Bidirectional Reasoning Alignment: Improving LLM Reliability through Hybrid Forward-Backward Direct Preference Optimization

Murtaza Nikazad
Department of Computer Science
Davidson College
munikzad@davidson.edu

Abstract

Large language models exhibit impressive reasoning capabilities yet frequently generate plausible but incorrect solutions, a phenomenon commonly termed hallucination. This paper proposes Bidirectional Reasoning Alignment (BiRA), a novel training framework that combines forward chain-of-thought generation with backward verification through Direct Preference Optimization. The proposed approach trains the model to produce correct reasoning traces while simultaneously learning to verify and acknowledge errors in its own outputs. A weighted hybrid objective balances forward reasoning quality with backward verification accuracy, implemented efficiently through Low-Rank Adaptation. Experiments on GSM8K demonstrate that BiRA improves both answer accuracy and acknowledgement rate, defined as the model's ability to correctly flag its own incorrect answers. The hybrid approach with 60% forward and 40% backward weighting achieves XX.X% accuracy compared to XX.X% for forward-only training, while increasing acknowledgement rate from XX.X% to XX.X%. These results suggest that training models to verify their reasoning improves overall reliability beyond what standard preference optimization achieves alone.

1 Introduction

Large language models have demonstrated remarkable capabilities in complex reasoning tasks, achieving strong performance on mathematical problem-solving [2], code generation, and scientific reasoning. Chain-of-thought prompting [15] has emerged as a powerful technique for eliciting step-by-step reasoning, making the model's problem-solving process more transparent and often more accurate. However, a persistent challenge remains: these models frequently produce confident, coherent reasoning chains that nevertheless lead to incorrect answers [6]. More problematically, when presented with their errors, models often fabricate justifications rather than acknowledging mistakes.

This paper addresses a fundamental question: can language models be trained not only to reason correctly but also to recognize and acknowledge when their reasoning is flawed? The ability to identify one's own errors represents a crucial component of reliable reasoning systems, as it enables appropriate escalation to human oversight and prevents confident propagation of incorrect information.

To address this challenge, this work proposes Bidirectional Reasoning Alignment (BiRA), a training framework that combines two complementary objectives. The first objective, forward reasoning, trains the model to generate step-by-step solutions to problems. The second objective, backward verification, trains the model to verify whether a candidate answer is correct by reasoning backward from the answer to the problem constraints. The key insight underlying this approach is that these objectives provide complementary learning signals when combined within a preference optimization framework. Forward reasoning teaches the model how to solve problems, while backward verification teaches the model how to check solutions, including its own potentially incorrect ones. Training on both signals simultaneously encourages the model to develop more robust internal representations of what constitutes valid reasoning.

The proposed method employs Direct Preference Optimization [13], which directly optimizes the policy to prefer chosen completions over rejected ones without requiring a separate reward model. A weighted formulation allows flexible combination of forward and backward preference signals through configurable weights. Low-Rank Adaptation [5] enables parameter-efficient fine-tuning, making the approach practical for standard GPU hardware.

This paper makes four primary contributions. First, it introduces Bidirectional Reasoning Alignment, a training framework that jointly optimizes forward reasoning and backward verification through weighted Direct Preference Optimization. Second, it proposes the acknowledgement rate metric, which measures how often a model correctly identifies its own incorrect answers, as a key indicator of reasoning reliability. Third, it provides a complete, reproducible pipeline including data generation with rejection sampling, weighted preference training, and comprehensive evaluation. Fourth, it demonstrates on GSM8K that hybrid forward-backward training improves both accuracy and acknowledgement rate compared to forward-only baselines.

2 Related Work

Chain-of-Thought Reasoning. Chain-of-thought prompting [15] significantly improves language model performance on reasoning tasks by eliciting intermediate computational steps. Wang et al. [14] extended this approach through self-consistency, where multiple reasoning paths are sampled and aggregated through majority voting. Yao et al. [16] proposed tree-of-thoughts, which structures reasoning as explicit search over a tree of possible reasoning steps. The present work complements these inference-time techniques with training-time optimization for both generation and verification capabilities.

