CUDA: Capturing Uncertainty and Diversity in Preference Feedback Augmentation

Sehyeok Kang* *kangsehyeok0329@kaist.ac.kr*
KAIST AI

Jaewook Jeong* *lockcept@kaist.ac.kr*
KAIST AI

Se-Young Yun† *yunseyoung@kaist.ac.kr*
KAIST AI

Reviewed on OpenReview: *https://openreview.net/forum?id=KWENSE1tC4*

Abstract

Preference-based Reinforcement Learning (PbRL) effectively addresses reward design challenges in Reinforcement Learning and facilitates human-AI alignment by enabling agents to learn human intentions. However, optimizing PbRL critically depends on abundant, diverse, and accurate human feedback, which is costly and time-consuming to acquire. Existing feedback augmentation methods aim to alleviate the scarcity of human preference feedback. However, they often neglect diversity, primarily generating feedback for high-confidence trajectory pairs with extreme differences. This approach leads to a biased augmented set that incompletely represents human preferences. To overcome this, we introduce Capturing Uncertainty and Diversity in preference feedback Augmentation (CUDA), a novel approach that comprehensively considers both uncertainty and diversity. CUDA enhances augmentation by employing ensemble-based uncertainty estimation for filtering and extracting feedback from diverse clusters via bucket-based categorization. These two mechanisms enable CUDA to obtain diverse and accurate augmented feedback. We evaluate CUDA on MetaWorld and DMControl offline datasets, demonstrating significant performance improvements over various offline PbRL algorithms and existing augmentation methods across diverse scenarios.

1 Introduction

Preference-based Reinforcement Learning (PbRL) offers an effective solution to the challenges of reward design in Reinforcement Learning (RL), enabling agents to align their actions with human intentions. However, ensuring PbRL performance requires a large amount of diverse and accurate feedback, but obtaining extensive human feedback faces considerable costs and time limitations. To address this, feedback augmentation methods (Park et al., 2022; Hwang et al., 2023; Choi et al., 2024) have emerged, allowing systems to leverage sparse human preferences.

Crucially, however, current augmentation strategies predominantly overlook the diversity aspect of the generated feedback, focusing almost exclusively on its accuracy. For instance, the strategies often opt to select only pairs with extremely confident estimated preference probabilities. This means augmented feedback primarily targets trajectory pairs already exhibiting pronounced differences. Such a selection criterion inherently restricts the diversity of the augmented dataset, creating a concentrated pool of feedback that fails

*Equal contribution. †Corresponding author.
Code available at: `https://github.com/lockcept/CUDA`

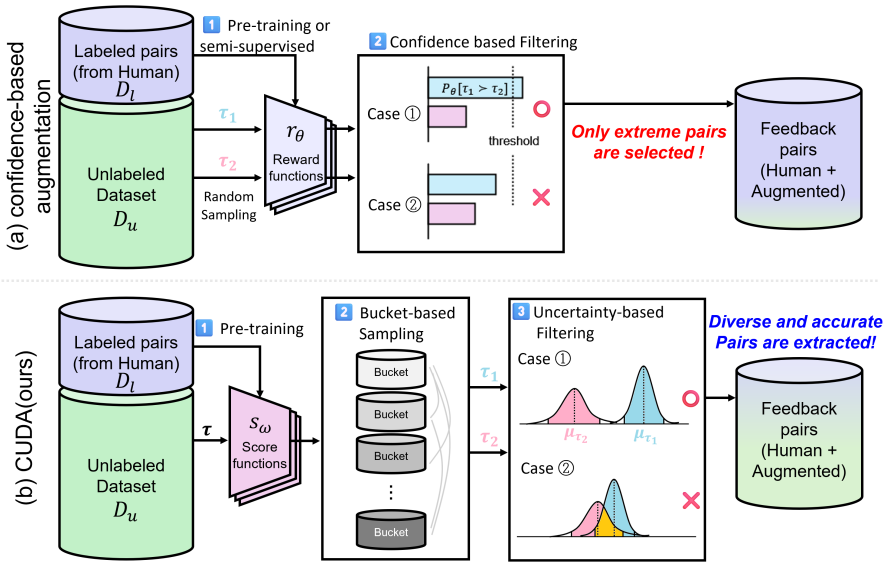


Figure 1: Comparison of CUDA and existing methods. (a) Conventional confidence-based feedback augmentation. (b) The CUDA framework. CUDA ensures diversity through bucket-based sampling, which groups estimated rewards and induces sampling from different buckets. Furthermore, uncertainty-based sampling enables CUDA to select pairs even without extreme differences, achieving diverse yet confident augmentation. CUDA's augmented feedback supports both reward-based and reward-free policy learning. Augmented feedback pairs can be applied to a wide range of PbRL methods (e.g., reward-model-free methods such as IPL, OPRL, and DPPO, as well as reward-based methods such as PT and MR).

to cover the full spectrum of human preferences and nuanced decision-making, and consequently, introduces bias into the augmented feedback. We later illustrate this limitation quantitatively in Section 3 (Figure 2), where conventional confidence-based augmentation is shown to concentrate on trajectory pairs with extreme reward differences, leading to a lack of diversity in the generated feedback.

To overcome these challenges, we introduce CUDA (**C**apturing **U**ncertainty and **D**iversity in preference feedback **A**ugmentation), a novel approach that comprehensively considers both uncertainty (confidence) and diversity in augmenting preference feedback. Figure 1 provides an overview of the overall CUDA framework, which contrasts with conventional confidence-based augmentation by introducing two key mechanisms that jointly promote diversity and reliability:

- **Bucket-based sampling**: CUDA utilizes a strategy that distinguishes and places unlabeled trajectories into multiple buckets based on estimated rewards, and then samples across the buckets to ensure diverse pairs.
- **Uncertainty-based filtering**: Instead of relying on conventional confidence-based filters, CUDA applies ensemble-based uncertainty filtering by selecting only cases where confidence intervals do not overlap. By taking into account the uncertainty of pretrained models and choosing only trajectories where they are confident, CUDA can augment with high accuracy even when the predicted reward gap is small.

Although each of the two mechanisms has been explored in other domains and is not entirely new, they have not been jointly applied to preference augmentation before. Their importance in feedback augmentation lies in the fact that, during feedback augmentation, both the reliability of feedback and its coverage with respect to unseen data are critical for effective reward model training. CUDA integrates these two mechanisms in a balanced manner to capture both aspects. As a result, CUDA obtains diverse and reliable augmented feedback. As will be shown in Section 3, CUDA's augmented feedback exhibits a significantly more diverse distribution than conventional confidence-based approaches, better aligning with the ground-truth reward landscape and ultimately contributing to a superior policy.

We evaluate the performance of CUDA on MetaWorld and DMControl offline datasets provided by Choi et al. (2024). The experimental results show that CUDA significantly outperforms a variety of offline PbRL algorithms and achieves substantial performance gains over augmentation-based methods across a wide range of scenarios.

Contributions. We propose CUDA, an effective method for augmenting preference feedback required in PbRL. By applying bucket-based sampling and uncertainty-based filtering, CUDA selects and augments feedback that is both diverse and novel (unseen), leading to superior empirical performance compared to existing augmentation approaches that consider only a single aspect of feedback generation.

2 Background

2.1 Offline Preference-Based Reinforcement Learning

Preference-Based Reinforcement Learning (PbRL) (Christiano et al., 2017) is a framework for training reinforcement learning agents without explicitly defining a reward function. The goal of offline PbRL is to optimize the policy function $\pi_\theta(a|s)$ using this pre-collected preference data, without any further interaction with the environment. Instead of directly designing a reward signal, a supervisor provides feedback in the form of preferences. Generally, a preference is composed of two trajectories (τ_1, τ_2) and a human feedback label (y) that encodes a comparison between them.

To model these preferences, the Bradley-Terry(Bradley & Terry, 1952) is commonly used in pairwise comparison problems. The probability that trajectory τ_1 is preferred over trajectory τ_2 is given by:

$$P[\tau_1 \succ \tau_2] = \frac{\varphi(f_\theta(\tau_1))}{\varphi(f_\theta(\tau_1)) + \varphi(f_\theta(\tau_2))} \tag{1}$$

Here, $f_\theta(\tau)$ represents the score assigned to trajectory τ by the model, which reflects its estimated quality or preference and φ is an activation function, commonly using either the exponential function or a linear function. The model parameters θ are learned by minimizing the cross-entropy loss between the predicted preference probabilities and the actual human feedback labels. The loss for a single pairwise comparison is defined as:

$$\mathcal{L} = -(y \log P[\tau_1 \succ \tau_2] + (1-y) \log P[\tau_2 \succ \tau_1]) \tag{2}$$

where $y = 1$ if τ_1 is preferred to τ_2, and $y = 0$ otherwise.

