2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Reducing Oracle Feedback with Vision-Language Embeddings for Preference-Based RL

Udita Ghosh[1*], Dripta S. Raychaudhuri[2], Jiachen Li[1], Konstantinos Karydis[1], Amit Roy-Chowdhury[1]

***Abstract*— Preference-based reinforcement learning can learn effective reward functions from comparisons, but its scalability is constrained by the high cost of oracle feedback. Lightweight vision-language embedding (VLE) models provide a cheaper alternative, but their noisy outputs limit their effectiveness as standalone reward generators. To address this challenge, we propose ROVED, a hybrid framework that combines VLE-based supervision with targeted oracle feedback. Our method uses the VLE to generate segment-level preferences and defers to an oracle only for samples with high uncertainty, identified through a filtering mechanism. In addition, we introduce a parameter-efficient fine-tuning method that adapts the VLE with the obtained oracle feedback in order to improve the model over time in a synergistic fashion. This ensures the retention of the scalability of embeddings and the accuracy of oracles, while avoiding their inefficiencies. Across multiple robotic manipulation tasks, ROVED matches or surpasses prior preference-based methods while reducing oracle queries by up to 80%. Remarkably, the adapted VLE generalizes across tasks, yielding cumulative annotation savings of up to 90%, highlighting the practicality of combining scalable embeddings with precise oracle supervision for preference-based RL. Project page: https://roved-icra-2026.github.io/**

I. Introduction

When a robot manipulator is assigned a task, it must possess the necessary skills to complete it effectively. Acquiring such skills through reinforcement learning (RL) is challenging, primarily because designing a reward function that reliably guides the agent toward the desired behavior is difficult. Hand-crafted rewards often require significant domain expertise, risk unintended shortcuts or reward hacking, and struggle to capture complex task goals. A widely adopted alternative is preference-based RL (PbRL) [1], [2], where a reward function is inferred from human or oracle feedback in the form of comparisons between agent behaviors. This approach removes the need for carefully engineered reward functions and provides a more direct way to align agent behavior with human intent. However, current PbRL methods typically demand large volumes of high-quality feedback, which is costly and time-consuming to obtain, thereby limiting their practicality and scalability.

To reduce human effort, recent work has explored vision–language embedding (VLE) models such as CLIP [3], which enable zero-shot reward estimation from natural language descriptions (*e.g.*, "open the door") by mapping text and images into a shared representation space [4], [5], [6], [7]. While scalable, these VLE-based rewards are often coarse and noisy, limiting their utility in precise domains such as robotic manipulation [8]. More recently, large vision–language models (VLM) like Gemini-Pro [9] have been proposed as automated oracles for preference learning [10], replacing human annotators entirely. Although accurate, VLMs incur high query costs and suffer from slow, auto-regressive inference. This motivates the need for a *method that retains the scalability of embedding based approaches and the accuracy of oracles, while avoiding their inefficiencies.*

Towards this goal, we introduce ROVED: **R**educing **O**racle Feedback using **V**ision-language **E**mbed**D**ings, a framework for efficient PbRL that combines lightweight VLE models with targeted oracle feedback (where the oracle may be a human or a VLM). Prior approaches rely exclusively on either embeddings [4] or oracle comparisons [1], [10]. In contrast, ROVED adopts a hybrid strategy: the VLE generates segment-level preferences and defers to the oracle only when confidence is low. This creates a feedback loop where the VLE improves from oracle supervision and becomes increasingly selective in querying, thereby preserving annotation quality while substantially reducing query costs.

To the best of our knowledge, we are the first to exploit VLEs as a supplementary source of noisy labels to reduce dependence on high-quality oracle feedback. Our approach rests on two components. *First*, we improve the quality of scalable preference labels through a parameter-efficient fine-tuning scheme that combines an unsupervised dynamics-aware objective with sparse oracle feedback. Prior efforts have attempted to denoise embedding-based rewards using expert demonstrations or environment shaping [8], but demonstrations are costly to collect [11] and often suffer from domain gaps [12]. In contrast, preference feedback is lightweight, accessible, and more broadly applicable [13]. *Second*, to minimize oracle queries, we adopt robust confidence-aware training [14], [15]: by constraining the KL divergence between the reward model and VLE predictions, the system automatically resolves confident cases and escalates only uncertain ones. Together, these mechanisms deliver both efficiency and precision in preference-based reward learning.

We evaluate ROVED on robotic manipulation tasks from Meta-World [16] and demonstrate that it learns reward functions capable of training effective policies. ROVED matches the performance of oracle-only methods while reducing annotation cost by **50-80%**. Moreover, a VLE fine-tuned on one task generalizes to related tasks with minimal additional supervision, yielding *cumulative annotation savings* of **75-90%**. This cross-task generalization—absent in prior

[1]Univeristy of California, Riverside; [2]AWS AI Labs (Work done outside AWS)
* Corresponding author: ughos002@ucr.edu

work—demonstrates that pairing scalable VLE models with selective oracle feedback is a promising direction for practical and efficient PbRL. The main contributions of this work are:

- We present ROVED, a novel framework for efficient PbRL that combines the scalability of VLEs with the precision of selective oracle supervision.
- We introduce two key techniques: (i) a parameter-efficient adaptation scheme that improves noisy VLE preference labels with dynamics-aware objectives and sparse oracle feedback, and (ii) an uncertainty-aware sample selection strategy that queries the oracle only for uncertain cases in an effort to reduce oracle feedback.
- We demonstrate that ROVED achieves oracle-level performance on Meta-World tasks while cutting annotation cost by 50–80%, and further achieves 75–90% cumulative savings through cross-task generalization.

II. Related Work

Designing reward functions. Traditional methods for reward design often rely on manual trial-and-error, requiring substantial domain expertise and struggling to scale to complex, long-horizon tasks. Inverse reinforcement learning offers an alternative by attempting to infer reward functions from expert demonstrations [17]. Evolutionary algorithms have also been explored for automated reward function discovery [18]. More recently, foundation models have enabled reward design from high-level task descriptions. Large language models (LLM) have been employed to derive reward functions directly from natural language task descriptions [19], [20], although these approaches assume access to the environment's source code. VLE models provide another path, aligning visual observations with textual descriptions to yield dense reward signals [4], [5], [6], [7]. These methods are fast and scalable, but VLE rewards are typically noisy and insufficiently precise for fine-grained control [8].

