CoAct: Co-Active LLM Preference Learning with Human-AI Synergy

Ruiyao Xu[⋆] **Mihir Parmar**[◇] **Tiankai Yang**[♣] **Zhengyu Hu**[♡]
Yue Zhao[♣] **Kaize Ding**[⋆]

[⋆]Northwestern University [◇]Google
[♣]University of Southern California [♡]University of Washington

Abstract

Learning from preference-based feedback has become an effective approach for aligning LLMs across diverse tasks. However, high-quality human-annotated preference data remains expensive and scarce. Existing methods address this challenge through either self-rewarding, which scales by using purely AI-generated labels but risks unreliability, or active learning, which ensures quality through oracle annotation but cannot fully leverage unlabeled data. In this paper, we present CoAct, a novel framework that synergistically combines self-rewarding and active learning through strategic human-AI collaboration. CoAct leverages self-consistency to identify both reliable self-labeled data and samples that are requiring oracle verification. Additionally, oracle feedback guides the model to generate new instructions within its solvable capability. Evaluated on three reasoning benchmarks across two model families, CoAct achieves average improvements of +13.25% on GSM8K, +8.19% on MATH, and +13.16% on WebInstruct, consistently outperforming all baselines. [1]

1 Introduction

Preference alignment has demonstrated remarkable performance across a wide array of tasks, including instruction following, question answering, math reasoning, and creative writing (Rafailov et al., 2024; Ouyang et al., 2022; Anthropic, 2022; Christiano et al., 2017; Hu et al., 2024b; Wang et al., 2025b). However, the *scarcity* of *high-quality* human-annotated pairwise preference data severely limits the effectiveness and scalability of preference learning methods (Luo et al., 2025; Yuan et al., 2024; Li et al., 2024b; Lee et al., 2024; Lei et al., 2026; Hu et al., 2026). Existing approaches (Huang et al., 2023a; Yuan et al., 2024; Shen et al., 2025; Das et al., 2025) tackle either *data scarcity* or *data quality* in isolation, each making distinct trade-offs.

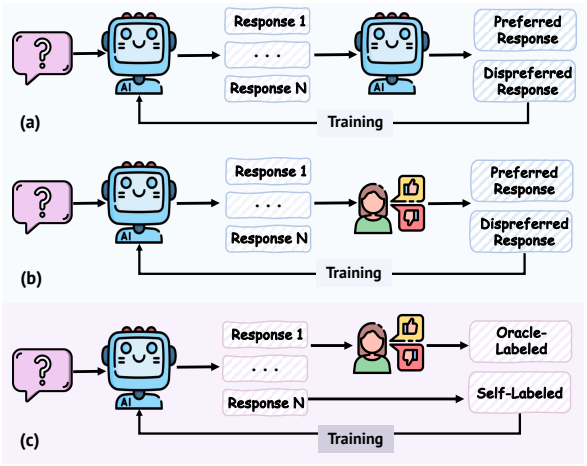


Figure 1: **(a)** Self-rewarding uses AI self-labeled data to construct preference pairs; **(b)** Active preference learning uses human annotation to ensure data quality; **(c)** Our framework CoAct combines both approaches through human-AI collaboration.

To address the challenge of *data scarcity* in preference learning, recent works explore "self-rewarding" pipelines that utilize LLMs themselves to generate new instructions, produce candidate responses, and construct preference pairs through self-evaluation (Huang et al., 2023a; Yuan et al., 2024; Chen et al., 2024; Rosset et al., 2024). By leveraging the model's own judgments to create training data, these self-rewarding frameworks dramatically reduce the need for human annotation or external reward models. While promising in their scalability, these approaches face a critical limitation: without external validation, they are prone to amplifying self-bias errors, where models reinforce their own errors and misconceptions through iterative self-training, potentially diverging from true human preferences (Laidlaw et al., 2025; Zhang et al., 2025; Shafayat et al., 2025; Ding et al., 2024; Huang et al., 2023b; Bansal and Sharma, 2023).

On the other hand, active preference learning has been explored to ensure *data quality* (Shen

[1]Our code is available at https://github.com/rux001/CoAct.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 16508–16525
July 2-7, 2026

et al., 2025; Das et al., 2025; Lin et al., 2026; Muldrew et al., 2024). These methods incorporate an iterative data acquisition and fine-tuning loop, where at each iteration, the most informative samples are strategically selected from an unlabeled pool for human annotation. However, active preference learning faces its own fundamental limitation: the constrained annotation budget prevents utilization of the vast amounts of remaining unlabeled data, and thus, potentially valuable samples are neglected. These two paradigms present a fundamental dilemma: *How can we achieve better data efficiency for LLM alignment by leveraging the synergy between human and AI?*

In this paper, we introduce CoAct, a novel framework that bridges self-rewarding and active learning for preference alignment through strategic human-AI synergy as shown in Figure 1. Our framework operates through an iterative process: at each round, we generate multiple responses for each unlabeled instruction and construct preference pairs through the idea based on self-consistency (Wang et al., 2023; Prasad et al., 2025; Jiao et al., 2025) – the most consistent response is considered as chosen and the least consistent one is used as rejected. Based on the consistency score of the chosen response, we partition samples into high-consistency and low-consistency subsets. For high-consistency samples, we further identify samples with potential self-consistent errors through k-NN distance metrics (Sun et al., 2022). Those samples will be routed for oracle labeling along with low-consistency subset. The remaining high-consistency samples are directly used as self-labeled training data. Remarkably, oracle feedback serves a dual purpose: ① providing reliable training signals through verified preference pairs, and ② guiding new instruction generation, where oracle-verified examples serve as in-context demonstrations to generate instructions within the model's solvable capability. Finally, both oracle-labeled and self-labeled preference pairs are combined to update the model using a modified DPO objective that incorporates both the DPO loss and an NLL regularization term (Pang et al., 2024).

In our experiments, we evaluate CoAct using two model families (`Llama3-8B` and `Qwen3-4B`) across three reasoning benchmarks (`GSM8K`, `MATH`, `WebInstruct`). Experimental results demonstrate that CoAct achieves substantial performance improvements over baseline methods. Specifically, CoAct achieves average gains of +13.25% on `GSM8K`, +8.19% on `MATH`, and +13.16% on `WebInstruct` across both models after four training iterations. Notably, CoAct outperforms the strongest baseline by 4-8 percentage points at the final iteration. Beyond in-domain performance, CoAct demonstrates strong generalization to out-of-domain benchmarks, consistently achieving the best performance on `GPQA` and `MMLU-Pro`. Moreover, we analyze the effectiveness of self-consistency for preference construction, observing strong Pearson correlations with accuracy.

2 Related Work

LLM Preference Alignment. Preference alignment aims to align LLMs with human preferences across dimensions including safety, helpfulness, factuality, reasoning, and scientific discovery (Askell et al., 2021; Ouyang et al., 2022; Hu et al., 2025a; Wang et al., 2025c, 2026). Reinforcement Learning from Human Feedback (RLHF) (Leike et al., 2018; Stiennon et al., 2020) is a prevalent approach that trains a reward model from human preferences and uses reinforcement learning algorithms such as PPO to optimize the language model (Anthropic, 2022; Christiano et al., 2017). Direct Preference Optimization (DPO) (Rafailov et al., 2024) has emerged as a more efficient alternative, eliminating the explicit reward model by directly optimizing preference probabilities. Several extensions have since been proposed, including KTO (Ethayarajh et al., 2024), GPO (Zhao et al., 2024), ΨPO (Azar et al., 2024), and ODPO (Amini et al., 2024).

However, high-quality annotated preference data remains limited and expensive to obtain. Recent work leverages LLMs themselves to generate or verify preference data, commonly referred to as RLAIF or self-rewarding (Lee et al., 2024; Yuan et al., 2024; Chen et al., 2024; Prasad et al., 2025; Shafayat et al., 2025; Wu et al., 2025; Liu et al., 2025b; Hu et al., 2025b). For instance, Yuan et al. (2024) propose self-rewarding language models, where the model acts as its own judge via LLM-as-a-Judge prompting to evaluate self-generated responses and iteratively improve itself. While this approach improves scalability, it introduces the risk of self-bias (Bansal and Sharma, 2023; Wang et al., 2024a, 2023; Xu and Ding, 2026). Prior works have explored human-AI collaboration for data generation in traditional NLP (Bartolo et al., 2022; Liu et al., 2022; Wang et al., 2024a). Recent

work (Liu et al., 2025a) explores human-AI collaboration for generating preference data for reward model training. However, their approach relies on a group of strong LLMs to aggregate preference judgments, which remains resource-intensive.

Active LLM Alignment. Active learning for LLM alignment seeks to maximize alignment quality while minimizing human annotation costs by strategically selecting queries for labeling. Early work on active preference learning focused on applications in robotics and autonomous systems (Sadigh et al., 2017; Biyik and Sadigh, 2018; Wang et al., 2025a). In the context of LLMs, recent approaches can be broadly categorized into heuristic methods and theoretically grounded frameworks. Heuristic methods leverage uncertainty-based metrics (Muldrew et al., 2024; Melo et al., 2024; Gleave and Irving, 2022; Hu et al., 2024a) or margin-based selection criteria (Muldrew et al., 2024) to identify high-value samples. From a theoretical perspective, Das et al. (2025) introduce Active Preference Optimization (APO), which frames active selection as a contextual dueling bandit problem and proves near-optimal sample complexity bounds. Similarly, Mehta et al. (2023) study dueling bandits for preference learning with theoretical guarantees. In contrast, Lin et al. (2026) proposes a selection criterion for non-linear reward functions that directly leverages the LLM itself to parameterize the reward model used for active data selection. Nevertheless, their approach has two key limitations: ① the selection criterion may be less effective for complex reasoning tasks, and ② they do not fully exploit the unlabeled data pool.

