbf{a}$, succeed on both task types. We provide a qualitative view with t-SNE in Appendix C.1. On walker-walk, quadruped-walk, and door-unlock, BRM's features are more ordered by reward than those of QPA, and the two are similar on drawer-open.

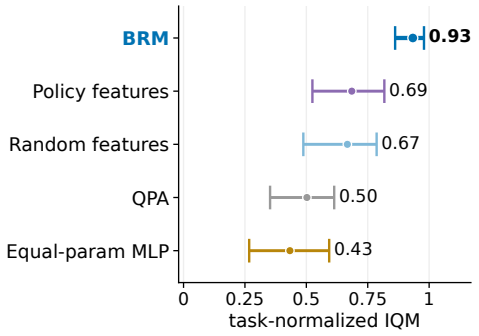

Figure 3: **The gain is due to the critic features, not model capacity** (QPA backbone, 5 seeds). Task-normalized IQM (95% CI) over four tasks for five reward-model variants. Full details in Appendix B.1.

Does the reward model generalize beyond the preference distribution? We hypothesized in Section 4.1 that critic features help the reward model generalize to states far from the labeled segments. To test this, we measure preference-prediction accuracy on three held-out evaluation sets across four environments (Table 3). *full* is a uniform sample of the replay buffer, *recent* is the last 20% of transitions, and *SAC* uses the replay buffer of a separately-trained agent that has access to the true reward. BRM improves on 9 of 12 cells, and the largest improvement is on the most distributionally shifted set (*SAC* door-unlock). We hypothesize that this is because the critic features summarize the agent's full state visitation, while QPA's MLP is trained on the preference-labeled segments alone. The three drawer-open cells are within 3 accuracy points in either direction, as expected for the task on which the return curves saturate near the PEBBLE ceiling (Section 5.2).

Table 3: **BRM improves reward accuracy, especially on out-of-distribution trajectories** (QPA backbone, 5 seeds, end of training). *full*: uniform replay sample. *recent*: last 20% of transitions. *SAC*: replay buffer of a SAC agent trained on the true reward. Per-cell winner in **bold**.

	walker-walk			cheetah-run			door-unlock			drawer-open		
Method	full	recent	SAC	full	recent	SAC	full	recent	SAC	full	recent	SAC
QPA	0.70	0.71	0.68	0.83	0.85	0.93	0.84	0.82	0.41	**0.86**	0.85	**0.65**
QPA + BRM	**0.77**	**0.82**	**0.78**	**0.93**	**0.92**	**0.95**	**0.91**	**0.87**	**0.72**	0.84	0.85	0.62

Do shallower features transfer better? We fix the feature source at the SAC critic and ablate the head architecture and feature depth (Table 4). Variants differ along three axes: linear vs. nonlinear head, with vs. without the skip connection to raw $(\mathbf{s}, \mathbf{a})$, and layer 1 vs. layer 2 of the critic. Most of the gain is due to the critic features, which alone raise the IQM from 0.59 to 0.79 (Appendix A.4). The skip connection with a nonlinear head is the best overall combination. Some single-axis variants match or surpass BRM on individual tasks but degrade on others (Table 9). Reinitializing the heads at every preference update helps on average (Table 9), with the largest improvement on door-unlock and reversals on drawer-open and walker-run. We hypothesize that this is because heads trained against one feature snapshot co-adapt to it, following the fragile co-adaptation reported for supervised transfer (Yosinski et al., 2014). Reading the second hidden layer instead of the first degrades performance on all four DMC tasks, which is the depth pattern that motivates BRM's first-layer choice. The full 9×7 design table is in Appendix A.4. We discuss limitations in Appendix E.

Table 4: **The skip connection with a nonlinear head generalizes across task types** (QPA backbone, 10 seeds, mean $\pm$ SE). All variants use frozen Q_{θ_1} features unless noted; BRM = NL head + skip connection. The IQM column is over these four tasks (the 7-task IQM is in Table 9).

	DMC (return)		MW (success %)		Aggregate
Variant	walker-walk	quad-walk	door-unlock	door-open	IQM
QPA Baseline	751 ± 50	523 ± 94	47.8 ± 10.5	54.5 ± 9.1	0.60
BRM (Skip NL)	$\mathbf{929 \pm 18}$	$\mathbf{838 \pm 46}$	$\mathbf{94.4 \pm 5.4}$	$\mathbf{69.7 \pm 15.2}$	**0.96**
No Reinit (NL)	924 ± 29	665 ± 139	58.6 ± 12.6	54.8 ± 20.9	0.78
Deep Features (NL)	909 ± 44	285 ± 60	60.2 ± 23.0	40.0 ± 24.5	0.59

6 CONCLUSION

In this work, we hypothesized that the data asymmetry in PbRL can be turned into an opportunity by transferring the critic's features to the reward model. Our experiments suggest that this is the case: BRM improves over previous PbRL methods across tasks and backbones, with the largest gains on states held out from the preference distribution. Our ablations further suggest that most of the gain is due to the critic features themselves, and not to added model capacity. We hope that the connection between the critic and the reward model explored in this work can motivate further use of transfer learning between RL components in preference-based settings.

ACKNOWLEDGMENTS

Acknowledgments omitted for double-blind review.

REPRODUCIBILITY STATEMENT

We describe all tasks, feedback budgets, and hyperparameters in Appendix A.1, and the evaluation protocol, seeds, and compute in Appendix A.2. For every baseline with a public implementation, we use that code with its published settings, and we list the few settings that differ in Appendix A.1. Unless otherwise stated, results use 10 seeds per method and task (5 seeds for ablations), and we report significance tests in Appendix A.3. We will release our code upon publication.