Preference-based RL. Preference feedback has been widely explored across domains, including natural language processing [21] and robotics [2]. In RL, Christiano *et al.* pioneered the use of human trajectory comparisons [1], with subsequent works improving sample efficiency and generalization. For instance, PEBBLE [2] combined human preferences with unsupervised exploration, SURF [22] augmented preference datasets using semi-supervised learning, and MRN [23] incorporated policy performance into reward model updates. These methods leverage the fact that relative judgments are easier for humans than explicit reward specification, but they remain constrained by costly human annotation. To alleviate this bottleneck, recent work such as RL-VLM-F [10] uses large VLMs, such as Gemini-Pro, as preference oracles for preference learning. While promising in accuracy, this introduces new challenges: VLMs are expensive to query due to API costs and slow because of their auto-regressive nature.

Given these scalability challenges, we propose a hybrid framework that selectively queries expensive oracles only when fast, automatic supervision is unreliable. Specifically, we use a lightweight VLE model to generate initial preference labels, deferring to the oracle based on a model uncertainty criterion. Unlike prior methods that fully commit to either oracle or automated feedback, our approach dynamically balances scalability and precision by allocating supervision where it is most needed, following principles from uncertainty-based active learning [24]. This targeted strategy substantially reduces annotation costs without sacrificing reward quality.

Learning from noisy labels. The challenge of learning from noisy or imprecise labels has been extensively studied in supervised learning, particularly with the rise of machine-generated annotations [25]. Robust training methods tackle this issue through diverse strategies, including regularization techniques [26], [27], specialized loss functions [28], and sample-selection mechanisms [29]. In our framework, the preference labels generated by the VLE model may be noisy due to its inherent limitations. To mitigate this, we adopt a sample-selection strategy inspired by RIME [15], which identifies confident labels for training while flagging uncertain samples for oracle refinement. This combination of robust training and targeted oracle feedback ensures that our reward model remains accurate while keeping preference learning efficient.

III. Preliminaries

We consider the standard RL setup [30], where an agent interacts with an environment in discrete time. At each timestep t, the agent observes a state s_t and selects an action a_t according to its policy π. The environment then provides a reward r_t and transitions the agent to the next state s_{t+1}. The return $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$ represents the discounted sum of future rewards starting at t, where $\gamma \in [0, 1)$ is the discount factor. The goal of RL is to maximize the expected return from each state s_t.

Preference-based RL. In the PbRL setup, an oracle provides feedback in the form of preferences between trajectory segments of the agent. The agent uses these preferences to construct an internal reward model r_θ. Formally, a trajectory segment σ is defined as a sequence of observations and actions: $\sigma = \{(s_1, a_1), (s_2, a_2), ..., (s_T, a_T)\}$. Given a pair of segments (σ_0, σ_1), preferences are expressed as $y \in \{(0, 1), (0.5, 0.5), (1, 0)\}$, where $(1, 0)$ indicates $\sigma_0 \succ \sigma_1$, $(0, 1)$ indicates $\sigma_1 \succ \sigma_0$, and $(0.5, 0.5)$ indicates a tie. Here, $\sigma_i \succ \sigma_j$ denotes that segment i is preferred to segment j. The probability of one segment being preferred over another is modeled via the Bradley-Terry model [31]:

$$P_\theta[\sigma_1 \succ \sigma_0] = \frac{\exp \sum_t r_\theta(s_t^1, a_t^1)}{\sum_{i \in \{0,1\}} \exp \sum_t r_\theta(s_t^i, a_t^i)} \ . \tag{1}$$

To train the reward function r_θ, we minimize the cross-entropy loss between the predicted preferences P_θ and the observed preferences y,

$$\mathcal{L}_{\text{pref}} = -\mathbb{E}_{(\sigma_0, \sigma_1, y) \sim D} \big[y(0) P_\theta[\sigma_0 \succ \sigma_1] + y(1) P_\theta[\sigma_1 \succ \sigma_0] \big] \ . \tag{2}$$

The policy π_ϕ and the reward function r_θ are updated in an alternating fashion. First, the reward function is updated using the sampled preferences. Next, the policy is optimized to maximize the expected cumulative reward under the learned reward function using standard RL algorithms.

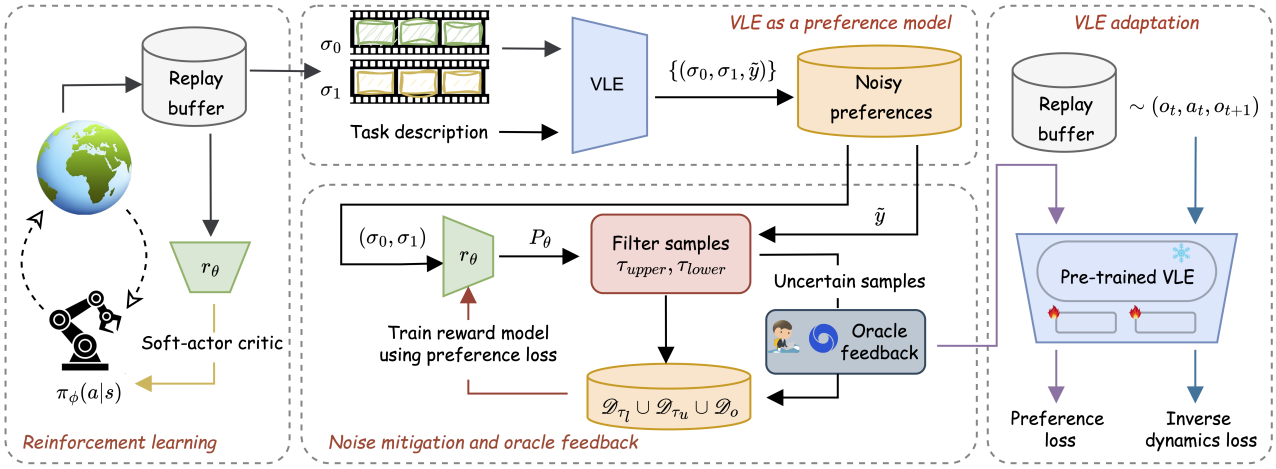

Fig. 1: **Overview of our approach.** Given a task description, ROVED iteratively updates the policy π_ϕ via reinforcement learning using the reward model r_θ. Trajectory segments from the replay buffer are sampled and labeled with VLE-generated preferences. These samples are then classified as clean or noisy using thresholds τ_{upper} and τ_{lower}. A budgeted subset of noisy samples is sent for oracle annotation. The reward model is trained on both VLE and oracle-labeled preferences, while the VLE is fine-tuned using oracle annotations and replay buffer samples.