3 Preliminary

In this section, we establish the notation and formalize the problem setting for active preference learning:

Definition 1 (Active Preference Learning)**.** *Let $\mathcal{D}_U = \{x_j\}_{j=1}^{N_U}$ denote an unlabeled instruction pool and $\mathcal{D}_L = \{(x_i, y_i^+, y_i^-)\}_{i=1}^{N_L}$ an initial set of labeled preference pairs with $y_i^+ \succ y_i^-$. An Active Preference Learning algorithm proceeds in iterations: at each step t, given a batch $\mathcal{B}_t \subset \mathcal{D}_U$ of instructions, generates multiple candidate responses per instruction using the current model θ_t, select the top M pairs for oracle preference labeling to obtain $\mathcal{D}_{oracle}^{(t)}$ based on an acquisition function. The labeled set is augmented as $\mathcal{D}_L \leftarrow \mathcal{D}_L \cup \mathcal{D}_{oracle}^{(t)}$, and model parameters are updated via a preference-learning objective.*

We now formalize *Co-Active Preference Learning*, which integrates human and AI supervision:

Definition 2 (Co-Active Preference Learning)**.** *Building upon APL, Co-Active Preference Learning augments the active learning loop with self-generated supervision. At each iteration t, given batch $\mathcal{B}_t \subset \mathcal{D}_U$, the algorithm constructs self-labeled preference pairs $\mathcal{D}_{AI}^{(t)} = \{(x, \tilde{y}^+, \tilde{y}^-) \mid x \in \mathcal{B}_t\}$ alongside selecting M pairs for oracle labeling to obtain $\mathcal{D}_{oracle}^{(t)}$. The labeled set is augmented as $\mathcal{D}_L \leftarrow \mathcal{D}_L \cup \mathcal{D}_{oracle}^{(t)} \cup \mathcal{D}_{AI}^{(t)}$, and model parameters are updated via a preference-learning objective.*

4 CoAct: Human-AI Co-Active Preference Learning

Self-Consistency Preference Construction. For each instruction x in the current batch $\mathcal{B}_t$, we use temperature-based sampling with the current model θ_t to generate k diverse responses:

$$y_x = \{y_1, y_2, \ldots, y_k\} \sim \theta_t(\cdot|x) \quad (1)$$

where each response y_i includes both the reasoning process and the final answer. This diverse sampling enables us to capture the model's uncertainty (He et al., 2025) and reasoning variations for the same instruction. Previous work has demonstrated the promising ability of self-consistency properties of LLMs to improve answer accuracy (Wang et al., 2023; Huang et al., 2023a; Yang et al., 2024; Prasad et al., 2025; Xu et al., 2026). The core intuition behind this approach is that while LLMs may produce individual incorrect responses, it is significantly more difficult for them to consistently generate the same erroneous answer across multiple independent sampling attempts.

Therefore, we employ a consistency function $C(\cdot)$ to measure response consistency and then further construct preference pairs based on self-consistency. The consistency function extracts the final answer from each response $y \in y_x$ via $\text{ans}(\cdot)$ and computes the relative frequency:

$$C(y) = \frac{1}{k} \sum_{m=1}^{k} \mathbf{1}\{\text{ans}(y_m) = \text{ans}(y)\} \quad (2)$$

Using this consistency function, we create initial preference pairs $\mathcal{D}_{self}^{(t)}$ for the current batch $\mathcal{B}_t$ by selecting responses with extreme consistency scores. Specifically, we identify the most consistent response as the chosen response

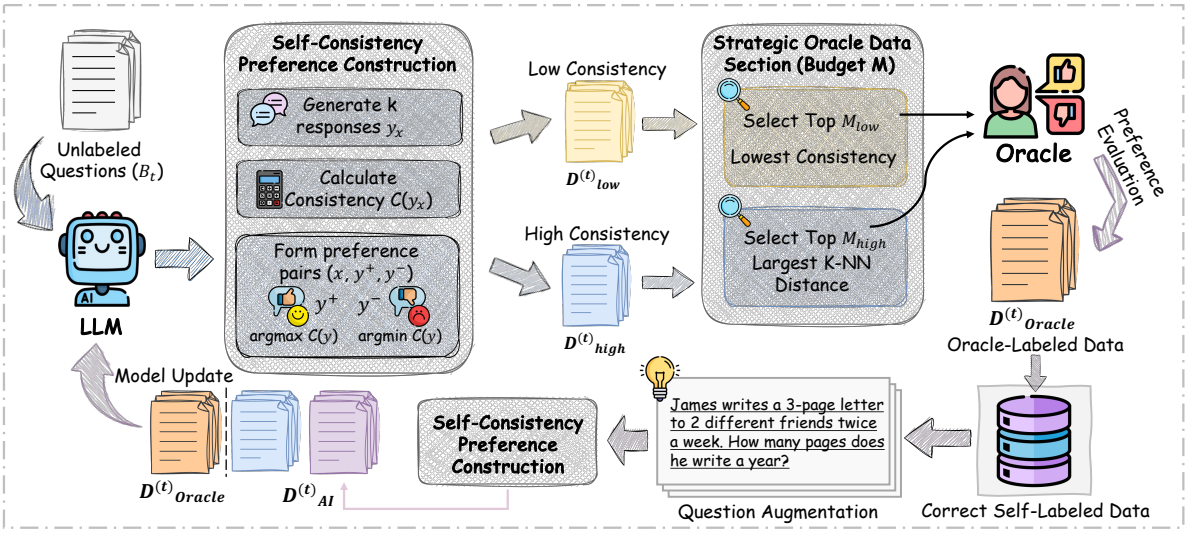


Figure 2: Overview of the CoAct framework. CoAct combines three key components: ① self-consistency-based preference construction, ② strategic oracle annotation selection, and ③ oracle-guided instruction augmentation to generate new training data within the model's capability.

$y^+ = \arg\max_{y \in y_x} C(y)$ and the least consistent response as the rejected response $y^- = \arg\min_{y \in y_x} C(y)$. The preference pairs are then constructed as:

$$\mathcal{D}^{(t)}_{self} = \{(x, y^+, y^-) \mid x \in \mathcal{B}_t\} \quad (3)$$

Based on the consistency score of the chosen response, we further partition the preference pairs into two subsets:

$$\mathcal{D}^{(t)}_{high} = \{(x, y^+, y^-) \in \mathcal{D}^{(t)}_{self} : C(y^+) \geq \tau\}$$
$$\mathcal{D}^{(t)}_{low} = \{(x, y^+, y^-) \in \mathcal{D}^{(t)}_{self} : C(y^+) < \tau\} \quad (4)$$

where τ is a consistency threshold that separates high-confidence self-labels from uncertain cases.

Data Selection for Oracle Annotation. In traditional active learning pipelines, the goal is to select the most informative samples for annotation under strict budget constraints (Li et al., 2024a; Shen et al., 2025; Lin et al., 2026). However, this approach faces a fundamental limitation: the constrained labeling budget prevents the utilization of potentially valuable information contained in the remaining unlabeled data. This raises a critical question: *Given limited annotation budgets, can we leverage both oracle-labeled data and LLM self-labeled data together to synergistically boost model performance?*

Oracle Feedback Protocol

Preference Evaluation: Verify whether $y^+ \succ y^-$ is correct; flip to $y^- \succ y^+$ if incorrect

Given an oracle labeling budget of M samples for iteration t, we strategically allocate this budget between low-consistency and high-consistency subsets to maximize information gain. Our approach partitions the budget as $M = M_{low} + M_{high}$, where M_{low} and M_{high} represent the number of samples selected from $\mathcal{D}^{(t)}_{low}$ and $\mathcal{D}^{(t)}_{high}$, respectively.

- *Low-Consistency Selection for Oracle Labeling:* For the low-consistency subset, we select the top M_{low} samples with the lowest consistency scores for their chosen responses, as these represent the most uncertain cases:

$$\mathcal{S}^{(t)}_{low} = \text{TopK}_{\text{lowest}}(\mathcal{D}^{(t)}_{low}, M_{low}, C(y^+)) \quad (5)$$

  where $\text{TopK}_{\text{lowest}}$ selects the M_{low} samples with the smallest consistency scores $C(y^+)$.

- *High-Consistency Selection for Oracle Labeling:* While high consistency typically indicates higher reliability, LLMs may still generate self-consistent but incorrect responses. These errors can be especially harmful because they reinforce flawed reasoning patterns (Tan et al., 2025; Duan et al., 2024). To identify such errors in the high-consistency subset $\mathcal{D}^{(t)}_{high}$, we adopt a non-parametric k-nearest neighbors (k-NN) approach inspired by research in out-of-distribution (OOD) detection (Sun et al., 2022; Xu and Ding, 2025; Yang et al., 2025b, 2022).

