Post-Training Large Language Models via Reinforcement Learning from Self-Feedback

Anonymous ACL submission

Abstract

Large Language Models (LLMs) often produce plausible but poorly-calibrated answers, limiting their reliability on reasoning-intensive tasks. Recent research suggests that Chain-of-Thought (CoT) reasoning paths are inherent in pre-trained LLMs and can be elicited by simply altering the decoding process, where the presence of a CoT path correlates with higher answer confidence.

Building on these insights, we present Reinforcement Learning from Self-Feedback (RLSF), a post-training stage that utilises the model's intrinsic confidence as a self-generated reward. By generating multiple CoT decoding beams from a frozen LLM, we compute the confidence of each final answer span and rank the resulting traces accordingly to create synthetic preferences. These preferences are subsequently utilised to fine-tune the policy through standard preference optimisation, requiring no human labels, gold answers, or externally curated rewards.

RLSF simultaneously (i) refines the model's probability estimates–restoring well-behaved calibration–and (ii) strengthens step-by-step reasoning, yielding improved performance on arithmetic reasoning and multiple-choice question answering. By converting a model's own uncertainty into structured self-feedback, RLSF affirms reinforcement learning on intrinsic model behaviour as a principled and data-efficient component of the LLM post-training pipeline. Our results demonstrate that leveraging these inherent reasoning capabilities provides a robust path for enhancing model reliability without manual prompt engineering or external supervision.[1]

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in text generation and comprehension (Brown et al., 2020; Ouyang et al., 2022). However, performance often suffers on tasks requiring extended logical chains, where consistency and accuracy are difficult to maintain during multi-step inference (Kambhampati, 2024). Furthermore, LLM outputs are frequently poorly calibrated, as expressed confidence rarely predicts actual accuracy (Bai et al., 2022; Tian et al., 2023; OpenAI et al., 2024a). This miscalibration leads to overconfidence in complex tasks, making it challenging to identify when a model's output is trustworthy.

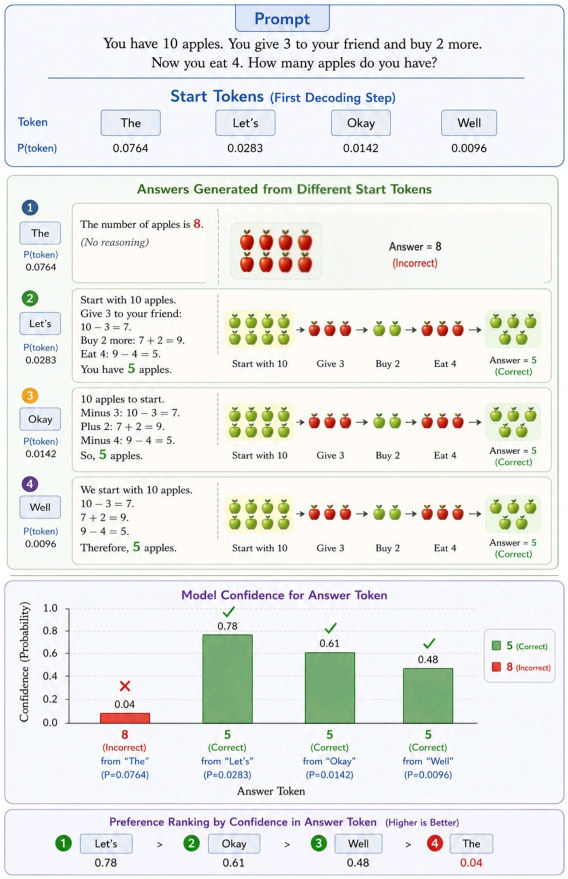

Figure 1: An overview of the RLSF pipeline.

Reliable confidence estimates are vital; a model

[1]Code will be released upon publication.

claiming 90% certainty should be correct in 90% of cases. Yet, standard post-training methods like RLHF, while improving performance, often further degrade calibration (Bai et al., 2022; Tian et al., 2023; OpenAI et al., 2024a). This paradox means that common alignment techniques can make models more helpful but simultaneously less reliable.

Inherent Reasoning and Decoding To improve reasoning, Chain-of-Thought (CoT) prompting encourages intermediate steps (Wei et al., 2022; Kojima et al., 2022), while newer architectures are specifically trained to produce reasoning traces (OpenAI et al., 2024b). Recent research indicates that CoT paths are inherent in pre-trained models and can be elicited simply by altering the decoding process without explicit prompting (Wang and Zhou, 2024). Crucially, the presence of these inherent CoT paths correlates with significantly higher confidence in the final answer. This suggests that token-level confidence is a robust proxy for reasoning quality, effectively separating sound logical paths from flawed sequences.

While CoT-decoding effectively elicits these abilities, the required inference-time compute is an order of magnitude higher than standard methods. Moreover, most existing calibration techniques rely on labelled correctness data, which severely limits their scalability (Tian et al., 2023; Kapoor et al., 2024).

Learning from Inherent Confidence We build upon the insight that inherent CoT paths are linked to higher answer confidence. Instead of using this metric only for inference-time selection, we use it as a signal to generate preference data for model alignment. By ranking generated reasoning traces based on answer span confidence, we can train a reward model that captures the characteristics of high-quality reasoning. Similarly, humans can use internal confidence as a feedback mechanism for learning in the absence of external supervision (Ptasczynski et al., 2022).

We introduce **Reinforcement Learning from Self-Feedback (RLSF)**, a post-training stage using the model's own confidence as an intrinsic reward (Figure 1). RLSF advances self-training in three ways: (1) it uses the link between decoding confidence and reasoning paths to create synthetic preferences without human labels; (2) it explicitly optimises for calibration by refining probability estimates via self-generated rewards; (3) it strengthens step-by-step reasoning through reinforcement learning, achieving the benefits of CoT-decoding at baseline inference costs.

RLSF augments standard post-training pipelines, including supervised fine-tuning and preference optimisation. While generating CoT beams for training data increases initial costs, the final model maintains a $1\times$ inference speed, making it highly efficient.