AI USE STATEMENT

Our primary usage of generative AI tools in this work has been with writing code, including implementing one baseline that lacked a public codebase, writing training and plotting scripts, and formatting figures and tables. We also used generative AI tools for editing the paper text. All code and text have been verified by the authors. We have not used generative AI tools for experiment design or formulating the research contributions. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

David Abel, David Hershkowitz, and Michael Littman. Near optimal behavior via approximate state abstraction. In *International Conference on Machine Learning (ICML)*, 2016.

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

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Fengshuo Bai, Rui Zhao, Hongming Zhang, Sijia Cui, Shao Zhang, Lei Han, Ying Wen, Yaodong Yang, et al. Star: Efficient preference-based reinforcement learning via dual regularization. *Advances in Neural Information Processing Systems*, 38:168265–168300, 2026.

André Barreto, Will Dabney, Rémi Munos, Jonathan J. Hunt, Tom Schaul, Hado van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, pp. 4055–4065, 2017.

Ralph A. Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Jie Cheng, Gang Xiong, Xingyuan Dai, Qinghai Miao, Yisheng Lv, and Fei-Yue Wang. RIME: Robust preference-based reinforcement learning with noisy preferences. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 8229–8247. PMLR, 2024.

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, pp. 4299–4307, 2017.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*. PMLR, 2014.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. DeepMDP: Learning continuous latent space models for representation learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2170–2179. PMLR, 2019.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

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*, pp. 1861–1870. PMLR, 2018.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2019.

Alex Hernández-García and Peter König. Data augmentation instead of explicit regularization. *arXiv preprint arXiv:1806.03852*, 2018.

Xiao Hu, Jianxiong Li, Xianyuan Zhan, Qing-Shan Jia, and Ya-Qin Zhang. Query-policy misalignment in preference-based reinforcement learning. In *Proceedings of the 12th International Conference on Learning Representations*, 2024.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations*, 2017.

Changyeon Kim, Jongjin Park, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. Preference transformer: Modeling human preferences using transformers for RL. In *Proceedings of the 11th International Conference on Learning Representations*, 2023.

Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2661–2671, 2019.

Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 4, 1991.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5639–5650. PMLR, 2020.

Kimin Lee, Laura Smith, Anca Dragan, and Pieter Abbeel. B-Pref: Benchmarking preference-based reinforcement learning. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021a.

Kimin Lee, Laura M. Smith, and Pieter Abbeel. PEBBLE: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6152–6163. PMLR, 2021b.

Xinran Liang, Katherine Shu, Kimin Lee, and Pieter Abbeel. Reward uncertainty for exploration in preference-based reinforcement learning. In *International Conference on Learning Representations*, 2022.

Runze Liu, Fengshuo Bai, Yali Du, and Yaodong Yang. Meta-reward-net: Implicitly differentiable reward learning for preference-based reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22270–22284, 2022.

Katherine Metcalf, Miguel Sarabia, Natalie Mackraz, and Barry-John Theobald. Sample-efficient preference-based reinforcement learning with dynamics aware rewards. In *Proceedings of the 7th Conference on Robot Learning (CoRL)*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1484–1532. PMLR, 2023.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A universal visual representation for robot manipulation. In *Proceedings of the 6th Conference on Robot Learning*, 2022.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning (ICML)*, volume 99, pp. 278–287, 1999.

Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*. PMLR, 2022.

Jongjin Park, Younggyo Seo, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. SURF: Semi-supervised reward learning with data augmentation for feedback-efficient preference-based reinforcement learning. In *International Conference on Learning Representations*, 2022.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*. PMLR, 2021.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

Max Schwarzer, Ankesh Anand, Rishab Goel, R. Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. *Advances in neural information processing systems*, 35:9460–9471, 2022.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9870–9879. PMLR, 2021.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Jeremy Tien, Jerry Zhi-Yang He, Zackory Erickson, Anca D. Dragan, and Daniel S. Brown. Causal confusion and reward misidentification in preference-based reward learning. In *International Conference on Learning Representations*, 2023.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Reinforcement learning with prototypical representations. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11920–11931. PMLR, 2021.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, volume 27, pp. 3320–3328, 2014.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2021.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*, 2017.

Yiwen Zhu, Jinyi Liu, Pengjie Gu, Yifu Yuan, Zhenxing Ge, Wenya Wei, Zhou Fang, Yujing Hu, and Bo An. Improving reward models with proximal policy exploration for preference-based reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Appendix

BRM: Better Reward Modeling via Critic Feature Transfer in Continuous Control

Contents

A Experimental Details and Extended Results 15

A.1 Hyperparameters 15

A.2 Evaluation Protocol, Seeds, and Compute 16

A.3 Significance Tests 17

A.4 Extended Design Choice Analysis 17

A.5 Per-Algorithm Orthogonality Results 18

A.6 Scope of comparison 18

B Why the Method Works 19

B.1 Feature Source Ablation Details 19

B.2 Circular Dependency Discussion 19

C Qualitative Evidence of Representation Quality 20

C.1 BRM vs QPA Feature Geometry 20

C.2 Linear Reward Correlation Across Training 20

D Mechanism and Robustness 21

D.1 Regularization Is Not a Substitute for Representation 21

D.2 Dual-Critic Ablation 21

D.3 Robustness to Noisy Preference Labels 22

E Limitations 23

A Experimental Details and Extended Results

In this section, we provide the hyperparameters, extended design-choice tables, and per-algorithm learning curves used in the main-text experiments.