By optimizing this loss function over a set of labeled trajectory pairs, the model aligns its predictions with human preferences, effectively inferring a reward signal without the need for manual design.

There have been several approaches to optimizing policies in offline PbRL. Preference Transformer (PT) (Kim et al., 2023) models human preferences using a Transformer-based architecture and proposes a method to learn non-Markovian rewards as a weighted sum, extending existing Markovian reward assumptions. This approach effectively reflects temporal dependencies and enables the solution of complex control tasks. Offline Preference-based Reward Learning (OPRL) (Shin et al., 2023) approached offline PbRL by selecting queries with high Value of Information through active learning. Direct Preference-based Policy Optimization without reward modeling (DPPO) (An et al., 2023) learns policies directly from preference by utilizing a contrastive learning framework. Inverse Preference Learning (IPL) (Hejna & Sadigh, 2024) proposes an efficient algorithm that directly learns preference data using Q-functions without explicitly learning a reward function. This design allows for effective learning with a simpler structure and fewer parameters.

2.2 Feedback Augmentation in PbRL

Feedback augmentation is a key technique for improving the performance of PbRL by maximizing the information gained from limited human feedback.

Table 1: Performance difference between confidence-based feedback augmentation and real feedback. **Confidence** refers to training with 10,000 augmented pairs generated using a reward model trained on 500 human feedback, via confidence-based augmentation. In contrast, **Real** represents obtaining 10,000 feedback samples using a scripted teacher model. Although the augmented feedback achieved near-perfect accuracy, Real and Confidence still show a significant performance difference, highlighting that accuracy alone is insufficient for effective policy learning.

Method	button-press	box-close	sweep	Average
Confidence	0.55 ± 0.18	0.63 ± 0.27	0.76 ± 0.10	0.65
Real	0.73 ± 0.32	0.89 ± 0.13	0.86 ± 0.15	0.83
Differences	-0.18	-0.26	-0.10	-0.18

SURF (Park et al., 2022) first introduces feedback augmentation to PbRL, particularly within an online learning framework. Its methodology involves augmenting preference pairs by filtering for those where the estimated reward based preference probability exceeds a specific threshold, subsequently leveraging these augmented feedback for semi-supervised learning. The reliance of SURF on only confidence-based filtering limits its augmented data to trajectory pairs exhibiting significant disparities. Our approach, however, can include pairs with even subtle differences between trajectories, leading to the creation of a richer and more diverse set of augmented feedback.

SeqRank (Hwang et al., 2023) utilizes sequential preference ranking to efficiently generate additional feedback through comparisons between defender and challenger. LiRE (Choi et al., 2024) proposed a method to improve the performance of the reward model without additional data collection by constructing a Ranked List of Trajectories (RLT) using second-order preferences. SeqRank and LiRE use trajectories for augmentation only if those trajectories have been included in a feedback ranking at least once. On the other hand, our approach can include unlabeled trajectories in the feedback generation process. This allows the model to infer preferences for previously unseen trajectories, which can greatly expand the scope of data augmentation. In addition, unlike existing methods, which require sequential and online feedback collection, our method can learn from arbitrary and simultaneous feedback without being bound by strict order.

Beyond PbRL for control, the trade-off between preference pair difficulty and downstream policy learning has also been studied in the context of LLM post-training and alignment. For example, RLCD (Yang et al., 2023) generates preference pairs from contrastive prompts to produce cleaner labels with larger preference gaps, while Curriculum-RLAIF (Li et al., 2025) progressively incorporates pairs of increasing difficulty to improve reward model generalization. These works highlight that how preference pairs are selected, in terms of both difficulty and coverage, has a direct effect on downstream learning dynamics, a perspective that aligns with the motivation of our approach.

3 Problem Definition

3.1 Offline Preference-based Reinforcement Learning

In PbRL, an agent learns a policy by utilizing human feedback rather than explicit reward functions. For the offline setting, we assume the following conditions. Given an offline dataset $\mathcal{D}$, it comprises a large quantity of unlabeled trajectories $\mathcal{D}_u$ and a limited number of trajectory pairs $\mathcal{F}_l$ labeled by human feedback. The goal is to learn an optimal policy $\pi_\theta(a|s)$ that maximizes the inferred reward signal derived from these preference labels.

3.2 Insufficient diversity in feedback augmentation

Optimizing the reward function with only a limited number of labeled pairs ($\mathcal{F}_l$) poses challenges. Consequently, some researchers augment preference feedback based on estimated rewards. They implement confidence-based approaches that calculate probabilities using estimated rewards from Eq. (1) and add a pair as augmented feedback if the value exceeds a certain threshold. However, this method does not guar-

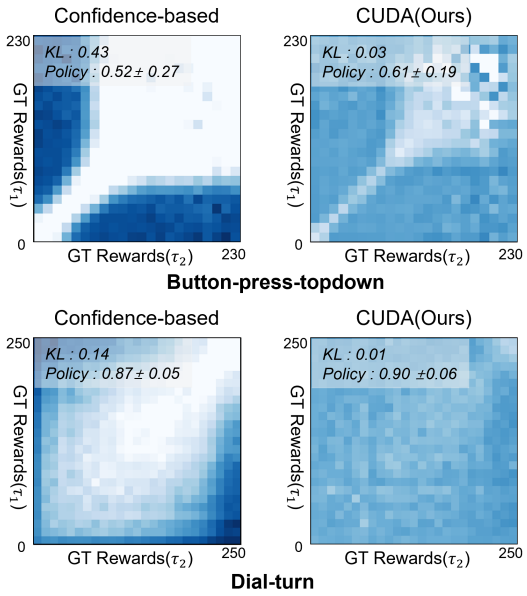


Figure 2: **Comparison of conventional augmentation methods (confidence-based) and CUDA.** The heatmap displays the distribution of ground truth rewards for the augmented pairs each model generates. The intensity of the heatmap indicates concentration. Confidence-based augmentation concentrates augmented feedback where two trajectories have large differences. In contrast, CUDA's augmented pairs show a dispersed distribution. *KL* denotes the KL divergence between the distribution of augmented feedback and that of the offline dataset, computed through ground-truth preference labels, which indicates whether the feedback distribution is skewed relative to the offline dataset. *Policy* indicates the success rate of the policy trained with augmented feedback. CUDA, notably, both aligns more closely with the ground truth reward and makes a more robust policy.

antee diverse augmented feedback. Typically, only pairs with extremely large trajectory differences exceed the threshold, meaning the augmented data does not include common cases, leading to inherent bias in the augmented feedback. Figure 2 illustrates this point, with the left (confidence-based) showing the results of feedback augmentation using conventional strategies that filter using the difference between trajectories. In these scenarios, augmentation predominantly occurs for trajectory pairs where the reward difference is extreme, leading to a lack of diversity in the generated feedback. As a result, it shows a significant performance difference compared to using real feedback. Although confidence-based augmentation achieves high agreement with real labels, it leads to significantly lower policy performance compared to using 10,000 real feedbacks (Table 1). This analysis indicates that diversity, in addition to confidence, plays a crucial role in shaping policy performance. Therefore, we require a novel feedback augmentation approach that resolves the diversity limitations and provides both diverse and confident feedback.

4 Method

4.1 Our Method: CUDA

We propose **C**apturing **U**ncertainty and **D**iversity in preference feedback **A**ugmentation (CUDA), a unified framework that combines score model learning with feedback augmentation. Unlike existing augmentation methods that operate on individual data samples, CUDA performs *relational augmentation* by combining existing trajectories to construct new pairsets with expected preferences. CUDA follows a two-stage pipeline. In the first stage, we train a **score function** on the labeled feedback and use it for feedback augmentation, including ensemble-based uncertainty estimation and bucket construction over trajectory-level returns. In the second stage, we train a **reward function** from scratch on the combined (original and augmented) feedback and use its per-step outputs to train the policy via IQL. The two functions share the same network architecture but differ in the final-layer activation, reflecting the distinct requirements of each stage. While our overall goal is to capture both reliability and coverage in the augmented feedback, we achieve this through three main components:

(1) We train an ensemble of score functions with random initializations on a limited set of labeled feedback to enable robust return prediction and uncertainty estimation.