  We hypothesize that LLMs generate high-consistency errors on instructions deviating from correctly solved problems. Using oracle-verified correct preferences from previous iterations as

in-distribution (ID) data: $\mathcal{D}_{\text{ID}}^{(t)} = \{(x, y^+, y^-) \in \mathcal{S}_{\text{oracle}}^{(t-1)} : y^+ \succ y^- \text{ correct}\}$. For each $x_i \in \mathcal{D}_{\text{high}}^{(t)}$, we extract normalized penultimate hidden states: $z_i = \phi(x_i)/\|\phi(x_i)\|_2$, where ϕ is the model's feature encoder. We compute the k-NN distance: $r_k(z_i) = \min_{z \in \mathcal{Z}_{\text{ID}}^{(t)}} \|z_i - z\|_2$ where $\mathcal{Z}_{\text{ID}}^{(t)} = \{\phi(x)/\|\phi(x)\|_2 : (x, y^+, y^-) \in \mathcal{D}_{\text{ID}}^{(t)}\}$. Larger k-NN distances indicate likely OOD samples with self-consistent errors. We select:

$$\mathcal{S}_{\text{high}}^{(t)} = \text{TopK}_{\text{largest}}(\mathcal{D}_{\text{high}}^{(t)}, M_{\text{high}}, r_k(z_i)) \quad (6)$$

The final oracle evaluation set $\mathcal{D}_{oracle}^{(t)} = \mathcal{S}_{low}^{(t)} \cup \mathcal{S}_{high}^{(t)}$ undergoes human assessment to obtain preference labels.

Question Augmentation with Oracle Feedback. Prior work has demonstrated that expanding question diversity to cover a broader range of unseen scenarios effectively improves performance (Yu et al., 2024; Prasad et al., 2025). To this end, we exploit oracle annotation for a dual purpose: it provides gold labels for samples where the LLM is most uncertain and reveals cases where the model can provide accurate responses. We leverage these oracle-verified examples as in-context demonstrations to guide the LLM in generating new, diverse instructions that remain within its solvable capability. Specifically, we extract the subset of high-consistency samples with correct preferences:

$$\mathcal{D}_{\text{correct}}^{(t)} = \{(x, y^+, y^-) \in \mathcal{S}_{\text{high}}^{(t)} : y^+ \succ y^- \text{ is correct}\} \quad (7)$$

We randomly sample n instruction examples from $\mathcal{D}_{\text{correct}}^{(t)}$ and prompt the model to generate new instructions:

$$\mathcal{D}_{\text{new}}^{(t)} = \{x_i' \sim \theta_t(\cdot \mid \text{ICL}(\mathcal{D}_{\text{correct}}^{(t)}, n))\}_{i=1}^{N_{\text{new}}} \quad (8)$$

where $\text{ICL}(\mathcal{D}_{\text{correct}}^{(t)}, n)$ constructs an in-context learning prompt from n sampled instructions, generating N_{new} new instructions $\{x_i'\}_{i=1}^{N_{\text{new}}}$. For each newly generated instruction $x_i' \in \mathcal{D}_{\text{new}}^{(t)}$, we apply the same self-consistency preference construction procedure to generate k responses and construct preference pairs $(x_i', y_i'^+, y_i'^-)$. We filter these pairs by the consistency threshold and combine them with the original high-consistency self-labeled pairs to form the final AI-labeled dataset:

$$\mathcal{D}_{\text{AI}}^{(t)} = (\mathcal{D}_{\text{high}}^{(t)} \setminus \mathcal{S}_{\text{high}}^{(t)}) \cup \{(x_i', y_i'^+, y_i'^-) : x_i' \in \mathcal{D}_{\text{new}}^{(t)}, C(y_i'^+) \geq \tau\} \quad (9)$$

Model Update. We combine oracle-labeled and AI-labeled pairs to construct $\mathcal{D}_{\text{final}}^{(t)} = \mathcal{D}_{\text{oracle}}^{(t)} \cup \mathcal{D}_{\text{AI}}^{(t)}$. We update θ_t using a modified DPO objective (Pang et al., 2024):

$$\mathcal{L}(\theta_t) = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}_{\text{final}}^{(t)}} \Bigg[\log \sigma \Big(\beta \log \frac{\theta_t(y^+|x)}{\theta_0(y^+|x)} - \beta \log \frac{\theta_t(y^-|x)}{\theta_0(y^-|x)} \Big) - \alpha |y^+| \log \theta_t(y^+|x) \Bigg] \quad (10)$$

where $\beta > 0$ controls preference learning strength, $\alpha \geq 0$ regulates the likelihood term, and $|y^+|$ weights by response length. The updated model θ_{t+1} is then used to generate responses for the next iteration $(t+1)$, enabling iterative improvement where each round builds upon the refined capabilities of the previous model. This AI-human supervision strategy is theoretically grounded:

When Does Mixed Supervision Help?

Theorem (see §D for proof): Let $\mathcal{D}_{\text{oracle}}$ be clean with size N_o and $\mathcal{D}_{\text{AI}}$ have size N_{ai} with symmetric noise $\epsilon_{ai} < \frac{1}{2}$. Let $\text{Gap}(\pi) = V^*(\pi^*) - V^*(\pi)$ denote the policy sub-optimality. Then

$$\frac{\text{Gap}(\pi_{\text{oracle}})}{\text{Gap}(\pi_{\text{mix}})} \geq \sqrt{1 + \frac{N_{ai}(1-2\epsilon_{ai})^2}{N_o}}.$$

5 Experiments

5.1 Experimental Setup

Datasets. To evaluate the effectiveness of our proposed CoAct, we utilize three commonly used reasoning benchmarks for our main experiment: `GSM8K` (Cobbe et al., 2021) for grade school mathematical reasoning, `MATH` (Hendrycks et al., 2021) for advanced competition-level mathematics, and `WebInstruct` (Ma et al., 2025) for physics reasoning. To assess generalization to out-of-domain data, we further evaluate on `AIME` (Veeraboina, 2023) for advanced mathematical problem-solving, `GPQA` (Rein et al., 2024) for graduate-level science questions, and `MMLU-Pro` (Wang et al., 2024b) for multi-domain knowledge. Detailed dataset statistics are provided in Appendix A.

Baselines. We compare our approach against several active learning methods for preference alignment, focusing on different data selection strategies for oracle annotation while maintaining consistent training procedures across all methods: (1) *Random* randomly selects preference pairs without informativeness criteria; (2) *Entropy* (Muldrew et al., 2024) selects samples with highest prediction

Table 1: Performance across active learning iterations on reasoning benchmarks with different base models. Results show accuracy (%). Numbers with arrows indicate improvement (↑) or decline (↓) over base model. We highlight the best and second best results per iteration.

Base Model	Dataset	Methods	Random	Entropy	Pref Certainty	Pref + Ent	CoAct
Llama3 8B	GSM8K	Base Model	23.53	23.53	23.53	23.53	23.53
		+ Iteration 1	25.34 ↑1.81	26.43 ↑2.90	27.15 ↑3.62	28.92 ↑5.39	31.95 ↑8.42
		+ Iteration 2	28.05 ↑4.52	30.41 ↑6.88	32.48 ↑8.95	31.76 ↑8.23	37.56 ↑14.03
		+ Iteration 3	31.31 ↑7.78	33.21 ↑9.68	34.62 ↑11.09	35.89 ↑12.36	40.63 ↑17.10
		+ Iteration 4	34.57 ↑11.04	36.56 ↑13.03	37.28 ↑13.75	39.41 ↑15.88	43.58 ↑20.05
	MATH	Base Model	4.62	4.62	4.62	4.62	4.62
		+ Iteration 1	11.44 ↑6.82	11.24 ↑6.62	11.02 ↑6.40	11.34 ↑6.72	8.46 ↑3.84
		+ Iteration 2	11.78 ↑7.16	11.76 ↑7.14	10.84 ↑6.22	11.45 ↑6.83	10.88 ↑6.26
		+ Iteration 3	11.89 ↑7.27	12.01 ↑7.39	11.72 ↑7.10	11.94 ↑7.32	13.21 ↑8.59
		+ Iteration 4	12.07 ↑7.45	12.94 ↑8.32	11.08 ↑6.46	13.07 ↑8.45	14.46 ↑9.84
	WebInstruct	Base Model	7.69	7.69	7.69	7.69	7.69
		+ Iteration 1	7.05 ↓0.64	7.51 ↓0.18	9.23 ↑1.54	10.26 ↑2.57	11.54 ↑3.85
		+ Iteration 2	4.69 ↓3.00	7.93 ↑0.24	13.46 ↑5.77	12.18 ↑4.49	15.38 ↑7.69
		+ Iteration 3	5.13 ↓2.56	8.82 ↑1.13	11.97 ↑4.28	13.08 ↑5.39	13.82 ↑6.13
		+ Iteration 4	9.62 ↑1.93	10.11 ↑2.42	14.53 ↑6.84	14.87 ↑7.18	15.97 ↑8.28
Qwen3 4B	GSM8K	Base Model	88.39	88.39	88.39	88.39	88.39
		+ Iteration 1	92.35 ↑3.96	93.51 ↑5.12	94.12 ↑5.73	92.67 ↑4.28	93.44 ↑5.05
		+ Iteration 2	93.48 ↑5.09	93.08 ↑4.69	94.58 ↑6.19	93.21 ↑4.82	94.75 ↑6.36
		+ Iteration 3	94.14 ↑5.75	93.67 ↑5.28	94.93 ↑6.54	94.31 ↑5.92	95.02 ↑6.63
		+ Iteration 4	93.57 ↑5.18	94.02 ↑5.63	94.03 ↑5.64	94.58 ↑6.19	94.84 ↑6.45
	MATH	Base Model	69.17	69.17	69.17	69.17	69.17
		+ Iteration 1	68.78 ↓0.39	68.24 ↓0.93	68.46 ↓0.71	69.12 ↓0.05	73.91 ↑4.74
		+ Iteration 2	69.28 ↑0.11	68.94 ↓0.23	69.24 ↑0.07	69.87 ↑0.70	74.39 ↑5.22
		+ Iteration 3	69.43 ↑0.26	69.38 ↑0.21	69.01 ↓0.16	71.68 ↑2.51	75.64 ↑6.47
		+ Iteration 4	70.89 ↑1.72	70.14 ↑0.97	70.52 ↑1.35	70.21 ↑1.04	75.71 ↑6.54
	WebInstruct	Base Model	35.92	35.92	35.92	35.92	35.92
		+ Iteration 1	37.68 ↑1.76	39.14 ↑3.22	41.25 ↑5.33	42.87 ↑6.95	44.23 ↑8.31
		+ Iteration 2	40.15 ↑4.23	43.26 ↑7.34	47.92 ↑12.00	46.73 ↑10.81	50.38 ↑14.46
		+ Iteration 3	42.37 ↑6.45	45.89 ↑9.97	49.64 ↑13.72	50.21 ↑14.29	52.77 ↑16.85
		+ Iteration 4	44.12 ↑8.20	47.83 ↑11.91	51.25 ↑15.33	52.48 ↑16.56	53.96 ↑18.04

entropy to target model uncertainty; (3) *Pref Certainty* (Muldrew et al., 2024) prioritizes samples with low confidence in preference predictions; and (4) *Pref + Ent* (Muldrew et al., 2024) combines preference uncertainty with prediction entropy. All baselines follow identical training protocols using the same DPO objective and hyperparameters, differing only in their oracle annotation selection strategies. Detailed descriptions of each baseline method are summarized in Appendix B.