A.1 Hyperparameters

Table 5: Per-environment configuration. Feedback budgets and query intervals follow those reported by QPA (Hu et al., 2024), and by PPE (Zhu et al., 2025) for the tasks QPA does not use. The only change is on sweep-into, where we use 3000 queries over 1M steps instead of the 10000 over 2M reported by PPE. All methods share these settings.

Environment	Training steps	Feedback budget	Query interval	Batch size
walker-walk	500k	100	20k	1024
cheetah-run	500k	100	20k	1024
walker-run	750k	250	20k	1024
quadruped-walk	750k	1000	30k	1024
quadruped-run	1M	1000	30k	1024
humanoid-stand	2M	10000	5k	1024
door-unlock	1M	2000	20k	512
door-open	1M	3000	5k	512
drawer-open	1M	3000	5k	512
button-press	1M	2000	5k	1024
window-open	500k	400	5k	1024
sweep-into	1M	3000	5k	1024

Table 6: Shared hyperparameters across all experiments.

Parameter	Value
SAC critic hidden dims	$[1024, 1024]$
SAC actor hidden dims	$[1024, 1024]$
SAC learning rate	10^{-4}
Discount γ	0.99
Segment length H	50
Reward ensemble size K	3
Reward head hidden dim	256
Frozen feature dim d_h	1024
Reward learning rate	3×10^{-4}

Baseline reward-model configurations. Each baseline keeps its published configuration; Table 6 lists the default shared with BRM and the ablation arm. PEBBLE, RUNE, SURF, PPE, RIME, and STAR use this default (K=3 ensemble, no temporal-cropping augmentation). QPA's published configuration is K=1 with 20× temporal-cropping augmentation, used in Table 1. For the feature-source ablations (Figure 3, Table 10), we re-run QPA at K=3 with no augmentation so that all variants (equal-parameter MLP, random features, policy features, BRM) share the same ensemble and augmentation, and the comparison isolates the feature-source contribution.

RIME and STAR. Both run on the same SAC backbone as PEBBLE. STAR releases no code, so we reimplemented it from the equations in its paper and ran it under our protocol and feedback budgets, following those of recent online PbRL methods (QPA, PPE). Two settings differ from the published ones. First, STAR's preference margin ε is 0.005 instead of 0.05. At our segment length H=50, ε=0.05 flattened the learned reward and stopped learning in early runs, so we selected ε from a small sweep. Its policy-regularization weights keep their published values (β=6, η=0.1). Second, RIME on one DMC task uses 10,000 reward-update steps per round, as in its reference scripts, where every other method uses 2,000.

STAR's own setup. STAR (Bai et al., 2026) regularizes both the reward and the policy learning objectives. Its evaluation is mainly offline. Twelve of its eighteen continuous-control tasks are offline (D4RL and Robosuite, with real human preferences) and six are online (three DMC, three MetaWorld). In the online setting it compares against PEBBLE, SURF, and MRN, and reports scores normalized by SAC and averaged over tasks. Its six online tasks are all in our benchmark, but with different feedback budgets and training lengths (Table 7). Our results therefore measure STAR under the protocol shared by all methods in Table 1, and are not directly comparable to the scores it reports.

Table 7: **STAR's online setup and ours** on the six tasks STAR evaluates online. Feedback: total preference queries.

	Feedback		Training steps	
Task	STAR	Ours	STAR	Ours
cheetah-run	100	100	1M	500k
walker-walk	200	100	1M	500k
quadruped-walk	1000	1000	1M	750k
button-press	100	2000	1M	1M
window-open	100	400	1M	500k
sweep-into	4000	3000	1M	1M

Hyperparameter selection. We did not tune BRM-side hyperparameters. The reward-model defaults in Table 6 (K=3 ensemble, hidden width 256, segment length H=50, reward learning rate 3×10^{-4}) are inherited from QPA's published values; the only BRM-specific choice, the frozen-feature width d_h=1024, follows the SAC critic's first-layer width. For the regularization baselines (Appendix D.1), L_2 weight decay 10^{-4} and dropout rate 0.1 are single values applied uniformly across all environments, taken at standard published values without per-environment search. The only baseline hyperparameter selected on our setup is STAR's margin ε (see above).

A.2 Evaluation Protocol, Seeds, and Compute

Evaluation protocol. For each logged checkpoint we roll out evaluation episodes with the training policy and record either the mean episode return (DeepMind Control) or the fraction of successful episodes (MetaWorld, where success is a binary per-episode event defined by the benchmark). Learning curves are sampled at a fixed cadence matched to the training step budget in Table 5; the end-of-training values reported in Tables 1–2 are the final checkpoint's evaluation mean. Policy evaluation uses the policy's mean action on DMC and sampled actions on MetaWorld, following the conventions of PEBBLE (Lee et al., 2021b) and QPA (Hu et al., 2024).

Seeds. Main comparisons (Tables 1, 2) and design-choice analyses (Tables 4, 9) use 10 seeds per method-environment cell. Feature-source ablations (Figure 3, Table 10) use 5 seeds per variant: the three ablation methods (equal-parameter MLP, random features, policy features) are run natively at 5 seeds, and BRM and the QPA baseline are reported on a fixed 5-seed subsample of their 10-run set to match. The preference-accuracy analysis (Table 3) uses 5 seeds.