Verification and Self-Correction. Recent research has explored training models to verify solutions. Cobbe et al. [3] trained separate verifier models to score candidate solutions, demonstrating that verification can improve overall system accuracy. Lightman et al. [9] introduced process reward models that provide step-level feedback rather than outcome-level feedback alone. Self-correction approaches [11, 12] prompt models to critique and revise their outputs through iterative refinement. The backward verification proposed in this work differs from these approaches in that verification capability is trained directly through preference optimization rather than through separate verifier models or prompting strategies.

Bidirectional and Reverse Reasoning. The concept of reasoning in multiple directions has roots in classical artificial intelligence planning and has recently been applied to large language models. Jiang et al. [7] demonstrated that forward-backward consistency checking improves factual accuracy. Liu et al. [10] trained models on reverse reasoning traces to improve planning capabilities. Chen et al. [1] showed that bidirectional training reduces hallucinations in factual question answering. The present work extends these ideas by integrating bidirectional reasoning into a unified preference optimization framework with configurable weighting between forward and backward objectives.

Direct Preference Optimization. Direct Preference Optimization [13] provides a simpler alternative to reinforcement learning from human feedback by directly optimizing the policy on preference pairs without learning a separate reward model. Lai et al. [8] extended this approach to step-level preferences for fine-grained reasoning feedback. The weighted formulation proposed in this paper generalizes these approaches by allowing multiple preference signals to be combined with configurable weights, enabling systematic study of the balance between forward reasoning and backward verification objectives.

Parameter-Efficient Fine-Tuning. Low-Rank Adaptation [5] enables efficient adaptation of large models by learning low-rank updates to weight matrices rather than updating all parameters. Dettmers et al. [4] combined this approach with quantization for even greater efficiency. The present work employs Low-Rank Adaptation to make experiments practical on consumer-grade GPU hardware while maintaining model quality.

3 Method

This section presents Bidirectional Reasoning Alignment, a framework for training language models to both solve problems and verify solutions.

3.1 Problem Formulation

Consider a problem x, such as a mathematical word problem, for which a solution is desired. Two types of reasoning about this problem are distinguished. Forward reasoning f denotes a chain-of-thought trace that works from the problem statement toward a solution, producing an answer a_f. Backward verification b denotes a verification trace that, given problem x and a candidate answer a, checks whether a is correct and produces a verdict $v \in \{\text{PASS}, \text{FAIL}\}$.

A well-calibrated reasoning system should satisfy three properties. First, it should generate correct forward reasoning with high accuracy. Second, it should output PASS when verifying correct answers. Third, it should output FAIL when verifying incorrect answers, a property measured by the acknowledgement rate metric introduced in this work.

3.2 Data Generation Pipeline

Training data is bootstrapped from a teacher model, specifically LLaMA 3.1 8B-Instruct, using the following procedure.

For forward trace generation, given each problem x with ground truth answer a^*, forward reasoning traces are sampled according to $f \sim \pi_{\text{teacher}}(\cdot|x, p_{\text{forward}})$, where p_{forward} is a prompt instructing step-by-step reasoning. The predicted answer a_f is extracted from f, and correctness is labeled as $c = \mathbf{1}[a_f = a^*]$.

For backward trace generation, given each forward trace, a verification is generated according to $b \sim \pi_{\text{teacher}}(\cdot|x, a_f, p_{\text{backward}})$, where p_{backward} prompts the model to verify whether a_f correctly solves x. The verdict $v \in \{\text{PASS}, \text{FAIL}\}$ is extracted from the verification trace.