Implementation Details. We use `Llama3-8B` (Grattafiori et al., 2024) and `Qwen3-4B` (Yang et al., 2025a) as backbone models to demonstrate effectiveness across different model families and scales. For each iteration, we set the oracle budget to $M = 300$ and train all models using the modified DPO objective for 10 epochs with learning rate 5×10^{-6} and effective batch size 16. We set the DPO hyperparameter $\beta = 0.5$, NLL regularization coefficient $\alpha = 1$, and the number of sampled responses $k = 8$. When generating multiple responses and questions, we sample using temperatures from the set $\{0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7\}$ to encourage diverse reasoning paths. Additional details are in Appendix C.

5.2 Main Results

Table 1 presents the performance of CoAct compared to baseline active learning methods across three reasoning benchmarks and two model fam-

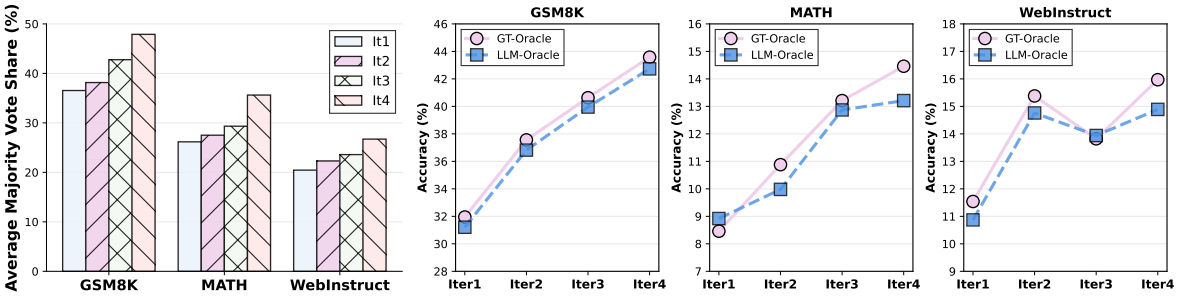


Figure 3: Left: Average majority vote share across iterations, showing the percentage of samples where the most frequent answer from k sampled responses matches the ground truth. Right: Comparison of GT-Oracle vs LLM-Oracle across iterations.

ilies. We make three key observations: **Observation ①: CoAct consistently outperforms baselines in later iterations.** Across both model families and all three datasets, CoAct achieves the best performance by iteration 4. On `Llama3-8B`, CoAct improves over the base model by 20.05% on `GSM8K`, 9.84% on `MATH`, and 8.28% on `WebInstruct`. However, we observe that on `MATH` with `Llama3-8B`, CoAct underperforms baselines in early iterations. This is likely due to the base model's low initial capability, causing generated responses to be too noisy for reliable self-labeling. **Observation ②: Performance variance decreases with stronger base models.** When the base model has strong initial capability, performance differences across methods become less pronounced. On `Qwen3-4B` with `GSM8K`, the performance spread at iteration 4 is only 1.27%. In contrast, on `Llama3-8B` with `GSM8K`, the spread is 9.01%. This suggests that strategic data selection matters more when the base model has substantial room for improvement, while stronger models benefit more uniformly from additional preference data regardless of selection strategy. **Observation ③: Strategic mixing of oracle and self-labeled data is crucial.** The variance in baseline performance highlights the importance of data utilization. Random sampling often shows the weakest performance, particularly on `WebInstruct` with `Llama3-8B` where it degrades performance.

5.3 More Analysis

To provide deeper insights into CoAct, we analyze ① self-consistency evolution across iterations, ② consistency-accuracy correlation, ③ oracle design choices, and ④ out-of-domain generalization, all using `Llama3-8B`.

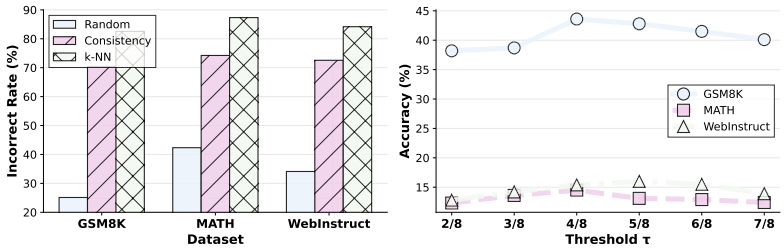
Self-Consistency Change Over Iterations. From Figure 3 (left), we observe that models become more consistent across iterations, with the average vote share $\mathcal{C}(y^+)$ increasing steadily on all datasets. Notably, we find that datasets where the model achieves higher performance also exhibit stronger self-consistency. This positive correlation between consistency and accuracy validates our approach of using self-consistency as a reliable proxy for response quality in constructing preference pairs.

Table 2: Pearson correlation between self-consistency $\mathcal{C}(y)$ and accuracy on test set across iterations.

Dataset	Iteration 1	Iteration 2	Iteration 3	Iteration 4
`GSM8K`	0.8654	0.8721	0.9582	0.9745
`MATH`	0.9135	0.9429	0.9601	0.9756
`WebInstruct`	0.7815	0.8623	0.9183	0.9544

Self-Consistency vs. Accuracy. After each iteration, we evaluate the trained model on the test set by generating k responses per instruction and constructing preference pairs using our self-consistency criterion. For each test instruction x, we compute the consistency score $C(y^+)$ of the chosen response and compare it against the ground-truth correctness. Table 2 reports the Pearson correlation between self-consistency scores and test accuracy across training iterations on `GSM8K`, `MATH`, and `WebInstruct`. Across all datasets, self-consistency exhibits a strong positive correlation with accuracy, and this correlation consistently strengthens over successive iterations.

Choice of Oracle. We compare two oracle scenarios to validate our design choices. In our main experiments, we employ GPT-5 as the oracle LLM and also provide ground-truth answers as references to judge preference pairs. We compare this approach with an alternative where GPT-5 serves as the oracle based purely on its own judgment. Figure 3 presents the performance comparison across `GSM8K`, `MATH`, and `WebInstruct` datasets over four training iterations. Across all three benchmarks, both oracle configurations yield comparative performance. These results indicate that powerful LLMs



Aug.	Self	GSM8K	MATH	Web.
○	○	36.38	11.95	13.53
✔	○	39.85	13.34	13.89
○	✔	41.23	12.12	15.21
✔	✔	**43.58**	**14.46**	**15.97**

Figure 4: Sensitivity analysis and ablation study. Left: Incorrect rate for different high-consistency selection methods. Middle: Performance across consistency threshold τ values. Right: Ablation study of key modules.

can serve as effective substitutes for humans, substantially reducing annotation costs while maintaining strong performance.

Out-of-Domain Generalization. We evaluate out-of-domain generalization on GPQA and MMLU-Pro using models trained on GSM8K, MATH, and WebInstruct. As shown in Figure 5, CoAct consistently outperforms all baselines across training datasets, demonstrating robust transfer to unseen domains. We hypothesize that oracle-guided instruction augmentation contributes to this improved generalization by diversifying the training distribution within the model's solvable range. A more comprehensive analysis is provided in Appendix E.

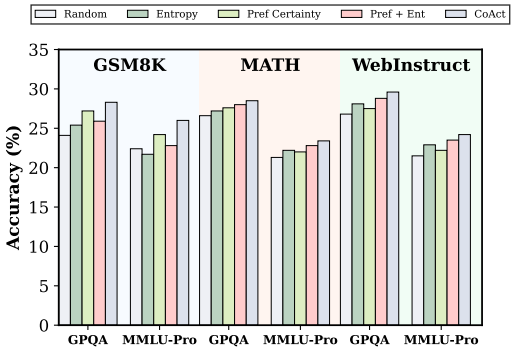


Figure 5: Out-of-domain generalization on GPQA and MMLU-Pro.

5.4 Ablation and Sensitivity Analysis

In this section, we conduct sensitivity analysis of key hyperparameters and evaluate the effectiveness of the key modules in CoAct.

Effectiveness of k-NN Selection. To validate the effectiveness of k-NN distance for detecting self-consistent errors in high-consistency samples, we compare different selection strategies. Figure 4 (left) shows the oracle incorrect rate for samples selected by each method at Iteration 4 on Llama3-8B. The k-NN distance approach consistently identifies samples with significantly higher error rates across all datasets, achieving 82.56% on GSM8K, 87.32% on MATH, and 84.17% on WebInstruct. In contrast, random selection yields substantially lower error rates, while selecting the lowest consistency samples in this subset results in 70.42%, 74.23%, and 72.56%. This demonstrates that k-NN distance successfully identifies OOD samples where the model generates self-consistent but incorrect responses.