(2) We partition the unlabeled trajectories into multiple buckets based on mean predictions, and sample feedback from inter-bucket pairs to ensure diversity.

(3) To maintain reliability, we filter out trajectory pairs with high predictive uncertainty.

Bucket-based sampling and uncertainty-based filtering address complementary aspects of feedback augmentation: bucket-based sampling extends the augmented distribution beyond extreme pairs to improve diversity, while uncertainty-based filtering preserves label accuracy even for the low-margin comparisons that bucketing surfaces. Neither component alone is sufficient, as confirmed empirically in Figure 4a.

This integrated pipeline allows us to selectively generate preference labels that are both diverse and reliable, enhancing reward model performance with minimal human effort.

4.2 Score Functions from Labeled Feedback

We use the Markovian Reward (MR) model as our score function architecture. During training, we apply an exponential transformation $\varphi(x) = e^x$, which is used to define the probability in the loss function.

We train an ensemble of N score functions with different random initializations on the labeled data. For each state-action pair (s_t, a_t) in a trajectory $\tau = \{(s_1, a_1), \ldots, (s_T, a_T)\}$, each model predicts a score $r_n(s_t, a_t)$, and the ensemble mean is given by:

$$\bar{r}_t = \frac{1}{N} \sum_{n=1}^{N} r_n(s_t, a_t) \tag{3}$$

The predicted return for the trajectory is computed as:

$$\hat{R}(\tau) = \sum_{t=1}^{T} \bar{r}_t \tag{4}$$

To estimate uncertainty, we compute an uncertainty score $u(\tau)$ by summing the variance of the score predictions at each timestep:

$$u(\tau) = \left(\sum_{t=1}^{T} \left(\frac{1}{N} \sum_{n=1}^{N} \left(r_n(s_t, a_t) - \bar{r}_t \right)^2 \right) \right)^{1/2} \tag{5}$$

This state-wise variance captures local prediction disagreement across the ensemble, and its sum provides a trajectory-level uncertainty estimate used for data filtering.

4.3 Bucket-based Sampling

To promote diversity in the augmented feedback, we cluster the set of unlabeled trajectories into k buckets using k-means++ (Arthur & Vassilvitskii, 2006) clustering based on their predicted return $\hat{R}(\tau)$. Instead of fixing k in advance, we search for the optimal number of clusters within the range $[\frac{1}{2}k, k]$, and select the value of k that maximizes the silhouette score (Rousseeuw, 1987). The method ensures more natural groupings of trajectories with similar rewards. Appendix B.2 empirically supports this adaptive bucketing (with vs. without k-means) and visualizes the resulting bucket structure.

To construct pairs, we allocate the number of feedback pairs for each bucket pair (i, j) with $i < j$ as

$$N_{ij} = \alpha \, |\mathcal{B}_i| \, |\mathcal{B}_j|, \tag{6}$$

where $\mathcal{B}_i$ denotes the set of trajectories in bucket i. The normalization factor α is chosen such that the total number of generated pairs is approximately n:

$$\alpha = \frac{n}{\sum_{i<j} |\mathcal{B}_i| \, |\mathcal{B}_j|}. \tag{7}$$

The strategy emphasizes high-volume bucket pairs while preserving return diversity, resulting in a balanced and representative feedback set for reward model training. A bucket pair (i, j) admits $|\mathcal{B}_i| \cdot |\mathcal{B}_j|$ cross-bucket

comparisons, so allocating the sampling budget in proportion to this quantity covers the available pair space in a way that reflects the empirical structure of the unlabeled dataset.

We restrict augmentation to inter-bucket pairs. Since buckets are formed by k-means++ clustering on predicted returns, trajectories within the same bucket have similar score estimates, so intra-bucket pairs tend to have small score differences and are therefore less suitable for constructing binary preference labels under the Bradley-Terry model.

4.4 Uncertainty-based Filtering

To ensure the confidence of augmented feedback, we apply uncertainty based filtering to the pairs selected from bucket-based sampling. For each trajectory τ, we use the uncertainty score $u(\tau)$ defined in Eq. (5).

When constructing a pair between two trajectories τ_i and τ_j from bucket $i < j$, we include the pair only if their predicted returns $\hat{R}(\tau_i)$ and $\hat{R}(\tau_j)$ are separated by a sufficient margin relative to their uncertainties. Specifically, the pair is chosen if it satisfies the following condition:

$$\hat{R}(\tau_i) + z \cdot u(\tau_i) < \hat{R}(\tau_j) - z \cdot u(\tau_j) \tag{8}$$

where z is a margin parameter that controls the tolerance for uncertainty. This criterion ensures that the relative preference between τ_i and τ_j is inferred only when the pretrained score functions are confident about the comparison, thereby filtering out pairs with low certainty.

We combine the original labeled feedback with the augmented preference pairs generated from unlabeled trajectories to construct the final training set of preference feedback.

4.5 Reward Learning and Policy Optimizing

In Section 4.2, we first train the score function using the original feedback and then use it to augment additional preference feedback. On the combined (original and augmented) feedback, we then train a *reward function* from scratch. The reward function shares the backbone of the score function but uses a different final-layer activation, as explained below.

While exponential transformations ($\varphi(x) = e^x$) tend to distribute rewards as evenly as possible, linear transformations ($\varphi(x) = x$) emphasize distributions with a few high-reward trajectories and many low-reward ones, which provides an advantage in policy performance. For uncertainty estimation, however, we employ an exponential transformation. We empirically verify this difference in Appendix C, where the linear transformation yields a sharper separation of high- and low-reward trajectories compared to the exponential one.

The use of a linear activation function is also motivated by its ability to better capture second-order preferences (e.g., if $A \succ B$ and $B \succ C$, then $A \succ C$) (Choi et al., 2024). Since the augmented feedback contains more such transitive preference patterns than the original data, a model with a linear activation function and a positive-valued output layer is more suitable for learning from the augmented feedback.

Once trained, the reward model is used to generate pseudo-rewards for the unlabeled offline dataset. These pseudo-rewards are then used to train a policy using Implicit Q-Learning (IQL) (Kostrikov et al., 2021) , allowing the agent to optimize behavior aligned with the learned preferences.

5 Experimental Results

5.1 Experimental Settings

5.1.1 Offline Dataset

We evaluate CUDA on two widely used benchmark datasets in offline reinforcement learning: Meta-World (Yu et al., 2020) and DeepMind Control Suite (DMControl) (Tassa et al., 2018). These environments provide

Table 2: Average success rates on Metaworld. The blue background indicates the best performance, while the gray background denotes the second-best performance.

	Algorithms	button-press-topdown	box-close	dial-turn	sweep	button-press-topdown-wall	sweep-into	drawer-open	lever-pull	avg
	Ground Truth*	0.69 ± 0.11	0.75 ± 0.02	0.51 ± 0.04	0.68 ± 0.06	0.30 ± 0.05	0.43 ± 0.05	0.20 ± 0.16	0.25 ± 0.19	0.48
Offline PbRL	MR-linear*	0.33 ± 0.23	0.44 ± 0.35	0.40 ± 0.14	0.95 ± 0.04	0.13 ± 0.09	0.25 ± 0.08	0.16 ± 0.16	0.51 ± 0.06	0.40
	MR-exp*	0.03 ± 0.03	0.35 ± 0.37	0.30 ± 0.29	0.92 ± 0.11	0.05 ± 0.05	0.31 ± 0.11	0.15 ± 0.05	0.79 ± 0.24	0.33
	PT*	0.11 ± 0.19	0.01 ± 0.03	0.26 ± 0.27	0.47 ± 0.04	0.02 ± 0.04	0.33 ± 0.12	0.02 ± 0.01	0.84 ± 0.07	0.26
	IPL*	0.10 ± 0.07	0.01 ± 0.01	0.11 ± 0.06	0.06 ± 0.07	0.01 ± 0.01	0.14 ± 0.04	0.19 ± 0.10	0.03 ± 0.03	0.09
	OPRL†	0.12 ± 0.06	0.04 ± 0.03	0.54 ± 0.11	0.94 ± 0.06	0.00 ± 0.00	0.26 ± 0.08	0.94 ± 0.06	0.54 ± 0.12	0.42
	DPPO†	0.04 ± 0.04	0.10 ± 0.11	0.27 ± 0.22	0.10 ± 0.16	0.01 ± 0.01	0.23 ± 0.07	0.36 ± 0.11	0.10 ± 0.12	0.15
Augmentation	Random *	0.54 ± 0.21	0.60 ± 0.18	0.81 ± 0.10	0.79 ± 0.16	0.28 ± 0.15	0.34 ± 0.10	0.07 ± 0.06	0.83 ± 0.08	0.53
	SURF(offline)*	0.37 ± 0.26	0.33 ± 0.36	0.38 ± 0.17	0.85 ± 0.28	0.13 ± 0.07	0.19 ± 0.07	0.09 ± 0.08	0.53 ± 0.15	0.36
	SeqRank*	0.34 ± 0.26	0.01 ± 0.02	0.26 ± 0.12	0.35 ± 0.33	0.02 ± 0.04	0.25 ± 0.08	0.27 ± 0.13	0.17 ± 0.25	0.20
	LiRE*	0.57 ± 0.21	0.79 ± 0.18	0.82 ± 0.16	0.68 ± 0.33	0.28 ± 0.07	0.33 ± 0.09	0.01 ± 0.06	0.82 ± 0.28	0.54
	CUDA (Ours)	0.59 ± 0.27	0.64 ± 0.23	0.90 ± 0.06	0.88 ± 0.07	0.29 ± 0.11	0.36 ± 0.09	0.04 ± 0.06	0.84 ± 0.05	0.56

