Uncertainty-Aware Iterative Preference Optimization for Enhanced LLM Reasoning

Lei Li[1*], **Hehuan Liu**[1*], **Yaxin Zhou**[2], **Zhaoyang Gui**[1], **Xudong Weng**[1], **Yi Yuan**[1], **Zheng Wei**[1†], **Zang Li**[1]

[1] Platform and Content Group, Tencent
[2] University of Electronic Science and Technology of China
{arleyli, mermaidliu, hemingwei}@tencent.com

Abstract

Direct Preference Optimization (DPO) has recently emerged as an efficient and effective method for aligning large language models with human preferences. However, constructing high-quality preference datasets remains challenging, often necessitating expensive manual or powerful LM annotations. Additionally, standard DPO exhibits suboptimal performance in complex reasoning tasks, such as mathematical and code reasoning. In this paper, we introduce an approach to collect preference pairs through iterative sampling and execution feedback, tailored to the current learning state (*e.g.* well-learned, mis-learned, and unlearned) of the policy model. To alleviate the failures of DPO and improve its applicability in reasoning tasks, we propose IUPO, an iterative uncertainty-aware preference optimization method that achieves fine-grained preference control by assessing model confidence. We validate our approach across three reasoning tasks, incorporating five established reasoning datasets and one self-curated dataset. Our experimental results demonstrate an overall improvement of 3.6% over the standard DPO method and show the model exhibits promising generalizability.

1 Introduction

Preference optimization has emerged as a crucial ingredient in the post-training process to advance the development of large language models (LLMs) (Christiano et al., 2017; Tunstall et al., 2023; Dubey et al., 2024). The early approaches utilize reinforcement learning (RL) to align the LLM policy with human feedback or AI-generated feedback against a reward model, denoted as RLHF (Nakano et al., 2021; Ouyang et al., 2022; OpenAI, 2023) or RLAIF (Bai et al., 2022b; Lee et al., 2024; Wang et al., 2024). To streamline this process, Rafailov et al. (2023) proposes an offline direct preference optimization method, termed DPO, which aligns the policy directly with feedback without reward modeling. Benefiting from its simplicity and efficiency, DPO has shown impressive results in various applications, including summarization (Stiennon et al., 2020), dialogue assistance (Bai et al., 2022a; Anil et al., 2023), and chat benchmarks (Tunstall et al., 2023).

However, in complex reasoning tasks such as code reasoning and long-chain mathematical reasoning tasks, DPO often achieves only moderate gains or even impairs performance. We conjecture that this performance gap can be primarily attributed to (1) the scarcity of high-quality preference data and (2) the limitations inherent in the alignment method for improving the complex reasoning capabilities of large language models. Specifically, while long-chain complex reasoning tasks require numerous reasoning steps to solve, most alignment data are at the instance level and cannot pinpoint specific errors in incorrect answers, thus hindering the improvement of reasoning abilities. Although some researchers explore more fine-grained preference data, such as step-level (Lai et al., 2024) preferences and preference trees (Yuan et al., 2024a), they are often costly to collect and present scalability challenges. Besides, Feng et al. (2024) points out another drawback of DPO: it can reduce the probabilities and rewards of both preferred and undesirable outputs, thereby increasing the likelihood of errors in long-chain reasoning (Yuan et al., 2024a). Pal et al. (2024) further investigate the failure mode of DPO when the preferred and dispreferred outputs are minimally contrastive, finding that DPO increases the probability of the token(s) that differ, yet decreases the probability of subsequent tokens. Meanwhile, another significant area of research focuses on iterative or online preference optimization, which aims to alleviate the distribution shift problem in offline

[*] Equal contribution and shared co-first authorship.
[†] Corresponding Author.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 23996–24012
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

DPO (Yuan et al., 2024b; Guo et al., 2024; Pang et al., 2024). However, the performance of these methods remains suboptimal, due to the challenges in ensuring the quality of preference data.

In this paper, we propose **I**terative **U**ncertainty-based **P**reference **O**ptimization (IUPO), a method that iteratively optimizes policy through response sampling and execution feedback. Initially, our approach employs both policy and naive language models to generate multiple responses to a given query. Subsequently, we establish a virtual executable environment for the code reasoning task, and deploy answer extractors for the mathematical reasoning task, allowing us to verify the correctness of responses without reward models or verifiers. Following this, we utilize the validated data to construct preference pairs, taking into account the learning state of the policy model to improve the quality of the alignment data. The crucial advantages of the above process are outlined as follows: (1) The generation of preference data relies exclusively on pre-existing models without additional manual or more powerful model annotations. (2) The preference data is continuously updated during the iteration process, ensuring that the data remains in-distribution for policy model, which has been shown to be more effective than out-of-distribution data (Lai et al., 2024). (3) Our approach generates preference pairs with minimal contrastive (*i.e.* preferred and undesirable responses have a low edit distance), providing a better learning signal for policy optimization (D'Oosterlinck et al., 2024).

Additionally, we find the uncertainty measure (Jiang and Gupta, 2019; Wang and Zhou, 2024) strongly correlates with the performance of language models. Models tend to exhibit higher error rates when they display low confidence in certain tokens. Building on this observation, we leverage token-level uncertainty measures to achieve fine-grained control during preference optimization. Specifically, we mine the tokens that exhibit lower uncertainty measures and adjust the probability of the subsequent derailed tokens, which mitigates the decrease in the preferred probability issue. Our experimental results substantiate that the average confidence of the model is improved after optimization.

We comprehensively evaluate our method across a diverse spectrum of tasks, encompassing text-to-SQL reasoning (SQL and BIRD (Li et al., 2023)), code reasoning (Human Eval (Chen et al., 2021) and MBPP (Austin et al., 2021)), and mathematical reasoning (GSM8k (Cobbe et al., 2021a) and MATH (Hendrycks et al., 2021b)). Our experimental results demonstrate that IUPO yields a 3.6% improvement after three iterations compared to standard DPO, and consistently outperforms other baselines including SFT and DPO-Positive. In addition, our weak-to-strong and cross-model generalization experiments indicate that both our method and the generated preference data exhibit notable generalization capabilities. We also present a detailed analysis of how the uncertainty measure and iterative optimization influence the data distribution, training trajectory, and model performance.

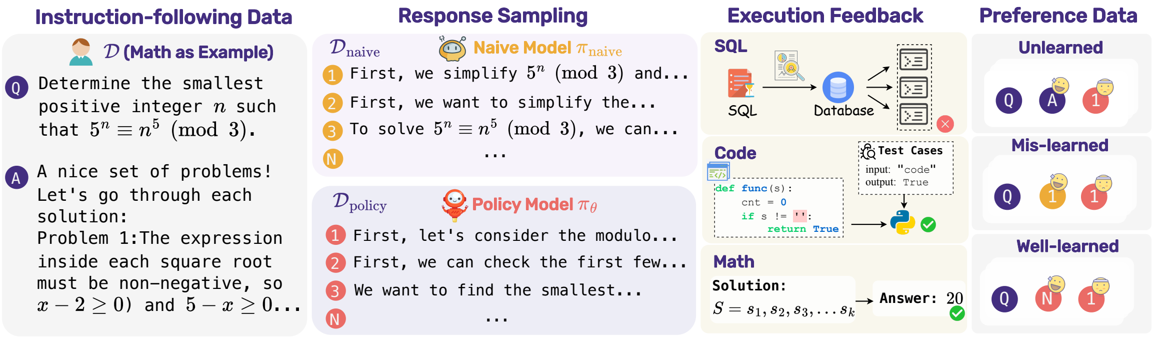
To summarize, our key contributions are encapsulated as follows: (1) We extend the direct preference optimization methods with uncertainty measure and iterative learning, resulting in IUPO. This method endows the standard preference optimization method with fine-grained control and alleviates its distribution shift issue. (2) We introduce an automatic strategy for preference data generation through response sampling and execution feedback, which considers the learning state of the policy model without requiring additional manual or more powerful model annotations. (3) We substantiate our contributions through experimental evaluations conducted using Llama3 and Mistral models across three reasoning tasks, which conclusively demonstrate the effectiveness and generalization capability of our approach in enhancing the reasoning ability of LLMs.