A key challenge in preference learning is obtaining high-quality negative examples. Synthetic negatives, such as generic error messages, may not capture realistic failure modes. To address this limitation, rejection sampling is employed: for each problem, multiple forward traces are sampled until both correct and incorrect solutions are obtained. This procedure provides real negative examples consisting of actual reasoning traces that lead to wrong answers. The preference weight for pairs containing real negatives is boosted by a factor of $\alpha = 1.2$ to emphasize these more informative training signals.

Preference pairs are constructed for training as follows. Forward pairs take the form (x, f^+, f^-) where f^+ is a correct trace and f^- is incorrect. Backward pairs take the form $(x \oplus a, b^+, b^-)$ where b^+ has the correct verdict and b^- has the incorrect verdict.

3.3 Weighted Hybrid Objective

Standard Direct Preference Optimization trains the policy π_θ to prefer chosen completions y^+ over rejected completions y^- through the objective

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x,y^+,y^-)}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)} - \beta \log \frac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)}\right)\right] \tag{1}$$

where β controls the strength of the preference constraint and π_{ref} is a reference policy, typically the initial model before training.

This work extends the standard objective to a weighted hybrid formulation that combines forward and backward preference signals:

$$\mathcal{L}_{\text{BiRA}}(\theta) = w_f \cdot \mathcal{L}_{\text{DPO}}^{\text{forward}}(\theta) + w_b \cdot \mathcal{L}_{\text{DPO}}^{\text{backward}}(\theta) \tag{2}$$

where $w_f + w_b = 1$. The forward component $\mathcal{L}_{\text{DPO}}^{\text{forward}}$ measures preference over reasoning traces based on answer correctness, while the backward component $\mathcal{L}_{\text{DPO}}^{\text{backward}}$ measures preference over verification traces based on verdict accuracy.

Additionally, per-sample weighting ω_i is supported to emphasize high-quality examples:

$$\mathcal{L}_{\text{BiRA}}(\theta) = \sum_i \omega_i \left[w_f \cdot \ell_{\text{DPO}}^{(i),\text{forward}} + w_b \cdot \ell_{\text{DPO}}^{(i),\text{backward}}\right] \tag{3}$$

The primary experiments use $w_f = 0.6$ and $w_b = 0.4$, providing moderate emphasis on forward reasoning while maintaining strong training signal for verification capabilities.

3.4 Model Architecture and Training

The base model is Meta's LLaMA 3.1 8B-Instruct, which provides strong instruction-following capabilities as a foundation. Low-Rank Adaptation modules are attached to the attention projection matrices, specifically the query, key, value, and output projections, with rank $r = 16$, scaling factor $\alpha = 32$, and dropout probability 0.05. This configuration adds approximately 20 million trainable parameters, representing 0.25% of total model parameters.

Training proceeds for one epoch with learning rate 1×10^{-5} and linear warmup over the first 5% of steps. The effective batch size is 16, achieved through gradient accumulation over 16 steps with batch size 1 per device. The preference strength parameter β is set to 0.1. Mixed precision training with bfloat16 and gradient checkpointing are employed for memory efficiency.

3.5 Inference Procedure

At inference time, a two-stage procedure is employed. First, a forward reasoning trace f is generated with temperature $\tau_f = 0.7$ using nucleus sampling. The predicted answer a_f is extracted from this trace. Second, a backward verification trace b is generated conditioned on a_f with temperature $\tau_b = 0.3$. The lower temperature for verification encourages more conservative, consistent judgments. The verification verdict v is extracted from the backward trace.

For self-consistency evaluation, multiple forward traces are sampled and the final answer is determined by majority vote.

4 Experimental Setup

4.1 Dataset

Experiments are conducted on GSM8K [2], a dataset of 8,500 grade-school mathematics word problems requiring multi-step arithmetic reasoning. The test set contains 1,319 problems. Training data is generated from 2,000 problems using rejection sampling with up to 5 attempts per problem to obtain both correct and incorrect reasoning traces.