VLE as a reward model. VLE models comprise a language encoder $\mathcal{F}_L$ and an image encoder $\mathcal{F}_I$, which map text and image inputs into a shared latent space respectively. Using contrastive learning on image-caption pairs, often augmented with task-specific or dynamics-aware objectives [5], VLEs align textual and visual representations effectively. Given the image observation o_t corresponding to a state s_t, and the language description of the task l, most works [5], [7] define the reward as:

$$r_t^{vle} = \frac{\langle \mathcal{F}_L(l), \mathcal{F}_I(o_t) \rangle}{\|\mathcal{F}_L(l)\| \cdot \|\mathcal{F}_I(o_t)\|} \,. \tag{3}$$

While this reward captures some aspects of task progress, it is often too coarse for fine-grained tasks such as manipulation. Although higher rewards tend to correspond to states closer to task completion, the signal is noisy and does not fully reflect nuanced progress [8]. This motivates our approach: VLEs provide a scalable starting point for generating preference labels, which can then be refined with targeted oracle supervision. By improving VLE accuracy selectively, we reduce the need for expensive preference queries while maintaining high-quality reward signals.

IV. Method

In this section, we present ROVED, a framework designed to minimize the reliance on expensive oracle supervision in preference-based RL by leveraging VLEs. We first outline how VLEs are utilized to provide preference feedback for training a reward model (Sec. IV-A). Then, we address the limitations of directly applying VLEs to new environments and propose a data-efficient adaptation approach that combines self-supervised learning with minimal oracle feedback to align the VLE with the environment's dynamics (Sec. IV-B). Finally, to ensure robust training in the presence of noisy machine-generated feedback, we introduce a filtering mechanism that identifies noisy VLE predictions and refines them with targeted oracle input - both to reduce annotation cost and to correct unreliable preference labels (Sec. IV-C). Fig. 1 provides an overview of our approach.

A. Scalable preference generation using VLE

To leverage VLEs for generating preference feedback, we begin by extracting image sequences corresponding to each trajectory segment. Specifically, for a given pair of segments (σ_0, σ_1), we extract the image sequences (O_0, O_1), where each sequence is defined as $O_i = \{o_0^i, o_1^i, o_2^i, \ldots, o_T^i\}$ for $i \in \{0, 1\}$. Here, o_t represents the image observation of the state s_t at the t-th time step.

Using the language description of the task l, we compute the return for each segment as $R_i = \sum_{t=0}^{T} r_t^{vle}$, with r_t^{vle} the reward at time step t derived from the VLE via Eq. 3. Based on these returns, the preference label $\tilde{y}$ is assigned as:

$$\tilde{y} = \begin{cases} (0,1), & \text{if } R_1 > R_0, \\ (1,0), & \text{if } R_1 < R_0, \\ (0.5, 0.5), & \text{otherwise.} \end{cases} \tag{4}$$

These preferences are then used to train a reward model $r_\theta(s_t, a_t)$ by minimizing the preference loss in Eq. 2. The trained reward model can be integrated into a PbRL algorithm for policy optimization. In this work, we specifically leverage PEBBLE [2], a PbRL framework that combines unsupervised pre-training with off-policy learning using Soft Actor-Critic [32].

B. VLE adaptation to improve preference quality

A key challenge in applying VLEs to downstream RL tasks is the domain gap between pretraining data and the target environment [33], [8], which often leads to noisy feedback. We address this with a data-efficient adaptation strategy that aligns the VLE to environment dynamics using minimal oracle feedback and self-supervised learning.

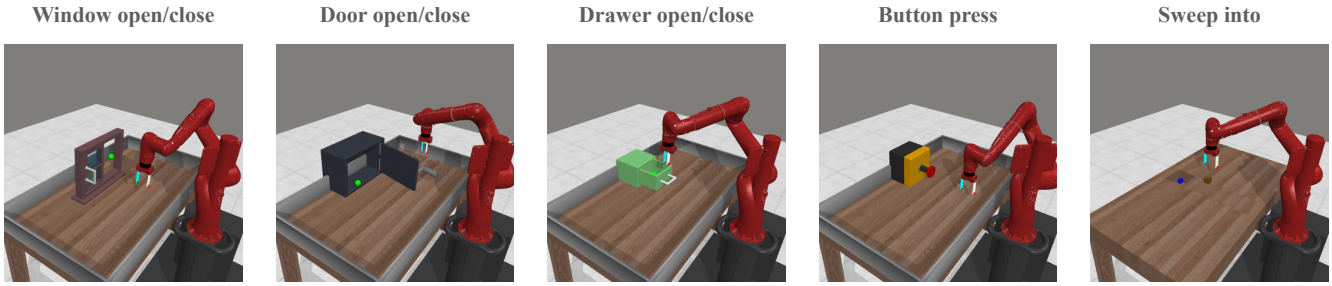

Fig. 2: **Tasks.** An illustration of the different manipulation tasks from Meta-World on which we evaluate our approach.

We freeze the VLE and introduce two learnable layers, $\mathcal{G}_L$ and $\mathcal{G}_I$, on top of the language and image embedding layers, respectively. These layers are fine-tuned to adapt the VLE. The layer $\mathcal{G}_L$ processes the language embedding $\mathcal{F}_L(l)$ of the task description l and outputs an adapted text embedding, $\mathcal{G}_L \circ \mathcal{F}_L(l)$. Similarly, for each image observation o_t, the adapted image embedding is generated by $\mathcal{G}_I \circ \mathcal{F}_I(o_t)$. With these adapted embeddings, the reward for preference feedback is updated as:

$$r_t^{vle} = \frac{\langle \mathcal{G}_L \circ \mathcal{F}_L(l), \mathcal{G}_I \circ \mathcal{F}_I(o_t) \rangle}{\|\mathcal{G}_L \circ \mathcal{F}_L(l)\| \cdot \|\mathcal{G}_I \circ \mathcal{F}_I(o_t)\|}. \tag{5}$$

A small number of oracle-provided preferences are sampled to fine-tune the VLE using the preference loss in Eq. 2. The dense rewards for training are derived from the updated similarity measure (Eq. 5). The methodology for selecting oracle feedback samples is detailed in Sec. IV-C.