2 Preference Dataset

2.1 Data Generation

Traditional methods for generating high-quality preference datasets rely heavily on human labor (Ouyang et al., 2022) or strong LLMs (Bai et al., 2022b), which is time-consuming and expensive. Additionally, the precision and clarity of the resulting preference signals may be compromised, as the preference pairs are often minimally contrastive. In this section, we introduce a simple yet effective method for building preference datasets. As shown in Figure 1, our approach includes response sampling and execution feedback and can be subdivided into the following four key steps:

Step 1: Initialization. We begin by initializing with an instruction-following dataset $\mathcal{D}$, which consists of sets of (x, y) pairs, a naive model π_{naive}, and a policy model π_θ initialized from π_{naive} and then supervised fine-tuned on the dataset $\mathcal{D}$.


Instruction-following Data
Response Sampling
Execution Feedback
Preference Data
$\mathcal{D}$ (Math as Example)
Q
A
$\mathcal{D}_{\text{naive}}$
Naive Model π_{naive}
1
2
3
N
$\mathcal{D}_{\text{policy}}$
Policy Model π_θ
SQL
SQL
Database
Code
Test Cases
input: "code"
output: True
Math
Unlearned
Mis-learned
Well-learned

Determine the smallest positive integer n such that $5^n \equiv n^5 \pmod 3$.

A nice set of problems! Let's go through each solution:
Problem 1:The expression inside each square root must be non-negative, so $x - 2 \geq 0$) and $5 - x \geq 0$...

1. First, we simplify $5^n \pmod 3$ and...
2. First, we want to simplify the...
3. To solve $5^n \equiv n^5 \pmod 3$, we can...

N ...

1. First, let's consider the modulo...
2. First, we can check the first few...
3. We want to find the smallest...

N ...

```
def func(s):
    cnt = 0
    if s != '':
        return True
```

Solution:
$S = s_1, s_2, s_3, \ldots s_k$

Answer: 20

Figure 1: An illustration of the data creation pipeline. 😀 refers to the chosen and 😟 refers to rejected.

Step 2: Response Sampling. For each query x_i in $\mathcal{D}$, we sample N responses from both π_θ and π_{naive}, forming the two new set $\mathcal{D}_\theta = \{(x_i, y_j)\}_{j=1}^N$ and $\mathcal{D}_{\text{naive}} = \{(x_i, y'_j)\}_{j=1}^N$.

Step3: Execution Feedback. In scenarios involving code reasoning and mathematical reasoning, we simulate a virtual environment to execute synthetic responses. We then compare these execution results with the ground-truth answers to eliminate unfortunate instances. Each pair from $\mathcal{D}_\theta$ and $\mathcal{D}_{\text{naive}}$ is assigned a reward $r \in \{0, 1\}$, where $r = 1$ indicates that the response is correct.

Step4: Preference Pairs Construction. We construct the final preference pairs focusing on three learning states of π_θ: (1) *Unlearned* $(y \in \mathcal{D}, y_j \in \mathcal{D}_\theta | r_j = 0)$. We let the ground-truth answer as chosen and the error response generated by the policy π_θ as rejected, highlighting the fallibility of the model. (2) *Mis-learned* $(y'_j \in \mathcal{D}_{naive}, y_j \in \mathcal{D}_\theta | r'_j = 1, r_j = 0)$. We select the correct response from the naive model as chosen to steer the deviations in the policy model. (3) *Well-learned* $(y_i \in \mathcal{D}_\theta, y_j \in \mathcal{D}_\theta | r_i = 1, r_j = 0)$. In this part, we directly use the responses generated by the policy model to compose preference pairs, similar to self-rewarding (Yuan et al., 2024b).

Given that the policy π_θ undergoes continuous optimization during preference learning, we can naturally iterate the aforementioned steps to update preference data progressively. It is important to note that this method is not only efficient - eliminating the need for additional LMs or human labor, but also effective - it generates in-distribution preference data for the policy model. Furthermore, this approach facilitates an iterative online preference optimization process. The complete algorithm process is detailed in Algorithm 1.

Dataset	Prompt Length	Response Length	# Pairs	Normalized Levenshtein (↑)
SQL	49.5	280.9	16,627	87%
BIRD	189.2	213.0	29,939	78%
Math	265.5	1587.4	13,918	38%
Code	1448.8	872.3	28,430	50%

Table 1: Average character-level Levenshtein edit-distance between chosen and rejected answers for four preference datasets.

2.2 Data Statistics

Regarding the instruction-following dataset, we select APPS+ (Dou et al., 2024) for code reasoning, GSM8K (Cobbe et al., 2021b) and Math (Hendrycks et al., 2021b) for mathematical reasoning, and BIRD (Li et al., 2023) for text-to-SQL reasoning. We also curated a new text-to-SQL dataset that mirrors real-world distributions. Then we apply our preference dataset generation strategy to these datasets. The statistical data and comparisons across reasoning tasks are presented in Table 1 and Figure 7. We observe that the preference pairs exhibit minimal contrast since they have low Levenshtein distance (*i.e.* edit distance), which provides more clear learning signals. For more details, please refer to Appendix B.

3 Method

3.1 Revisited Direct Preference Optimization (DPO)

Direct Preference Optimization (DPO) (Rafailov et al., 2023) is a computationally lightweight alignment method that directly optimizes the language model to human preferences without explicit reward modeling. Specifically, given an input prompt x and a preference pair (y_w, y_l), DPO aims to maximize the probability of the preferred output y_w and

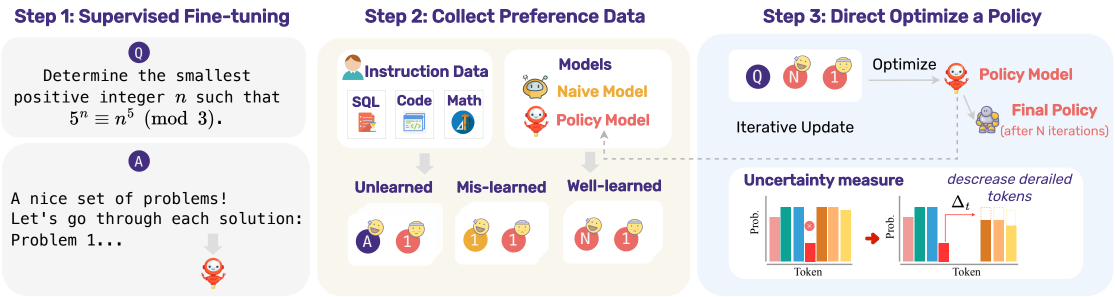


Figure 2: Overview of our IUPO framework. We first use the instruction-following data to fine-tune an LM policy. Then we collect preference data based on the learning state of the policy. Finally, we optimize the policy model with the preference data via uncertainty measure. This whole procedure is then iterated N times. The circle N and circle 1 come from Figure 1 refers to the $n-th$ response and the first response of the policy output.

minimize that of the undesirable output y_l:

$$u(x, y_w, y_l) = \beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)},$$
$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}[\log \sigma(u(x, y_w, y_l))]$$

where $\mathcal{D}$ is the preference data, $\pi_\theta(\cdot)$ is the policy model to be optimized, π_{ref} is the reference model kept frozen during training, and β is a parameter that controls deviation from reference policy π_{ref}.