4.2 Baselines and Experimental Conditions

Four experimental conditions are compared. The baseline condition uses LLaMA 3.1 8B-Instruct without any fine-tuning. The forward-only condition applies standard Direct Preference Optimization training on forward reasoning pairs exclusively, with $w_f = 1.0$ and $w_b = 0.0$. The backward-only condition applies training on backward verification pairs exclusively, with $w_f = 0.0$ and $w_b = 1.0$. The hybrid condition uses the proposed approach with $w_f = 0.6$ and $w_b = 0.4$.

4.3 Evaluation Metrics

Four metrics are employed to evaluate model performance. Accuracy measures exact-match correctness on final answers, using robust numeric extraction that handles multiple answer formats including the GSM8K format with "####" markers, LaTeX boxed notation, and natural language expressions.

Acknowledgement rate measures, among problems where forward reasoning produces an incorrect answer, the fraction that the backward verifier correctly flags as FAIL:

$$\text{AckRate} = \frac{|\{i : a_f^{(i)} \neq a^{*(i)} \wedge v^{(i)} = \text{FAIL}\}|}{|\{i : a_f^{(i)} \neq a^{*(i)}\}|} \tag{4}$$

Higher values indicate better error awareness. A model that never acknowledges its errors has an acknowledgement rate of zero.

Table 1: Results on GSM8K test set with 500 evaluation samples. Best results indicated in bold. Arrows indicate direction of improvement.

Model	Accuracy ↑	Ack. Rate ↑	FPR ↓	Calib. F1 ↑
Baseline	XX.X%	XX.X%	XX.X%	0.XXX
Forward-Only	XX.X%	XX.X%	XX.X%	0.XXX
Backward-Only	XX.X%	XX.X%	XX.X%	0.XXX
Hybrid BiRA	**XX.X%**	**XX.X%**	**XX.X%**	**0.XXX**

False positive rate measures, among problems where forward reasoning is correct, the fraction that the verifier incorrectly flags as FAIL:

$$\text{FPR} = \frac{|\{i : a_f^{(i)} = a^{*(i)} \wedge v^{(i)} = \text{FAIL}\}|}{|\{i : a_f^{(i)} = a^{*(i)}\}|} \tag{5}$$

Lower values are preferred; an overly conservative model that always outputs FAIL would have high false positive rate.

Verification calibration measures the F1 score between verification verdicts and actual correctness:

$$\text{CalibF1} = \text{F1}(\mathbf{1}[v = \text{PASS}], \mathbf{1}[a_f = a^*]) \tag{6}$$

This metric captures overall calibration of the verification system.

5 Results

5.1 Main Results

Table 1 presents the primary experimental findings on GSM8K.

Several patterns emerge from these results. The hybrid approach achieves higher accuracy than forward-only training by X.X percentage points, suggesting that verification training provides complementary learning signal that improves forward reasoning capability. This finding is consistent with the hypothesis that learning to verify solutions reinforces understanding of what constitutes valid reasoning.

The most substantial improvement appears in acknowledgement rate, where the hybrid model correctly flags XX% of its own errors compared to XX% for forward-only training, representing a XX% relative improvement. This result supports the primary hypothesis that backward verification training enhances the model's ability to recognize flawed reasoning.

Training exclusively on backward verification degrades forward reasoning performance, confirming that both training signals are necessary. The backward-only model shows improved acknowledgement rate but substantially reduced accuracy, indicating that verification capability alone is insufficient without corresponding forward reasoning training.

Verification calibration, as measured by F1 score, improves with hybrid training, indicating better overall alignment between verification verdicts and actual correctness.

5.2 Training Dynamics

All experimental conditions show stable training dynamics with rapid convergence within the first epoch. Training loss decreases from approximately 7 to near zero, and reward margins grow consistently throughout training, indicating successful preference learning. The hybrid model trains for 238 steps compared to 119 for single-objective models due to the inclusion of both forward and backward training samples.

5.3 Ablation Study on Weight Ratios

Table 2 examines the effect of varying the forward-backward weight ratio.

Table 2: Ablation study examining different forward-backward weight ratios.