Contributions We evaluate RLSF on logical reasoning and multiple-choice tasks where accuracy and calibration are typically low. Our main contributions, confirmed in Sections 4 and 5, are:

1. We show that token-level confidence from inherent reasoning paths is an effective reward signal for improving answer-level calibration.

2. We demonstrate that RLSF provides a practical alternative to CoT-decoding by offering similar performance gains at standard inference costs.

3. We provide evidence that meaningful reward models can be derived without human labels or external demonstrations using only task-specific prompts.

4. We confirm that RLSF does not amplify safety-related biases, allowing for safe integration into standard post-training workflows.

2 Related Work

2.1 Confidence Estimation in LLMs

Well-calibrated confidence estimates are essential for the reliability of LLMs in decision-support tasks (Wang and Zhou, 2024). While supervised fine-tuning often preserves token-level calibration (Kuhn et al., 2022; Xiao et al., 2022), alignment via RLHF frequently degrades it, as reward signals prioritise human preference over calibrated uncertainty (Tian et al., 2023; OpenAI et al., 2024a). Recent work has moved from single-token metrics to span-level confidence, which better reflects the correctness of complex reasoning chains (Kojima et al., 2022; Wang and Zhou, 2024). However, existing methods to restore calibration—such as verbalised confidence (Tian et al., 2023), auxiliary predictors (Huang et al., 2025), or calibration-aware tuning (Kapoor et al., 2024; Stangel et al., 2025)—remain limited by their dependence on labelled correctness data.

2.2 Chain-of-Thought Reasoning

Chain-of-Thought (CoT) techniques improve reasoning by prompting models to articulate intermediate steps (Wei et al., 2022; Kojima et al., 2022) or through specific training on reasoning traces (Chung et al., 2024; Zelikman et al., 2022). Notably, Wang and Zhou (2024) demonstrated that CoT paths are inherent in pre-trained models and can be elicited at inference time by selecting decoding paths with the highest token-level confidence. While this CoT-decoding bypasses the need for intensive prompt engineering or task-specific fine-tuning, it incurs an order-of-magnitude increase in inference cost, limiting its practical deployment.

2.3 Preference Learning and Self-Training

Standard alignment involves reinforcement learning from human feedback (RLHF; Christiano et al., 2017; Ouyang et al., 2022) or Direct Preference Optimisation (DPO; Rafailov et al., 2023). To reduce reliance on human labels, RLAIF (Lee et al., 2023) and AutoPM (Huang et al., 2023) use AI-generated feedback to construct preference datasets. Self-training methods, such as REST (Gulcehre et al., 2023) and AfD (Sun and van der Schaar, 2025), further leverage model-generated outputs to improve task accuracy. RLSF builds on these approaches by using intrinsic confidence as a zero-shot preference signal, eliminating the need for external LLM judges or gold labels.

2.4 Intrinsic Motivation and Self-Reward

Intrinsic rewards allow reinforcement learning to proceed in environments where extrinsic feedback is sparse or absent (Chentanez et al., 2004; Barto, 2013). In the context of LLMs, this has been explored through self-generated signals where models evaluate their own outputs (Klissarov et al., 2024). Drawing from cognitive science, confidence-based learning is recognised as a fundamental mechanism for human learning without external supervision (Ptasczynski et al., 2022), suggesting a principled basis for using internal confidence as a training signal.

2.5 Bias and Safety Evaluation

Evaluating whether post-training methods amplify harmful biases is critical for responsible deployment. Benchmarks like XSTest (Röttger et al., 2024) detect exaggerated safety behaviours, while AlpacaEval (Li et al., 2023) provides scalable approximations to human evaluation using LLM-as-judge frameworks. Ensuring that RLSF maintains safety profiles while enhancing reasoning is a key component of our evaluation.

3 Methodology

Our method builds on the observation from Wang and Zhou (2024) that in well-calibrated models, answer confidence correlates with reasoning quality and accuracy. We extend this finding by using confidence as an intrinsic reward signal to train models that generate well-calibrated responses. This is formulated as a sequential decision-making process optimised via reinforcement learning to maximise the confidence of answer tokens.

Notation An auto-regressive large language model can be viewed as a policy $\pi_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$ in the reinforcement learning framework. At the initial time step, the state s_0 is the initial input (prompt or a question) $\mathtt{q}$. At each following time step t, given state s_t (the prompt and previously generated tokens), the policy selects action a_t (the next token $w \in \mathcal{V}$) according to the distribution $\pi_{\boldsymbol{\theta}}(a \mid s)$. The state transitions to $s_{t+1} = s_t \odot a_t$, where $\odot$ denotes concatenation. Policy parameters are optimised via proximal policy optimisation (PPO; Schulman et al., 2017).

3.1 Chain-of-Thought Decoding

Chain-of-Thought (CoT) decoding (Wang and Zhou, 2024) generates multiple reasoning hypotheses for a given input $\mathtt{q}$, then selects the most confident answer.

For input $\mathtt{q}$, CoT decoding samples the K top-probability tokens $w^1, \ldots, w^K$ at the first decoding step from $\pi_{\boldsymbol{\theta}}(w|\mathtt{q})$. Each token w^k seeds a hypothesis $\mathtt{h}^k$, which is then extended via greedy auto-regressive decoding:

$$\mathtt{h}^k \leftarrow \mathtt{h}^k \odot \arg\max_{w} \pi_{\boldsymbol{\theta}}(w|\mathtt{h}^k)$$

until the end-of-sequence token is generated.

Answer Span Identification To identify the answer span $\mathtt{g}^k$ within hypothesis $\mathtt{h}^k$, the text "`So the answer is`" is appended to prompt the model to generate answer tokens $\mathtt{g}'_k$. The answer span $\mathtt{g}^k$ is then located in $\mathtt{h}^k$ via string matching with $\mathtt{g}'_k$.