3.2 Failure Mode of DPO

Although DPO has achieved many impressive results in various tasks and has become one of the most popular alignment methods, it only makes moderate gains or even decreases the performance on standard reasoning tasks such as code and mathematical reasoning, especially when y_w and y_l have low edit distance. The reasons may be attributed to the following points:

1. **Coarse-grained preference signal.** Code and mathematical reasoning are recognized as critical domains, requiring complex, long-chain reasoning abilities. However, the optimization of DPO operates at the instance level, where most preference data signals are coarse-grained, making the model struggle to identify detailed errors in incorrect answers.

2. **Decrease in preferred probability.** When preferred and undesirable responses share many similar tokens, DPO may decrease the probabilities of both the undesirable and preferred (Pal et al., 2024). Feng et al. (2024) also theoretically demonstrates DPO loss significantly impacts $\pi_\theta(y_l|x)$ due to the larger gradient, as opposed to its effect on $\pi_\theta(y_w|x)$.

3. **Frozen reward and offline learning**. Standard DPO is an offline method that relies on a pre-collected preference dataset. While the policy is continuously updated, the reward distribution remains static, leading to distribution shift and reward hacking problems.

3.3 IUPO

To alleviate the above failures of DPO in Section 3.2, we propose the **I**terative **U**ncertainty-based **P**reference **O**ptimization (IUPO) method, which utilize "uncertainty" to measure model confidence to achieve fine-grained control, and iterative collect preference data to optimize policy.

Uncertainty. Uncertainty is employed to measure model confidence (Wang and Zhou, 2024) by calculating probability disparity between the top and secondary tokens, which is similar to the minimum-margin approach (Jiang and Gupta, 2019). The uncertainty measures of the whole response y is $\Delta(y) = \{\Delta(y^t)|t = 1, ..., |y|\}$, where $\Delta(y^t)$ refers to the uncertainty of token t, defined as follows:

$$\Delta(y^t) = p(y_1^t|y^{<t}, x) - p(y_2^t|y^{<t}, x) + \epsilon \quad (1)$$

where $\Delta(*^t) \in [\epsilon, 1]$, and ϵ is a small number to prevent the result from being zero, y_1^t and y_2^t represent the top two tokens at the t-th generation step, chosen for their maximum post-softmax probabilities from the vocabulary. Wang and Zhou (2024) utilizes the uncertainty measure $\Delta(y)$ as a reliable indicator in CoT-decoding, yielding a significant boost on the model's reasoning performance. Furthermore, our experimental observations outline the key characteristics of the uncertainty measure as follows: (1) The uncertainty measure strongly correlates with the model's performance. (2) A low uncertainty measure Δ_t, indicates a lack of confidence in the model. This condition often coincides with the model's propensity to make errors, or the different tokens between preferred and undesirable. (3) The confidence of the model at the current time step has minimal impact on the generation of subsequent tokens, indicating that the model may continue along an erroneous trajectory with high confidence. Based on this, we propose integrating the uncertainty measure into the preference optimization process. On the one hand, the token-level uncertainty measure can **enable fine-grained optimization control**. On the other hand, we can leverage this measure to identify tokens where the model is prone to errors and **adjust the probability of subsequent derailed tokens** to mitigate the decrease in the preferred probability issue. Specifically, we select tokens with uncertainty measure below a fixed threshold τ and adjust the confidence of tokens within their subsequent window K:

$$\Delta(y^{t+k}) = (1 - \frac{k}{K}) \cdot \Delta(y^t), \quad k \in [1, K] \tag{2}$$

where k is the relative distance (number of token intervals) with the token t, and K is a hyperparameter refers to window size (number of tokens). Tokens that are closer to token t within the window are more significantly influenced. Then we employ the measure to adjust the probabilities of the subsequent tokens, as illustrated in Figure 2, and the modified DPO loss can be seen as follows:

$$u_{\Delta}(x, y_w, y_l) = \beta \log \frac{\pi_{\theta,\Delta}(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_{\theta,\Delta}(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \tag{3}$$

$$\mathcal{L}_{\text{UPO}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}[\log \sigma(u_{\Delta}(x, y_w, y_l))]$$

where $\pi_{\theta,\Delta}(y|x) = \prod_t^{|y|} \pi_\theta(y^t|y^{<t}, x) \cdot \Delta(y^t)$, Since $\Delta(y^t)$ is less than 1, the probability of the token after the difference with preferred in $\pi_\theta(y_l|x)$ will decrease, and the corresponding gradient will be lower, thus alleviating the decrease in the preferred probability issue.

Iterative. To improve the performance and alleviate the reward distribution shift problem, we optimize the policy model π_θ iteratively, in which the policy model and the preference data are both fresh during each iteration. Specifically, we initialize the policy model π_θ and the reference model π_{ref} with the supervised fine-tuned model π_{sft}. The initial preference data is also generated based on π_{sft} and π_θ, and is subsequently utilized to optimize the policy model π_θ using Equation 3. Then the preference data is regenerated based on the updated policy model π_θ, as described in Section 2.

$$u^i_{\Delta}(x, y_w, yl) = \beta \log \frac{\pi^i_{\theta,\Delta}(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi^i_{\theta,\Delta}(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \tag{4}$$

$$\mathcal{L}_{\text{IUPO}}(\theta) = -\sum_i^I \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}^i}[\log \sigma(u^i_{\Delta}(x, y_w, yl))]$$

where $i \in [1, I]$ is the current iteration and I is the total iterations, which is set to 3 in our paper. Note that the reference model π_{ref} kept frozen during the preference optimization process.

4 Experiments

4.1 Experimental Settings

Datasets & Baselines. We conduct experiments using LLaMA3 series (LLaMA3-8B and LLaMA3-70B) (Dubey et al., 2024) and Mistral-7B (Jiang et al., 2023). In the supervised fine-tuning stage, we utilize the training set of BIRD (Li et al., 2023), APPS+ (Dou et al., 2024), Dart-Math (Tong et al., 2024) and self-curated text-to-SQL dataset SQL as the fine-tuning data. The preference data is iteratively generated based on the above dataset as described in Section 2. We evaluate our method on text-to-SQL reasoning tasks (SQL and BIRD), code reasoning tasks (Human Eval (Chen et al., 2021) and MBPP (Austin et al., 2021)), and mathematical reasoning tasks (GSM8K and MATH), compared with GPT series models and preference methods (DPO (Rafailov et al., 2023) and DPOP (Pal et al., 2024)). For more details, refer to the Appendix B.

Setup. In each preference data generation iteration, we generate 10 responses per question using sampling with temperature 0.7. In our IUPO method, we set the uncertainty threshold τ as 0.3 and the uncertainty windows K as 5. All preference methods in our experiments use the same β, epoch, batch size, as detailed in Appendix C.