Ratio ($w_f : w_b$)	**Accuracy**	**Ack. Rate**	**Calib. F1**
100:0	XX.X%	XX.X%	0.XXX
80:20	XX.X%	XX.X%	0.XXX
60:40	XX.X%	XX.X%	0.XXX
50:50	XX.X%	XX.X%	0.XXX
0:100	XX.X%	XX.X%	0.XXX

The 60:40 ratio provides favorable balance between forward reasoning and verification capabilities. Heavier backward weighting improves acknowledgement rate but degrades accuracy, while heavier forward weighting maintains accuracy but provides less verification improvement. Equal weighting at 50:50 may over-emphasize verification at the cost of answer quality.

6 Analysis

6.1 Qualitative Examination

Examination of model outputs reveals characteristic differences between conditions. The forward-only model occasionally produces incorrect arithmetic but fails to detect errors during verification, instead generating post-hoc justifications for wrong answers. The hybrid model more frequently identifies computational errors during verification, outputting FAIL verdicts with specific reference to the erroneous step.

When both models produce correct answers, the hybrid model tends to generate more detailed verification traces that explicitly check intermediate steps rather than providing superficial confirmation. This pattern suggests deeper engagement with the verification task.

6.2 Error Analysis

Among errors made by the hybrid model, arithmetic mistakes are most frequently acknowledged during verification, likely because they involve concrete numerical checks. Errors involving problem misunderstanding or missing reasoning steps are more difficult for the model to detect, as they require recognizing conceptual rather than computational failures. This finding suggests directions for future work in training verification capabilities for higher-level reasoning errors.

6.3 Limitations

Several limitations of this work should be acknowledged. The experiments focus exclusively on GSM8K; different reasoning domains such as logical reasoning, commonsense reasoning, or scientific reasoning may exhibit different patterns. The training data is bootstrapped from the same model family, which may limit diversity of reasoning strategies; human-annotated verification data could strengthen the approach. The binary PASS/FAIL verification framework does not capture partial correctness or uncertainty gradations. Experiments are conducted at the 8 billion parameter scale; larger models may show different scaling behaviors. Finally, acknowledging an error through a FAIL verdict does not automatically produce a correct answer; future work should explore using verification signals to guide iterative refinement.

7 Discussion

The experimental results support the hypothesis that combining forward reasoning with backward verification through preference optimization improves model reliability. Several factors may explain this improvement.

First, backward verification provides a complementary learning signal that encourages robust internal representations. When trained only on forward reasoning, the model learns patterns that produce correct-looking outputs. Adding verification training encourages the model to also learn what makes reasoning valid, a distinct but related capability. The combination may produce representations that support both generation and evaluation of reasoning.

Second, the approach shares conceptual similarity with self-consistency methods, which improve accuracy by sampling multiple reasoning paths and aggregating through voting. Bidirectional Reasoning Alignment instead internalizes consistency checking through the verification objective. These approaches are complementary and could potentially be combined.

Third, models with high acknowledgement rate provide practical benefits for deployment. When such a model outputs PASS, users can have greater confidence in the answer. When it outputs FAIL, appropriate escalation to human review becomes possible. This capability enables more effective human-AI collaboration than systems that cannot identify their own errors.

Regarding broader implications, improving model reliability in reasoning tasks has positive applications in education, scientific discovery, and decision support. However, even models with high acknowledgement rates can still confidently produce errors. The present work reduces but does not eliminate the need for human oversight in high-stakes applications.

8 Conclusion

This paper introduced Bidirectional Reasoning Alignment, a training framework that combines forward chain-of-thought reasoning with backward verification through weighted Direct Preference Optimization. The key contribution is demonstrating that training models to verify reasoning, including recognizing their own errors, improves overall reliability beyond what forward-only training achieves.

On GSM8K, the hybrid approach improves accuracy by X.X percentage points while substantially increasing acknowledgement rate from XX% to XX%. These results suggest that verification training provides complementary signal that encourages more robust reasoning capabilities.

