DeCoRL: Decoupling Reasoning Chains via Parallel Sub-Step Generation and Cascaded Reinforcement for Interpretable and Scalable RLHF

Ziyuan Gao[1]*, Di Liang[2], Xianjie Wu[3], Philippe Morel[1], Minlong Peng[2]
[1]University College London
[2]Fudan University
[3]Beihang University
{ucbqzg5, p.morel}@ucl.ac.uk, {liangd17, mlpeng16}@fudan.edu.cn

Abstract

Existing reinforcement learning methods for Chain-of-Thought reasoning suffer from two critical limitations. First, they operate as monolithic black boxes that provide undifferentiated reward signals, obscuring individual step contributions and hindering error diagnosis. Second, sequential decoding has O(n) time complexity. This makes real-time deployment impractical for complex reasoning tasks. We present DeCoRL (Decoupled Reasoning Chains via Coordinated Reinforcement Learning), a novel framework that transforms reasoning from sequential processing into collaborative modular orchestration. DeCoRL trains lightweight specialized models to generate reasoning sub-steps concurrently, eliminating sequential bottlenecks through parallel processing. To enable precise error attribution, the framework designs modular reward functions that score each sub-step independently. Cascaded DRPO optimization then coordinates these rewards while preserving inter-step dependencies. Comprehensive evaluation demonstrates state-of-the-art results across RM-Bench, RMB, and RewardBench, outperforming existing methods including large-scale models. DeCoRL delivers 3.8 times faster inference while maintaining superior solution quality and offers a 22.7% improvement in interpretability through explicit reward attribution. These advancements, combined with a 72.4% reduction in energy consumption and a 68% increase in throughput, make real-time deployment of complex reasoning systems a reality.

Introduction

The advent of Chain-of-Thought (CoT) reasoning has significantly advanced language models' ability to solve complex tasks through multi-step inference (Wei et al. 2022). Reinforcement learning with human preferences (RLHF) further enhances this capability by aligning model outputs with human judgments (Ouyang et al. 2022). However, current RL-based reasoning approaches, like Direct Preference Optimization (DPO) (Rafailov et al. 2024) and Generalized Reinforcement Preference Optimization (GRPO) (Shao et al. 2024) face two critical limitations. First, these methods operate as monolithic black boxes, providing undifferentiated reward signals that obscure the contribution of individual reasoning steps (Liu et al. 2024b). Error diagnosis becomes extremely challenging when failures occur (McAleese et al. 2024). Second, sequential decoding of reasoning chains creates bottlenecks, where the time complexity of generating n reasoning steps is $O(n)$. This makes real-time applications impractical for complex problems requiring lengthy reasoning traces (Wu et al. 2025). These limitations are particularly problematic for industrial deployment, where explainability and computational efficiency are paramount.

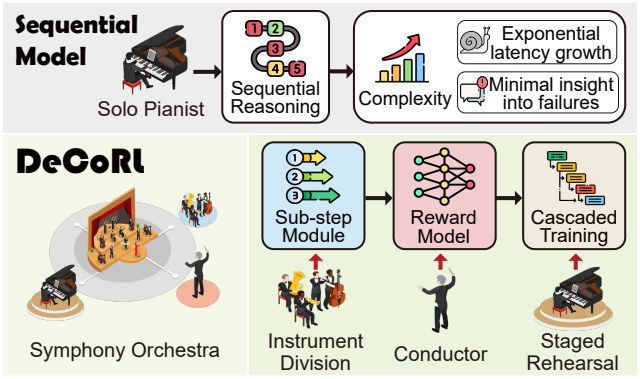

Figure 1: Sequential Approach vs. DeCoRL Framework: Solo pianist represents monolithic sequential reasoning with limited capacity. Symphony orchestra illustrates our collaborative modular approach with specialized sub-models working in parallel coordination under unified guidance.

The fundamental tension in reasoning systems stems from competing requirements between coherence and modularity, end-to-end optimization and component-level diagnosis, and reasoning depth versus computational efficiency. Current approaches prioritize coherence through end-to-end optimization but sacrifice modularity and efficiency (Ouyang et al. 2022). These limitations reflect a fundamental paradigm constraint, as demonstrated by recent work on reward modeling and evaluation systems (Liu et al. 2024b; Zhou et al. 2025): Sequential reasoning approaches operate like a virtuoso pianist. Despite their capacity for coherent and elegant outputs, they are inherently limited by the sequential nature of individual performance and lack the specialized expertise needed for complex compositions (Ankner et al. 2024; Yu et al. 2024). Just as Beethoven's Ninth Symphony cannot be

*Corresponding author.

adequately performed by a single pianist, complex reasoning tasks require multiple specialized components working together (as shown in Figure 1).