We also fine-tune the VLE using an unsupervised objective designed to align the VLE embeddings with environment dynamics. Given the current observation o_t, action a_t, and the next observation o_{t+1}, we train the VLE to predict the action which leads to the transition between observations:

$$\|f\left(\mathcal{G}_I \circ \mathcal{F}_I(o_t), \mathcal{G}_I \circ \mathcal{F}_I(o_{t+1})\right) - a_t\|^2, \tag{6}$$

where f is a linear layer. This encourages the adapted embeddings to capture task-relevant dynamics, improving the precision of preference feedback.

C. Noise mitigation and targeted oracle feedback

The VLE generated preferences, while scalable, are prone to noise and lack the reliability afforded by oracle-provided annotations. To ensure robust training, it is thus crucial to distinguish between accurate and noisy samples. This categorization not only improves the stability of the reward model training but also optimizes the use of oracle feedback by focusing on refining the noisy samples.

Identifying noisy samples. Our approach builds on findings from noisy training [15], which show that deep networks fit clean samples before memorizing noisy labels. We prioritize samples with lower training losses as clean, while treating high-loss samples as potential candidates for oracle review.

Specifically, consider the preference loss defined in Eq. 2 for training the reward model r_θ. Assuming the loss for clean samples is bounded by ρ, Cheng *et al.* [15] show that the KL divergence between the predicted preference distribution P_θ and label $\tilde{y}$ for a noisy sample (σ_0, σ_1) is lower bounded as:

$$D_{KL}(\tilde{y} \| P_\theta) \geq -\ln \rho + \frac{\rho}{2} + \mathcal{O}(\rho^2) \ . \tag{7}$$

To filter out unreliable samples based on Eq. 7, we take a lower bound on the KL divergence, $\tau_{base} = -\ln \rho + \alpha\rho$, where ρ is the maximum loss calculated on the filtered samples from the latest update, and $\alpha \in (0, 0.5]$ is a hyperparameter. To ensure tolerance for clean samples under distribution shifts during policy learning, we introduce an additional uncertainty term in the lower bound, $\tau_{\text{unc}} = \beta_t \cdot s_{KL}$, where s_{KL} is the standard deviation of KL divergence across encountered samples, and β_t decays linearly over time. Specifically, $\beta_t = \max(\beta_{min}, \beta_{max} - kt)$, where $\beta_{\min}$, $\beta_{\max}$, and k are hyperparameters controlling the decay. The final dynamic lower bound is given by $\tau_{lower} = \tau_{base} + \tau_{unc}$.

This adaptive thresholding enables greater tolerance for noisy samples early in training, while becoming more conservative as learning progresses.

Handling noisy samples. Samples with a KL divergence below τ_{lower} are considered clean and are incorporated into the reward model training:

$$\mathcal{D}_{\tau_l} = \{(\sigma^0, \sigma^1, \tilde{y}) : D_{KL}(\tilde{y} \| P_\theta(\sigma_0, \sigma_1)) < \tau_{lower}\} \ . \tag{8}$$

Conversely, samples with a KL divergence exceeding a higher threshold τ_{upper} are presumed to contain noisy labels with high certainty. To preserve their utility, we relabel these samples by flipping their predicted labels and include them in a separate dataset:

$$\mathcal{D}_{\tau_u} = \{(\sigma_0, \sigma_1, \mathbf{1} - \tilde{y}) : D_{KL}(\tilde{y} \| P_\theta(\sigma_0, \sigma_1)) > \tau_{upper}\} \ . \tag{9}$$

The remaining samples, with KL divergence between τ_{lower} and τ_{upper}, are deemed *uncertain*, and we sample from them based on the available oracle annotation budget. These samples are particularly valuable, as both the VLE and reward model struggle to assign reliable labels. By focusing annotation on this subset, $\mathcal{D}_o$, we ensure that annotations address the most uncertain cases.

Training with selective feedback. The reward model is trained on $N_{\text{vle}} = |\mathcal{D}_{\tau_l}| + |\mathcal{D}_{\tau_u}|$ machine-labeled samples, supplemented by $N_{\text{oracle}} = |\mathcal{D}_o|$ oracle-labeled samples from uncertain cases. The N_{oracle} samples are used to update the VLE. This targeted feedback mechanism improves the reward model and the VLE while significantly reducing the overall annotation burden.