Compute. Each training run is allocated a single NVIDIA V100 (32 GB) with 2 CPU cores and the partition's default memory; no multi-GPU or multi-node setup is required. A representative DMC run (walker-walk, 500k steps) takes about 2 hours of wall-clock time per seed; the longer DMC tasks (walker-run, quadruped-walk, 750k steps) take about 3 hours, quadruped-run (1M) about 5 hours, and humanoid-stand (2M) about 9 hours. The MetaWorld tasks take 6–7 hours at 1M steps, and about half that for window-open (500k). STAR's additional in-sample critic makes it about 1.3× slower than PEBBLE. The reward model accounts for under 5% of total wall-clock time across all methods, with the SAC actor–critic update dominating, so BRM's frozen-feature heads and the baselines' deep MLPs incur comparable per-step cost.

A.3 Significance Tests

Table 8 reports two-sided Welch's t-test p-values for BRM vs. each of the seven baselines across all twelve environments (10 seeds per cell), computed from the raw per-seed final-performance values used throughout the paper. We report raw p-values without multiple-comparison correction; 52 of 84 pairwise comparisons are significant at $\alpha = 0.05$.

Table 8: **Welch's t-test p-values** for BRM vs. each baseline (10 seeds per cell, two-sided). Bold: $p < 0.05$. The results match Figure 2 and Table 1.

Environment	vs. QPA	vs. PEBBLE	vs. RUNE	vs. PPE	vs. SURF	vs. RIME	vs. STAR
walker-walk	**0.007**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**
cheetah-run	**0.011**	**0.003**	**0.003**	0.094	**<0.001**	**<0.001**	**<0.001**
walker-run	0.405	0.077	**0.012**	0.068	**<0.001**	**0.009**	**<0.001**
quadruped-walk	**0.010**	**<0.001**	**<0.001**	**0.023**	**<0.001**	**<0.001**	**<0.001**
quadruped-run	**0.002**	**<0.001**	**<0.001**	**0.015**	**<0.001**	**<0.001**	**<0.001**
humanoid-stand	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**
door-unlock	**0.002**	0.277	0.070	**0.026**	0.169	**<0.001**	0.078
door-open	0.405	0.444	0.869	0.507	0.945	**0.049**	0.755
drawer-open	0.452	0.829	0.448	0.578	0.131	**0.005**	0.863
button-press	0.094	**<0.001**	**0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**
window-open	0.301	0.131	0.263	**<0.001**	0.238	**0.011**	0.280
sweep-into	0.489	0.364	**0.012**	0.107	0.188	0.638	0.634

The pattern matches the main-text discussion in Section 5.2. BRM is significant vs. QPA on all six DMC environments except walker-run and on door-unlock on MetaWorld. On door-open and drawer-open, where BRM's mean is below a baseline's (STAR on both, PEBBLE on drawer-open), no significant difference is detected.

A.4 Extended Design Choice Analysis

Table 9 reports the complete design choice comparison across all 9 variants and all 7 environments on the QPA backbone.

Table 9: Design choice comparison on the QPA backbone. DMC: episode return. MW: success rate (%). Aggregate IQM as in Section 5.1. BRM = NL + Skip (highlighted row).

	DeepMind Control (return)				MetaWorld (success %)			Aggregate
Variant	walker-walk	cheetah-run	quad-walk	walker-run	door-unlock	door-open	drawer-open	IQM
QPA Baseline	751 ± 50	489 ± 67	523 ± 94	617 ± 47	47.8 ± 10.5	54.5 ± 9.1	50.0 ± 16.5	0.59
Linear	938 ± 12	616 ± 46	675 ± 76	$\mathbf{764 \pm 11}$	78.6 ± 11.8	40.0 ± 16.3	11.2 ± 9.9	0.72
Nonlinear (NL)	$\mathbf{952 \pm 2}$	556 ± 71	676 ± 93	741 ± 17	77.7 ± 9.5	59.5 ± 16.2	47.3 ± 14.4	0.79
Skip Linear	897 ± 40	599 ± 82	668 ± 85	667 ± 48	$\mathbf{99.3 \pm 0.6}$	36.7 ± 15.3	26.1 ± 13.7	0.75
Skip NL (BRM)	929 ± 18	704 ± 22	$\mathbf{838 \pm 46}$	684 ± 62	94.4 ± 5.4	$\mathbf{69.7 \pm 15.2}$	67.1 ± 14.9	**0.87**
No Reinit (Linear)	914 ± 46	676 ± 57	535 ± 109	747 ± 11	62.8 ± 16.6	64.4 ± 15.1	79.6 ± 19.9	0.77
No Reinit (NL)	924 ± 29	696 ± 35	665 ± 139	739 ± 10	58.6 ± 12.6	54.8 ± 20.9	$\mathbf{86.8 \pm 13.2}$	0.78
Deep Feat (Linear)	825 ± 55	624 ± 32	454 ± 135	535 ± 67	74.4 ± 11.4	36.2 ± 22.4	16.4 ± 16.4	0.58
Deep Feat (NL)	909 ± 44	626 ± 82	285 ± 60	487 ± 120	60.2 ± 23.0	40.0 ± 24.5	40.0 ± 23.3	0.56

BRM (Skip NL) achieves the strongest aggregate performance. It ranks first on quadruped-walk and is competitive on every other task, while variants that excel on individual tasks degrade elsewhere. Deep features consistently degrade performance, and no-reinitialization trades DMC for MetaWorld.

Critic features alone already help. BRM concatenates the critic features with the raw state-action pair (Eq. 7), so one may ask whether the gain is due to the features or to the larger input. The variants without the skip connection answer this question. When the critic features are the sole input, the nonlinear head reaches an IQM of 0.79 and the linear head 0.72, both above QPA (0.59). Adding the skip connection further improves the nonlinear head to 0.87. In other words, the critic features account for most of the gain, and the skip connection makes it more robust across tasks.