$*$ shows our implementation results, and † presents results from Choi et al. (2024)

Table 3: Performance on DMControl. The blue background indicates the best performance, while the gray background denotes the second-best performance.

Algorithm	hopper-hop	walker-walk	humanoid-walk	avg
Ground Truth†	157.95 ± 9.64	839.6 ± 36.57	250.9 ± 11.62	416.15
MR-linear*	127.69 ± 26.22	635.02 ± 94.35	89.92 ± 12.78	284.21
SURF*	127.60 ± 29.00	696.56 ± 78.61	100.95 ± 15.21	308.37
SeqRank†	80.84 ± 27.67	698.81 ± 91.71	80.68 ± 14.67	286.77
LiRE†	99.14 ± 12.28	822.27 ± 50.83	104.08 ± 17.45	341.83
CUDA (Ours)	132.50 ± 19.68	784.64 ± 34.21	148.06 ± 35.94	355.07

$*$: our implementation; †: results from Choi et al. (2024)

diverse tasks for evaluating both locomotion and robotic manipulation. We use the dataset collected by Choi et al. (2024) from Meta-World and DMControl. For our experiments, we utilize feedback generated by a scripted teacher based on this dataset. The scripted teacher, a widely used method in PbRL (Choi et al., 2024; Hwang et al., 2023; Kim et al., 2023), replaces human feedback with rewards from the environment to pose preferences. These environmental rewards are solely used for preference calculation and not for model training.

5.1.2 Baselines

To evaluate the effectiveness of our proposed method, we compare it against multiple baselines, including both offline PbRL methods and feedback augmentation-based PbRL methods. We implement a Markovian Reward (MR) baseline, where the reward model is trained under the assumption that the reward depends only on the current state-action pair (s, a), following the standard Markov decision process (MDP) formulation. We also evaluate an MR-linear baseline, which differs from MR-exp in the activation functions used. Specifically, while MR-exp employs an exponential activation($\varphi(x) = e^x$) in Eq. (1), MR-linear adopts a linear activation ($\varphi(x) = x$) and applies $\tanh(x) + 1$ in the last layer of the reward model. Moreover, we compare CUDA against Preference Transformer (PT) Kim et al. (2023), OPRL Shin et al. (2023), DPPO An et al. (2023), and IPL Hejna & Sadigh (2024) to evaluate its performance relative to other PbRL methods. Furthermore, we compare CUDA with feedback augmentation methods such as Random, which selects arbitrary segment pairs and labels them using a trained reward model, as well as SURF Park et al. (2022), SeqRank Hwang

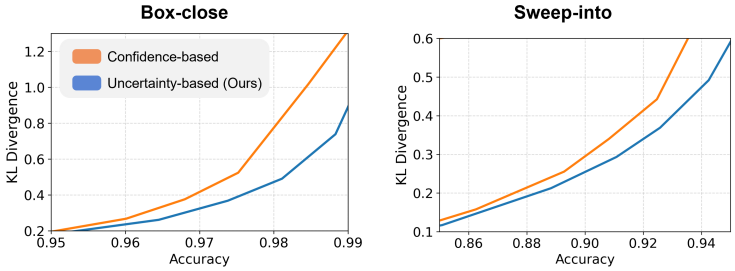


Figure 3: KL divergence of the selected feedback distributions compared to random sampling, measured at matched accuracy levels by adjusting the confidence threshold (for confidence-based methods) or the z-value (for uncertainty-based methods). Lower KL indicates greater trajectory diversity and better coverage of the offline dataset, enabling richer preference inference.

et al. (2023), and LiRE Choi et al. (2024). We adapt and re-implement the online version of SURF for use in the offline setting.

5.1.3 Implementation Details

Both CUDA and all baselines utilize a 500 feedback pair dataset per scenario for model training. Also, all reward models—including those used in baselines and CUDA—are trained using the MR-linear architecture. For policy learning, we employ Implicit Q-Learning (IQL) (Kostrikov et al., 2021) as the policy optimization algorithm. Each experiment is run with ten different random seeds, and the final results are reported as the mean and standard deviation across these runs. Since policy performance is dependent on the distribution of the pair set, we ensure fairness by using the same pair set and unlabeled trajectory dataset configuration across all experiments. Further details on hyperparameters settings are provided in the Appendix A.2.

5.2 Main Results

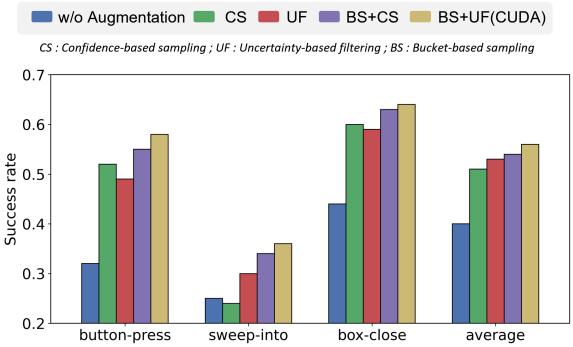
To evaluate the performance of CUDA, we conduct a comparative study against existing offline PbRL approaches and PbRL methods that utilize feedback augmentation. Table 2 presents the average success rates of various methods across different tasks in the MetaWorld environment. The results indicate significant performance variations among the methods, highlighting the effectiveness of different algorithms. CUDA demonstrates superior performance across multiple tasks, achieving the highest success rates in five tasks. Additionally, the average value for CUDA across all scenarios surpasses other baselines. Notably, CUDA outperforms offline PbRL methods.

Moreover, it also demonstrates superiority when compared to augmentation methodologies. The Random strategy, which applies augmentation to arbitrarily selected samples, also shows improved performance compared to the no-augmentation baseline. However, due to the lower accuracy of the augmented feedback, its performance remains inferior to that of CUDA. While most augmentation methods generally lead to improved performance, they can sometimes degrade performance in tasks where accurate feedback is critical—such as in sweep environments. Nevertheless, CUDA consistently outperforms other augmentation strategies, even in such challenging settings. Overall, CUDA exhibits both robustness and adaptability, consistently outperforming alternative augmentation strategies, including in tasks that demand high-precision feedback.

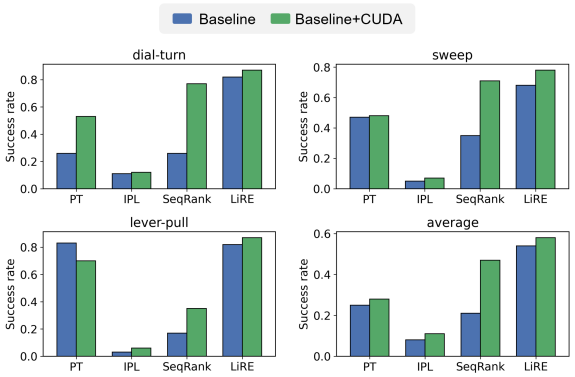
Additionally, our method demonstrates better performance on the DMControl environment, as shown in Table 3, which presents the average rewards of different methods on DMControl. CUDA consistently shows improved performance against baselines. These results collectively prove that CUDA is capable of generating better augmented feedback in diverse scenarios and across different environments.