Confidence Disparity For an answer span $\mathtt{g}$ at position m consisting of M tokens, the confidence

disparity c is:

$$c = \frac{1}{M} \sum_{i=0}^{M-1} \Big[\underbrace{\max_{w} \pi_{\boldsymbol{\theta}}(w|s_{m+i})}_{\text{most likely token probability}} - \underbrace{\max_{w \neq \arg\max \pi_{\boldsymbol{\theta}}} \pi_{\boldsymbol{\theta}}(w|s_{m+i})}_{\text{second most likely token probability}} \Big] \quad (1)$$

where $s_{m+i} = \mathtt{q} \odot \mathtt{h}_{1:m+i}$ is the context up to the token being evaluated.

Probability disparity measures the model's certainty by computing the gap between the top two token probabilities at each position in the answer span, then averaging across all answer tokens. A large disparity (e.g., 0.9 vs 0.05 = 0.85) indicates the model strongly prefers one token; a small disparity (e.g., 0.4 vs 0.35 = 0.05) indicates uncertainty between alternatives.

Why disparity rather than raw probability? Raw token probability can be misleadingly high due to vocabulary distribution effects—a token with 0.6 probability might seem confident, but if the second-highest is 0.55, the model is actually uncertain. Disparity captures this uncertainty by considering the full distribution shape. It has been shown to be more reliable than raw probability (Wang and Zhou, 2024).

3.2 Reinforcement Learning from Self-Feedback

Given an input $\mathtt{q}$, CoT decoding produces K hypotheses $\mathtt{h}^1, \ldots, \mathtt{h}^K$, each with confidence disparity $c_1, \ldots, c_K$. We rank these hypotheses by confidence to construct a preference dataset $\mathcal{D}$, where higher-confidence responses are treated as preferred over lower-confidence ones.

RLSF with PPO We train a reward model R_ϕ on $\mathcal{D}$ using the Bradley-Terry preference framework (Bradley and Terry, 1952), then optimise the policy $\pi_{\boldsymbol{\theta}}$ via PPO (see Appendix A for details). This approach, which we term RLSF with PPO, follows the standard RLHF pipeline but uses self-generated preferences rather than human annotations.

RLSF with DPO Alternatively, $\mathcal{D}$ can be used to directly fine-tune the model via Direct Preference Optimisation (DPO; Rafailov et al., 2023), eliminating the separate reward model. We term this RLSF with DPO, though strictly speaking it does not involve reinforcement learning.

Computational Cost Training RLSF requires generating K CoT beams per prompt to construct $\mathcal{D}$, incurring K-fold cost during the preference data collection phase. However, once trained, the RLSF model performs standard greedy decoding at inference time with no additional overhead—unlike CoT-decoding which maintains K-fold cost at inference.

Figure 1 illustrates the complete RLSF pipeline.

4 Experimental Setup

We evaluate RLSF on mathematical reasoning and multiple-choice question answering tasks to assess its impact on both calibration and accuracy.

4.1 Datasets

Mathematical Reasoning We use Multi-Arith and GSM8K (Cobbe et al., 2021), both containing arithmetic word problems requiring multistep computation and reasoning. Prompts consist of the question only, with no additional formatting.

Multiple-Choice Question Answering We use CommonsenseQA (Talmor et al., 2019) and ARC Easy (Clark et al., 2018), which test common-sense and elementary-level reasoning in a constrained answer format. Prompts contain the question followed by a list of possible answers.

Reward Model Evaluation We evaluate reward models on the RewardBench (Lambert et al., 2024) mathematical reasoning subset. Each instance contains two AI-generated responses to the same question, labelled as preferred or rejected by human annotators. We use only the mathematical reasoning subset for models trained on mathematical tasks.

4.2 Models

We evaluate RLSF across models with varying calibration quality to assess robustness:

Multiple-choice experiments:

- Phi-2 (Hughes, 2023) – a poorly-calibrated model on CommonsenseQA
- Gemma 2 2B IT (Riviere et al., 2024) – a well-calibrated model on ARC Easy

Mathematical reasoning experiments:

- Gemma 2 2B IT (Riviere et al., 2024) – well-calibrated baseline

Base Model	Further optimisation	Reward Model	Prompting Strategy	Decoding Strategy	Accessible Reward Model	MultiArith		GSM8K	
						Accuracy ↑	ECE ↓	Accuracy ↑	ECE ↓
QWEN 2.5 7B	None	DeepSeek	Default	Greedy	N	**79.44**	20.75	53.67	42.86
	None	DeepSeek	Default	CoT (10)	N	77.77	22.22	**58.52**	**41.46**
	None	DeepSeek	CoT	Greedy	N	67.77	31.61	44.73	50.23
	RLHF(PPO)	URM	Default	Greedy	Y	74.13	23.24	46.82	51.24
	RLSF(DPO)	Ours	Default	Greedy	Y	71.62	25.12	50.73	38.15
	RLSF(PPO)	Ours	Default	Greedy	Y	78.06	**18.35**	**58.62**	**41.92**
Gemma 2 2B	None	Gemma	Default	Greedy	N	98.12	7.43	85.57	12.24
	None	Gemma	Default	CoT (10)	N	**99.01**	**4.12**	**89.18**	**10.94**
	None	Gemma	CoT	Greedy	N	98.08	7.18	85.61	12.83
	RLHF(PPO)	URM	Default	Greedy	Y	97.83	12.73	82.43	17.83
	RLSF(DPO)	Ours	Default	Greedy	Y	96.13	10.52	84.74	17.43
	RLSF(PPO)	Ours	Default	Greedy	Y	**98.83**	7.81	88.14	12.54

Table 1: Comparison of RLSF variants and baselines on MultiArith and GSM8K. *Accuracy* and *Estimated Calibration Error (ECE)* are reported across models, decoding strategies, and training paradigms. All models are based on the RLHF by PPO variants of the base model.

- QWEN 2.5 7B (DeepSeek-AI et al., 2025) – fine-tuned with DeepSeek R1 distillation

Model selection deliberately spans well-calibrated and poorly-calibrated cases to test RLSF's effectiveness under different initial conditions.