A.5 Per-Algorithm Orthogonality Results

Figures 4 and 5 show learning curves for PEBBLE and RUNE with and without BRM across the seven environments. BRM generally improves performance and reduces variance on DMC tasks. On MetaWorld, the improvement is largest on door-unlock, while drawer-open (where PEBBLE already performs well) shows diminishing returns.

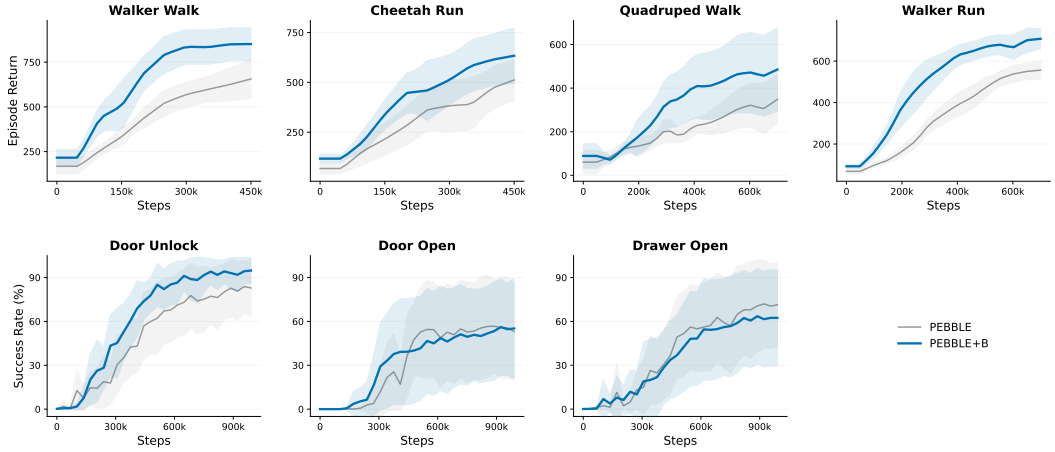

Figure 4: Learning curves for PEBBLE (gray) vs. PEBBLE+B (blue) across the seven environments (10 seeds, shaded: 95% CI). Top row: DMC (episode return). Bottom row: MW (success rate %).

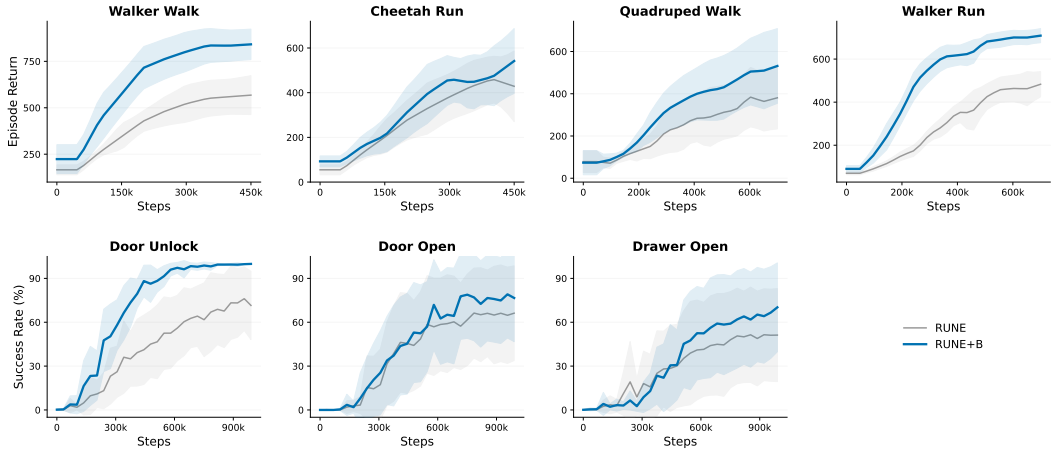

Figure 5: Learning curves for RUNE (gray) vs. RUNE+B (blue) across the seven environments (10 seeds, shaded: 95% CI). Top row: DMC (episode return). Bottom row: MW (success rate %).

A.6 Scope of Comparison

We benchmark against seven PbRL baselines (QPA, PEBBLE, RUNE, SURF, PPE, RIME, STAR) and verify orthogonality on three of them (QPA, PEBBLE, RUNE; Table 2). This subsection notes the methods we do not run head-to-head and why.

MRN. MRN (Liu et al., 2022) acts on the reward-model training objective rather than the input representation, meta-learning the reward parameters via a bilevel loop with Q-function preference accuracy as the outer loss. MRN acts on a different component than BRM; we leave a head-to-head to future work.

Preference Transformer. Preference Transformer (Kim et al., 2023) departs from the MLP entirely, modeling preferences as a learned weighted sum of non-Markovian rewards via a transformer. Recent PbRL methods sharing our benchmarking lineage (PEBBLE, RUNE, SURF, QPA, PPE) do not run PT as a baseline; we follow that convention.

Orthogonality coverage on SURF and PPE. SURF and PPE each overlap with QPA on their main design choice. SURF's temporal cropping overlaps with QPA's cropping augmentation, and PPE's coverage-shaped query selection with QPA's visitation-aware selection. The QPA+BRM result therefore probes the components those two methods change, and we expect similar orthogonality patterns for SURF+BRM and PPE+BRM.

B WHY THE METHOD WORKS

In this section, we present the full feature-source ablation and a discussion of the circular dependency between critic and reward model.