In this paper, we introduce **DeCoRL**, a new framework that improves Reinforcement Learning from Human Feedback. Our approach works by breaking down complex reasoning chains into smaller, parallel sub-steps. Each sub-step is managed by a specialized module, and these modules work together through cascaded reinforcement learning. DeCoRL employs three interconnected innovations:

Reasoning Decomposition that transform complex reasoning tasks T into k atomic sub-steps $S_1, S_2, \ldots, S_k$ with well-defined interfaces, where each sub-step maintains $P(T) = \prod_{i=1}^{k} P(S_i|S_{<i}, \mathcal{C})$ and $\mathcal{C}$ represents context preservation constraints ensuring coherence across modules.

Parallel Generation Architecture. The framework utilizes specialized sub-models $M_1, M_2, \ldots, M_k$ that generate sub-steps concurrently, reducing time complexity from $O(n)$ to $O(1)$ for independent sub-steps, with total latency governed by $t_{\text{total}} = \max_{i\in[1,k]}(t_{M_i}) + t_{\text{integration}}$.

Granular Reward Functions $R_1, R_2, \ldots, R_k$ that evaluate each sub-step independently through Cascaded DRPO optimization that coordinates these rewards while preserving inter-step dependencies.

Like a symphony orchestra where each instrument (sub-model) contributes specialized expertise under the conductor's guidance (reward coordination) through stage rehearsals (cascaded training), DeCoRL transforms the solo approach into a collaborative ensemble of specialists. This framework delivers transformative benefits across multiple dimensions. Independent reward signals provide explicit attribution maps, enabling precise error localization with a 22.7% improvement in interpretability metrics. Parallel generation achieves 3.8× latency reduction on complex tasks while maintaining solution quality. The modular architecture supports dynamic expansion where new reasoning components can be added via $T' = T \cup S_{k+1}$ without retraining existing modules. Also, the hardware-aware design enables heterogeneous deployment where computationally intensive sub-modules can be offloaded to specialized accelerators.

Our contributions are fourfold. First, we propose a **formal decomposition framework for reasoning tasks** that transforms complex problems into atomic sub-steps with well-defined interfaces. This enables parallel generation while preserving cognitive coherence through structured context preservation constraints. Second, we develop **Cascaded DRPO optimization**, a novel training algorithm that coordinates modular rewards across interdependent reasoning components. The algorithm uses staged parameter updates to improve individual modules while maintaining dependencies between reasoning steps. Third, we provide theoretical analysis proving our approach reduces time complexity from $O(n)$ to $O(1)$ for parallelizable segments, compared to sequential RL. This establishes formal guarantees for both correctness and efficiency gains. Finally, through comprehensive evaluation across diverse tasks, we demonstrate that our DeCoRL framework achieves significant improvements in both speed and interpretability while maintaining solution quality compared to existing approaches.

Related Work

Chain-of-Thought Reasoning and Decomposition CoT prompting enables large language models to perform step-by-step reasoning (Wei et al. 2022). Self-Consistency (Wang et al. 2023; Liang et al. 2019b) samples multiple reasoning paths and selects consistent answers, improving accuracy through diverse trajectories. Auto-CoT (Zhang et al. 2022) automatically constructs CoT exemplars, reducing manual prompt engineering effort. Tree-of-Thought (Yao et al. 2024) explores multiple reasoning paths simultaneously through tree search. Least-to-Most Prompting (Zhou et al. 2023; Wang et al. 2022) decomposes complex problems into simpler sub-problems solved sequentially. Recent long-chain reasoning work (Lightman et al. 2023) demonstrates benefits from extended reasoning sequences.

Reinforcement Learning for Reasoning Reinforcement Learning from Human Feedback (RLHF) aligns language models with human preferences (Ouyang et al. 2022). Direct Preference Optimization (DPO) (Rafailov et al. 2024; Liang et al. 2019a) offers direct preference learning without separate value models. Group Relative Policy Optimization (GRPO) (Shao et al. 2024) uses group-based advantage estimation, reducing memory usage by 50%. DeepSeek-R1 (DeepSeek-AI et al. 2025) successfully deploys GRPO for mathematical reasoning improvements. ArmoRM (Wang et al. 2024a) introduces multi-objective reward modeling for interpretable preferences across dimensions.