Impact of Consistency Threshold. We analyze the sensitivity of CoAct to the consistency threshold τ that partitions samples into high and low-consistency subsets. Figure 4 (middle) shows performance across different threshold values at Iteration 4 on Llama3-8B. Performance peaks around $\tau = 4/8$ to $5/8$ across datasets, with degradation at both lower and higher values. When τ is too low, unreliable samples contaminate the self-labeled training data; when too high, the effective training set size remains low.

Ablation Study. To understand the contribution of each component in CoAct, we conduct an ablation study by removing key components and evaluating performance at the final iteration on Llama3-8B. Table 2 presents the results.Without both question augmentation (Aug.) and self-labeling, the framework relies solely on oracle-labeled data from active selection, achieving 36.38% on GSM8K, 11.95% on MATH, and 13.53% on WebInstruct. This oracle-only baseline demonstrates that limited annotation budget alone is insufficient to fully unlock model capabilities. Interestingly, on MATH, using question augmentation outperforms using self-labeling alone, suggesting that oracle-guided instruction generation within the model's solvable range is more effective than leveraging self-consistency on the original unlabeled instructions if the original data is too hard.

6 Conclusion

In this paper, we introduce CoAct, a framework that bridges self-rewarding and active preference learning through strategic human-AI collaboration.

Our approach addresses the fundamental dilemma in preference alignment: balancing the scalability of AI-generated preference data with the quality assurance of human annotations. Experimental results demonstrate that CoAct achieves demonstrative performance compared to existing active preference learning pipelines. These findings establish CoAct as a practical solution for preference alignment in resource-constrained settings.

Limitations

While CoAct demonstrates strong empirical performance, we acknowledge several limitations. First, generating multiple responses per instruction for self-consistency increases computational overhead. Second, our evaluation focuses on reasoning tasks with objective answers. Third, the consistency threshold requires dataset-specific tuning, though performance remains relatively stable across reasonable ranges. These limitations suggest promising directions for extending CoAct to broader domains and more efficient consistency estimation methods.

References

Afra Amini, Tim Vieira, and Ryan Cotterell. 2024. Direct preference optimization with an offset. *arXiv preprint arXiv:2402.10571*.

Anthropic. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. In *arXiv preprint arXiv:2204.05862*.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, and 1 others. 2021. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *AISTATS*.

Parikshit Bansal and Amit Sharma. 2023. Large language models as annotators: Enhancing generalization of nlp models at minimal cost. *arXiv preprint arXiv:2306.15766*.

Max Bartolo, Tristan Thrush, Sebastian Riedel, Pontus Stenetorp, Robin Jia, and Douwe Kiela. 2022. Models in the loop: Aiding crowdworkers with generative annotation assistants. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3754–3767, Seattle, United States. Association for Computational Linguistics.

Erdem Biyik and Dorsa Sadigh. 2018. Batch active preference-based learning of reward functions. In *CoRL*.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. In *ICML*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *NeurIPS*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Nirjhar Das, Souradip Chakraborty, Aldo Pacchiano, and Sayak Ray Chowdhury. 2025. Active preference optimization for sample efficient rlhf. In *ECML PKDD*.

Kaize Ding, Xiaoxiao Ma, Yixin Liu, and Shirui Pan. 2024. Divide and denoise: Empowering simple models for robust semi-supervised node classification against label noise. In *KDD*.

Jinhao Duan, Hao Cheng, Shiqi Wang, Alex Zavalny, Chenan Wang, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. 2024. Shifting attention to relevance: Towards the predictive uncertainty quantification of free-form large language models. In *ACL*.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. In *ICML*.

Adam Gleave and Geoffrey Irving. 2022. Uncertainty estimation for language reward models. *arXiv preprint arXiv:2203.07472*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Jianfeng He, Linlin Yu, Changbin Li, Runing Yang, Fanglan Chen, Kangshuo Li, Min Zhang, Shuo Lei, Xuchao Zhang, Mohammad Beigi, and 1 others. 2025. Survey of uncertainty estimation in llms-sources, methods, applications, and challenges. *Information Fusion*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Arjun Agarwal, Steven Levi, Kevin Ellis, and 1 others. 2021. Measuring mathematical problem solving with the math dataset. In *NeurIPS*.

Zhengyu Hu, Yichuan Li, Zhengyu Chen, Jingang Wang, Han Liu, Kyumin Lee, and Kaize Ding. 2024a. Let's ask gnn: Empowering large language model for graph in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1396–1409.

Zhengyu Hu, Jianxun Lian, Zheyuan Xiao, Max Xiong, Yuxuan Lei, Tianfu Wang, Kaize Ding, Ziang Xiao, Nicholas Jing Yuan, and Xing Xie. 2025a. Population-aligned persona generation for llm-based social simulation. *arXiv preprint arXiv:2509.10127*.

Zhengyu Hu, Jianxun Lian, Zheyuan Xiao, Seraphina Zhang, Tianfu Wang, Nicholas Jing Yuan, Xing Xie, and Hui Xiong. 2025b. Unveiling the learning mind of language models: A cognitive framework and empirical study. *arXiv preprint arXiv:2506.13464*.

Zhengyu Hu, Linxin Song, Jieyu Zhang, Zheyuan Xiao, Tianfu Wang, Zhengyu Chen, Nicholas Jing Yuan, Jianxun Lian, Kaize Ding, and Hui Xiong. 2024b. Explaining length bias in llm-based preference evaluations. *arXiv preprint arXiv:2407.01085*.

Zhengyu Hu, Jieyu Zhang, Zhihan Xiong, Alexander Ratner, Kaize Ding, and Ranjay Krishna. 2026. Towards acyclic preference evaluation of language models via multiple evaluators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 21903–21911.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023a. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023b. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Fangkai Jiao, Geyang Guo, Xingxing Zhang, Nancy F. Chen, Shafiq Joty, and Furu Wei. 2025. Preference optimization for reasoning with pseudo feedback. In *ICLR*.

Cassidy Laidlaw, Shivam Singhal, and Anca Dragan. 2025. Correlated proxies: A new definition and improved mitigation for reward hacking. In *ICLR*.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and 1 others. 2024. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. In *ICML*.

Yuxuan Lei, Tianfu Wang, Jianxun Lian, Zhengyu Hu, Defu Lian, and Xing Xie. 2026. Humanllm: Towards personalized understanding and simulation of human nature. *arXiv preprint arXiv:2601.15793*.

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. 2018. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*.

Dongyuan Li, Zhen Wang, Yankai Chen, Renhe Jiang, Weiping Ding, and Manabu Okumura. 2024a. A survey on deep active learning: Recent advances and new frontiers. *IEEE Transactions on Neural Networks and Learning Systems*.

Yichuan Li, Kaize Ding, Jianling Wang, and Kyumin Lee. 2024b. Empowering large language models for textual data augmentation. In *Findings of ACL*.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, and 1 others. 2022. Competition-level code generation with alphacode. *Science*.

Xiaoqiang Lin, Arun Verma, Zhongxiang Dai, Daniela Rus, See-Kiong Ng, and Bryan Kian Hsiang Low. 2026. Activedpo: Active direct preference optimization for sample-efficient alignment. In *ICLR*.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022. WANLI: Worker and AI collaboration for natural language inference dataset creation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6826–6847, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, and 1 others. 2025a. Skywork-reward-v2: Scaling preference data curation via human-ai synergy. *arXiv preprint arXiv:2507.01352*.

Yue Liu, Hongcheng Gao, Shengfang Zhai, Yufei He, Jun Xia, Zhengyu Hu, Yulin Chen, Xihong Yang, Jiaheng Zhang, Stan Z Li, and 1 others. 2025b. Guardreasoner: Towards reasoning-based llm safeguards. *arXiv preprint arXiv:2501.18492*.

Junyu Luo, Bohan Wu, Xiao Luo, Zhiping Xiao, Yiqiao Jin, Rong-Cheng Tu, Nan Yin, Yifan Wang, Jingyang Yuan, Wei Ju, and Ming Zhang. 2025. A survey on efficient large language model training: From data-centric perspectives. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30904–30920, Vienna, Austria. Association for Computational Linguistics.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. 2025. General-reasoner: Advancing llm reasoning across all domains. In *NeurIPS*.

Viraj Mehta, Vikramjeet Das, Ojash Neopane, Yijia Dai, Ilija Bogunovic, Jeff Schneider, and Willie Neiswanger. 2023. Sample efficient reinforcement learning from human feedback via active exploration. *arXiv preprint arXiv:2312.00267*.

Luckeciano C Melo, Panagiotis Tigas, Alessandro Abate, and Yarin Gal. 2024. Deep bayesian active learning for preference modeling in large language models. In *NeurIPS*.

William Muldrew, Peter Hayes, Mingtian Zhang, and David Barber. 2024. Active preference learning for large language models. In *ICML*.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. 2013. Learning with noisy labels. In *NeurIPS*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. In *NeurIPS*.

Archiki Prasad, Weizhe Yuan, Richard Yuanzhe Pang, Jing Xu, Maryam Fazel-Zarandi, Mohit Bansal, Sainbayar Sukhbaatar, Jason Weston, and Jane Yu. 2025. Self-consistency preference optimization. In *ICML*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. 2024. Direct nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*.

Dorsa Sadigh, Anca D. Dragan, S. Shankar Sastry, and Sanjit A. Seshia. 2017. Active preference-based learning of reward functions. In *Robotics: Science and Systems*.