4.2 Main Results

IUPO improves over baselines. The main performance results of all models are shown in Table 2. Across all three reasoning scenarios, it is evident that our IUPO consistently outperforms the supervised fine-tuned model (SFT) and direct preference optimization method (DPO), exhibiting an improvement of +2.1% in the Mistral-7B model and +3.6% in the Llama3-8B model. Additionally,

Model	Phase	Text-to-SQL (EX)		Code (Pass@1)		Math (EM)		Avg.↑
		SQL	BIRD	Human Eval	MBPP	GSM8K	MATH	
GPT-3.5-Turbo	-	24.1	47.2	64.9	77.0	92.0	42.5	54.1
GPT-4-Turbo-0409	-	46.4	53.4	87.6	80.2	94.5	73.4	71.1
GPT-4o-0513	-	42.4	56.1	90.2	81.4	95.8	76.6	72.2
Mistral-7B	Base	5.2	27.1	34.2	47.5	45.9	16.5	29.4
	SFT	50.9	54.1	24.4	46.7	82.3	42.3	50.1
	DPO	49.1	54.2	23.8	45.9	83.6	42.3	49.8
	DPOP	50.0	54.4	25.0	**47.9**	83.2	42.5	50.5
	IUPO-1	53.5	54.4	28.7	43.6	83.5	42.2	51.0
	IUPO-2	54.3	54.7	29.9	44.2	83.6	42.5	51.5
	IUPO-3	**55.1** ↑6.0	**55.1** ↑0.9	**30.5** ↑6.7	44.4 ↓1.5	**83.8** ↑0.2	**42.8** ↑0.5	**51.9** ↑2.1
Llama3-8B	Base	9.5	32.9	59.2	53.3	51.0	21.2	37.8
	SFT	50.0	52.7	40.9	57.6	82.5	43.5	54.5
	DPO	50.8	52.5	38.4	55.3	82.6	43.5	53.9
	DPOP	51.2	52.5	36.6	57.2	83.2	43.9	54.1
	IUPO-1	51.7	54.2	47.6	56.0	83.2	43.9	56.1
	IUPO-2	52.6	54.6	48.8	58.8	83.5	43.8	57.0
	IUPO-3	**52.6** ↑1.8	**56.1** ↑3.6	**49.0** ↑10.6	**59.1** ↑3.8	**83.8** ↑1.2	**43.9** ↑0.4	**57.4** ↑3.6

Table 2: The main results of our IUPO against other baselines across three reasoning tasks.

we find that DPO underperforms compared to SFT across multiple datasets, particularly in scenarios where the edit distances between preferred and dispreferred examples are minimal. DPOP adds an additional penalty term to the DPO loss function to incentivize maintaining a high log-likelihood of the preferred completions. While this approach yields slightly better performance over the standard DPO, it remains less effective compared to our IUPO.

Iterations of IUPO yield improved reasoning. Our observations indicate that our IUPO yields performance improvements over its training iterations in most scenarios. Specifically, the average performance increases from 56.1% to 57.0% to 57.4% across each iteration. However, the magnitude of improvement diminishes with each iteration, as evidenced by the gains of 1.6%, 0.9%, and 0.4%, respectively. This trend suggests the presence of an upper limit on learning capacity across iterations, which is explored in detail in Section 4.4.

4.3 Analysis of the Uncertainty

Evolution of Model Uncertainty. To understand the impact of uncertainty measures on model performance, we compare the uncertainty value between supervised fine-tuning and our IUPO approach. Additionally, we analyze the uncertainty values for both correct and incorrect model predictions across three distinct reasoning tasks. As shown in Figure 3, the average uncertainty measures of LLM across all four reasoning tasks are at a high level, which indicates that LLMs generally exhibit confidence in the content they generate. This observation aligns with the discussion in Section 3.3, where the uncertainty measure for the correctly predicted sample is consistently higher than that for the incorrectly predicted one. This pattern underscores the effectiveness of the uncertainty measure in identifying areas where the model is prone to making errors. Moreover, compared to supervised fine-tuning, our IUPO approach significantly boosts the confidence of the model, particularly in scenarios where the predictions are correct.

Training Trajectory. In Figure 4, we study the training trajectories of chosen/rejected rewards on the four reasoning tasks for DPO, DPOP, and our IUPO alignment methods. Firstly, the reward margin between the chosen and rejected of all three methods increases during the training process, indicating that these alignment methods help distinguish preferred and dispreferred responses. Secondly, the training trajectories of the three methods exhibit distinct characteristics. Specifically, DPO can reduce the reward of the chosen when the preferred and dispreferred have minimal differences. DPOP mitigates this issue but leads to an increase in the rejected rewards. In contrast, our IUPO produces a more reasonable phenomenon that the rewards of chosen grow up to positive and the rewards of rejected steadily decline. Lastly, our IUPO achieves a larger margin between preferred and dispreferred responses compared to other alignment methods within the same training steps.

4.4 Analysis of the Iterations

Model performance for various iterations To better understand the role and impact of the iterations, we visualize the relations between performance

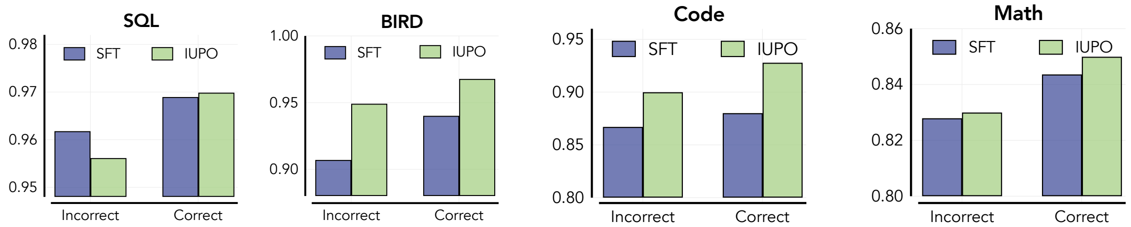


Figure 3: The uncertainty measures of SFT and IUPO between correct and incorrect answers in the four reasoning datasets. The y-axis refers to the uncertainty measure Δ_t, where larger means more confidence in the model.

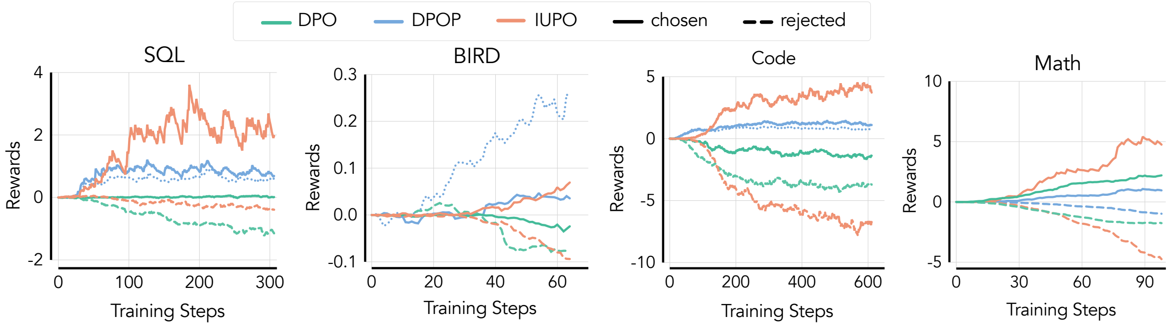


Figure 4: The reward for Llama3-8B on each reasoning task, trained using DPO, DPOP, or our IUPO alignment methods. Different methods use different colors.