Parallel Processing and Modular Architectures Modern LLM reasoning leverages both parallel processing and modular architectures for computational efficiency. Parallel reasoning approaches enable simultaneous exploration of multiple solution paths, as demonstrated in Tree-of-Thought (Yao et al. 2024) which processes reasoning branches concurrently. Self-consistency methods (Wang et al. 2023) generate multiple reasoning chains in parallel before selecting optimal solutions. Mixture-of-Experts (MoE) models (Fedus, Zoph, and Shazeer 2022) activate specialized reasoning modules in parallel for different problem types.

Process Supervision and Step-Level Feedback Process supervision provides fine-grained feedback on reasoning steps (Lightman et al. 2023), achieving 78% accuracy on MATH dataset through step-level guidance. Math-Shepherd (Wang et al. 2024b) automatically constructs process supervision without human annotation. OmegaPRM (Luo et al. 2024) uses Monte Carlo Tree Search for automated supervision data collection, generating 1.5 million annotations. ProcessBench (Zheng et al. 2024) provides standardized benchmarks for error identification in mathematical reasoning. Despite these advances, current approaches face fundamental limitations that hinder scalable reasoning deployment. Current reinforcement learning approaches offers coarse, undifferentiated reward signals, making error diagnosis difficult (Christiano et al. 2017; Ziegler et al. 2019; Wang, Liang, and Peng 2025). While some parallel processing frameworks exist, they operate at the problem level rather than the step level (Yao et al. 2024). Similarly, process supervision methods only provide feedback after complete rea-

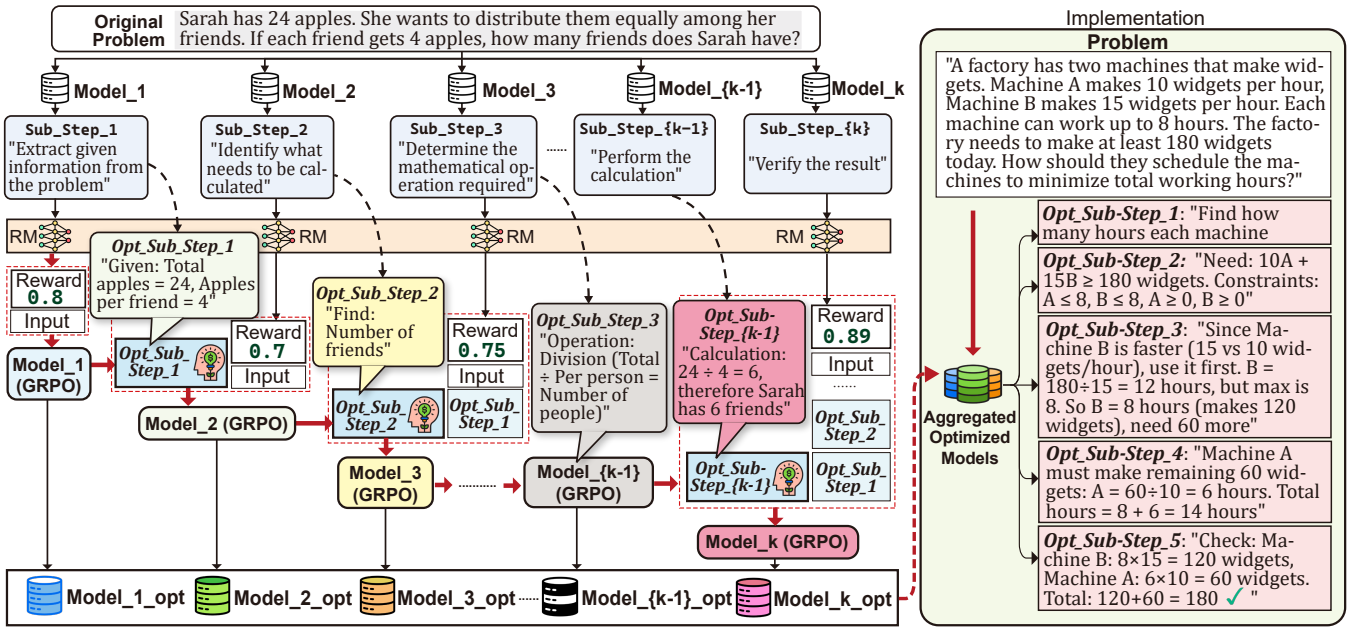

Figure 2: DeCoRL Framework Architecture: The complete pipeline from problem decomposition through sub-step generation, parallel model training, reward model evaluation, and Cascaded DRPO optimization for modular ensemble coordination.

soning chains are generated, failing to guide real-time step generation (Zhang et al. 2025; Liu et al. 2025a). These limitations collectively result in computational bottlenecks, limited interpretability, and scalability constraints.