Sheikh Shafayat, Fahim Tajwar, Ruslan Salakhutdinov, Jeff Schneider, and Andrea Zanette. 2025. Can large reasoning models self-train? *arXiv preprint arXiv:2505.21444*.

Yunyi Shen, Hao Sun, and Jean-Francois Ton. 2025. Active reward modeling: Adaptive preference labeling for large language model alignment. In *ICML*.

Freda Shi, Daniel Fried, Marjan Ghazvininejad, Luke Zettlemoyer, and Sida I. Wang. 2022. Natural language to code translation with execution. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3533–3546, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. In *NeurIPS*.

Yiyou Sun, Yifei Ming, Xiaojin Zhu, and Yixuan Li. 2022. Out-of-distribution detection with deep nearest neighbors. In *ICML*.

Hexiang Tan, Fei Sun, Sha Liu, Du Su, Qi Cao, Xin Chen, Jingang Wang, Xunliang Cai, Yuanzhuo Wang, Huawei Shen, and Xueqi Cheng. 2025. Too consistent to detect: A study of self-consistent errors in LLMs. In *EMNLP*.

Hemish Veeraboina. 2023. Aime problem set 1983-2024.

Tianfu Wang, Yi Zhan, Jianxun Lian, Zhengyu Hu, Nicholas Jing Yuan, Qi Zhang, Xing Xie, and Hui Xiong. 2025a. Llm-powered multi-agent framework for goal-oriented learning in intelligent tutoring system. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 510–519.

Xinru Wang, Hannah Kim, Sajjadur Rahman, Kushan Mitra, and Zhengjie Miao. 2024a. Human-llm collaborative annotation through effective verification of llm labels. In *CHI*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *ICLR*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *NeurIPS*.

Ziqing Wang, Yibo Wen, Abhishek Pandy, Han Liu, and Kaize Ding. 2026. Molmem: Memory-augmented agentic reinforcement learning for sample-efficient molecular optimization. *Preprint*, arXiv:2604.12237.

Ziqing Wang, Yibo Wen, William Pattie, Xiao Luo, Weimin Wu, Jerry Yao-Chieh Hu, Abhishek Pandey, Han Liu, and Kaize Ding. 2025b. Polo: Preference-guided multi-turn reinforcement learning for lead optimization. *arXiv preprint arXiv:2509.21737*.

Ziqing Wang, Kexin Zhang, Zihan Zhao, Yibo Wen, Abhishek Pandey, Han Liu, and Kaize Ding. 2025c. A survey of large language models for text-guided molecular discovery: from molecule generation to optimization. *arXiv preprint arXiv:2505.16094*.

Tianhao Wu, Weizhe Yuan, Olga Golovneva, Jing Xu, Yuandong Tian, Jiantao Jiao, Jason E Weston, and Sainbayar Sukhbaatar. 2025. Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge. In *EMNLP*.

Ruiyao Xu and Kaize Ding. 2025. Large language models for anomaly and out-of-distribution detection: A survey. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5992–6012, Albuquerque, New Mexico. Association for Computational Linguistics.

Ruiyao Xu and Kaize Ding. 2026. GNN-as-judge: Unleashing the power of LLMs for graph learning with GNN feedback. In *ICLR*.

Ruiyao Xu, Noelle I. Samia, and Han Liu. 2026. DS2-instruct: Domain-specific data synthesis for large language models instruction tuning. In *Findings of the Association for Computational Linguistics: EACL 2026*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Jingkang Yang, Pengyun Wang, Dejian Zou, Zitang Zhou, Kunyuan Ding, Wenxuan Peng, Haoqi Wang, Guangyao Chen, Bo Li, Yiyou Sun, and 1 others. 2022. Openood: Benchmarking generalized out-of-distribution detection. In *NeurIPS*.

Tiankai Yang, Yi Nian, Li Li, Ruiyao Xu, Yuangang Li, Jiaqi Li, Zhuo Xiao, Xiyang Hu, Ryan A. Rossi, Kaize Ding, Xia Hu, and Yue Zhao. 2025b. AD-LLM: Benchmarking large language models for anomaly detection. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 1524–1547, Vienna, Austria. Association for Computational Linguistics.

Yuqing Yang, Yan Ma, and Pengfei Liu. 2024. Weak-to-strong reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8350–8367, Miami, Florida, USA. Association for Computational Linguistics.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *ICLR*.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. 2024. Self-rewarding language models. In *ICML*.

Yanzhi Zhang, Zhaoxi Zhang, Haoxiang Guan, Yilin Cheng, Yitong Duan, Chen Wang, Yue Wang, Shuxin Zheng, and Jiyan He. 2025. No free lunch: Rethinking internal feedback for llm reasoning. *arXiv preprint arXiv:2506.17219*.

Siyan Zhao, John Dang, and Aditya Grover. 2024. Group preference optimization: Few-shot alignment of large language models. In *ICLR*.

A Datasets

We evaluate CoAct on three reasoning benchmarks covering different domains and difficulty levels:

- **GSM8K (Cobbe et al., 2021).** GSM8K (Grade School Math 8K) is a dataset of 8,500 grade school math word problems that require multi-step arithmetic reasoning. Each problem is accompanied by a natural language solution with intermediate reasoning steps and a final numerical answer. Following previous work (Prasad et al., 2025), we use a data split of 6.7K/0.8K/1.3K for train/dev/test sets respectively. We use the training set as our unlabeled instruction pool, the dev set for validation, and report final performance on the test set. For each iteration, we sample a batch of 1,675 instructions from the unlabeled pool. Performance is measured using exact match accuracy, where a response is considered correct only if the final numerical answer exactly matches the ground truth.
- **MATH (Hendrycks et al., 2021).** The MATH dataset contains 12,500 challenging competition-level mathematics problems from high school math competitions. Problems span various topics including algebra, counting and probability, geometry, intermediate algebra, number theory, prealgebra, and precalculus. Each problem includes a detailed solution with step-by-step reasoning. We hold out a portion of the training set to create a dev set for model selection and hyperparameter tuning, resulting in train/dev/test splits of 6.7K/0.8K/5K problems respectively. We use the training set as our unlabeled instruction pool and report exact match accuracy on the test set. For each iteration, we sample a batch of 1,675 instructions from the unlabeled pool.
- **WebInstruct (Ma et al., 2025).** WebInstruct is a reasoning dataset designed to evaluate models' ability to solve complex science problems across diverse domains. In this paper, we focus on the physics domain. We use the WebInstruct-verified dataset from TIGER-Lab and filter samples by answer type `float` to obtain numerical physics problems. This results in train/dev/test splits of 8K/1K/156 samples respectively. The training

set is further divided into 4 iterations of 2K samples each. For each iteration, we sample a batch of 500 instructions from the current iteration's pool. We report exact match accuracy on the test set.

Dataset Statistics. Table 3 summarizes key statistics for the three reasoning benchmarks used in our experiments. Table 4 reports the total training data size per iteration for both `Llama3-8B` and `Qwen3-4B`, which includes oracle-labeled samples, self-labeled samples, and augmented questions. Table 5 presents the agreement rate between GPT-5 judgments and the original preference labels using self-consistency construction. We observe consistently high agreement rates across all settings, with `Qwen3-4B` achieving higher agreement due to its stronger initial capabilities.

Table 3: Dataset statistics for the three reasoning benchmarks used in our experiments.

Dataset	Train	Dev	Test	Domain
GSM8K	6,700	800	1,319	Grade school math
MATH	6,700	800	5,000	Competition math
WebInstruct	8,000	1,000	156	Physics reasoning

Table 4: Total training data size per iteration for `Llama3-8B` and `Qwen3-4B`.

Dataset	Model	It 1	It 2	It 3	It 4
GSM8K	`Llama3-8B`	865	948	1065	1034
	`Qwen3-4B`	1792	1356	1495	1614
MATH	`Llama3-8B`	869	1124	915	971
	`Qwen3-4B`	1206	1032	1281	1130
WebInstruct	`Llama3-8B`	1421	1350	1382	1415
	`Qwen3-4B`	1655	1558	1617	1672

Table 5: Agreement (%) between GPT-5 judgments and preference labels in the constructed dataset across models, datasets, and iterations.

Dataset	Model	It 1	It 2	It 3	It 4
GSM8K	Llama3-8B	79.67	81.33	82.56	81.12
	Qwen3-4B	91.14	92.34	93.00	92.68
MATH	Llama3-8B	77.78	79.12	79.16	82.37
	Qwen3-4B	86.02	89.45	90.13	89.54
WebInstruct	Llama3-8B	86.43	87.24	87.78	90.18
	Qwen3-4B	93.36	94.45	94.35	94.22

B Baselines

We compare CoAct against four active learning baselines for preference alignment. All methods share the same iterative training framework and DPO objective, differing only in their data selection strategies for oracle annotation. At each iteration t, each method samples a batch $\mathcal{B}_t$ from the unlabeled pool, selects M samples from $\mathcal{B}_t$ for oracle labeling, and trains the model using the modified DPO objective.