B.1 FEATURE SOURCE ABLATION DETAILS

Table 10 reports the full results for the feature source ablation (Section 5.4) across all four environments. The equal-parameter MLP is QPA's reward MLP at $K{=}3$ with no augmentation (Appendix A.1), rebuilt with hidden width 370 instead of 256, giving it about 286k trainable parameters (i.e., matched to BRM's head at $\sim$ 270k and roughly $2\times$ the same MLP at width 256), while still receiving only the raw $(\mathbf{s}, \mathbf{a})$ input. BRM achieves the best performance on every environment, while all ablation variants degrade on drawer-open.

Table 10: **Feature source ablation** (QPA backbone, 5 seeds). DMC: episode return. MW: success rate (%). On drawer-open, all ablation variants fall below the QPA baseline; BRM is the only method that succeeds.

	DMC (return)		MetaWorld (success %)	
Variant	walker-walk	cheetah-run	door-unlock	drawer-open
QPA Baseline	679 ± 52	536 ± 48	27.2 ± 10.9	40.4 ± 24.3
Equal-param MLP	584 ± 127	552 ± 97	57.6 ± 19.5	3.6 ± 3.6
Random features	827 ± 77	600 ± 34	61.6 ± 18.0	34.0 ± 21.4
Policy features	913 ± 26	525 ± 58	78.8 ± 14.2	18.2 ± 18.2
BRM (ours)	$\mathbf{926 \pm 33}$	$\mathbf{672 \pm 17}$	$\mathbf{99.6 \pm 0.2}$	$\mathbf{80.0 \pm 20.0}$

B.2 CIRCULAR DEPENDENCY DISCUSSION

On the circular dependency between critic and reward model. BRM introduces a *circular dependency*. The reward model $\hat{r}_\psi$ provides the reward signal that trains the SAC critics $Q_{\theta_1}, Q_{\theta_2}$, yet the critics' frozen first-layer features $\varphi(\mathbf{s}, \mathbf{a})$ are used as input to $\hat{r}_\psi$ (Eq. 6). We argue that this circularity is benign, for three reasons.

The skip connection provides a safe fallback. Because the raw $(\mathbf{s}, \mathbf{a})$ pair is always present in $\mathbf{z}$ (Eq. 7), the reward heads can recover a fully raw-input baseline by learning to ignore φ. Early in training, when the critic has been trained on few transitions, the heads naturally rely on $[\mathbf{s}; \mathbf{a}]$.

No large feedback effect detected. The dual-critic ablation (Appendix D.2) breaks the loop by giving the policy a separately-trained critic that is never touched by the reward head. BRM's improvement does not differ significantly from this decoupled version on any of the seven tasks (Welch's t-test, $\min p = 0.14$ at 10 seeds). We conclude that the shared feature path's benefit is concentrated in reward-model quality, though small feedback effects cannot be excluded under the current seed budget.

Reinitialization prevents stale co-adaptation. Reward heads may co-adapt to a particular snapshot of φ that becomes stale as the critic evolves (Yosinski et al., 2014). We address this by reinitializing

all reward heads from scratch when new preference labels arrive (Section 4.3), so each round of reward training starts fresh with the latest critic features.

C Qualitative Evidence of Representation Quality

In this section, we provide two qualitative views of the critic's learned features, complementing the quantitative ablation of Section 5.4.

C.1 BRM vs QPA Feature Geometry

In this section, we visualize critic features with t-SNE on four tasks (Figure 6). We use 10,000 state-action pairs from the final checkpoint, colored by the true reward, and compare the frozen critic features of BRM with the hidden layer of the QPA reward model. On walker-walk, quadruped-walk, and door-unlock, the features of BRM are more ordered by reward than those of QPA. On drawer-open, the two are similar.

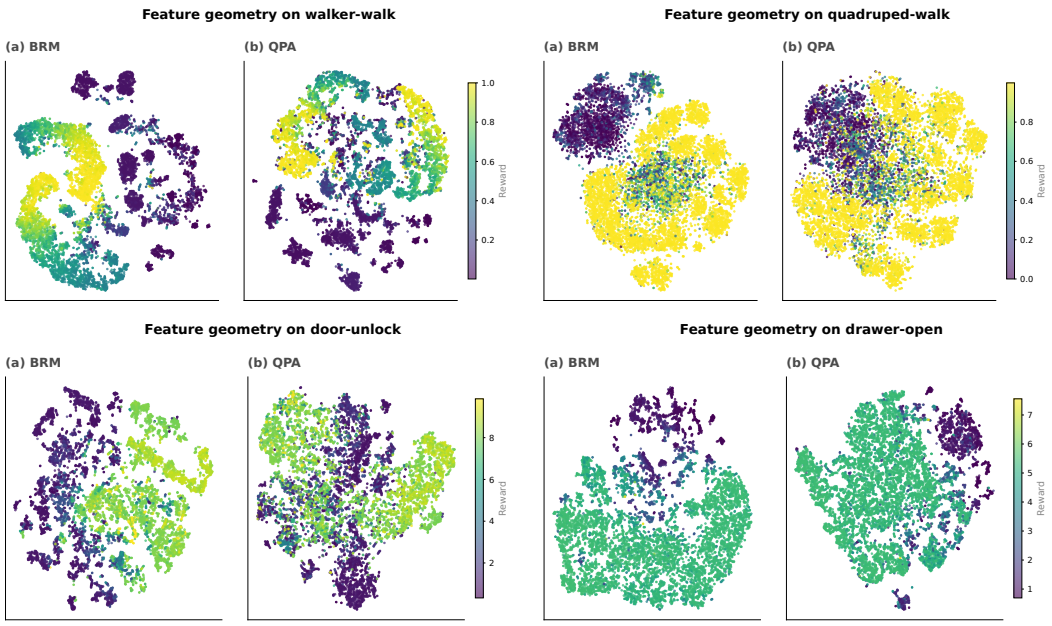

Figure 6: **Critic features are more ordered by reward on three of four tasks.** t-SNE of 10,000 state-action pairs colored by the true reward (one seed). (a) The frozen critic features of BRM. (b) The hidden layer of the QPA reward model.