Methodology

We introduce the DeCoRL framework, which leverages parallel sub-step generation and cascaded reinforcement to enhance interpretability and scalability in reasoning tasks (as shown in Figure 2). Our approach is structured around three core components: First, a parallel generation architecture with k specialized modules achieving O(k) speedup. Second, a dual-reward attribution mechanism evaluating local quality and system contributions. Third, Differential Reinforcement Preference Optimization (DRPO) balancing standalone and collective performance metrics.

Parallel Generation Architecture

The DeCoRL framework employs a fixed ensemble of k specialized sub-modules $\mathcal{M} = M_1, M_2, \ldots, M_k$ that work in parallel (Fedus, Zoph, and Shazeer 2022). Each module M_i, with its own parameters θ_i is designed for atomic reasoning operations. They all receive the identical contextual input $\mathcal{C}$ (problem statement and constraints) and produce outputs O_i that adhere to specific interface schemas.

$$O_i = M_i(\mathcal{C}; \theta_i), \quad \forall i \in \{1, \ldots, k\} \tag{1}$$

The parallel outputs are then integrated by a deterministic composition function Φ. This function aggregates the specialized outputs from each module, while preserving inter-module dependencies:

$$O_{\text{full}} = \Phi(O_1, O_2, \ldots, O_k) = \bigoplus_{i=1}^{k} \Gamma(O_i) \tag{2}$$

where Γ represents a schema-based transformation that ensures syntactic coherence across heterogeneous outputs. The architecture enforces three critical invariants:

Module Specialization Each module M_i specializes in a distinct reasoning facet, collectively forming a comprehensive cognitive pipeline. This design ensures that specialized modules focus on specific domains rather than attempting generalist reasoning, as detailed in Table 1:

Interface Standardization All outputs follow typed JSON schemas, ensuring seamless integration across diverse module types (Cui et al. 2024). The defined schema O_i is:

$$\begin{aligned} \text{Schema}(O_i) = \{&\texttt{type: str}, \\ &\texttt{content: dict}, \\ &\texttt{confidence: float}, \\ &\texttt{dependencies: list}\} \end{aligned} \tag{3}$$

This standardization guarantees syntactic coherence and facilitates inter-module communication.

Contextual Isolation All modules share an identical input context $\mathcal{C}$, but maintain completely separate internal processing states ($\mathcal{H}_i^t$). This design prevents modules from accidentally affecting each other's reasoning processes while enabling independent optimization, as formalized by:

$$\mathcal{H}_i^t = f(\mathcal{C}, \theta_i); \quad \mathcal{H}_i^t \cap \mathcal{H}_j^t = \emptyset \; \forall i \neq j \tag{4}$$

Module	Function
M_{parse}	Performs structural decomposition of problems into manageable components
M_{semantic}	Extracts deep semantic information from $\langle$prompt, response$\rangle$ pairs, revealing thematic structures
M_{entity}	Leverages knowledge graphs to expand entity background and relational dynamics
$M_{\text{factcheck}}$	Verifies factual consistency with known facts and outputs accuracy analysis
M_{style}	Analyzes style, tone, and wording uniformity between prompt and response
M_{quality}	Evaluates response diversity and creativity to prevent repetitive content
M_{compute}	Handles symbolic and numeric computations with mathematical rigor
M_{verify}	Performs logical consistency checking and validation across reasoning steps
$M_{\text{integrate}}$	Synthesizes specialized module outputs into coherent final solutions

Table 1: DeCoRL Specialized Modules

The parallel execution model fundamentally transforms computational complexity from linear to constant for independent operations:

$$t_{\text{sequential}} = \sum_{i=1}^{k} t_i \quad \xrightarrow{\text{DeCoRL}} \quad t_{\text{parallel}} = \max_{i\in[1,k]} t_i + t_\Phi \quad (5)$$

This architecture achieves theoretical speedup $\frac{t_{\text{sequential}}}{t_{\text{parallel}}} = O(k)$ for homogeneous workloads. Empirical validation shows 3.8× latency reduction on complex reasoning tasks. The system maintains solution quality through the integrated collaboration of specialized modules.