- **Random.** This baseline randomly selects M samples from the unlabeled instruction pool.
- **Entropy (Muldrew et al., 2024).** This method leverages the predictive entropy of the language model as an uncertainty measure. For each instruction x in the batch $\mathcal{B}_t$, we sample k responses and approximate the predictive entropy as:

$$H_{p_\theta}(y|x) \approx -\frac{1}{k}\sum_{i=1}^{k} \log p_\theta(y_i|x), \quad y_i \sim p_\theta(\cdot|x) \tag{11}$$

  The method selects the top M samples with the highest entropy, prioritizing instructions where the model exhibits the greatest uncertainty.
- **Preference Certainty (Pref Certainty) (Muldrew et al., 2024).** This method focuses on the model's confidence in preference predictions under the Bradley-Terry model. For each instruction $x \in \mathcal{B}_t$, we sample two responses $y_1, y_2 \sim p_\theta(\cdot|x)$ and compute the difference in implicit rewards:

$$\begin{aligned} \text{certainty}(x) &= |\hat{r}(x, y_1) - \hat{r}(x, y_2)|, \\ \text{where } \hat{r}(x, y) &= \beta \log \frac{p_\theta(y|x)}{p_{\theta_0}(y|x)} \end{aligned} \tag{12}$$

  The method selects the top M samples with the lowest certainty scores, targeting cases where the model is most uncertain about preference ordering.
- **Preference + Entropy (Pref + Ent) (Muldrew et al., 2024).** This hybrid approach combines entropy and preference certainty to exploit their complementary strengths. The selection operates in two stages:
  1. **Entropy Filtering:** Rank all instructions in $\mathcal{B}_t$ by predictive entropy and select the top K samples with highest entropy, where $K > M$.
  2. **Preference Selection:** For the filtered K samples, generate response pairs and compute preference certainty scores. Select the top M samples with lowest certainty.

This two-stage design is based on the hypothesis that high-entropy instructions are more likely to yield uncertain preference predictions. We set $K = 2M$ following Muldrew et al. (2024).

C Implementation Details

C.1 Environment and Models

All experiments are conducted on 4 NVIDIA A100 GPUs with 80GB memory. We use `Llama3-8B-Base` and `Qwen3-4B` as backbone models to demonstrate effectiveness across different model families and scales. Unless otherwise specified, all experiments share the same hardware configuration and training pipeline.

C.2 Training Setup

Models are optimized using AdamW with a learning rate of 5×10^{-6} and weight decay 0.01. The effective batch size is 16. Training is performed for up to 10 epochs per iteration, and checkpoints are selected based on accuracy on the development set at the end of each epoch. All models are trained using the modified DPO objective, with DPO temperature $\beta = 0.5$ and NLL regularization coefficient $\alpha = 1.0$.

For active learning, we allocate an oracle budget of $M = 300$ preference pairs per iteration, evenly split between low-consistency and high-consistency subsets, with $M_{\text{low}} = 150$ and $M_{\text{high}} = 150$. For each prompt, we sample $k = 8$ candidate responses. The dataset-specific consistency threshold τ is set to $4/8$ for GSM8K and MATH, and $5/8$ for WebInstruct. We adopt LoRA with LoRA rank set to $r = 8$ and LoRA alpha set tp $\alpha_{\text{LoRA}} = 16$. Model training is implemented using the LLaMA-Factory framework.[2]

C.3 Inference and Evaluation

During response generation, we use nucleus sampling with top-$p = 0.9$ and sample temperatures from the set $\{0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7\}$ to encourage diverse reasoning paths. For final evaluation, we generate responses using three different random seeds and report results averaged across these runs. In this setting, we fix the generation temperature to 0.7 and top-p to 0.9. Inference is accelerated using `vLLM`.[3] All baseline methods use identical inference settings for fair comparison.

[2] https://github.com/hiyouga/LlamaFactory

[3] https://github.com/vllm-project/vllm

D Theoretical Analysis

We formalize preference-based LLM alignment under noisy feedback and establish conditions under which mixed supervision, combining clean oracle preferences with noisy AI-generated preferences, provably improves over oracle-only training.

D.1 Preliminaries and Assumptions

Assumption 1 (Preference Generation and BTL Model)**.** *Prompts $s \sim \rho$ and candidate responses $(a, a') \sim \pi_{\text{ref}}(\cdot \mid s)$. Preferences are generated under the Bradley–Terry–Luce model (Bradley and Terry, 1952):*

$$\mathbb{P}[a \succ a' \mid s] = \sigma(r^*(s, a) - r^*(s, a')),$$

where $\sigma(z) = (1 + e^{-z})^{-1}$ and r^* is the latent reward.

Assumption 2 (KL-Regularized Optimal Policy)**.** *The optimal policy π^* solves*

$$\max_{\pi} \mathbb{E}_{s\sim\rho,\, a\sim\pi}\left[r^*(s, a) - \beta \log \frac{\pi(a|s)}{\pi_{\text{ref}}(a|s)}\right],$$

with $\beta > 0$.

Following Rafailov et al. (2024), DPO is equivalent to binary logistic regression with implicit reward difference $h_\theta(x)$ and clean label probability $\mu_\theta(x) = \sigma(h_\theta(x))$, where $x = (s, a_w, a_l)$ and $y = 1$ indicates $a_w \succ a_l$.

Assumption 3 (Regularity)**.** *h_θ is twice continuously differentiable in θ, the clean Fisher information*

$$I_{\text{clean}}(\theta) = \mathbb{E}_x\left[\mu_\theta(1 - \mu_\theta)\nabla h_\theta \nabla h_\theta^\top\right]$$

is positive definite at θ^* with effective dimension d, and r^* is L_r-Lipschitz in θ near θ^*.

D.2 Fisher Information Under Symmetric Noise

Let $\tilde{y}$ be obtained by flipping y with probability $\epsilon \in [0, \frac{1}{2})$. Then $\tilde{\mu}_\theta(x) = (1 - 2\epsilon)\mu_\theta(x) + \epsilon$.

Lemma 1 (Noise-Attenuated Fisher Information)**.** *Under Assumption 3 and $\epsilon \in [0, \frac{1}{2})$,*

$$I_{\text{noisy}}(\theta) \preceq (1 - 2\epsilon)^2\, I_{\text{clean}}(\theta),$$

where $\preceq$ denotes the Loewner order.

Proof. Since $\nabla\tilde{\mu}_\theta = (1-2\epsilon)\,\mu_\theta(1-\mu_\theta)\,\nabla h_\theta$, the score of the noisy likelihood is

$$\nabla_\theta \log \tilde{p}_\theta(\tilde{y} \mid x) = \tfrac{(1-2\epsilon)\,\mu_\theta(1-\mu_\theta)}{\tilde{\mu}_\theta(1-\tilde{\mu}_\theta)}\,(\tilde{y}-\tilde{\mu}_\theta)\,\nabla h_\theta.$$

Taking the outer-product expectation with $\mathbb{E}[(\tilde{y}-\tilde{\mu}_\theta)^2 \mid x] = \tilde{\mu}_\theta(1-\tilde{\mu}_\theta)$,

$$I_{\text{noisy}}(\theta) = (1-2\epsilon)^2\,\mathbb{E}_x\left[\tfrac{(\mu_\theta(1-\mu_\theta))^2}{\tilde{\mu}_\theta(1-\tilde{\mu}_\theta)}\nabla h_\theta \nabla h_\theta^\top\right].$$

Since $\tilde{\mu}_\theta$ is a convex combination of μ_θ and $\frac{1}{2}$, concavity of $f(t) = t(1-t)$ gives $\tilde{\mu}_\theta(1-\tilde{\mu}_\theta) \geq \mu_\theta(1-\mu_\theta)$. Substituting yields the claim. □

Lemma 1 recovers the classical $(1-2\epsilon)$ signal attenuation for learning with noisy labels (Natarajan et al., 2013) in the DPO setting.

D.3 Mixed Supervision: Error Bound via Fisher Additivity

Let $\mathcal{D}_{\text{oracle}}$ be clean with N_o samples and $\mathcal{D}_{\text{AI}}$ be self-labeled with N_{ai} samples and symmetric noise rate $\epsilon_{ai} \in [0, \frac{1}{2})$, both drawn i.i.d. from ρ. By Fisher additivity,

$$\begin{aligned} I_{\text{mix}}(\theta) &= N_o \bar{I}_{\text{clean}} + N_{ai}\bar{I}_{\text{noisy}} \\ &\preceq N_{\text{eff}}\,\bar{I}_{\text{clean}}(\theta), \end{aligned}$$

where $N_{\text{eff}} \triangleq N_o + N_{ai}(1-2\epsilon_{ai})^2$ denotes the *effective sample size*.

Lemma 2 (Parameter Estimation Rate)**.** *Under Assumption 3, as $N_o + N_{ai} \to \infty$,*

$$\mathbb{E}\|\hat{\theta}-\theta^*\|_2 \leq (1+o(1))\sqrt{\tfrac{d}{N_{\text{eff}}\,\lambda_{\min}(\bar{I}_{\text{clean}})}}.$$

Proof. By standard M-estimator asymptotics, the MLE satisfies $\sqrt{N_o+N_{ai}}(\hat{\theta}-\theta^*) \xrightarrow{d} \mathcal{N}(0,\Sigma)$ with $\Sigma^{-1} = \frac{N_{\text{eff}}}{N_o+N_{ai}}\bar{I}_{\text{clean}}(\theta^*)$. Thus $\mathbb{E}\|\hat{\theta}-\theta^*\|_2^2 = \frac{1}{N_{\text{eff}}}\text{tr}(\bar{I}_{\text{clean}}^{-1}) + o(N_{\text{eff}}^{-1})$. Using $\text{tr}(\bar{I}_{\text{clean}}^{-1}) \leq d/\lambda_{\min}(\bar{I}_{\text{clean}})$ and Jensen's inequality yields the bound. □