4.3 Metrics

Answer Accuracy The percentage of model responses that exactly match correct answers, assessed via numeric equivalence for arithmetic tasks and exact string matching for multiple-choice questions.

Expected Calibration Error (ECE) Measures alignment between confidence and correctness by binning predictions by confidence level and computing the weighted average of absolute differences between confidence and accuracy per bin. For arithmetic tasks, ECE is computed on answer span tokens; for multiple-choice, on the choice token. Lower ECE indicates better calibration.

Reward Model Accuracy On RewardBench, the percentage of instance pairs where the reward model correctly assigns a higher score to the human-preferred response. This evaluates whether RLSF-derived reward models capture meaningful quality signals despite lacking access to ground-truth labels or preferences.

4.4 Implementation Details

Preference Data Construction For each training prompt, we generate $K = 10$ candidate responses via CoT decoding (Section 3.1). Candidates are ranked by answer confidence disparity to form preference pairs for training.

Reward Model Training Each reward model is initialized from the corresponding base LLM with a scalar regression head attached. The model is fine-tuned using Low-Rank Adaptation (LoRA; Hu et al., 2021) via the TRL library (von Werra et al., 2020). Raw reward values are linearly rescaled to $[-1, 1]$ for stable training dynamics.

Policy Optimization For RLSF with PPO, we optimize the base model using the trained reward model following standard RLHF procedures. For RLSF with DPO, we directly optimize the base model on the preference dataset using the DPO loss. Detailed hyperparameters are provided in Appendix B.

Baselines We compare RLSF against:

- **Base model with greedy decoding** – Standard inference baseline
- **CoT prompting** – Prepending "Let's think step by step" to prompts
- **CoT decoding (K=10)** – Generating 10 reasoning paths and selecting the highest-confidence answer (Wang and Zhou, 2024)
- **RLHF with URM** – Training with the state-of-the-art URM reward model (Lou et al., 2025)

All experiments use greedy decoding at inference unless otherwise specified. We do not use sampling-based decoding (temperature, top-p) as our focus is calibration rather than maximum task accuracy, and preliminary experiments showed similar calibration patterns across decoding strategies.

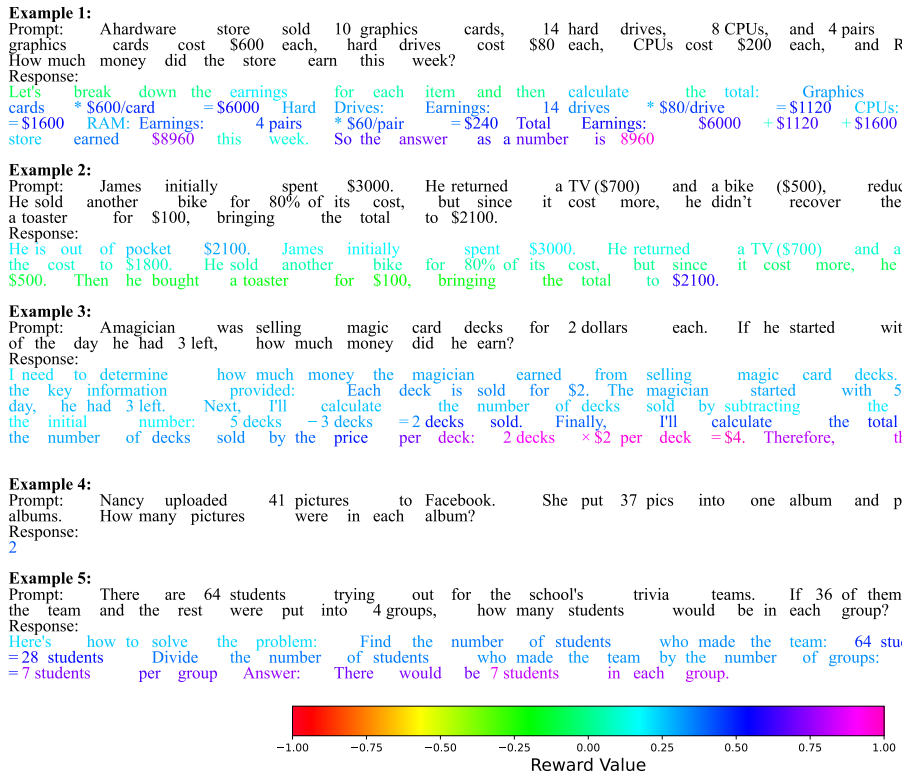

Figure 2: Example responses from the RLSF fine-tuned Gemma 2 model. The response words are colored based on the rewards obtained from the RLSF reward model.

5 Results

We first evaluate RLSF's impact on calibration and accuracy (Sections 5.1 and 5.2), then validate that RLSF-derived reward models capture meaningful quality signals (Section 5.3) and finally assess bias amplification (Section 5.4).

5.1 Mathematical Reasoning Tasks

Table 1 compares RLSF against baselines on MultiArith and GSM8K. We evaluate two base models: QWEN 2.5 7B (fine-tuned with DeepSeek R1 distillation) and Gemma 2 2B (instruction-tuned). Baselines include CoT prompting, CoT decoding (K=10), and RLHF with URM.

QWEN 2.5 7B (Poorly-Calibrated Base) RLSF with PPO achieves the best GSM8K accuracy (58.62%, matching CoT decoding) and lowest MultiArith ECE (18.35%). URM underperforms RLSF on both metrics despite higher RewardBench scores. CoT prompting performs worst, confirming that inference-time prompting alone is ineffective (Saparov and He, 2022). RLSF with DPO improves calibration modestly but underperforms the PPO variant.

Gemma 2 2B (Well-Calibrated Base) CoT decoding achieves highest accuracy (99.01% MultiArith, 89.18% GSM8K) and best calibration (4.12% ECE MultiArith). RLSF with PPO ranks second (98.83% acc, 7.81% ECE) but eliminates the K-fold inference cost. This represents a calibration-efficiency trade-off: RLSF trades slight calibration degradation (7.81% vs 4.12% ECE) for order-of-magnitude inference speedup.