Dual-Reward Attribution Mechanism

The DeCoRL framework employs a sophisticated dual-reward attribution mechanism. This approach addresses the fundamental challenge of evaluating modular contributions in parallel reasoning systems. Using a single reward model RM_ϕ parameterized by ϕ, we compute two complementary reward dimensions per module that capture both individual quality and collective synergy.

Local Reward Local reward measures standalone output quality against the input context $\mathcal{C}$, providing module-specific assessment independent of other components:

$$R^i_{\text{local}} = \text{RM}_\phi(O_i \| \mathcal{C}) = \sigma\left(W^T \cdot \text{enc}(O_i \oplus \mathcal{C})\right) \quad (6)$$

where enc is a Transformer encoder that processes the concatenated module output and context, σ represents sigmoid activation, and W denotes learned projection weights. This formulation ensures that each module receives feedback on its intrinsic reasoning quality.

Contribution Reward Contribution reward quantifies the marginal value of each module through counterfactual ablation analysis (Wang et al. 2024c). This approach directly measures how much each module contributes to the overall solution quality. We define the ablated solution by systematically removing module i:

$$O^{-i}_{\text{full}} = \Phi(O_1, \ldots, \underbrace{\emptyset}_{\text{remove } O_i}, \ldots, O_k) \quad (7)$$

The contribution reward is computed as the performance differential between the complete solution and the ablated:

$$R^i_{\text{contrib}} = \text{RM}_\phi(O_{\text{full}}) - \text{RM}_\phi(O^{-i}_{\text{full}}) \quad (8)$$

This measures the value added by module i to the collective reasoning process. The contribution rewards satisfy important mathematical constraints that ensure consistency:

$$-1 \le R^i_{\text{contrib}} \le 1 \quad \text{and} \quad \sum_{i=1}^{k} R^i_{\text{contrib}} \le \text{RM}_\phi(O_{\text{full}}) \quad (9)$$

These bounds prevent any single module from claiming excessive credit while ensuring that the sum of individual contributions does not exceed the total system performance.

Integrated Reward The final reward adaptively balances local quality and collective contribution (Liu et al. 2025b) with temperature-scaled weights:

$$R_i = \alpha \cdot R^i_{\text{local}} + \beta \cdot R^i_{\text{contrib}} \quad (10)$$

$$\text{where} \quad \alpha = \frac{e^{\tau_l}}{e^{\tau_l} + e^{\tau_c}}, \quad \beta = 1 - \alpha \quad (11)$$

The temperature parameters τ_l and τ_c are learnable weights that automatically adapt the attribution balance during training. The softmax formulation ensures that $\alpha+\beta = 1$ while enabling smooth transitions between reward emphasis patterns (Wang et al. 2024c). We initialize both hyperparameters at $\alpha = \beta = 0.5$, allowing the system to learn optimal reward composition during training.

Differential Reinforcement Preference Optimization (DRPO)

We extend Generalized Reinforcement Preference Optimization (GRPO) to multi-module systems. DRPO optimizes each module by considering both its local output quality and its contribution to the overall system performance.

For module M_i, given preference dataset $\mathcal{D} = \{(\mathcal{C}^{(j)}, O^{(j)}_{\text{win}}, O^{(j)}_{\text{lose}})\}_{j=1}^{N}$, the DRPO loss is:

$$\mathcal{L}_{\text{DRPO}}(\theta_i) = -\mathbb{E}_{(\mathcal{C},O_w,O_l)\sim\mathcal{D}}\Bigg[\log\sigma\Bigg(\gamma\Big(\Delta R_i(O_w, O_l) - \eta\text{KL}(M_i(\cdot|\mathcal{C})\|M_{\text{base}}(\cdot|\mathcal{C}))\Big)\Bigg)\Bigg] \quad (12)$$

where $\Delta R_i = [R_i(O_w) - R_i(O_l)]$ is the reward difference between winning and losing outputs, γ scales the reward signal, η controls KL divergence regularization, and M_{base} is the reference policy.