C.2 Linear Reward Correlation Across Training

We track how the linear correlation between critic features and the true reward evolves during training on four tasks (Figure 7). Second-layer features are often more correlated with the reward than first-layer ones, most clearly on quadruped-walk. However, layer 2 is a worse input to the reward model. Replacing layer 1 with layer 2 (Deep Feat (NL) in Table 9) reduces the quadruped-walk return from 838 to 285 and degrades walker-run ($684 \rightarrow 487$) and door-unlock ($94.4\% \rightarrow 60.2\%$). In other words, a higher linear correlation with the reward does not make features more useful to BRM. We hypothesize that this is because deeper features specialize to the current targets of the critic, which change at every feedback round, and therefore transfer less well to the next reward model (Section 4.1).

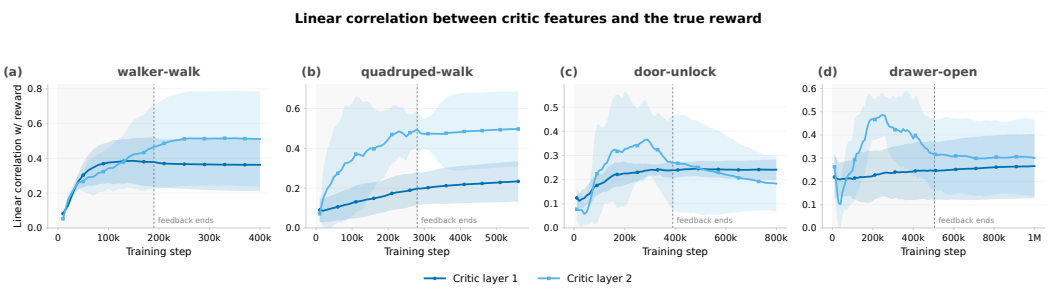

Figure 7: **Second-layer critic features are often more correlated with the reward.** Linear correlation between critic features and the true reward (mean and 95% CI over 5 seeds). The dashed line marks the last feedback round.

D Mechanism and Robustness

In this section, we present three experiments that probe the mechanism behind BRM's gain and test its robustness under label noise.

D.1 Regularization Is Not a Substitute for Representation

A natural alternative hypothesis is that the reward model's generalization failure stems from overfitting to the seen preference pairs. In that case, standard regularization should close the gap. We test this by augmenting the QPA reward model with two standard remedies, namely L_2 weight decay (coefficient 10^{-4} applied to all reward-model parameters) and dropout (rate 0.1 between hidden layers). All other training details are identical to the QPA baseline. We use 5 seeds per configuration across four environments spanning DMC locomotion and MetaWorld manipulation. For apples-to-apples comparison, QPA and BRM are subsampled to the same 5 seeds used in Table 10.

Table 11: **Regularization does not close the gap with BRM** (mean $\pm$ SE, 5 seeds each). L_2 uses weight decay 10^{-4}; dropout uses rate 0.1 between hidden layers. BRM included for reference; QPA and BRM rows use the same five seeds as Table 10. Best per task in **bold**.

	DMC (return)		MetaWorld (success %)	
Method	walker-walk	cheetah-run	door-unlock	drawer-open
QPA	679 ± 52	536 ± 48	27.2 ± 10.9	40.4 ± 24.3
QPA + L_2	833 ± 54	590 ± 48	62.0 ± 12.2	37.6 ± 22.0
QPA + dropout	810 ± 91	560 ± 87	73.8 ± 18.2	40.0 ± 24.5
BRM (ours)	$\mathbf{926 \pm 33}$	$\mathbf{672 \pm 17}$	$\mathbf{99.6 \pm 0.2}$	$\mathbf{80.0 \pm 20.0}$

Table 11 reports the results. L_2 and dropout yield modest gains on DMC tasks (e.g., walker-walk $+23\%$ / $+19\%$ over QPA), but their behavior on MetaWorld is inconsistent. In particular, L_2 reduces drawer-open success from 40.4% to 37.6%, and door-unlock remains far below BRM (62.0% and 73.8% versus 99.6%). This matches the capacity-control ablation (Section 5.4), where an equal-parameter MLP without critic features degraded performance despite having as many trainable parameters as BRM's heads. We conclude that the reward model's failure mode is poor extrapolation to unlabeled state-action pairs. Reducing overfitting on the labeled ones does not address it.

D.2 Dual-Critic Ablation

The previous ablations establish that sharing features helps the *reward model* (Sec. 5.4). We now ask whether the actor also benefits from training against a critic whose features the reward model consumes, or whether the gain is routed entirely through reward-model accuracy.

Method. We construct BRM-DC (*dual-critic*), a variant identical to BRM except that we add an auxiliary double-Q critic for the actor. The *main* critic is unchanged and still supplies φ to the reward

model (i.e., the feature-producing critic from Algorithm 1). The *aux* critic does not share weights or features with the main critic; the actor's policy gradient is computed against the aux critic only. Both critics use identical SAC TD losses, the same reward-model output, and the same target-network update rate, but each draws a separate minibatch from the replay buffer so that their updates are statistically uncorrelated at the batch level. We run BRM-DC on all seven environments with 10 seeds, holding all other hyperparameters at the BRM defaults.