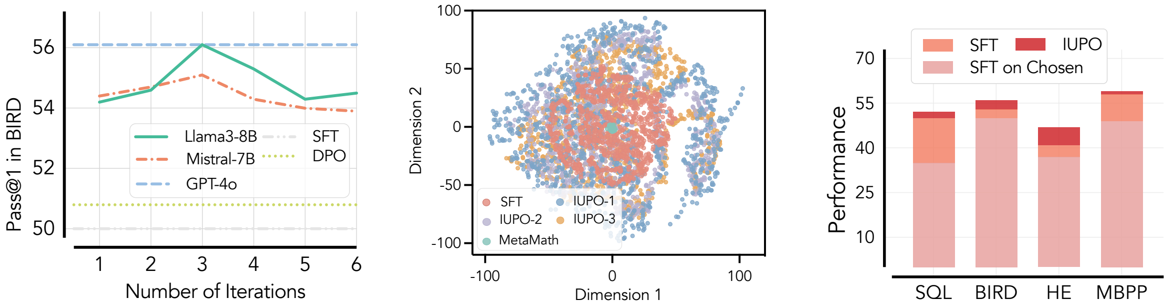


Figure 5: **Left**: Model performance for various training iterations. **Middle**: Visualization of the response distribution of SFT, IUPO, and MetaMath data. We select the MetaMath data related to GSM8K and MATH for comparison. **Right**: Comparison of SFT, SFT on chosen and IUPO.

Phase	BIRD	Human Eval	MBPP
IUPO-1	54.2	47.6	56.0
w/ twice data	54.2	47.3	58.0
w/ triple data	55.1	47.8	58.8
IUPO-2	54.6	48.8	58.8
IUPO-3	**56.1**	**47.0**	**59.1**

Table 3: The comparative results between one iteration with more data and more iterations with updating data on Llama3-8B.

and iterations as well as the distribution of preference data for each iteration. As shown in Figure 5, the performance of models on the BIRD dataset increases and then flattens out with the iterations, indicating that there is a performance ceiling when relying solely on iterative answer augmentation. To further improve, it may be beneficial to introduce synchronization in the diversity of questions.

Data Distribution. To visualize the distribution of the preference dataset, we first utilize the model to generate the pooled representation for the responses in the mathematical dataset. Afterward, we use t-SNE (Van der Maaten and Hinton, 2008) to map the representation into two-dimensional space, as shown in Figure 5 Middle. The data visualized includes the supervised fine-tuning data, iterative generated preference data, and the selected data related to GSM8k and MATH from MetaMath (Yu et al., 2023). It is clear to find that the data from MetaMath is aggregated at the center while our iterative data broadens the boundaries of SFT data.

Preference optimization vs. SFT on preferred. To determine whether the performance improvements come from increased training data or the efficacy of the preference optimization algorithm,

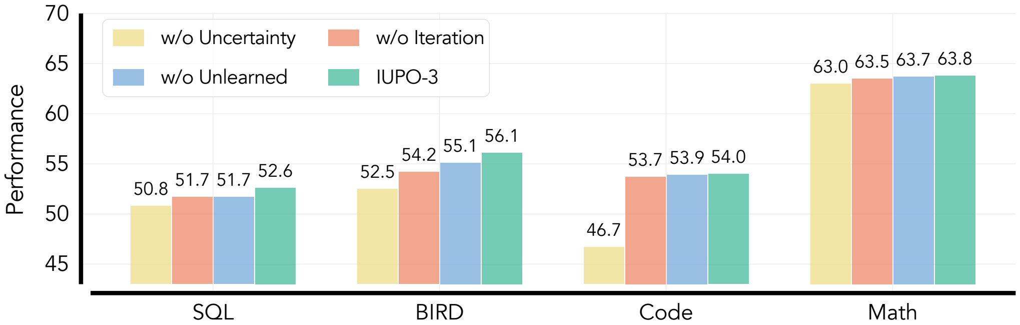


Figure 6: Ablation study across all the reasoning tasks using Llama3-8B. We show the averaged performance for the datasets of Code and Math.

we aggregate the preferred responses curated by the model in each iteration with the supervised fine-tuning data for supervised fine-tuning. However, as shown in Figure 5 Right, merely augmenting the dataset with preferred examples in a related manner did not help and even led to performance degradation, which is consistent with the findings in (Yuan et al., 2024b). In contrast, optimizing the model in the preference alignment manner with both preferred and dispreferred examples significantly improves the performance.

Iterative vs. More Preference data. We conduct a comparative experiment between one iteration with more data and more iterations with updating data to verify the effectiveness of iterative optimization. Specifically, we augment the preference data by doubling or tripling the sampling number N, and execute our IUPO method one iteration with the increased data. As shown in Table 3, while there is a noticeable improvement with the augmented preference dataset, the gains are not as substantial as performing optimization in two or three iterations. This observation underscores that the performance improvements achieved by IUPO are primarily driven by iterative optimization rather than increasing preference data volume alone.

4.5 Ablation Study

To elucidate the individual contributions of each component within our IUPO, we conducted an ablation study, and the results are depicted in Figure 6. In this study, we systematically discard key components: the iteration process (w/o Iteration), the uncertainty measure (w/o Uncertainty, degraded to DPO), and the preference pairs that models unlearned (w/o Unlearned). The results clearly demonstrate that each component of our method produces a positive effect on performance improvement, especially the iterative optimization and the uncertainty measure.

5 Related Work

5.1 Preference Dataset Curation and Augmentation

Preference dataset collection is an important step in LLM alignment. A common preference dataset is a set of prompts paired with a preferred and dispreferred response, where the preferred embodies the instructions, intentions, preferences, and values that humans intend for the LLM to internalize and replicate. Human labeling (Christiano et al., 2017; Ouyang et al., 2022) is a crucial tool for high-quality preference dataset construction. However, it is labor intensive and necessitates a certain level of knowledge of the annotator, which increases the cost and hinders the scalability of the data scale. Recently, LLMs have shown a high degree of alignment with human judgment (Gilardi et al., 2023), some researchers focus on Reinforcement Learning from AI Feedback (RLAIF) (Bai et al., 2022b; Lee et al., 2024), which leverages strong LLMs (*e.g.* GPT-4) to generate preference labels and achieves comparable performance to human labors. In this paper, we propose an effective method to build a preference dataset via iterative sampling based on the policy model and execution feedback to verify the correctness, which is efficient and effective.

5.2 Preference Optimization of LLMs

Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022) has emerged as a cornerstone in aligning LLMs with human preferences, providing a mechanism to enhance LLMs' comprehension of human requirements and refining their responses for improved alignment. To simplify this process, Rafailov et al. (2023) proposes DPO, which directly uses the pairwise data for model optimization without reward modeling. While DPO has achieved impressive results in various scenarios, it only makes moderate gains or even decreases performance for mathematical or code reasoning. (Feng et al., 2024) analyzes the failure modes of Direct Preference Optimization (DPO) and finds that the optimization process can inadvertently reduce the number of preferred examples. To alleviate this issue, Pal et al. (2024) adds a penalty term to DPO loss to incentivize maintaining a high log-likelihood of the preferred completions. In contrast, we utilize uncertainty to measure model confidence to achieve fine-grained control. Furthermore, we optimize the policy model in an iterative manner to realize online learning.

6 Conclusion

In this paper, we introduce IUPO, an iterative uncertainty-aware preference optimization method via response sampling and execution feedback. Our contribution also includes an automatic preference data generation strategy without additional manual or more powerful model annotations while considering the learning state of the policy model. Through comprehensive experimentation across three reasoning tasks and in-depth analysis of the components of our method, we have demonstrated the substantial benefits of IUPO in augmenting the reasoning ability of LLMs.