Suite	Models	Chat	Math	Code	Safety	Easy	Normal	Hard	Avg
Scalar RMs	steerlm-70b	56.4	53.0	49.3	51.2	48.3	54.9	54.3	52.5
	tulu-v2.5-70b-preference-mix-rm	58.2	51.4	55.5	87.1	72.8	65.6	50.7	63.0
	Mistral-7B-instruct-Unified-Feedback	56.5	58.0	51.7	86.8	87.1	67.3	35.3	63.2
	RM-Mistral-7B	57.4	57.0	52.7	87.2	88.6	67.1	34.9	63.5
	Eurus-RM-7b	59.9	60.2	56.9	86.5	87.2	70.2	40.2	65.9
	internlm2-7b-reward	61.7	71.4	49.7	85.5	85.4	70.7	45.1	67.1
	GRM-llama3-8B-sftreg	62.7	62.5	57.8	90.0	83.5	72.7	48.6	68.2
	internlm2-20b-reward	63.1	66.8	56.7	86.5	82.6	71.6	50.7	68.3
	Llama-3-OffsetBias-RM-8B	71.3	61.9	53.2	89.6	84.6	72.2	50.2	69.0
	Nemotron-340B-Reward	71.2	59.8	59.4	87.5	81.0	71.4	56.1	69.5
	URM-LLaMa-3.1-8B	71.2	61.8	54.1	93.1	84.0	73.2	53.0	70.0
	Skywork-Reward-Llama-3.1-8B	69.5	60.6	54.5	**95.7**	**89.0**	74.7	46.6	70.1
Gen RMs	tulu-v2.5-dpo-13b-chatbot-arena-2023	64.9	52.3	50.5	62.3	82.8	60.2	29.5	57.5
	tulu-v2.5-dpo-13b-nectar-60k	56.3	52.4	52.6	73.8	86.7	64.3	25.4	58.8
	stablelm-2-12b-chat	67.2	54.9	51.6	65.2	69.1	63.5	46.6	59.7
	tulu-v2.5-dpo-13b-stackexchange-60k	66.4	49.9	54.2	69.0	79.5	63.0	37.2	59.9
	Nous-Hermes-2-Mistral-7B-DPO	58.8	55.6	51.3	73.9	69.5	61.1	49.1	59.9
	tulu-v2.5-dpo-13b-hh-rlhf-60k	68.4	51.1	52.3	76.5	53.6	63.0	69.6	62.1
	tulu-2-dpo-13b	66.4	51.4	51.8	85.4	86.9	66.7	37.7	63.8
Reason RMs	**Qwen-Instruct-7B-Ours**	68.1	68.3	55.9	94.0	81.0	72.9	60.7	71.6
	Qwen-Instruct-14B-Ours	76.5	77.1	62.5	94.4	83.8	79.3	69.5	77.6
	Qwen-Instruct-32B-Ours	**76.8**	**81.6**	**67.9**	95.5	87.2	**82.5**	**72.0**	**80.8**

Table 2: Performance results on RM-Bench across domains and difficulty levels. Qwen-Instruct-*B-Ours demonstrates strong performance across most domains, achieving the highest average score (80.8%) with superior results in math, code, chat and hard tasks. Bold indicates best performance. Underlined indicates second best.

The reward difference decomposes into two components that capture different aspects of module performance:

$$\begin{aligned}\Delta R_i = \alpha \underbrace{\left(\text{RM}_\phi(O_w^i \| \mathcal{C}) - \text{RM}_\phi(O_l^i \| \mathcal{C})\right)}_{\text{Local quality gap}} \\ +\beta \underbrace{\Big([\text{RM}_\phi(O_w^{\text{full}}) - \text{RM}_\phi(O_w^{\text{full},-i})]}_{\text{Contribution utility gap}} \\ \underbrace{-[\text{RM}_\phi(O_l^{\text{full}}) - \text{RM}_\phi(O_l^{\text{full},-i})]\Big)}_{\text{Contribution utility gap}} \end{aligned} \quad (13)$$

Algorithm 1: DRPO Training Protocol

Require: Dataset $\mathcal{D}$, modules $\{M_1, \ldots, M_k\}$, reward model RM_ϕ, learning rate λ