When Does RLSF Help? Results reveal a clear pattern: RLSF provides substantial improvements for poorly-calibrated models (QWEN: 53.67%→58.62% acc, 42.86%→41.92% ECE on GSM8K) but offers primarily computational benefits for well-calibrated models (Gemma 2). RLSF is most valuable when: (1) base calibration is degraded post-RLHF, (2) K-fold decoding cost is prohibitive, or (3) calibration is prioritised over maximum accuracy.

Reward Model Visualization Figure 2 shows token-level rewards from the RLSF reward model. Correct answers receive high rewards, with intermediate reasoning steps also rewarded (Examples 1, 5). Verbose explanations receive lower scores (Example 3), indicating learned preference for con-

Model	Decoding Strategy	Commonsense QA		ARC Easy	
		Accuracy ↑	ECE ↓	Accuracy ↑	ECE ↓
Phi-2	Greedy	54.46	25.12	85.18	27.03
	CoT (10)	58.91	23.11	86.10	24.83
Gemma 2	Greedy	79.6	19.01	96.96	16.12
	CoT (10)	80.03	15.49	96.28	3.03

Table 2: Comparison of decoding strategies for Phi-2 and Gemma 2 9B IT across multiple-choice tasks.

Training Strategy	Decoding Strategy	Accuracy ↑	ECE ↓
No	Greedy	96.96	16.12
No	CoT (10)	96.28	**3.03**
RLSF (DPO)	Greedy	**97.05**	18.83
RLSF (PPO)	Greedy	**97.04**	5.12

Table 3: Performance on ARC Easy using Gemma 2 under different training and decoding strategies.

ciseness. The model sometimes exhibits post-hoc rationalization (Example 2), suggesting imperfect reasoning.

Figure 3 demonstrates calibration improvement: both pre- and post-RLSF models answer incorrectly, but RLSF reduces overconfidence (0.993→0.712) and increases probability mass on the correct answer (0.004→0.138).

5.2 Multiple-Choice Tasks

Table 2 shows CoT decoding improves calibration across models, confirming prior findings (Wang and Zhou, 2024). We select Gemma 2 (well-calibrated on ARC Easy) and Phi-2 (poorly-calibrated on CommonsenseQA) for detailed evaluation.

Results (Tables 3, 5) mirror mathematical reasoning findings: RLSF substantially improves poorly-calibrated models (Phi-2: 54.46%→61.13% acc, 25.12%→19.64% ECE) whilst offering modest gains for well-calibrated models (Gemma 2: comparable accuracy, worse calibration than CoT decoding but at 1× inference cost).

5.3 Reward Model Evaluation

Table 4 compares RLSF reward models against methods using human preference labels. URM (Lou et al., 2025) and QRM (Dorka, 2024) achieve 97% accuracy using both prompts and preferences. AfD (Sun and van der Schaar, 2025) uses prompts and demonstration answers. RLSF uses prompts alone, achieving 76–81% accuracy.

This 16–21 point gap reflects the inherent limitation of self-supervised methods: without ground-truth preferences, RLSF relies entirely on base model confidence. However, RLSF shows lower variance across base models (76% vs 81%) compared to AfD (73% vs 89%), suggesting confidence disparity provides a more robust signal than output imitation. Critically, as shown in Table 1, RewardBench scores do not predict downstream utility: URM (97% on RewardBench) underperforms RLSF on actual tasks (74.13% acc vs 78.06% acc on QWEN MultiArith).

Reward Model	Prompt	Answer	Preference	Accuracy ↑
URM LLaMa 3.1 8B	Y	N	Y	97.00
QRM LLaMa 3.1 8B	Y	N	Y	96.80
QWEN 2.5 7B AfD	Y	Y	N	89.29
GEMMA 2 2B AfD	Y	Y	N	73.12
QWEN 2.5 7B RLSF	Y	N	N	76.13
GEMMA 2 2B RLSF	Y	N	N	81.43

Table 4: Reward model accuracy on RewardBench (Math Reasoning). Columns indicate which components (Prompt, Answer, Preference) were used as input to the reward model.

Prompt

You have to do a lot of thinking while studying a new subject, but it is how you gain what?
Select one of the following:
A. fatigue
B. depression
C. best way
D. weight
E. knowledge

Model Output (Before RLSF)

A. fatigue Gaining knowledge often requires mental effort and can lead to fatigue, as you're putting in the time and energy to understand and process new information.
Predicted: A, Probability: 0.993
Golden Answer: E, Probability: 0.004

Model Output (After RLSF)

A. fatigue When studying something new, the mental exertion involved can cause fatigue. While the ultimate goal is to gain knowledge, the immediate effect of extended cognitive effort can be tiredness.
Predicted: A, Probability: 0.712
Golden Answer: E, Probability: 0.138

Figure 3: Example of calibration improvement via RLSF. Both responses incorrectly answer "A" instead of the correct answer "E", but the post-RLSF model produces a better explanation and expresses lower confidence.

Training Strategy	Decoding Strategy	Accuracy ↑	ECE ↓
No	Greedy	54.46	25.12
No	CoT (10)	58.91	23.11
RLSF (DPO)	Greedy	59.79	30.91
RLSF (PPO)	Greedy	**61.13**	**19.64**

Table 5: Performance on Commonsense QA using Phi-2 under various training strategies.

Training Dataset	Evaluation Dataset	Preference (%) ↑
GSM8K + MultiArith	XSTest	50.73
CommonsenseQA	XSTest	51.82
XSTest	XSTest	63.24

Table 6: AlpacaEval win rates on the XSTest dataset.