Limitations

While our uncertainty-aware iterative preference optimization demonstrates promising results on complex reasoning tasks, this work presents several limitations. Firstly, the experimental validation is primarily confined to mathematical reasoning and code generation tasks. Secondly, our experiments are conducted on 7B and 8B models. Given the scaling laws of large language models, the effectiveness of our method when applied to substantially larger architectures warrants further investigation. This constraint stems from computational resource limitations, future work should address these scalability challenges while expanding the evaluation to broader reasoning paradigms.

References

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernández Ábrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan A. Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vladimir Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, and et al. 2023. Palm 2 technical report. *CoRR*, abs/2305.10403.

Jacob Austin, Augustus Odena, Maxwell I. Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. 2021. Program synthesis with large language models. *CoRR*, abs/2108.07732.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, Benjamin Mann, and Jared Kaplan. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *CoRR*, abs/2204.05862.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosiute, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemí Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022b. Constitutional AI: harmlessness from AI feedback. *CoRR*, abs/2212.08073.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harrison Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4299–4307.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021a. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021b. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. 2024. RLHF workflow: From reward modeling to online RLHF. *CoRR*, abs/2405.07863.

Karel D'Oosterlinck, Winnie Xu, Chris Develder, Thomas Demeester, Amanpreet Singh, Christopher Potts, Douwe Kiela, and Shikib Mehri. 2024. Anchored preference optimization and contrastive revisions: Addressing underspecification in alignment. *CoRR*, abs/2408.06266.

Shihan Dou, Yan Liu, Haoxiang Jia, Limao Xiong, Enyu Zhou, Wei Shen, Junjie Shan, Caishuang Huang, Xiao Wang, Xiaoran Fan, Zhiheng Xi, Yuhao Zhou, Tao Ji, Rui Zheng, Qi Zhang, Xuanjing Huang, and Tao Gui. 2024. Stepcoder: Improve code generation with reinforcement learning from compiler feedback. *CoRR*, abs/2402.01391.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. KTO: model alignment as prospect theoretic optimization. *CoRR*, abs/2402.01306.

Duanyu Feng, Bowen Qin, Chen Huang, Zheng Zhang, and Wenqiang Lei. 2024. Towards analyzing and understanding the limitations of DPO: A theoretical perspective. *CoRR*, abs/2404.04626.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd-workers for text-annotation tasks. *CoRR*, abs/2303.15056.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Ramé, Thomas Mesnard, Yao Zhao, Bilal Piot, Johan Ferret, and Mathieu Blondel. 2024. Direct language model alignment from online AI feedback. *CoRR*, abs/2402.04792.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. 2021a. Measuring coding challenge competence with APPS. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Heinrich Jiang and Maya R. Gupta. 2019. Minimum-margin active learning. *CoRR*, abs/1906.00025.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *CoRR*, abs/2406.18629.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2024. RLAIF vs. RLHF: scaling reinforcement learning from human feedback with AI feedback. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin Chen-Chuan Chang, Fei Huang, Reynold Cheng, and Yongbin Li. 2023. Can LLM already serve as A database interface? A big bench for large-scale database grounded text-to-sqls. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3214–3252. Association for Computational Linguistics.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, and Sayak Paul. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. Webgpt: Browser-assisted question-answering with human feedback. *CoRR*, abs/2112.09332.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. 2024. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *CoRR*, abs/2402.13228.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *CoRR*, abs/2404.19733.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize from human feedback. *CoRR*, abs/2009.01325.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *CoRR*, abs/2407.13690.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of LM alignment. *CoRR*, abs/2310.16944.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 9426–9439. Association for Computational Linguistics.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting. *CoRR*, abs/2402.10200.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023.

Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3911–3921. Association for Computational Linguistics.

Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, Zhenghao Liu, Bowen Zhou, Hao Peng, Zhiyuan Liu, and Maosong Sun. 2024a. Advancing LLM reasoning generalists with preference trees. *CoRR*, abs/2404.02078.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024b. Self-rewarding language models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

A Formal Analysis.

Drawing on the Equation 1, we can derive the gradient with respect to θ:

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\text{DPO}}(\theta) &\propto -\nabla_\theta[\log \pi_\theta(y_w|x) - \log \pi_\theta(y_l|x)] \\ \propto -\nabla_\theta[\log \prod_{t=1}^{T} \pi_\theta(y_w^t|y_w^{<t}, x) &- \log \prod_{t=1}^{T} \pi_\theta(y_l^t|y_l^{<t}, x)] \\ \propto -\nabla_\theta[\sum_{t=1}^{T} \log \pi_\theta(y_w^t|y_w^{<t}, x) &- \sum_{t=1}^{T} \log \pi_\theta(y_l^t|y_l^{<t}, x)] \\ \propto -\sum_{t=1}^{T} \nabla_\theta[\log \pi_\theta(y_w^t|y_w^{<t}, x) &- \log \pi_\theta(y_l^t|y_l^{<t}, x)]\end{aligned} \tag{5}$$

where T is the total number of tokens. Following (Pal et al., 2024), we consider an extreme scenario that the two responses with an edit distance of 1 which differ at the token m (*i.e.* $y_w = (y_1, ..., y_T)$ and $y_l = (y_1, ..., y'_m, y_{m+1}, ..., y_T)$). Since the parameters θ of models are numerous, we focus on the logits θ_j, which is input to softmax. We let s_i^x represent the probability of the i-th token in the vocabulary conditioned on the input x, then we can simplify the Equation 5 to:

$$\begin{aligned}&\nabla_{\theta_j}[\log \pi_\theta(y_w^t|y_w^{<t}, x) - \log \pi_{\theta_j}(y_l^t|y_l^{<t}, x)] \\ &= 1\{1 = j\} - s_j^{\{y_w^{<t}, x\}} - 1\{1 = j\} - s_j^{\{y_l^{<t}, x\}} \\ &= s_j^{\{y_l^{<t}, x\}} - s_j^{\{y_w^{<t}, x\}}\end{aligned} \tag{6}$$

Since the policy model is likely to be reasonably well optimized after SFT, we should have $s_j^{\{y_w^{<t}, x\}} \leq s_j^{\{y_l^{<t}, x\}}$ for $j \neq m$. Therefore, we see the gradient vector is increasing in the wrong logit dimensions, which shows the standard DPO may increase the probability of the incorrect token after the difference point m. Subsequently, the gradient of our IUPO can be derived as:

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\text{UPO}}(\theta) \propto -\sum_{t=1}^{T} \nabla_\theta[\log \pi_\theta(y_w^t|y_w^{<t}, x) \cdot \Delta(y_w^t) - \\ \log \pi_\theta(y_l^t|y_l^{<t}, x) \cdot \Delta(y_l^t)]\end{aligned} \tag{7}$$

And the gradient of the t-th token with respect to the j-th logit becomes:

$$\begin{aligned}&\nabla_{\theta_j}[\log \pi_\theta(y_w^t|y_w^{<t}, x) \cdot \Delta(y_w^t) - \log \pi_{\theta_j}(y_l^t|y_l^{<t}, x) \cdot \Delta(y_l^t)] \\ &= s_j^{\{y_l^{<t}, x\}} \cdot \Delta(y_l^t) - s_j^{\{y_w^{<t}, x\}} \cdot \Delta(y_w^t)\end{aligned} \tag{8}$$

Since $\Delta(y_l^t) \leq \Delta(y_w^t)$ in most cases , especially when t is the difference token(see Section 4.3), the gradient can be negative, thus alleviating the DPO issue described above.