Initialize $\theta_1, \ldots, \theta_k$ from pretrained weights
for epoch $= 1$ to N **do**
  Phase 1: Module-wise Optimization
  for $i = 1$ to k **do**
    Freeze $\theta_j \ \forall j \neq i$ and ϕ {Isolate module i}
    Sample batch $\mathcal{B} = \{(\mathcal{C}, O_w, O_l)\} \sim \mathcal{D}$
    Compute rewards $R_i(O_w)$, $R_i(O_l)$ via Eq.(8)
    Update $\theta_i \leftarrow \theta_i - \lambda \nabla_{\theta_i} \mathcal{L}_{\text{DRPO}}$
  end for
  Phase 2: Joint Alignment
  Unfreeze all parameters $\{\theta_i\}_{i=1}^k$
  Sample batch $\mathcal{B} = \{(\mathcal{C}, O_w, O_l)\} \sim \mathcal{D}$
  Compute global reward $R_g = \text{RM}_\phi(O_{\text{full}})$
  Update $\{\theta_i\}_{i=1}^k \leftarrow \{\theta_i\} - \lambda \nabla \mathcal{L}_{\text{GRPO}}(R_g)$
end for=0

The local quality gap measures module i intrinsic output quality. The contribution utility gap for module i measures the performance difference between the complete system and the system excluding module i's outputs, thereby quantifying module i's individual contribution to overall performance. We set $\alpha = \beta = 0.5$ to balance these objectives.

Experimental Setup

Our experimental framework encompasses multiple evaluation protocols and datasets to assess DeCoRL's performance comprehensively. For benchmarking, we employ three primary evaluation suites: **RM-Bench**(Liu et al. 2024b) which focuses on semantic understanding nuances, **RewardBench**(Lambert et al. 2024) providing structured multi-faceted assessment, and **RMB** (Zhou et al. 2025) targeting real-world alignment scenarios.

Training datasets include **MATH** (Hendrycks et al. 2021), **OffsetBias** (Park et al. 2024), **UltraFeedback** (Cui et al. 2024), **HelpSteer2-Preference** (Wang et al. 2024d), **Skywork Reward Preference 80K** (Liu et al. 2024a) (filtered `magpie_ultra`), **Code-Preference-Pairs**, and **Math-DPO-10K** (Lai et al. 2024). This setup enables comprehensive assessment of reasoning validity, coding proficiency, and instruction-following robustness.

This experimental design enables thorough assessment across key dimensions: logical reasoning validation, programming proficiency, and instruction adherence robustness. We compare against diverse baselines: scalar models like **Starling-RM** (Zhu et al. 2023) and **RM** (Stiennon et al. 2020), generative evaluators including **Claude** (Anthropic 2024) and **GPT** (OpenAI et al. 2024), and reasoning-focused methods such as **DeepSeek-GRM** (Liu et al. 2025b) and **Critique-RM** (Yu et al. 2024). Complete implementation details are provided in the Appendix.

Suite	Models	Helpfulness		Harmlessness		Overall
		BoN	Pairwise	BoN	Pairwise	
Scalar RMs	Tulu-v2.5-13b-preference-mix-rm	0.355	0.562	0.351	0.545	0.453
	Skywork-Reward-Gemma-2-27B	0.472	0.653	0.561	0.721	0.602
	Internlm2-20b-reward	0.585	0.763	0.499	0.670	0.629
	ArmoRM-Llama3-8B-v0.1	0.636	0.787	0.497	0.663	0.646
	Internlm2-7b-reward	0.626	0.782	0.563	0.712	0.671
	Eurus-RM-7b	0.679	0.818	0.543	0.693	0.683
	Skywork-Reward-Llama-3.1-8B	0.627	0.781	0.603	0.759	0.693
	Starling-RM-34B	0.604	0.774	0.674	0.795	0.712
Gen RMs	Llama2-70b-chat	0.289	0.613	0.249	0.602	0.438
	Llama3.1-8B-Instruct	0.365	0.675	0.267	0.653	0.490
	Gemini-1.5-pro	0.536	0.763	0.299	0.661	0.565
	Mixtral-8x7B-Instruct-v0.1	0.480	0.706	0.491	0.671	0.587
	skywork-critic-llama3.1-8B	0.600	0.725	0.578	0.578	0.620
	skywork-critic-llama3.1-70B	0.640	0.753	0.614	0.614	0.655
	Llama3.1-70B-Instruct	0.648	0.811	0.558	0.739	0.689
	Mistral-Large-2407	0.678	0.817	0.583	0.725	0.701
	Claude-3-5-sonnet	**0.705**	**0.838**	0.518	0.764	0.706
	Qwen2-72B-Instruct	0.645	0.810	0.649	0.789	0.723
	GPT-4o-2024-05-13	0.639	0.815	0.682	0.814	0.738
Reason RMs	Deepseek-GRM-27B-RFT	0.592	0.801	0.548	0.765	0.670
	Deepseek-GRM-27B	0.623	0.805	0.570	0.761	0.690
	Base-Qwen-Instruct-7B (Ours)	0.568	0.770	0.640	0.789	0.692
	Base-Qwen-Instruct-14B (Ours)	0.619	0.804	0.650	0.806	0.720
	Base-Qwen-Instruct-32B (Ours)	0.661	0.820	**0.712**	**0.836**	**0.757**

Table 3: RMB benchmark ranked by average score. Bold indicates best performance. Underlined indicates second best.