5.4 Bias Evaluation

Table 6 evaluates whether RLSF amplifies biases using XSTest prompts (Röttger et al., 2024) with GPT-4o as judge (Li et al., 2023). Models trained on task-specific data (GSM8K, CommonsenseQA) show 50% preference versus base models, indicating no bias amplification. When trained directly on XSTest, RLSF reinforces cautious tendencies (63.24% preference), suggesting safety improvements.

6 Discussion and Future Work

RLSF adopts a self-supervised paradigm similar to recent self-training methods but differs in three key aspects. First, while REST and self-consistency rely on discrete signals like heuristic filters or majority voting, RLSF utilises token-level probability disparity as a continuous signal for gradient-based calibration optimisation. Second, RLSF explicitly targets calibration alongside task accuracy; on already well-calibrated models such as Gemma 2, it prioritises inference efficiency over marginal accuracy gains. Third, RLSF uses RL to learn the reasoning process that generates confident answers rather than merely filtering outputs. This enables calibration improvements even in poorly-calibrated base models, where simply selecting confident outputs would otherwise amplify existing errors.

The primary trade-off of RLSF is the increased training cost incurred by K-fold CoT beam generation for preference data construction. However, once trained, the model performs standard greedy decoding at inference with no overhead, unlike CoT-decoding which maintains K-fold costs at deployment. Consequently, RLSF is particularly valuable for resource-constrained environments that require the reliability of confidence-based decoding without the associated latency.

As a self-supervised step, RLSF augments existing post-training pipelines (Kumar et al., 2025) but does not replace RLHF. It refines inherent behaviours rather than introducing fundamentally new capabilities or correcting deep-seated pre-training biases. Our evaluation (Section 5.4) indicates that RLSF does not amplify safety-related biases, suggesting it can be safely integrated into production workflows.

Future Work Future research could explore the iterative application of RLSF, where the model's improved calibration is used to generate even higher-quality preference data in successive rounds. Additionally, while we focused on arithmetic and multiple-choice tasks, investigating RLSF in more open-ended creative or technical writing domains could reveal how intrinsic confidence scales with subjective quality. Finally, combining RLSF with extrinsic rewards in a hybrid objective may allow for simultaneous optimisaton of human alignment and mathematical calibration.

7 Conclusion

We propose Reinforcement Learning from Self-Feedback (RLSF), a post-training method that improves LLM calibration by using token-level confidence disparity as an intrinsic reward signal. Unlike existing approaches requiring human labels or demonstrations, RLSF derives preferences solely from prompts via Chain-of-Thought decoding.

Experiments on mathematical reasoning (MultiArith, GSM8K) and multiple-choice tasks (CommonsenseQA, ARC Easy) demonstrate that RLSF improves calibration without harming accuracy. For poorly-calibrated models, RLSF provides substantial gains in both metrics. For well-calibrated models, RLSF offers a calibration-efficiency trade-off: comparable performance to inference-time aggregation methods but at standard inference cost.

RLSF with PPO outperforms DPO, demonstrating that reinforcement learning effectively incorporates intrinsic motivation for calibration. The method does not amplify biases when applied to task-specific domains, suggesting safe integration into production pipelines.

Limitations

While RLSF demonstrates significant improvements in calibration and reasoning efficiency, sev-

eral limitations and associated risks remain. First, RLSF primarily addresses single-step reasoning trajectories where the model generates a complete chain of thought before reaching a final answer. Extending this framework to tasks requiring multi-turn interaction or long-horizon planning remains future work, although intrinsic rewards have shown promise in augmenting long-term RL methods (Zhou et al., 2024) and goal-oriented dialogue (Wesselmann et al., 2018).

Second, the construction of preference data relies on the existence of inherent reasoning paths within the base model. If a model is severely undertrained or lacks the latent capability to generate coherent reasoning steps, the token-level confidence signal may be too noisy to form a meaningful reward. Furthermore, while our experiments span diverse model sizes and initial calibration qualities, our evaluation was conducted on models with up to 9B parameters. Validating RLSF on frontier models (exceeding 70B parameters) and even more complex reasoning benchmarks would be necessary to fully confirm its generalisability across the scaling curve.

Third, RLSF is a self-supervised method that refines a model's internal consistency; it does not introduce external "ground truth" knowledge. Consequently, if a model is confidently incorrect due to factual gaps in its pre-training data, RLSF may reinforce these errors rather than correcting them. This suggests that RLSF is best utilised as a refinement stage for reasoning logic rather than a mechanism for knowledge acquisition.

Finally, relying entirely on intrinsic confidence as a reward introduces specific alignment risks. Optimising a policy against its own internal probability distributions can create a closed-loop feedback system that risks over-fitting to the model's idiosyncratic biases or style preferences. While our evaluation shows no amplification of safety-related biases in task-specific domains, deploying an unconstrained self-rewarding loop could potentially cause the model to drift from human intent over multiple iterations, generating highly confident but unaligned or sycophantic reasoning paths.

References

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, and 32 others. 2022. Constitutional AI: Harmlessness from AI Feedback. *arXiv preprint*. ArXiv:2212.08073 [cs].

Andrew G. Barto. 2013. Intrinsic motivation and reinforcement learning. In Gianluca Baldassarre and Marco Mirolli, editors, *Intrinsically motivated learning in natural and artificial systems*, pages 17–47. Springer Science+Business Media.

Ralph Allan Bradley and Milton E. Terry. 1952. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, 39(3/4):324–345. Publisher: [Oxford University Press, Biometrika Trust].

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nuttapong Chentanez, Andrew Barto, and Satinder Singh. 2004. Intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 17. MIT Press.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, and 16 others. 2024. Scaling Instruction-Finetuned Language Models. *Journal of Machine Learning Research*, 25(70):1–53.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv preprint*. ArXiv:1803.05457 [cs].

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. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint*. ArXiv:2110.14168 [cs].

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang,

Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint*. ArXiv:2501.12948 [cs].

Nicolai Dorka. 2024. Quantile Regression for Distributional Reward Models in RLHF. *arXiv preprint*. ArXiv:2409.10164 [cs].