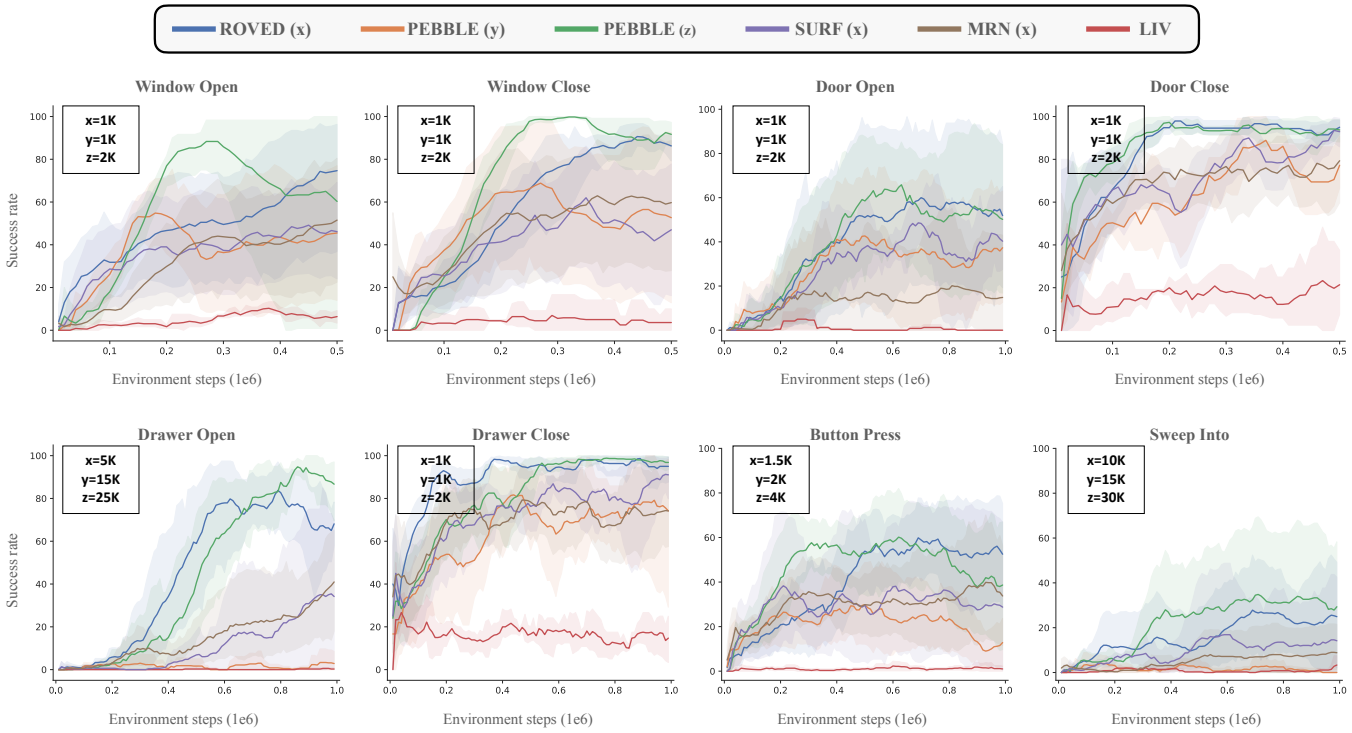

Fig. 3: **Improved feedback efficiency.** ROVED consistently outperforms all baselines with minimal oracle feedback, matching or exceeding PEBBLE's performance while requiring 50%-80% fewer annotations. At equal preference counts, ROVED also outperforms MRN and SURF. Variables (x, y, z) denote the number of oracle preferences used.

D. Overall algorithm

ROVED proceeds by initializing the policy π_ϕ, reward function r_θ, and additional layers $\mathcal{G}$ on top of the VLE randomly. Given a task description l, the method iteratively follows a structured cycle. First, the policy π_ϕ is updated using the reward function r_θ, interacting with the environment and storing observed trajectories in a buffer. From this buffer, trajectory segment pairs are sampled and assigned preferences using the VLE using Eq. 4 and Eq. 5. These labeled pairs are then categorized into clean and noisy samples based on the filtering strategy outlined in Sec. IV-C. A subset of the noisy samples is sent for oracle annotation, subject to a predefined budget. The reward model is updated using the preference-labeled pairs (VLE/oracle annotated) through Eq. 2, while the VLE is fine-tuned using the oracle-annotated samples, following the adaptation strategy outlined in Sec. IV-B.

V. Experiments

A. Experimental setup

Tasks. We evaluate ROVED on eight diverse manipulation tasks from Meta-World [16]: *door-open*, *door-close*, *drawer-open*, *drawer-close*, *window-open*, *window-close*, *button-press*, and *sweep-into*. The tasks are visualized in Fig. 2

Implementation. We use Soft Actor-Critic (SAC) [32] with state-based observations. Actor and critic models are MLPs (3 hidden layers, 256 units) trained with Adam using a learning rate (LR) of $1e{-}4$; the critic τ is set to the default value of $5e{-}3$. The reward model is an ensemble of 3 MLPs (3 hidden layers, 256 units, Leaky ReLU; tanh output), trained with Adam (LR $3e{-}4$, batch size 128) for 200 steps per iteration. The reward model is trained on 30K preference samples in total, combining the allocated oracle preference budget with the remaining preferences generated by the VLE.

We collect 1K random interactions, followed by 5K unsupervised interactions using state-entropy maximization. Policy, reward model, and VLE training begin after these 6K steps. For initialization, 250 oracle feedback samples are used to train both the reward model and the VLE. Subsequently, at every 3K steps, $N = 128$ preference pairs are sampled using segments of length 50 from the SAC replay buffer, following PEBBLE [2]. VLE labels for these pairs are added to the preference buffer $\mathcal{B}$. From $\mathcal{B}$, we sample $\mathcal{D}_{\tau_l}$ and $\mathcal{D}_{\tau_u}$ as described in Sec. IV-C. From the remaining uncertain pairs, up to $0.25N$ are selected for oracle feedback. This procedure is repeated every 3K steps until either the total preference budget or the oracle preference budget is exhausted.

To determine the oracle feedback budget, we first ran PEBBLE with varying budgets (1K–30K) across tasks, and identified the minimum budget to achieve a reasonable success rate ($> 50\%$) in each task. Although larger budgets generally improve performance, our goal is to show that incorporating VLEs can substantially reduce the amount of oracle feedback needed to reach comparable results. We set task-specific oracle budgets based on task complexity.

We experiment with both LIV [5] and DecisionNCE [7] as VLEs, and implement ROVED with LIV by default unless specified. The trainable layers, $\mathcal{G}_L$ and $\mathcal{G}_I$, are 2-layer MLPs (256, 64, ReLU), while the inverse dynamics head f is a

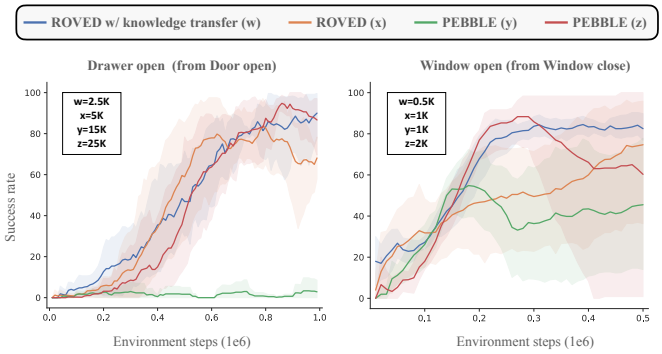

Fig. 4: **Knowledge transfer across tasks.** With knowledge transfer, ROVED matches or surpasses PEBBLE while reducing annotation requirements by 75–90%. This demonstrates effective transfer in both *same task, different object* (left) and *same object, different task* (right) settings. Variables (w, x, y, z) denote the number of preferences used.