Table 12: **Decoupling the actor's critic does not significantly change performance** (mean ± SE, 10 seeds each). BRM-DC uses a separate critic for the actor's policy gradient. Per-task best in **bold**; IQM is task-normalized (DMC/1000, MW/100). Welch's two-sided t-test between BRM and BRM-DC yields $p \in \{0.37, 0.73, 0.14, 0.75, 0.89, 0.31, 0.90\}$ across the seven tasks, in column order; equality is not rejected on any environment at $\alpha = 0.05$.

	DeepMind Control (return)				MetaWorld (success %)			Aggregate
Method	walker-walk	cheetah-run	quadruped-walk	walker-run	door-unlock	door-open	drawer-open	IQM
QPA	751 ± 50	489 ± 67	523 ± 94	617 ± 47	47.8 ± 10.5	54.5 ± 9.1	50.0 ± 16.5	0.59
BRM-DC (decoupled actor)	873 ± 57	693 ± 24	693 ± 80	$\mathbf{705 \pm 17}$	93.2 ± 6.4	$\mathbf{88.9 \pm 9.9}$	$\mathbf{69.7 \pm 15.2}$	0.86
BRM (shared, ours)	$\mathbf{929 \pm 18}$	$\mathbf{704 \pm 22}$	$\mathbf{838 \pm 46}$	684 ± 62	$\mathbf{94.4 \pm 5.4}$	69.7 ± 15.2	67.1 ± 14.9	**0.87**

Results. We find that decoupling the actor's critic from the reward-model feature source does not significantly change performance. Welch's two-sided t-test cannot reject equality on any environment ($\min p = 0.14$ on quadruped-walk), and the task-normalized IQM is 0.87 for BRM versus 0.86 for BRM-DC. We conclude that the shared-feature mechanism's benefit is concentrated in reward-model quality and is not load-bearing for actor or critic learning. This also justifies the simpler single-critic design we use as the main method. A second critic roughly doubles the cost of the SAC update and does not change the outcome significantly on any task we considered.

Note that Welch's test at 10 seeds has limited power against small effects, so we can claim the absence of a large feedback effect, though not equivalence in the strict sense. The largest residuals are on quadruped-walk ($+144$, ≈ 1.6 combined SE, in favour of sharing) and door-open (-19.2, ≈ 1.1 combined SE, in favour of decoupling); both directions appear, neither systematically.

D.3 Robustness to Noisy Preference Labels

The main results use a clean scripted oracle, where preference labels are produced from the ground-truth reward without error. A natural concern is that BRM's gain over QPA depends on this assumption and that the gap would close once the teacher makes mistakes. We test this on four environments (two DMC, two MetaWorld) by flipping each binary preference label independently with probability $\rho \in \{0.1, 0.2\}$ (i.e., 10% and 20% of labels are corrupted), holding the rest of the protocol identical to Section 5.1 and using 5 seeds per cell.

Table 13: **BRM is more robust to noisy preference labels than QPA** (mean ± SE, 5 seeds). Each label produced by the scripted oracle is flipped independently with probability ρ; everything else matches the main protocol. BRM achieves the higher mean on every cell. Welch's two-sided t-test p-values are reported in the rightmost column for each noise level.

	$\rho = 0.1$			$\rho = 0.2$		
Environment	QPA	BRM	p	QPA	BRM	p
DMC (episode return)						
walker-walk	473 ± 109	$\mathbf{677 \pm 161}$	0.33	370 ± 66	$\mathbf{429 \pm 179}$	0.77
cheetah-run	541 ± 69	$\mathbf{560 \pm 49}$	0.83	298 ± 57	$\mathbf{338 \pm 50}$	0.61
MetaWorld (success %)						
door-unlock	18.2 ± 16.5	$\mathbf{55.4 \pm 13.9}$	0.12	14.0 ± 10.1	$\mathbf{57.2 \pm 15.2}$	**0.05**
drawer-open	23.2 ± 11.1	$\mathbf{98.0 \pm 1.4}$	**0.002**	0.2 ± 0.2	$\mathbf{74.2 \pm 15.4}$	**0.009**

We find that BRM obtains the higher mean on every (env, ρ) cell. The qualitative pattern from the main results is preserved. The gains are largest on MetaWorld, smaller and noisier on DMC.

Specifically, on door-unlock and drawer-open the QPA reward model is largely degraded by label noise (success rates of 18%–23% at $\rho = 0.1$ and near-collapse at $\rho = 0.2$ on drawer-open), while BRM still achieves a usable policy ($\geq 55\%$ on every MetaWorld environment at $\rho = 0.1$). On DMC, both methods are hurt by noise, but BRM's point estimate stays above QPA's on every cell. Methods designed for noisy labels, such as RIME (Cheng et al., 2024), filter or relabel suspicious preferences. These methods act on the labels, not on the reward model's input, and can be combined with BRM.

E LIMITATIONS

BRM inherits the critic's features and is bounded by them; on tasks where the critic generalizes poorly, the reward model has no better starting point. On door-open and drawer-open, where STAR and PEBBLE stay above BRM without a significant gap, further improvement may require changing *what* is queried, not how the reward is parameterized. Layer 2 features are often more correlated with the true reward than layer 1 features, yet they make a worse input to the reward model (Appendix C.2). We hypothesize that this is because the targets keep changing while new preferences arrive, and deeper features may become a better input once feedback ends. A better understanding of this gap may lead to adaptive layer selection, which we leave for future work. We have evaluated BRM on continuous-control tasks with proprioceptive observations; whether the same transfer applies to pixel-based or discrete-action settings remains untested.