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. Reinforced Self-Training (ReST) for Language Modeling. *arXiv preprint*. ArXiv:2308.08998 [cs].

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models.

Liangjie Huang, Dawei Li, Huan Liu, and Lu Cheng. 2025. Beyond Accuracy: The Role of Calibration in Self-Improving Large Language Models. *arXiv preprint*. ArXiv:2504.02902 [cs] version: 1.

Shijia Huang, Jianqiao Zhao, Yanyang Li, and Liwei Wang. 2023. Learning Preference Model for LLMs via Automatic Preference Data Generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9187–9199, Singapore. Association for Computational Linguistics.

Alyssa Hughes. 2023. Phi-2: The surprising power of small language models.

Subbarao Kambhampati. 2024. Can large language models reason and plan? *Annals of the New York Academy of Sciences*, 1534(1):15–18. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/nyas.15125.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Arka Pal, Samuel Dooley, Micah Goldblum, and Andrew Wilson. 2024. Calibration-Tuning: Teaching Large Language Models to Know What They Don't Know. In *Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024)*, pages 1–14, St Julians, Malta. Association for Computational Linguistics.

Martin Klissarov, Pierluca D'Oro, Shagun Sodhani, Roberta Raileanu, Pierre-Luc Bacon, Pascal Vincent, Amy Zhang, and Mikael Henaff. 2024. Motif: Intrinsic motivation from artificial intelligence feedback. In *The Twelfth International Conference on Learning Representations*.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2022. Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation.

Komal Kumar, Tajamul Ashraf, Omkar Thawakar, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, Phillip H. S. Torr, Fahad Shahbaz Khan, and Salman Khan. 2025. LLM Post-Training: A Deep Dive into Reasoning Large Language Models. *Preprint*, arXiv:2502.21321.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2024. RewardBench: Evaluating Reward Models for Language Modeling. *Preprint*, arXiv:2403.13787.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Ren Lu, Thomas Mesnard, Johan Ferret, Colton Bishop, Ethan Hall, Victor Carbune, and Abhinav Rastogi. 2023. RLAIF: Scaling Reinforcement Learning from Human Feedback with AI Feedback.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. AlpacaEval: An Automatic Evaluator of Instruction-following Models.

Xingzhou Lou, Dong Yan, Wei Shen, Yuzi Yan, Jian Xie, and Junge Zhang. 2025. Uncertainty-aware Reward Model: Teaching Reward Models to Know What is Unknown. *arXiv preprint*. ArXiv:2410.00847 [cs].

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024a. GPT-4 Technical Report. *arXiv preprint*. ArXiv:2303.08774 [cs].

OpenAI, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, and 243 others. 2024b. OpenAI *o1* System Card. *Preprint*, arXiv:2412.16720.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll 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 *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744.

Lena Esther Ptasczynski, Isa Steinecker, Philipp Sterzer, and Matthias Guggenmos. 2022. The value of confidence: Confidence prediction errors drive value-based learning in the absence of external feedback. *PLOS Computational Biology*, 18(10):1–25.

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*, volume 36, pages 53728–53741.

Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, and 178 others. 2024. Gemma 2: Improving Open Language Models at a Practical Size. *arXiv preprint*. ArXiv:2408.00118 [cs].

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, Mexico City, Mexico. Association for Computational Linguistics.

Abulhair Saparov and He He. 2022. Language Models Are Greedy Reasoners: A Systematic Formal Analysis of Chain-of-Thought.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv preprint*. ArXiv:1707.06347 [cs].

Paul Stangel, David Bani-Harouni, Chantal Pellegrini, Ege Özsoy, Kamilia Zaripova, Matthias Keicher, and Nassir Navab. 2025. Rewarding Doubt: A Reinforcement Learning Approach to Confidence Calibration of Large Language Models. *arXiv preprint arXiv:2503.02623*.

Hao Sun and Mihaela van der Schaar. 2025. Inverse-RLignment: Large Language Model Alignment from Demonstrations through Inverse Reinforcement Learning. *arXiv preprint*. ArXiv:2405.15624 [cs].

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'99, pages 1057–1063, Cambridge, MA, USA. MIT Press.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. 2023. Just Ask for Calibration: Strategies for Eliciting Calibrated Confidence Scores from Language Models Fine-Tuned with Human Feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442, Singapore. Association for Computational Linguistics.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-Thought Reasoning Without Prompting. *arXiv preprint*. ArXiv:2402.10200 [cs].

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.

Paula Wesselmann, Yen-Chen Wu, and Milica Gašič. 2018. Curiosity-driven reinforcement learning for dialogue management. In *International Conference on Acoustics, Speech, and Signal Processing*.

Yuxin Xiao, Paul Pu Liang, Umang Bhatt, Willie Neiswanger, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2022. Uncertainty Quantification with Pre-trained Language Models: A Large-Scale Empirical Analysis. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7273–7284, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. STaR: Bootstrapping Reasoning With Reasoning. In *Advances in Neural Information Processing Systems*, volume 35, pages 15476–15488.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Aviral Kumar, and Sergey Levine. 2024. ArCHer: Training Language Model Agents via Hierarchical Multi-Turn RL.

A Reinforcement Learning for LLMs

The goal in RL-optimization is to find the parameters θ that maximise the expected discounted return starting from the initial state under the policy π_θ:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \sum_{t=1}^{T} \gamma^t r_t, \tag{2}$$

where T signifies a point where the chosen token is the end-of-sequence token. The policy gradient theorem (Sutton et al., 1999) provides the formulation of gradient for π_θ

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) A_{\pi_\theta}(s_t, a_t) \tag{3}$$

where $A_\pi(s,a)$ is the advantage function for policy π. Instead of directly working with the above gradient, PPO uses a gradient of the surrogate loss, which approximates the direction of the gradient above. The loss applies clipping and a KL penalty to prevent the small changes in the parameters θ from having unexpected changes on the policy π_θ. For full details, please refer to (Schulman et al., 2017) and Appendix A.3.