128–64–4 MLP with ReLU, with an LR of 3e−4. For label selection, we adopt RIME [15] hyperparameters: $\alpha = 0.5$, $\beta_{min} = 1$, $\beta_{max} = 3$, $k = 1/300$, and $\tau_{upper} = 3\ln(10)$.

VLM oracles. We also evaluate using a VLM-based oracle, following RL-VLM-F [10] with Gemini-Pro-1.5 [9]. Due to API costs[1], results are reported on a subset of tasks. We adopt the exact RL-VLM-F setup (segment length 1, two-step prompting, 20K budget), and additionally evaluate with a reduced budget of 10K. Note that RL-VLM-F requires more queries than PEBBLE as it operates on single-step segments.

Baselines. We compare ROVED against several baselines. *PEBBLE* [2] serves as a basic reference, learning a reward function from human preference feedback and optimizing policies using SAC. SURF [22] and MRN [23] improve preference learning efficiency via semi-supervised augmentation and policy-informed reward updates, respectively. For pure VLE-based approaches, we include *LIV* [5] and *DecisionNCE* [7]. Finally, *RL-VLM-F* [10] is evaluated as a VLM-based oracle. All results are averaged across five random seeds.

B. Main results

Does ROVED improve oracle feedback efficiency? We evaluate whether ROVED can achieve high task performance with substantially less oracle feedback. Fig. 3 shows the learning curves comparing task success rates. ROVED is evaluated using x oracle feedback samples, while PEBBLE, the baseline preference-based method, is assessed with y (similar or higher than x) and z (significantly higher) samples to highlight the impact of feedback efficiency. Efficiency-focused PbRL methods, SURF and MRN, are also evaluated using x oracle samples to examine the gains by ROVED.

Across most tasks, ROVED matches PEBBLE's performance while requiring only 50% fewer oracle queries. For more challenging tasks (*e.g.*, button-press, sweep-into, and drawer-open), the reduction increases to 63.5%, 66.6%, and 80%, respectively. Standalone VLE-based reward models

[1] https://ai.google.dev/gemini-api/docs/pricing#gemini-1.5-pro

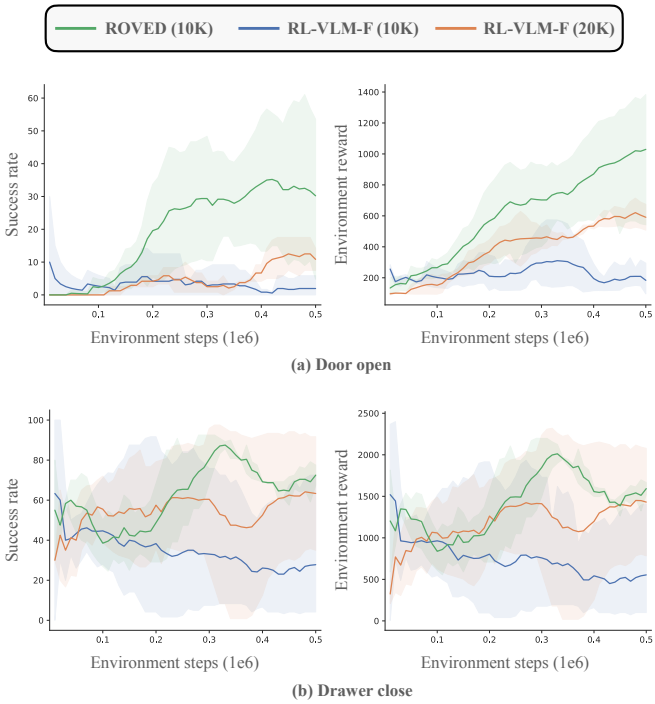

Fig. 5: **Experiments with VLM oracles.** ROVED achieves comparable or better performance than RL-VLM-F while using 50% fewer oracle preferences (denoted by the number in brackets), demonstrating its ability to generalize across different PbRL algorithms. Experiments are reported on a subset of environments due to the high API cost of VLMs.

(LIV and DecisionNCE) fail to achieve meaningful success due to noisy rewards when applied without supervision (see Sec. III), highlighting the need for minimal oracle feedback. While SURF and MRN show moderate gains over PEBBLE at reduced oracle budgets, they still lag behind ROVED across all tasks, emphasizing the effectiveness of our approach.

Can ROVED transfer knowledge across tasks? A key objective of ROVED is to refine the VLE with oracle feedback to reduce inherent noise. We test whether an adapted VLE can generalize to related tasks with minimal additional supervision. Two types of transfer are considered: (1) *same task, different object*: "door-open" → "drawer-open"; (2) *same object, different task*: "window-close" → "window-open". In both cases, the VLE for the target task is initialized with weights from the source task, while the rest of the algorithm remains unchanged. Fig. 4 shows that ROVED with knowledge transfer matches PEBBLE trained with 25K queries using as few as 2.5K, a 90% reduction in oracle queries. Even in the worst case, we observe at least 75% query reduction. This transferability enables cumulative annotation savings across tasks, a feature absent in existing methods.

Is ROVED effective with large vision-language models as oracles? We also evaluate ROVED in settings where large VLMs act as automated oracles, following the RL-VLM-F setup [10] with Gemini-Pro-1.5 [9]. As shown in Fig. 5, ROVED reduces the number of oracle queries by 50% without compromising policy performance and improves cumulative

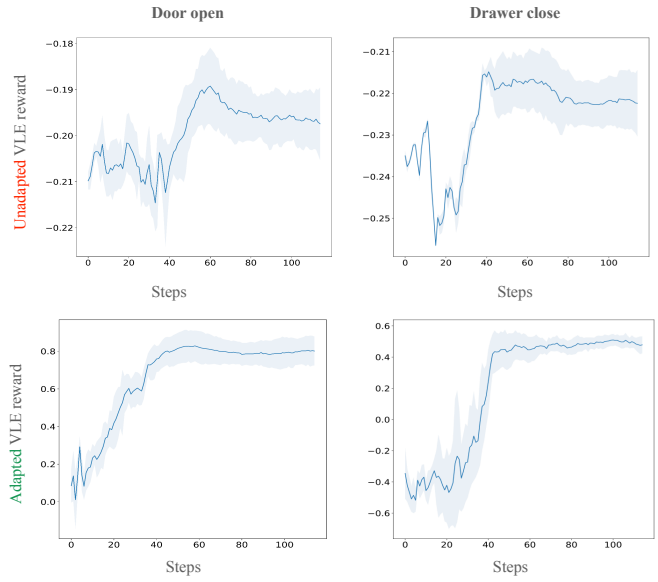

Fig. 6: **Impact of VLE adaptation.** VLE reward predictions on the door-open and drawer-close tasks before (top row) and after (bottom row) adaptation, averaged over the same 5 expert trajectories. After adaptation, the VLE aligns more closely with true task progress.

episode rewards. Although VLMs provide accurate preference labels, they are slow (∼5s per query) and incur high API costs. ROVED leverages lightweight VLEs to automatically generate most preference labels, reserving expensive VLM queries for uncertain cases. This approach produces feedback in under 0.009s—over three orders of magnitude faster—while cutting reliance on costly oracle queries by at least 50%, demonstrating that VLEs are essential for efficient PbRL.