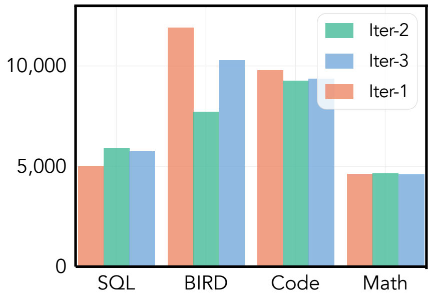


Figure 7: The number of data samples.

B Additional Dataset Information

B.1 The Supervised Fine-tuning Datasets

Table 4 shows the statistical details of the datasets used in supervised fine-tuning phases.

SQL Prompt

```
Generate a SQL query to answer this
question: ‘{question}‘

DDL statements:
{table_info}

The following SQL query best answers
the question ‘{question}‘:
```

SQL Since most of the prevalent Text-to-SQL benchmarks (*i.e.* WikiSQL (Zhong et al., 2017), and Spider (Yu et al., 2018)) focus on database schema with few rows of database values, we create a more challenging dataset for Text-to-SQL parsing to reduce the gap between academic study and real-world applications. In particular, we first select real-world databases with multiple rows and columns. Then we utilize GPT-4 (OpenAI, 2023) to generate user questions about the databases and the corresponding answers. All questions and answers are manually verified to ensure their quality.

BIRD Prompt

```
### Database scheme:
{table_info}

### Question:
{question}

### Match value:{match_value}

###SQL:
```

BIRD BIRD is another Text-to-SQL dataset developed by (Li et al., 2023). It contains 12,751 Text-to-SQL pairs and 95 databases with a total size of 33.4GB, spanning 37 professional domains, which highlights the challenges of dirty and noisy

Dataset	# Train	Task	Source
SQL	14,000	Text-to-SQL	Self-Curated
BIRD (Li et al., 2023)	12,751	Text-to-SQL	Open Source
APPS+ (Dou et al., 2024)	7,413	Code Reasoning	Open Source
DartMath (Tong et al., 2024)	591,000	Math Reasoning	Open Source

Table 4: Details about the supervised fine-tuning datasets.

database values, external knowledge grounding, and SQL efficiency, particularly in the context of massive databases.

APPS+ Prompt

```
### Instruction:
write an algorithm in python: {Task
description}
### Response:
```

APPS+ APPS+ is a clean version of APPS (Hendrycks et al., 2021a) created by (Dou et al., 2024). They excluded instances lacking input, output, and solutions of APPS, and standardized the formats of all instances. APPS+ contains 7,456 instances, including problem descriptions, canonical solutions, unit tests, and starter codes.

DartMath Prompt

```
Below is an instruction that describes a
task. Write a response that appropriately
completes the request.

###Instruction:
{query}

### Response:
```

DartMath DartMath is a synthetic dataset based on GSM8k (Cobbe et al., 2021a) and MATH (Hendrycks et al., 2021b) via difficulty-aware rejection sampling (Tong et al., 2024).

B.2 Details of Preference Dataset Generation Strategy

We show the details algorithm process in Algorithm 1.

Algorithm 1: Preference Dataset Generation Strategy

```
Input: naive model $\pi_{\text{naive}}$, policy model $\pi_\theta$,
    instruction-following dataset $\mathcal{D}$ consists of
    $N(x, y)$ pairs, iterations $\mathcal{I}$, sampling numbers
    $N$
Initialize $\pi_\theta$ from $\pi_{\text{naive}}$
Preference pairs dataset $\mathcal{D}_{\text{IUPO}} = \emptyset$
for $i = 1$ to $\mathcal{I}$ do
    for each pair $(x, y)$ in $\mathcal{D}$ do
        $\mathcal{D}_{\text{naive}} \leftarrow \emptyset$, $\mathcal{D}_\theta \leftarrow \emptyset$
        for $j = 1$ to $N$ do
            $y_j = \pi_\theta(x)$ // generate response from $\pi_\theta$
            $y'_j = \pi_{\text{naive}}(x)$ // generate response from $\pi_{\text{naive}}$
            $r = \text{EX}(x, y, y_j)$ // obtain the reward via execution feedback
            $r' = \text{EX}(x, y, y'_j)$
            Add pair $(x, y_j, r_j)$ to $\mathcal{D}_\theta$, and add pair $(x, y'_j, r'_j)$ to $\mathcal{D}_{\text{naive}}$
        end for
        Add all $(x, y, y_j)$ to $\mathcal{D}_{\text{IUPO}}$ where $y_j \in \mathcal{D}_\theta$ and $r_j = 0$ // Unleanred pairs
        Add all $(x, y'_j, y_j)$ to $\mathcal{D}_{\text{IUPO}}$ where $y_j \in \mathcal{D}_\theta, y'_j \in \mathcal{D}_{\text{naive}}$ and $r'_j = 1, r_j = 0$ // Mislearned pairs
        Add all $(x, y_i, y_j)$ to $\mathcal{D}_{\text{IUPO}}$ where $y_i, y_j \in \mathcal{D}_\theta$ and $r_i = 1, r_j = 0$ // Well-learned pairs
    end for
    $\pi_\theta$ = train_policy_model($\pi_\theta$, $\mathcal{D}_{\text{IUPO}}$) // training policy model
end for
Output: $\mathcal{D}_{\text{IUPO}}, \pi_\theta$
```

B.3 Details of the Evaluation Datasets

Table 5 shows the statistical details of the evaluation datasets. We use the same data configuration and assessment as the baseline.

Dataset	# Test	Task	Source
SQL	116	Text-to-SQL	Self-Curated
BIRD (Li et al., 2023)	1,533	Text-to-SQL	Open Source
Human Eval (Chen et al., 2021)	164	Code Reasoning	Open Source
MBPP (Austin et al., 2021)	257	Code Reasoning	Open Source
GSM8k (Cobbe et al., 2021b)	1,319	Math Reasoning	Open Source
MATH (Hendrycks et al., 2021b)	5,000	Math Reasoning	Open Source

Table 5: Details about the evaluation datasets.

C Details of Experiments

C.1 Detailed Evaluation Metrics

For all datasets, we compare the execution results between model predictions and ground truth. Specifically, we compute the **Execution Accuracy (EX)** for the Text-to-SQL reasoning task, which is defined as the proportion of examples in the evaluation set for which the executed results of both the predicted and ground-truth SQLs are identical, relative to the overall number of SQLs (Zhong et al., 2017). For code reasoning tasks, we compute the **pass@1** metric, where 1 code sample is generated per problem, and a problem is considered solved if the sample passes the unit tests. For the mathematical reasoning task, we extract the final answer from the generated solution and compare whether it is the same as the ground truth.

C.2 Implementation Details

All models of SFT and preference optimization phases are trained in the same environment (4 × 40G A100 GPUs.) During the preference dataset generation process, we generate $N = 10$ responses per question using sampling with temperature=0.7 and topp=1.0. Due to resource limitations, we adapt LoRA training in SFT and preference optimization phases with the PEFT Mangrulkar et al., 2022 framework. We set learning rate as 1e-4, training 4 epochs in SFT and 1 epoch in the alignment process. In DPO-Positive, we set the $\lambda = 50$ same as (Pal et al., 2024). In our IUPO, we set the uncertainty threshold $\tau = 0.3$ and the uncertainty windows $K = 5$.