A.1 Bradley-Terry reward model

When the policy choses the end of sequence token the system receives the reward from the reward model $r_T = R_\phi(s_{T-1} \odot a_{T-1})$, where s_{T-1} is the current context and a_{T-1} is the end of sequence token. $R_\phi(s)$ is another language model optimised with the Bradley-Terry (Bradley and Terry, 1952) loss

$$\mathcal{L}(h^1, h^2; \phi) = -\mathbb{E}_{\{\mathtt{q}, h^1, h^2\}} \log(\sigma(R_\phi(\mathtt{q} \odot h^1) - R_\phi(\mathtt{q} \odot h^2))), \tag{4}$$

where hypothesis h^1 is preferred over hypothesis h^2 for query q. Since the reward model is utilised only in the ultimate step, for its estimation, only the complete sequences are used.

A.2 Direct Preference Optimisation

Given a preference dataset and assuming the Bradley-Terry preference model, the policy can be directly optimised with the *direct preference optimisation* (DPO) loss given by

$$\mathcal{L}(h^1, h^2; \theta) = -\mathbb{E}_{\{\mathtt{q}, h^1, h^2\}} \Big[\log \sigma \Big(\beta \log \frac{\pi_\theta(h^1|\mathtt{q})}{\pi_{\text{ref}}(h^1|\mathtt{q})} - \beta \log \frac{\pi_\theta(h^2|\mathtt{q})}{\pi_{\text{ref}}(h^2|\mathtt{q})} \Big) \Big] \tag{5}$$

where for a query, q hypothesis h^1 is preferred over hypothesis h^2, $\pi(h|\mathtt{q}) \approx \pi(h_T|\mathtt{q} \odot h_{0:T-1})$ and π_{ref} is the reference LLM before DPO is applied. DPO bypasses the need to first learn the reward model and then perform reinforcement learning. Note, however, that discounting of the reward does not feature in DPO.

Direct preference optimisation bypasses the need for reinforcement learning by performing supervised learning with the objective given in Eq 5. Practically, it is still beneficial to perform some label smoothing to have better-calibrated responses.

A.3 Proximal policy optimisation

The advantage function $A_\pi(s,a)$ denotes the difference between the Q-function $Q_\pi(s,a)$, which is the expected return when taking action a in state s and from then on following the policy π and the value function $V_\pi(s)$, which is the expected return of state s for policy π:

$$\begin{aligned} A_\pi(s,a) &= Q_\pi(s,a) - V_\pi(s) \\ Q_\pi(s,a) &= \mathbb{E}_\pi \sum_{t=0}^{T-1} \gamma^l r_{t+1} | s_t = s, a_t = a \\ V_\pi(s) &= \mathbb{E}_\pi \sum_{l=0}^{T-t} \gamma^l r_t | s_t = s \end{aligned} \tag{6}$$

The surrogate loss, consisting of the clipping term and the KL-penalty term, is given by:

$$\begin{aligned} \mathcal{L}^{\text{CLIP+KL}}(\theta) = \mathbb{E}_t \Big[\min \Big(& \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t, \\ & \text{clip}\left(\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}, 1-\epsilon,\ 1+\epsilon \right) \hat{A}_t \Big) \\ & - \beta \cdot \sum_a \pi_{\theta_{\text{old}}}(a \mid s_t) \log \left(\frac{\pi_{\theta_{\text{old}}}(a \mid s_t)}{\pi_\theta(a \mid s_t)} \right) \Big], \end{aligned} \tag{7}$$

where $\text{clip}(x, a, b)$ ensures that x is within the interval $[a, b]$. The gradient of the surrogate loss (Eq 7) approximates the direction of the policy gradient (Eq 3).

To estimate the advantage function $\hat{A}_t$ needed for the surrogate loss, generalised advantage estimation is used:

$$\begin{aligned} \hat{A}_t &= \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \delta_{t+l} \quad \text{where} \\ \delta_t &= r_t + \gamma V_\pi(s_{t+1}) - V_\pi(s_t) \end{aligned} \tag{8}$$

Parameter $\lambda \in [0, 1]$ trades off bias (lower λ) and variance (higher λ). V_π is initialised with the reward model R_ϕ for complete sequences, and parameters ϕ are updated in such a way so that V_π estimates observed returns for partial sequences. In this way, the critic parameterised with ϕ influences the actor parameterised with θ.

PPO is an online learning algorithm, which means it needs to adopt exploration during the process of optimisation. That is achieved by applying a temperature parameter to the softmax layer of policy π_θ to induce more varied responses.

B Training Configurations

Here, we give relevant configurations to reproduce the results presented in this paper. The PPO configurations are given in Table 7 and are used for every RLSF with PPO set-up. The DPO configurations are given in Table 8 and are used in every RLSF with DPO configuration as well as for AfD. The supervised fine-tuning is used for AfD (Table 9).

Parameter	**Value**
Learning rate	5e-5
Epochs	5
Temperature for exploration	0.7
KL coefficient β	0.05
PPO clipping range ϵ	0.2
Discount factor γ	0.98
General advantage estimation λ	0.95

Table 7: PPO Training Hyperparameters

Parameter	**Value**
Learning rate	5e-5
Epochs	5
Label smoothing	0.01
DPO β	0.2

Table 8: DPO Training Hyperparameters

Parameter	**Value**
Learning rate	5e-5
Epochs	5

Table 9: Supervised Fine-tuning Hyperparameters

C Impact of the Discount Factor γ

See Table 10 for a comparison of different discount factors in PPO within our RLSF framework.

Variant	**MultiArith**		**GSM8K**	
	Accuracy ↑	ECE ↓	Accuracy ↑	ECE ↓
DPO	96.13	10.52	84.74	17.43
PPO ($\gamma = 1.0$)	98.52	8.12	87.13	12.49
PPO ($\gamma = 0.98$)	98.83	7.81	**88.14**	12.54

Table 10: Discount factor γ. Gemma 2 is fine-tuned using RLSF with PPO.