C. Ablation study

Impact of VLE adaptation. To understand how oracle feedback improves the VLE, we analyze reward outputs before and after adaptation. Fig. 6 shows VLE reward predictions on five expert trajectories for the door-open and drawer-close tasks. Ideally, rewards should increase along the trajectory, reflecting progress toward task completion. As shown, the adapted VLE better aligns with ground-truth task progress, producing higher reward values toward the end of successful trajectories without abrupt drops. This reduces reliance on expensive oracle feedback and enables the VLE to serve as a more effective and generalizable prior. Consequently, fine-tuned VLEs transfer across tasks more robustly, compounding annotation savings over time (see Sec. V-B). To further analyze the impact of this adaptation, we trained only the reward model using feedback from the VLE and the oracle, while keeping the VLE fixed (Sec. IV-B). As shown in Fig. 8, although the success rate initially increases, it later drops, since the VLE does not incorporate oracle feedback, leading to increasingly misaligned reward signals.

Robustness to choice of VLE. Our main experiments use LIV [5] as the default VLE, but we also evaluate ROVED with DecisionNCE [7] to test sensitivity to the choice of VLE. As

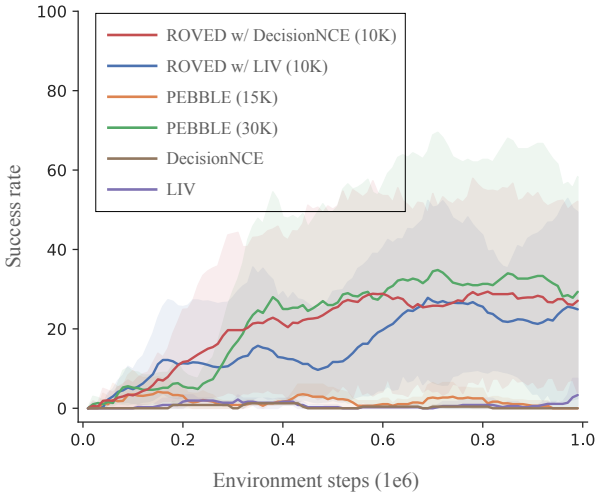

Fig. 7: **Robustness to choice of VLE.** ROVED maintains consistent performance (on the sweep-into task) across different VLE backbones, including LIV and DecisionNCE, highlighting robustness to the choice of embedding model.

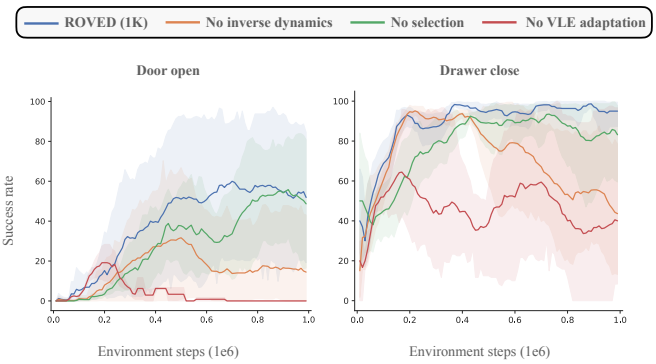

Fig. 8: **Ablation study.** Success rates when (i) removing the inverse dynamics loss from VLE adaptation, (ii) replacing selective feedback with random sampling or (iii) removing VLE adaptation. All ablations show clear performance degradation, underscoring the importance of each component.

shown in Fig. 7 on the sweep-into task, ROVED successfully leverages stronger pre-trained VLEs: DecisionNCE slightly improves performance over LIV. Importantly, DecisionNCE alone fails to solve these tasks without ROVED's uncertainty-aware hybrid components, validating the necessity of selective adaptation and oracle-guided feedback. This shows that ROVED's benefits generalize across VLE backbones and makes the approach adaptable to stronger or more specialized VLEs in future work.

Impact of inverse dynamics loss. To evaluate the efficacy of the self-supervised loss, we trained the VLE using only the preference loss in Eq. 2. As shown in Fig. 8, the success rate initially increases but later degrades. In the early stages, when oracle feedback is available, the VLE performs well by leveraging reliable supervision. However, without the inverse dynamics loss, its performance declines as the policy evolves. This occurs because, while the VLE learns from

oracle preferences, these preferences are derived from samples of a suboptimal policy. As the policy improves and the data distribution shifts, the VLE struggles to generalize due to limited understanding of environment dynamics. Incorporating the inverse dynamics loss allows the VLE to adapt to distribution shifts, maintaining stable performance throughout.

Impact of selective feedback. To evaluate the efficacy of the selection strategy, we evaluate a variant of ROVED where oracle feedback is collected on randomly selected pairs from the replay buffer instead of using the noise mitigation and selection strategy in Sec. IV-C. As shown in Fig. 8, removing this selection achieves reasonable performance and can even surpass PEBBLE with the same amount of feedback, but remains inferior to the full method. Random selection allows noisy labels to influence reward model training and introduces redundancy in oracle feedback, as uncertain segments for the VLE may not be prioritized. This shows the need of selective feedback in maximizing oracle efficiency.