D Additional Experiments

D.1 Comparison with preference optimization methods

KTO (Ethayarajh et al., 2024) is a powerful alignment method that directly maximizes the utility of generations instead of maximizing the log-likelihood of preferences. Since KTO does not require paired data, we split the preferred and dispreferred instances as the training data for a fair comparison. For all alignment methods, we optimize the policy model 3 iterations, and the results can be seen in Table 6. We find that KTO outperforms the vanilla DPO method and gains performance from iteration. This advantage may come from using prospect theory in KTO and its full utilization of preference samples. However, our IUPO remains the best performance due to its superior applicability to the contrastive preference data.

D.2 Extension to more reasoning scenarios

In some reasoning scenarios, we cannot use the execution environment to verify the correctness of the response. At this point, we can use the reward model to score the sampling responses and collect preference pairs by treating high-scoring responses as positive examples and low-scoring ones as dispreferred examples. To demonstrate the effectiveness of our method, following (Dong et al., 2024), we iteratively collect the preference pairs and optimize the policy model. Then we evaluate the model on TruthfulQA (Lin et al., 2022) and ARC-c (**?**) datasets. TruthfulQA. Among them, TruthfulQA is a benchmark dataset to measure the truthfulness of language models, which comprises 817 questions that span 38 categories, including health, law, finance, and politics. ARC-c is a dataset to measure the commonsense reasoning ability of models, which contains questions for science exams from grade 3 to grade 9. The results can be seen in Table 7, our IUPO obtains superior performance than DPO in each iteration, and the performance improves with iterations increase, proving that our approach is also feasible with the reward model.

Table 6: Comparison results for 3 iterations of DPO, KTO, and our IUPO methods.

Model	Phase	Text-to-SQL		Code		Math		Avg.↑
		SQL	BIRD	Human Eval	MBPP	GSM8K	MATH	
Llama3-8B	Base	9.5	32.9	59.2	53.3	51.0	21.2	37.8
	SFT	50.0	52.7	40.9	57.6	82.5	43.5	54.5
	DPO-1	50.8	52.5	38.4	55.3	82.6	43.5	53.9
	DPO-2	48.3	54.3	37.2	55.3	82.7	43.3	53.5
	DPO-3	49.1	53.0	37.2	55.6	83.1	43.2	53.5
	KTO-1	49.1	53.0	36.6	58.8	82.6	43.1	53.9
	KTO-2	48.2	53.7	37.2	57.6	82.6	42.6	53.7
	KTO-3	52.6	53.7	37.4	58.4	83.5	43.4	54.8
	IUPO-1	51.7	54.2	47.6	56.0	83.2	43.9	56.1
	IUPO-2	52.6	54.6	48.8	58.8	83.5	43.8	57.0
	IUPO-3	**52.6** ↑1.8	**56.1** ↑3.6	**49.0** ↑10.6	**59.1** ↑3.8	**83.8** ↑1.2	**43.9** ↑0.4	**57.4** ↑3.6

Table 7: Comparison results of DPO and our IUPO on ARC-c and TruthfulQA.

Dataset	SFT	DPO-1	DPO-2	DPO-3	IUPO-1	IUPO-2	IUPO-3
ARC-c	60.0	63.5	64.9	66.8	63.4	65.2	66.9
TruthfulQA	52.0	50.0	52.5	53.8	51.0	53.4	54.7

E Qualitative Examples

We show some qualitative examples of BIRD and Math tasks in Table 8 and Table 9. We compare the reasoning results of SFT, DPO, and our IUPO and highlight the wrong parts with distinct colors. In the Text-to-SQL task, we find that the results generated by different methods were remarkably similar, with only a few words differing. However, it is precisely these critical differences that lead to the opposite final outcomes. Benefiting from the contrasting preference data and fine-grained preference control, our IUPO makes correct judgments and predictions. Regarding to the mathematical reasoning task, the trajectory of logical reasoning is notably sensitive to nuanced variations, despite the apparent distinctions in formal derived from various methods. We find that the SFT model initially demonstrated correct reasoning but deviated towards incorrect conclusions in later stages. In contrast, both the DPO and UDPO models consistently made progress toward more accurate outcomes.

Question:	What are the foiled cards that are incredibly powerful when paired with non foiled cards? List the IDs.
Ground Truth:	SELECT id FROM cards WHERE cardKingdomId IS NOT NULL AND cardKingdomFoilId IS NOT NULL
SFT:	SELECT cardkingdomfoilid FROM cards WHERE cardkingdomid IS NOT NULL AND cardkingdomfoilid IS NOT NULL ✗
DPO:	SELECT cardkingdomfoilid FROM cards WHERE cardkingdomid IS NOT NULL AND cardkingdomfoilid IS NOT NULL ✗
IUPO:	SELECT id FROM cards WHERE cardkingdomfoilid IS NOT NULL AND cardkingdomid IS NOT NULL ✓

Table 8: A case example of BIRD dataset. ✓ refers to the answer is correct, while ✗ refers to the answer is incorrect.

Question:	Lillian's garden doesn't have any bird feeders in it so she wants to add some. She builds 3 and buys 3 others. Each bird feeder seems to attract 20 birds throughout the day until Lillian notices that the birds seem to prefer the feeders she made herself which attract 10 more birds each than the store-bought ones. How many birds can Lillian expect to see in her garden each day if the same amount keep coming to her bird feeders?
Ground Truth:	Lillian has 3 store-bought + 3 self-made bird feeders = «3+3=6»6 bird feeders in all. Lillian first notices each bird feeder attracts 20 birds, 6 x 20 = «6*20=120»120 birds. Then Lillian notices that her 3 self-made bird feeders attract 10 more birds each, 10 x 3 = «10*3=30»30 extra birds. Each day Lillian can expect to see 120 + 30 = «120+30=150»150 birds in her garden. The answer is: 150.
SFT:	Lillian builds 3 bird feeders and buys 3 others, so she has a total of 3 + 3 = 6 bird feeders. Each bird feeder she made herself attracts 10 more birds than the store-bought ones, so each of her bird feeders attracts 20 + 10 = 30 birds. The total number of birds that can be seen in her garden each day is the number of bird feeders times the number of birds each feeder attracts, so it's 6 * 30 = 180 birds. The answer is: 180 ✗
DPO:	Lillian builds 3 bird feeders and buys 3 others, so she has a total of 3 + 3 = 6 bird feeders. Each bird feeder she made herself attracts 20 + 10 = 30 birds each day. Each bird feeder she bought from the store attracts 20 birds each day. So, the total number of birds she can expect to see in her garden each day is 3 * 30 + 3 * 20 = 90 + 60 = 150 birds. The answer is: 150 ✓
IUPO:	Lillian builds 3 bird feeders and buys 3 others, so she has a total of 3 + 3 = 6 bird feeders. Each bird feeder she made herself attracts 10 more birds than the store-bought ones, so each of her bird feeders attracts 20 + 10 = 30 birds. The 3 bird feeders she made herself can therefore attract a total of 3 * 30 = 90 birds. The 3 store-bought bird feeders can attract a total of 3 * 20 = 60 birds. Adding these together, Lillian can expect to see a total of 90 + 60 = 150 birds in her garden each day. The answer is 150. ✓

Table 9: A case example of MATH dataset. ✓ refers to the answer is correct, while ✗ refers to the answer is incorrect.