Table 4: KL divergence under different strategies. **CS** = Confidence-based Sampling; **BS** = Bucket-based Sampling.

Method	button-press-topdown	sweep-into	box-close	avg
CS	0.429	0.268	0.316	0.331
CS+BS	**0.030** (↓0.39)	**0.011** (↓0.25)	**0.012** (↓0.30)	**0.025** (↓0.30)



(a) Performance comparison with and without application of each strategy. *CS* denotes the conventional augmentation method.



(b) Experimental results evaluating the compatibility of CUDA with various PbRL methods and feedback augmentation techniques.

Figure 4: (a) Performance evaluation of CUDA strategies and (b) CUDA compatibility tests.

5.3 The effectiveness of CUDA strategies

5.3.1 Uncertainty-based filtering

Figure 3 compares confidence-based and uncertainty-based filtering by plotting KL divergence against accuracy (agreement with ground-truth preference labels). KL divergence relative to random sampling serves as a measure of sampling bias, where lower values indicate better alignment with the offline dataset.

Across all tasks, uncertainty-based filtering achieves consistently lower KL divergence than confidence-based sampling at the same accuracy level. Unlike confidence, which relies solely on the difference in predicted rewards, incorporating pretrained model uncertainty allows maintaining comparable accuracy while sampling a more diverse set of trajectories. Such distributional diversity leads to a better approximation of the original dataset and ultimately supports improved generalization and policy performance. Simply lowering the confidence threshold does not close this gap: it shifts the operating point along the confidence-based curve in Figure 3, which remains above the uncertainty-based curve.

5.3.2 Bucket-based sampling

Table 4 shows the effect of applying bucket-based sampling. Bucket-based sampling consistently reduces KL divergence across environments. This suggests that bucket-based sampling enhances feedback diversity by preventing bias toward extreme pairs.

5.3.3 Overall Policy Performance Comparison

Figure 4a summarizes the overall policy performance under different combinations of strategies. **CUDA** (BS + UF) achieves the highest performance across environments, while using only confidence-based sampling (CS) results in the worst performance among the augmented variants.

Table 5: Experimental results using human feedback on the `button-press-topdown` task in Metaworld. The values in the table represent success rates.

Algorithms	**MR-linear**	**SeqRank**	**LiRE**
w/o CUDA	0.15 ± 0.07	0.22 ± 0.10	0.32 ± 0.10
with CUDA	$\mathbf{0.37 \pm 0.16}$	$\mathbf{0.41 \pm 0.19}$	$\mathbf{0.94 \pm 0.06}$
Differences	+0.22	+0.19	+0.62

Table 6: Sensitivity of CUDA to the initial number of labeled preference pairs on `box-close-v2`.

# of feedback	MR-linear	CUDA (ours)	Δ
100	0.04	0.34	+0.30
200	0.13	0.48	+0.35
500	0.44	0.64	+0.20
1000	0.60	0.66	+0.06

5.4 Comparability of CUDA

To evaluate CUDA's compatibility by assessing whether the augmented feedback it generates can enhance the performance of various existing offline PbRL methods, we apply CUDA to PT, IPL, SeqRank and LiRE. Figure 4b presents the results of the application. CUDA can be integrated into both reward model–dependent methods such as PT and reward model–free approaches like IPL, and it generally yields performance improvements across both types (indicated by green). Even for models like SeqRank and LiRE that already incorporate augmented data, CUDA's augmented data significantly contributes to improving policy performance. Therefore, CUDA is compatible with both preference-based RL and augmentation-based approaches.

5.5 Alignment with human feedback

To evaluate how well CUDA aligns with human preferences, we conduct experiments using human-provided feedback. Table 5 shows the performance in the `button-press-topdown` task in the Metaworld. We use 200 human preference feedback samples provided by Choi et al. (2024). CUDA not only improves performance when applied to MR-Linear alone, but also shows synergistic effects when combined with existing augmentation methods. Notably, its integration with LiRE yields a significant improvement in success rate.

5.6 Ablation Studies

5.6.1 Number of buckets(k)

CUDA leverages bucket-based sampling to achieve feedback diversity. Figure 5 (a) depicts policy performance changes relative to the number of buckets. We determine the optimal bucket count using the silhouette score, selecting a value within minimum and maximum bounds. Experiments show the best performance when k ranges from 10 to 20.

5.6.2 Number of augmented feedback(n)

Figure 5(b) illustrates how policy performance varies with the number of augmented feedback samples. While increasing the amount of augmented feedback generally improves performance, the gains saturate and peak at $n = 10{,}000$, after which performance degrades. This behavior is consistent with prior observations that excessive augmented or pseudo-labeled supervision can introduce noisy or low-quality labels, leading to diminished returns Lee (2013). Based on this trend, we select $n = 10{,}000$ as a suitable augmentation scale for our experiments.

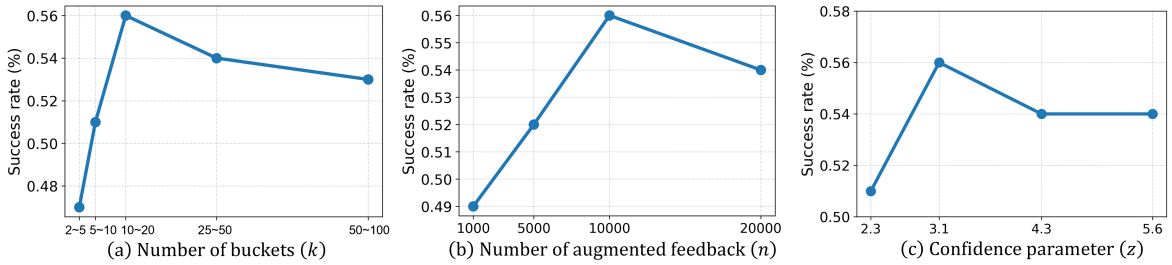


Figure 5: Ablation study findings. The success rate in the graph represents the average value across all tasks in Metaworld. (a) Success rate according to changes in the number of buckets (k). (b) Success rate based on the number of augmented feedback (n). (c) Success rate variation with respects to confidence parameter z

5.6.3 Confidence parameter(z)

We use z as the margin parameter for uncertainty-based filtering in Equation (8), which controls when pretrained reward models are regarded as confident about a comparison. Figure 5 (c) compares performance across different z values to assess its impact, showing that the best performance is achieved at $z = 3.1$.

5.6.4 Sensitivity to the initial feedback budget

Since CUDA is motivated by the limited-feedback setting, we further examine how its benefit varies with the number of original labeled preference pairs. We compare CUDA against the non-augmented MR-linear baseline on `box-close-v2` under feedback budgets of $\{100, 200, 500, 1000\}$ pairs. As shown in Table 6, CUDA consistently outperforms the baseline across all budgets, and the improvement is largest in the low-feedback regime (e.g., +0.30 at 100 pairs and +0.35 at 200 pairs, compared to +0.06 at 1000 pairs). This indicates that CUDA is particularly effective precisely in the regime it is designed for, namely when human preference labels are scarce. A complementary analysis of how reward-model quality scales with feedback size and label corruption is provided in Appendix D.2.

Additional sensitivity analyses related to the score-function ensemble size and the robustness of CUDA to label noise in the original feedback are provided in Appendix D.

6 Limitations

Since CUDA augments feedback based on a model trained with original feedback, it becomes crucial to perform effective initial sampling and obtain accurate true feedback. Inadequate feedback hinders its ability to perform well. Furthermore, maintaining the preference relationships between feedback pairs is essential for effective augmentation. For example, cyclic feedback has limitations in generating buckets, thus impairing performance. Therefore, as our future work, we are considering methods to obtain high-quality feedback during the initial feedback acquisition process, and methods to robustly train the score function even when feedback with cycles is given. In addition, CUDA employs an ensemble of score models for robust feedback augmentation, which increases computational and memory costs during training and may limit scalability to large-scale or resource-constrained settings.

7 Conclusion

In this work, we propose Capturing Uncertainty and Diversity in Preference Feedback Augmentation (CUDA), a method that addresses the challenge of limited human feedback in PbRL. CUDA introduces a novel approach that comprehensively integrates both uncertainty (confidence) and diversity into the augmentation process. It achieves this through two key mechanisms: utilizing ensemble-based uncertainty estimation for robust filtering, and promoting a broader range of augmented preferences by extracting feedback from diverse, bucket-categorized trajectory clusters. These strategies empower CUDA to generate augmented

feedback with high diversity and accuracy. Our extensive evaluations on MetaWorld and DMControl offline datasets clearly demonstrate CUDA's superior performance, outperforming both conventional offline PbRL algorithms and existing augmentation-based methods across a wide array of scenarios. This work highlights the critical importance of diversity alongside accuracy in feedback augmentation for advancing PbRL capabilities. In addition, CUDA is compatible with prior PbRL approaches and exhibits performance gains when used in conjunction with augmentation methods, indicating the presence of a synergistic effect.