Several directions merit future investigation. Extension to other reasoning domains would establish generality of the approach. Incorporation of human-annotated verification data could improve training quality. Development of methods that use verification signals to guide iterative refinement would address the limitation that error detection does not currently lead to correction. Exploration of step-level verification for fine-grained feedback represents another promising direction.

Reproducibility. Code, configurations, and trained model weights are available at `https://github.com/MurtazaKafka/reasoning`.

References

[1] Zhi Chen et al. Bidirectional Reasoning for Improved Factual Accuracy. *arXiv preprint*, 2025.

[2] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*, 2021.

[3] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*, 2021.

[4] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient Finetuning of Quantized LLMs. *arXiv preprint arXiv:2305.14314*, 2023.

[5] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*, 2022.

[6] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *arXiv preprint arXiv:2311.05232*, 2023.

[7] Weisen Jiang et al. FOBAR: Forward-Backward Reasoning in Language Models for Verification. *arXiv preprint*, 2024.

[8] Xin Lai et al. Step-DPO: Step-wise Preference Optimization for Long-chain Reasoning of LLMs. *arXiv preprint arXiv:2406.18629*, 2024.

[9] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step. *arXiv preprint arXiv:2305.20050*, 2023.

[10] Yinger Liu et al. Reverse Chain: A Generic-Rule for LLMs to Master Multi-API Planning. *arXiv preprint*, 2024.

[11] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-Refine: Iterative Refinement with Self-Feedback. *arXiv preprint arXiv:2303.17651*, 2023.

[12] Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. Automatically Correcting Large Language Models: Surveying the Landscape of Diverse Self-Correction Strategies. *arXiv preprint arXiv:2308.03188*, 2023.

[13] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[14] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models. *arXiv preprint arXiv:2203.11171*, 2023.

[15] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[16] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *arXiv preprint arXiv:2305.10601*, 2023.

A Implementation Details

A.1 Prompt Templates

The forward reasoning prompt instructs the model to solve problems step by step with explicit reasoning:

```
You are an expert problem solver. Solve the following problem step by step. Show
your reasoning clearly, then provide the final answer.

Problem: {question}

Solution:
```

The backward verification prompt instructs the model to verify candidate answers:

```
You are a careful verifier. Given a problem and a candidate answer, verify whether
the answer is correct by reasoning backwards from the answer.

Problem: {question}

Candidate Answer: {answer}

Verify the solution step by step, then conclude with either Verification: PASS if
the answer is correct, or Verification: FAIL if the answer is incorrect.

Verification:
```

A.2 Answer Extraction

Robust answer extraction handles multiple formats commonly encountered in mathematical reasoning. The GSM8K format uses "####" followed by the numeric answer. LaTeX boxed notation uses `\boxed{...}`. Natural language expressions such as "The answer is X" or "Final Answer: X" are also recognized. Fractional answers are converted to decimal form for comparison. Numeric comparison uses tolerance $\epsilon = 10^{-6}$ for floating-point answers.

A.3 Hyperparameter Configuration

Table 3 provides complete hyperparameter settings for reproducibility.

Table 3: Complete hyperparameter configuration.

Parameter	**Value**
Base model	LLaMA 3.1 8B-Instruct
LoRA rank r	16
LoRA scaling α	32
LoRA dropout	0.05
Target modules	W_q, W_k, W_v, W_o
Learning rate	1×10^{-5}
Warmup ratio	0.05
Weight decay	0.01
Effective batch size	16
Training epochs	1
DPO β	0.1
Forward weight w_f	0.6
Backward weight w_b	0.4
Real negative boost α	1.2
Forward temperature	0.7
Backward temperature	0.3
Maximum sequence length	512

A.4 Computational Resources

All experiments were conducted on a single NVIDIA RTX A6000 GPU with 48GB memory. Training each experimental condition requires approximately one hour. Total computational cost for all experiments including ablations is approximately 20 GPU-hours.