Lemma 3 (From Parameter Error to Policy Gap)**.** *Under Assumption 3, the policy gap $\text{Gap}(\theta) = V^*(\pi^*) - V^*(\pi_\theta)$ satisfies*

$$\text{Gap}(\theta) \leq 2L_r\,\|\theta-\theta^*\|_2.$$

Proof. Let $V_{r_\theta}^*$ denote the regularized value under r_θ. Adding and subtracting $V_{r_\theta}^*(\pi^*)$ and $V_{r_\theta}^*(\pi_\theta)$ gives

$$\begin{aligned} \text{Gap}(\theta) = \mathbb{E}_{\pi^*}[r^* - r_\theta] + \mathbb{E}_{\pi_\theta}[r_\theta - r^*] \\ + \left[V_{r_\theta}^*(\pi^*) - V_{r_\theta}^*(\pi_\theta)\right]. \end{aligned}$$

The third term is non-positive since π_θ maximizes $V_{r_\theta}^*$, and the first two are each bounded by $L_r\|\theta-\theta^*\|_2$ by the Lipschitz property of r. □

Theorem 1 (Improvement Condition for Mixed Supervision)**.** *Under Assumption 3, there exists $C > 0$ (independent of $N_o, N_{ai}, \epsilon_{ai}$) such that, as $N_o + N_{ai} \to \infty$,*

$$\begin{aligned} \text{Gap}_{\text{oracle}} &\leq \frac{C\sqrt{d}}{\sqrt{N_o}}, \\ \text{Gap}_{\text{mix}} &\leq \frac{C\sqrt{d}}{\sqrt{N_o + N_{ai}(1-2\epsilon_{ai})^2}}. \end{aligned}$$

Hence the mixed bound is strictly smaller than the oracle-only bound whenever $\epsilon_{ai} < \frac{1}{2}$, with relative improvement

$$\tfrac{\text{Gap}_{\text{oracle}}}{\text{Gap}_{\text{mix}}} \geq \sqrt{1 + \tfrac{N_{ai}(1-2\epsilon_{ai})^2}{N_o}}.$$

Proof. Combining Lemmas 2 and 3 with $C = 2L_r/\sqrt{\lambda_{\min}(\bar{I}_{\text{clean}})}$ yields both bounds; oracle-only corresponds to $N_{ai} = 0$. The improvement condition reduces to $N_{ai}(1-2\epsilon_{ai})^2 > 0$, i.e., $\epsilon_{ai} < \frac{1}{2}$. The ratio follows by direct computation. □

E More Experiments

E.1 Out-of-Domain Generalization

We evaluate the generalization capability of CoAct by testing models trained on in-domain datasets (`GSM8K`, `MATH`, `WebInstruct`) on challenging out-of-domain benchmarks. We assess performance on three diverse tasks: **AIME** (Veeraboina, 2023), **GPQA** (Rein et al., 2024) and **MMLU-Pro** (Wang et al., 2024b). Tables 6, 7, and 8 report out-of-domain performance for models trained on `GSM8K`, `MATH`, `WebInstruct` respectively. We compare CoAct against four active learning baselines: Random, Entropy, Pref Certainty, and Pref + Ent.

Table 6: Out-of-domain performance for models trained on `GSM8K`.

Method	AIME	GPQA	MMLU-Pro
Random	3.33	24.12	22.45
Entropy	3.33	25.38	21.67
Pref Certainty	2.22	27.14	24.18
Pref + Ent	3.33	25.89	22.76
CoAct	**6.67**	**28.35**	**25.99**

Table 7: Out-of-domain performance for models trained on MATH.

Method	AIME	GPQA	MMLU-Pro
Random	3.33	26.52	21.34
Entropy	0.00	27.18	22.15
Pref Certainty	3.33	27.48	22.08
Pref + Ent	3.33	27.85	22.76
CoAct	**6.67**	**28.46**	**23.42**

Table 8: Out-of-domain performance for models trained on WebInstruct.

Method	AIME	GPQA	MMLU-Pro
Random	0.00	26.83	21.54
Entropy	3.33	28.12	22.91
Pref Certainty	3.33	27.45	22.18
Pref + Ent	2.22	28.76	23.45
CoAct	**2.22**	**29.51**	**24.17**

F Extended Related Work

Self-Consistency for LLMs. Self-consistency was originally proposed as a test-time inference strategy to improve reasoning accuracy (Wang et al., 2023). The key intuition is that sampling multiple reasoning paths and aggregating answers that appear most frequently provides higher confidence that the consistent answer is correct (Wang et al., 2023; Shi et al., 2022; Li et al., 2022). Recently, researchers have adapted self-consistency from test-time inference to training-time pseudo-labeling for preference learning (Prasad et al., 2025; Jiao et al., 2025; Xu et al., 2026). The core idea is to generate multiple model responses and use consistency as a signal for quality: responses that consistently arrive at the same answer across different reasoning paths are treated as higher-quality, while inconsistent responses are considered lower-quality. However, existing work using self-consistency for preference learning does not consider how to actively select which queries benefit most from self-consistency-based labeling. Our work addresses this by integrating self-consistency into an active learning framework that selectively applies consistency-based pseudo-labeling to queries where it provides the strongest signal.

G Ethics Statement

We provide comprehensive methodological details to enable reproducibility. Our framework uses exclusively publicly available resources and does not collect or process personal information. ChatGPT, Gemini, and Claude were used solely for minor grammatical and formatting corrections, in accordance with their respective usage policies. While our framework is designed for benign research, we acknowledge potential risks such as bias propagation from automated data synthesis.

H Prompts

This section presents the prompts used throughout our experiments. We organize them into four categories: response generation for constructing preference pairs (green), instruction augmentation with oracle feedback (blue), oracle preference evaluation (red), and zero-shot evaluation on out-of-domain benchmarks (purple).

Response Generation Prompts

```
# ============================================
# GSM8K Response Generation
# ============================================
gsm8k_response_prompt: str = """
You are given a grade school math word problem involving basic arithmetic,
algebra, or geometry. Your task is to carefully read the problem and provide
a step-by-step solution for it.

Provide a step-by-step reasoning process and then write the final numerical
answer on a new line in the format:
final answer: <answer>
"""

# ============================================
# MATH Response Generation
# ============================================
math_response_prompt: str = """
You are given a competition-level mathematics problem. Your task is to provide
a detailed step-by-step solution demonstrating rigorous mathematical reasoning.

Present the final result inside a LaTeX boxed expression, i.e., write the answer as \\boxed{<answer>}.
"""

# ============================================
# WebInstruct Response Generation
# ============================================
webinstruct_response_prompt: str = """
You are given a physics problem that requires numerical reasoning.  Carefully read the problem and provide a
      step-by-step solution.
Show all calculations and clearly explain your reasoning.
Provide a step-by-step solution and write the final answer in the format:
final answer: <answer>
"""
"""
```

Question Augmentation with Oracle Feedback

```
# ============================================
# GSM8K Instruction Augmentation
# ============================================
gsm8k_augmentation_prompt: str = """
Based on the examples above, generate ONE solvable math word problem with
similar difficulty. Ensure all information needed to solve the problem is
included in the question.

Output the question and nothing else.
Q:
"""
# ============================================
# MATH Instruction Augmentation
# ============================================
math_augmentation_prompt: str = """
Based on the examples above, generate ONE challenging mathematics problem
with similar difficulty and topic. Ensure the problem is well-defined and
solvable with the given information.

Output the question and nothing else.
Q:
"""

# ============================================
# WebInstruct Instruction Augmentation
# ============================================
webinstruct_augmentation_prompt: str = """
Based on the examples above, generate ONE solvable physics problem with
similar difficulty and topic. The question should require numerical reasoning and may involve units or currency. Ensure
      all information needed to solve the problem is included.

Output the question and nothing else.
Q:
"""
```

Oracle Preference Evaluation

```
# ============================================
# Oracle Evaluation Prompt
# ============================================
oracle_evaluation_prompt: str = """
You are an expert evaluator for mathematical/pysical reasoning problems. Evaluate two
responses to a math question and output your evaluation as a JSON object.

Question: {question}

Response 1: {full_response1}

Response 2: {full_response2}

Given the ground truth full response: {ground_truth_full_response}

Evaluate the responses following this logic:
1. Check if Response 1's final answer is correct
2. Check if Response 2's final answer is correct
3. Determine preference based on correctness:
   - If only one response is correct, prefer the correct one
   - If both responses are correct, prefer the one with better reasoning/explanation
   - If both responses are incorrect, prefer the one with better reasoning/explanation

Output your evaluation as a JSON object with the following structure:
{{
    "response1_correct": true/false,
    "response2_correct": true/false,
    "response1_preferred": true/false,
    "reasoning": "Brief explanation of your evaluation"
}}

Only output the JSON object, no additional text.
"""
```

Zero-Shot Evaluation Prompts

```
# ============================================
# AIME Zero-Shot Prompt
# ============================================
aime_zeroshot_prompt: str = """
You are given an American Invitational Mathematics Examination (AIME) problem.
These are challenging olympiad-level problems requiring creative mathematical
thinking. Provide a rigorous solution with clear mathematical reasoning.

Provide a step-by-step solution and write the final answer in the format:
final answer: <answer>
"""

# ============================================
# GPQA Zero-Shot Prompt
# ============================================
gpqa_zeroshot_prompt: str = """
You are given a graduate-level multiple choice question from physics, chemistry, or biology. Analyze each option
      carefully based on established scientific principles.

Provide a step-by-step explanation and write the final answer in the format:
final answer: <A/B/C/D>
"""

# ============================================
# MMLU-Pro Zero-Shot Prompt
# ============================================
mmlu_pro_zeroshot_prompt: str = """
You are given a multiple-choice question that tests knowledge across various
domains. Analyze the question carefully, consider each option, and provide
your reasoning before selecting the best answer.

Provide a step-by-step explanation and write the final answer in the format:
final answer: <A/B/C/D>
"""
```