Broader Impact Statement

CUDA can improve the practicality of preference-based reinforcement learning by augmenting costly human preference data using both diversity (sampling pairs across predicted-reward buckets to avoid focusing only on extreme comparisons) and uncertainty (ensemble-based filtering to keep more reliable synthetic comparisons), potentially enabling more robust learning in domains like robotics and control with fewer labels. However, it may also amplify biases present in the initial human feedback, encourage over-reliance on automatically augmented comparisons that can reinforce mistakes under distribution shift. Responsible use should therefore include clear documentation of feedback sources and biases, targeted human re-checks of high-impact or borderline cases, strong privacy/consent practices when user data is involved, and efforts to reduce compute overhead.

Acknowledgement

This work was partly supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2022-II220311 Development of Goal-Oriented Reinforcement Learning Techniques for Contact-Rich Robotic Manipulation of Everyday Objects and Institute for Information & communications Technology Technology Planning & Evaluation(IITP)grant funded by the Korea government(MSIT)(RS-2019-II190075, Artificial Intelligence Graduate School Support Program(KAIST).

References

Gaon An, Junhyeok Lee, Xingdong Zuo, Norio Kosaka, Kyung-Min Kim, and Hyun Oh Song. Direct preference-based policy optimization without reward modeling. *Advances in Neural Information Processing Systems*, 36:70247–70266, 2023.

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

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

Heewoong Choi, Sangwon Jung, Hongjoon Ahn, and Taesup Moon. Listwise reward estimation for offline preference-based reinforcement learning. *arXiv preprint arXiv:2408.04190*, 2024.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Joey Hejna and Dorsa Sadigh. Inverse preference learning: Preference-based rl without a reward function. *Advances in Neural Information Processing Systems*, 36, 2024.

Minyoung Hwang, Gunmin Lee, Hogun Kee, Chan Woo Kim, Kyungjae Lee, and Songhwai Oh. Sequential preference ranking for efficient reinforcement learning from human feedback. *Advances in Neural Information Processing Systems*, 36:49088–49099, 2023.

Changyeon Kim, Jongjin Park, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. Preference transformer: Modeling human preferences using transformers for rl. *arXiv preprint arXiv:2303.00957*, 2023.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop*, 2013.

Kimin Lee, Laura Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021.

Mengdi Li, Jiaye Lin, Xufeng Zhao, Wenhao Lu, Peilin Zhao, Stefan Wermter, and Di Wang. Curriculum-rlaif: Curriculum alignment with reinforcement learning from ai feedback. *arXiv preprint arXiv:2505.20075*, 2025.

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. *arXiv preprint arXiv:2203.10050*, 2022.

Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

Daniel Shin, Anca D Dragan, and Daniel S Brown. Benchmarks and algorithms for offline preference-based reward learning. *arXiv preprint arXiv:2301.01392*, 2023.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. Rlcd: Reinforcement learning from contrastive distillation for language model alignment. *arXiv preprint arXiv:2307.12950*, 2023.

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 *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

A Experimental Details

A.1 Dataset

In this study, we utilized the dataset originally collected in the LiRE(Choi et al., 2024). The dataset was generated by collecting replay buffers during the training of online reinforcement learning agents. Specifically, the online Soft Actor-Critic (SAC) algorithm (Haarnoja et al., 2018), implemented in PEBBLE(Lee et al., 2021), using ground-truth rewards.

Replay buffers are collected by evaluating the online policy's success rate every 50,000 steps. We use data gathered until the policy's success rate approaches 50%, same as the one used in the source paper. This approach ensures that the performance is compared fairly under the same conditions as described in the source paper.

A.2 Hyperparameters and Setup

Table 7: Hyperparameters of the reward model and the baselines.

Hyperparameter	**Value**
Reward Model	
Optimizer	Adam
Learning rate	1e-3
Batch size	32
Hidden layer dim	256
Hidden layers	2
Activation function	ReLU
Final activation	Tanh
Epochs	200
# of ensembles (labeled / augmented)	7 / 3
Ensemble aggregation Method	Average
IQL (Kostrikov et al., 2021)	
Optimizer	Adam
Critic, Actor, Value hidden dim	256
Critic, Actor, Value hidden layers	2
Critic, Actor, Value activation function	ReLU
learning rate	3e-4
Batch size	256
Discount factor γ	0.99
Soft Update Rate τ	0.05
Temperature β	3.0
Expectile	0.7

Setup We use a single NVIDIA RTX A5000 GPU and 96 CPU cores (Intel Xeon Gold 5220R @ 2.20GHz) in our experiments. The system runs on Ubuntu 22.04 with Linux kernel 6.5.0 and CUDA 11.8.

A.3 CUDA Algorithm Pseudocode

Algorithm 1 CUDA Algorithm

Require: Unlabeled dataset $\mathcal{D}_u$, Labeled feedback $\mathcal{F}_\ell$, Number of buckets K
Ensure: Feedback set to train policy $\mathcal{F}_{\text{refined}}$

1. Train an ensemble of reward models $\mathcal{R}_\theta = \{r_1, \ldots, r_N\}$ on labeled feedback $\mathcal{F}_\ell$

2. For each trajectory $\tau = \{(s_t, a_t)\}_{t=1}^{T}$ in $\mathcal{D}_u$, compute the ensemble-averaged reward at each timestep:

$$\bar{r}_t = \frac{1}{N} \sum_{n=1}^{N} r_n(s_t, a_t)$$

and the predicted return:

$$\hat{R}(\tau) = \sum_{t=1}^{T} \bar{r}_t$$

3. For each trajectory τ, compute an uncertainty score:

$$u(\tau) = \left(\sum_{t=1}^{T} \left(\frac{1}{N} \sum_{n=1}^{N} \left(r_n(s_t, a_t) - \bar{r}_t \right)^2 \right) \right)^{1/2}$$

4. Cluster the trajectories in $\mathcal{D}_u$ into K buckets $\{\mathcal{B}_1, \ldots, \mathcal{B}_K\}$ using k-means++ on the predicted return $\hat{R}(\tau)$.
The optimal number of clusters k^* is selected as:

$$k^* = \arg \max_{k \in [\frac{1}{2}K, K]} \texttt{silhouette\_score}\left(k, \mathcal{D}_u; \mathcal{R}_\theta\right)$$

5. Assign sampling weights $w_{ij} \propto |\mathcal{B}_i| \cdot |\mathcal{B}_j|$ for all unordered bucket pairs (i, j) with $i < j$, and normalize such that:

$$\sum_{i<j} w_{ij} \approx n$$

6. For each bucket pair (i, j) with $i < j$, sample trajectory pairs $(\tau_i, \tau_j) \in \mathcal{B}_i \times \mathcal{B}_j$ until w_{ij} valid feedback pairs are collected, subject to the confidence-separated preference condition:

$$\hat{R}(\tau_i) + z \cdot u(\tau_i) < \hat{R}(\tau_j) - z \cdot u(\tau_j)$$

Let $\mathcal{F}_{\text{aug}}$ be the set of all such trajectory pairs.

7. Combine the original labeled feedback with the augmented set:

$$\mathcal{F}_{\text{refined}} = \mathcal{F}_\ell \cup \mathcal{F}_{\text{aug}}$$

B Bucket-based sampling and uncertainty-based filtering

B.1 KL Divergence of Augmented Feedback from Random Sampling

Table 8: KL divergence of reward models under different feedback selection strategies across Metaworld tasks. **CS** = Confidence-based Sampling, **UF** = Uncertainty-based Filtering, **BS+UF (CUDA)** = Bucket-based Sampling combined with Uncertainty Filtering (our method).