VI. Conclusion

In this work, we introduced ROVED, a hybrid framework for PbRL that combines the scalability of VLEs with the precision of selective oracle supervision. By leveraging an uncertainty-aware sample selection strategy and a parameter-efficient VLE adaptation mechanism, we reduce reliance on costly human or VLM feedback. Across several manipulation tasks, we achieve comparable or superior policy performance while cutting annotation costs by 50–80%. Moreover, our fine-tuned VLE transfers effectively across related tasks, enabling cumulative annotation savings of 75–90%. These results highlight ROVED as a practical step toward more efficient and scalable PbRL.

Acknowledgement. This work was partially supported by The NSF grants under Award No. 2326309 and 2312395, and the UC Multi-campus Research Programs Initiative.

References

[1] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *NeurIPS*, 2017. 1, 2

[2] K. Lee, L. Smith, and P. Abbeel, "Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training," *arXiv preprint arXiv:2106.05091*, 2021. 1, 2, 3, 5, 6

[3] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *ICML*, 2021. 1

[4] J. Rocamonde, V. Montesinos, E. Nava, E. Perez, and D. Lindner, "Vision-language models are zero-shot reward models for reinforcement learning," *arXiv preprint arXiv:2310.12921*, 2023. 1, 2

[5] Y. J. Ma, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman, "Liv: Language-image representations and rewards for robotic control," in *ICML*, 2023. 1, 2, 3, 5, 6, 7

[6] S. Sontakke, J. Zhang, S. Arnold, K. Pertsch, E. Bıyık, D. Sadigh, C. Finn, and L. Itti, "Roboclip: One demonstration is enough to learn robot policies," *NeurIPS*, 2024. 1, 2

[7] J. Li, J. Zheng, Y. Zheng, L. Mao, X. Hu, S. Cheng, H. Niu, J. Liu, Y. Liu, J. Liu, *et al.*, "Decisionnce: Embodied multimodal representations via implicit preference learning," in *ICML*, 2024. 1, 2, 3, 5, 6, 7

[8] Y. Fu, H. Zhang, D. Wu, W. Xu, and B. Boulet, "Furl: Visual-language models as fuzzy rewards for reinforcement learning," *arXiv preprint arXiv:2406.00645*, 2024. 1, 2, 3

[9] G. Team, P. Georgiev, V. I. Lei, R. Burnell, L. Bai, A. Gulati, G. Tanzer, D. Vincent, Z. Pan, S. Wang, *et al.*, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," *arXiv preprint arXiv:2403.05530*, 2024. 1, 6

[10] Y. Wang, Z. Sun, J. Zhang, Z. Xian, E. Biyik, D. Held, and Z. Erickson, "Rl-vlm-f: Reinforcement learning from vision language foundation model feedback," in *ICML*, 2024. 1, 2, 6

[11] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz, "Keyframe-based learning from demonstration: Method and evaluation," *International Journal of Social Robotics*, vol. 4, pp. 343–355, 2012. 1

[12] L. Smith, N. Dhawan, M. Zhang, P. Abbeel, and S. Levine, "Avid: Learning multi-stage tasks via pixel-level translation of human videos," *arXiv preprint arXiv:1912.04443*, 2019. 1

[13] D. J. Hejna III and D. Sadigh, "Few-shot preference learning for human-in-the-loop rl," in *CoRL*, 2023. 1

[14] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning from noisy labels with deep neural networks: A survey," *IEEE transactions on neural networks and learning systems*, vol. 34, no. 11, pp. 8135–8153, 2022. 1

[15] J. Cheng, G. Xiong, X. Dai, Q. Miao, Y. Lv, and F.-Y. Wang, "Rime: Robust preference-based reinforcement learning with noisy preferences," *arXiv preprint arXiv:2402.17257*, 2024. 1, 2, 4, 6

[16] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *CoRL*, 2020, pp. 1094–1100. 1, 5

[17] J. Ho and S. Ermon, "Generative adversarial imitation learning," *NeurIPS*, 2016. 2

[18] H.-T. L. Chiang, A. Faust, M. Fiser, and A. Francis, "Learning navigation behaviors end-to-end with autorl," *IEEE RA-L*, vol. 4, no. 2, pp. 2007–2014, 2019. 2

[19] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar, "Eureka: Human-level reward design via coding large language models," *arXiv preprint arXiv:2310.12931*, 2023. 2

[20] R. Wang, D. Zhao, Z. Yuan, I. Obi, and B.-C. Min, "Prefclm: Enhancing preference-based reinforcement learning with crowdsourced large language models," *IEEE RA-L*, 2025. 2

[21] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *NeurIPS*, 2024. 2

[22] J. Park, Y. Seo, J. Shin, H. Lee, P. Abbeel, and K. Lee, "Surf: Semi-supervised reward learning with data augmentation for feedback-efficient preference-based reinforcement learning," *arXiv preprint arXiv:2203.10050*, 2022. 2, 6

[23] R. Liu, F. Bai, Y. Du, and Y. Yang, "Meta-reward-net: Implicitly differentiable reward learning for preference-based reinforcement learning," *NeurIPS*, vol. 35, 2022. 2, 6

[24] D. Li, Z. Wang, Y. Chen, R. Jiang, W. Ding, and M. Okumura, "A survey on deep active learning: Recent advances and new frontiers," *Transactions on Neural Networks and Learning Systems*, 2024. 2

[25] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi, "Self-instruct: Aligning language models with self-generated instructions," *arXiv preprint arXiv:2212.10560*, 2022. 2

[26] M. Lukasik, S. Bhojanapalli, A. Menon, and S. Kumar, "Does label smoothing mitigate label noise?" in *ICML*, 2020. 2

[27] U. Ghosh, D. S. Raychaudhuri, J. Li, K. Karydis, and A. Roy-Chowdhury, "Robust offline imitation learning from diverse auxiliary data," *Transactions on Machine Learning Research*, 2025. 2

[28] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," *NeurIPS*, 2018. 2

[29] W. Wang, F. Feng, X. He, L. Nie, and T.-S. Chua, "Denoising implicit feedback for recommendation," in *Proceedings of the 14th ACM international conference on web search and data mining*, 2021, pp. 373–381. 2

[30] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018. 2

[31] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons," *Biometrika*, vol. 39, no. 3/4, pp. 324–345, 1952. 2

[32] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *ICML*, 2018. 3, 5

[33] D. S. Raychaudhuri, S. Paul, J. Vanbaar, and A. K. Roy-Chowdhury, "Cross-domain imitation from observations," in *ICML*, 2021. 3