Method	button-press -topdown	box-close	dial-turn	sweep	button-press -topdown-wall	sweep-into	drawer-open	lever-pull	avg
CS	0.4291	0.3166	0.1411	0.2902	0.3836	0.2686	0.2566	0.3122	**0.2997**
UF	0.0617	0.0287	0.0360	0.0354	0.0545	0.0261	0.0403	0.0486	**0.0414**
BS+CS	0.0302	0.0126	0.0141	0.0216	0.0273	0.0115	0.0232	0.0466	**0.0234**
BS+UF (CUDA)	0.0305	0.0124	0.0138	0.0248	0.0257	0.0109	0.0227	0.0477	**0.0236**

B.2 Performance without k-Means

Table 9: Performance comparison with and without k-Means bucketing across environments. Results are reported as mean ± standard deviation. The better performing condition per row is highlighted.

Environment	**Without k-Means**	**With k-Means**
button-press-topdown	0.6108 ± 0.1916	0.5885 ± 0.4432
box-close	0.6220 ± 0.2349	0.6396 ± 0.2297
dial-turn	0.8984 ± 0.0596	0.9028 ± 0.0616
sweep	0.8544 ± 0.0914	0.8856 ± 0.0744
button-press-topdown-wall	0.2544 ± 0.1406	0.2956 ± 0.1164
sweep-into	0.3488 ± 0.1018	0.3608 ± 0.0985
drawer-open	0.0536 ± 0.0584	0.0408 ± 0.0631
lever-pull	0.8608 ± 0.0731	0.8420 ± 0.0581
Average	0.5629 ± 0.3231	0.5635 ± 0.3193

B.3 Agreement Rate of Augmented Preference Pairs with True Rewards

Table 10: Pairwise preference prediction accuracy of reward models under different feedback selection strategies across Metaworld tasks. **CS** = Confidence-based Sampling, **UF** = Uncertainty-based Filtering, **BS+UF (CUDA)** = Bucket-based Sampling combined with Uncertainty Filtering (our method).

Method	button-press -topdown	box-close	dial-turn	sweep	button-press -topdown-wall	sweep-into	drawer-open	lever-pull	avg
CS	0.9990	0.9853	0.9621	0.9905	0.9967	0.9400	0.9951	0.9951	**0.9830**
UF	0.9809	0.9119	0.9000	0.9290	0.9742	0.8025	0.9564	0.9543	**0.9262**
BS+CS	0.9706	0.8883	0.8576	0.9205	0.9638	0.7510	0.9457	0.9520	**0.9062**
BS+UF (CUDA)	0.9696	0.8910	0.8575	0.9207	0.9627	0.7525	0.9453	0.9522	**0.9064**

B.4 True Reward Heatmap of Augmented Preference Pairs

Figure 6 shows the true reward heatmaps of the selected augmented preference pairs for each environment under CS and CUDA sampling strategies. The heatmaps visualize the relative sampling density of each trajectory pair compared to random sampling, with the color intensity indicating values roughly between 0.6 and 1.4.

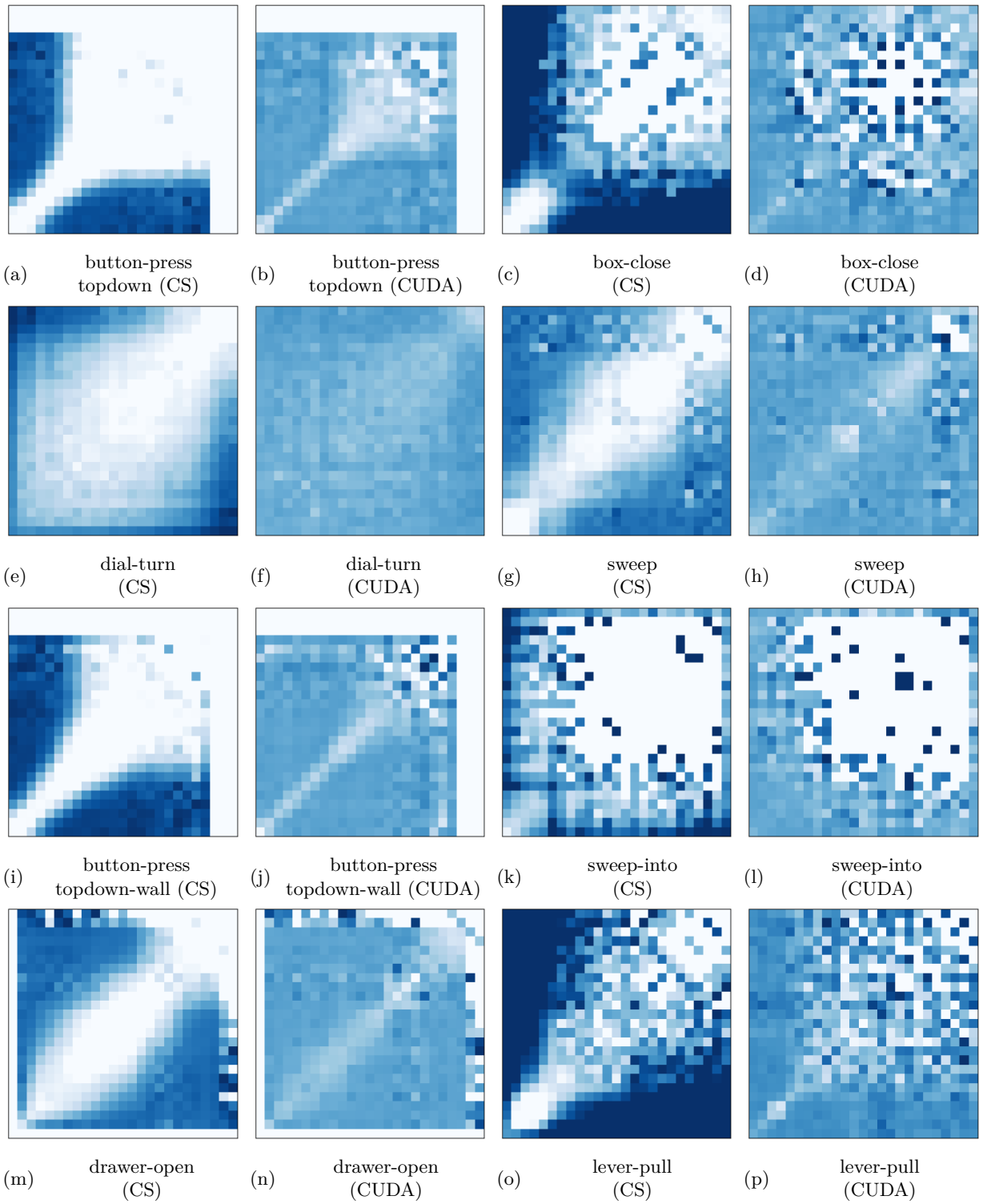
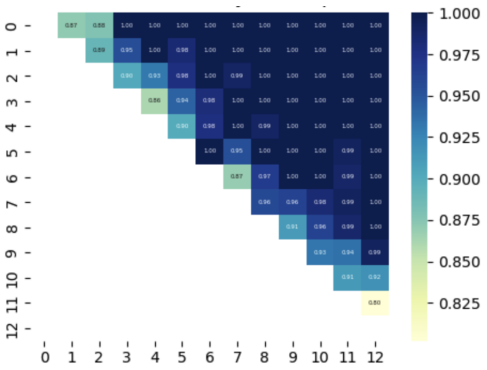


Figure 6: Heatmaps of true rewards for augmented preference pairs.

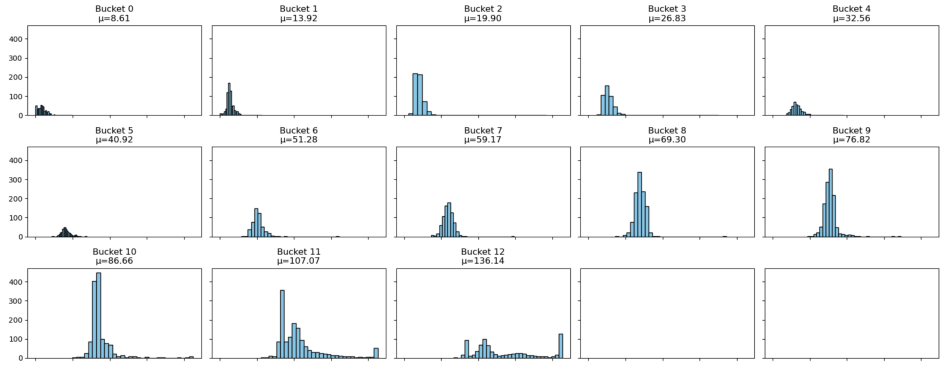
B.5 True Reward Distribution per Bucket

Figure 7 shows a detailed visualization of the bucket structure in the *button-press-topdown* environment.

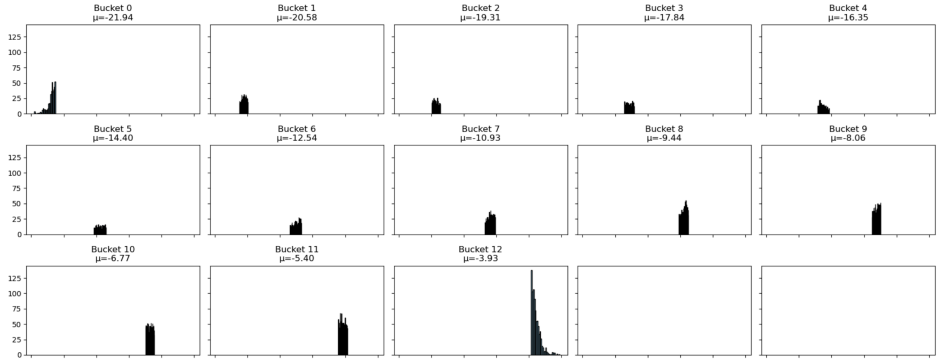
Subfigure (a) presents the pairwise preference prediction accuracy between buckets, with values closer to 1 indicating strong ordering agreement. Subfigures (b) and (c) show the distributions of true and predicted rewards in each bucket, respectively, highlighting the effectiveness of k-means-based bucketing in separating trajectories with different reward profiles.



(a) Inter bucket preference accuracy matrix



(b) True reward distribution per bucket

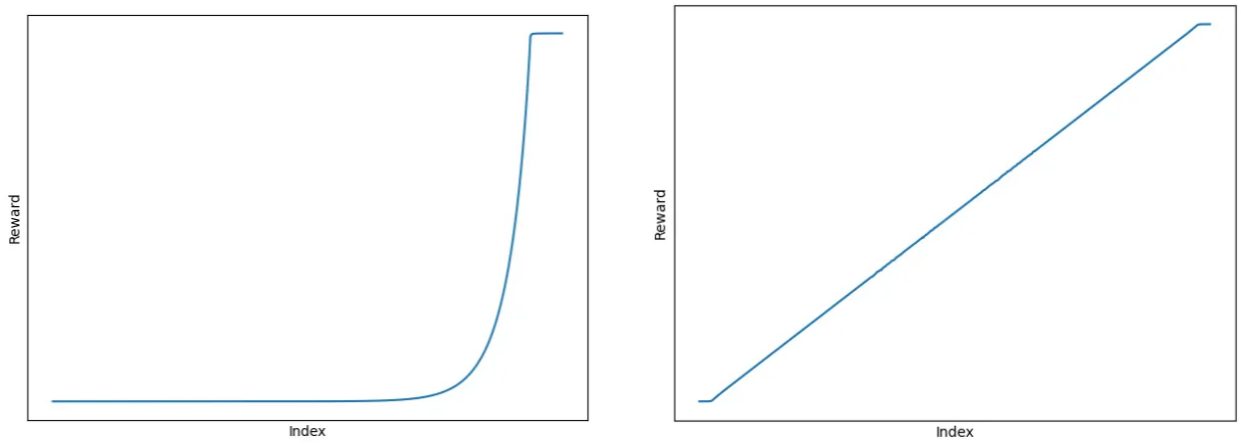


(c) Predicted reward distribution per bucket

Figure 7: Analysis of the kmeans bucket structure in the *button press topdown* environment.

C Comparison of Linear and Exponential Activations

In this section, we empirically compare the reward distributions learned under linear and exponential activation functions. To isolate the effect of the activation choice, we construct a controlled experiment with synthetic reward data. Specifically, we generate 1,000 distinct states, each assigned a ground-truth reward value, and define a binary label for every possible state pair indicating the correct ordering between the two rewards. The reward model is then trained to minimize the binary cross-entropy loss for each pairwise comparison. For this experiment, we also constrain the range of possible state rewards to match the overall reward scale observed in the main experiments. Specifically, the output range is set to $(0, 50)$ for the linear activation and $(-25, 25)$ for the exponential activation. This setup allows a direct comparison of how each activation maps the same preference structure into reward magnitudes.



(a) Linear activation ($\varphi(x) = x$)

(b) Exponential activation ($\varphi(x) = e^x$)

Figure 8: Comparison between linear and exponential activation functions in the reward model. The linear transformation produces a sharper separation between high- and low-reward trajectories, while the exponential transformation yields smoother and more evenly distributed rewards.

D Additional Sensitivity Analyses

In this appendix we provide additional sensitivity analyses that complement the main experiments. These analyses examine how CUDA behaves under three axes that are not directly evaluated in the main body: (i) label noise in the initial labeled feedback, (ii) the size of the initial labeled feedback combined with label corruption, and (iii) the ensemble size of the score function. All three analyses are conducted on a single representative environment (`box-close-v2` in MetaWorld) and are intended as diagnostic studies rather than full benchmark evaluations.

D.1 Sensitivity to Label Noise in the Initial Labeled Feedback

Because CUDA builds augmented comparisons on top of a score function learned from the original labeled feedback, noise in the initial labels can propagate through the entire augmentation pipeline. To examine how severe this effect is, we inject symmetric label noise into the 500 original labeled feedback pairs on `box-close-v2` by flipping each preference label independently with probability equal to the reported noise ratio, and compare CUDA against the non-augmented MR-linear baseline.

Table 11 reports the results. CUDA outperforms the non-augmented baseline across all noise levels. The value at noise ratio 0.15 is slightly higher than at 0.10; we attribute this non-monotonicity to variance across random seeds rather than a systematic trend, since the overall pattern decreases as noise grows. CUDA remains more robust than no augmentation across noise levels, but its absolute performance still degrades as noise increases.

Table 11: Policy performance on `box-close-v2` under different levels of symmetric label noise in the 500 original labeled feedback pairs. CUDA outperforms the non-augmented MR-linear baseline at every noise level.

Noise ratio	MR-linear	CUDA
0.00	0.44	0.64
0.05	0.26	0.48
0.10	0.17	0.31
0.15	0.18	0.45
0.20	0.00	0.26
0.25	0.00	0.11

D.2 Feedback Size and Reward Model Quality under Label Corruption

The sensitivity of CUDA to the *size* of the initial labeled feedback set is analyzed in the main body (Section 5.6.4). As a complementary diagnostic, we also measure how the quality of the learned reward function itself scales with feedback size and label corruption, independently of the augmentation step. Concretely, we vary the number of original labeled feedback pairs and, for each budget, inject symmetric label corruption at several rates, then report the Pearson correlation between the learned per-step reward and the ground-truth environment reward on `box-close-v2`.

Table 12 summarizes the results. As the number of labeled feedback pairs grows, the learned reward becomes both more accurate (higher correlation in the clean column) and less sensitive to corruption (the correlation degrades more gracefully as the corruption rate increases). This supports the general motivation behind CUDA: augmenting a limited labeled set with additional preference comparisons can improve reward-model robustness to labeling errors.

Table 12: Pearson correlation between the learned reward and the ground-truth environment reward on `box-close-v2`, as a function of the number of original labeled feedback pairs (rows) and the label corruption rate (columns).

# of feedback	0.00	0.01	0.05	0.10	0.20
100	0.70	0.67	0.31	0.21	0.27
500	0.75	0.70	0.69	0.69	0.27
1000	0.86	0.87	0.86	0.69	0.11
2000	0.87	0.86	0.89	0.79	0.68
10000	0.95	0.95	0.92	0.90	0.86

D.3 Ensemble Size of the Score Function

CUDA uses an ensemble of score functions to estimate trajectory-level uncertainty. Since the ensemble size N affects both the uncertainty estimate and the computational cost of the augmentation stage, we examine how policy performance changes as we vary N. In our setting the ensemble is trained on only 500 labeled feedback pairs by default, so moderately large ensembles incur only a modest additional training cost.

Table 13 reports policy performance on `box-close-v2` for $N \in \{3, 5, 7, 10\}$, with $N = 7$ corresponding to the default setting used throughout the paper. The best performance is obtained at $N = 7$, while both smaller and larger ensembles perform worse in this setting, indicating that simply increasing the ensemble size does not monotonically improve performance.

Table 13: Effect of the ensemble size N of the score function on policy performance (`box-close-v2`). The default setting used in the main experiments is $N = 7$.

Ensemble size N	Policy performance
3	0.60
5	0.55
7 (ours)	0.64
10	0.60