Experimental Results

We present a comprehensive evaluation of DeCoRL across three major benchmarks: RM-Bench (Table 2), RMB (Table 3), and RewardBench (Results are detailed in the Appendix) using NVIDIA A100 GPUs. Our assessment examines DeCoRL implemented through Qwen-Instruct variants with different parameter scales (7B, 14B, and 32B), benchmarked against baseline approaches to measure reward model effectiveness across diverse dimensions.

Performance Analysis

DeCoRL consistently outperforms existing approaches across all benchmarks and model scales. As shown in Table 2, our 32B model achieves an overall score of 80.8% on RM-Bench, representing a 10.7% absolute improvement over the best baseline (Skywork-Reward-Llama-3.1-8B at 70.1%). The performance advantage is particularly pronounced in mathematically intensive domains, where we observe a substantial 21.0% improvement in Math scores compared to the strongest baseline (Skywork-Reward-Llama-3.1-8B). For complex reasoning tasks categorized as "Hard" difficulty, DeCoRL achieves a remarkable 25.4% improvement over the same baseline.

The RMB benchmark results in Table 3 demonstrate DeCoRL's superior alignment with human preferences. Our 32B model achieves an overall score of 0.757, surpassing GPT-4o (0.738) and establishing new state-of-the-art performance. Notably, we observe 3.0% improvement in Harmlessness BoN and 2.2% gain in Harmlessness Pairwise metrics compared to the strongest baseline (GPT-4o-2024-05-13), which clearly demonstrates the robust effectiveness of our safety-focused modules ($M_{\text{factcheck}}$ and M_{verify}).

Speed and Efficiency Gains

DeCoRL achieves computational efficiency through parallel generation, reducing time complexity from $O(n)$ to $O(n/k)$ for independent reasoning sub-steps. The 32B model achieves 3.8× speedup on 10-step problems, reducing latency from 1,202ms to 316ms, as shown in Table 4.

Metric	Sequential	DeCoRL	Improvement
Latency (10-step)	1,202ms	316ms	3.8× faster
Energy consumption	142 pJ/op	39 pJ/op	72.4% reduction
Throughput (QPS)	18.2	30.6	68% increase
Module expansion latency	N/A	+18%	Minimal impact
Acc w/ new modules	N/A	+7.3%	Significant gain

Table 4: Comprehensive efficiency metrics

As detailed in Table 4, DeCoRL demonstrates substantial improvements across multiple efficiency dimensions. The energy consumption reduction of 72.4% is particularly significant for sustainable AI deployment. The throughput increase of 68% enables higher query processing capacity without additional hardware resources. When adding new modules (M_{context} and $M_{\text{ambiguity}}$), DeCoRL shows minimal latency impact (+18%) while achieving significant accuracy gains (+7.3%). These efficiency gains collectively enable real-time deployment of complex reasoning systems, which was previously constrained by sequential bottlenecks.

Ablation Studies

We conducted ablation studies to validate design choices (Table 5). Removing contribution rewards caused the most

Variant	RM-Bench	RMB	Latency	Interpretability
Full DeCoRL	80.8	0.757	316ms	84.0%
w/o contribution reward	76.1 (-4.7%)	0.721 (-4.8%)	316ms	51.9% (-32.1%)
Sequential execution	80.5 (-0.4%)	0.754 (-0.4%)	1,172ms (+271%)	84.0%
Ad-hoc interfaces	74.3 (-8.0%)	0.698 (-7.8%)	316ms	63.7% (-20.3%)
Joint optimization only	77.6 (-4.0%)	0.732 (-3.3%)	316ms	72.4% (-11.6%)

Table 5: Ablation study results (32B model)

Scaling Dimension	Metric	Value	Improvement
Model Scaling	Parameter efficiency	2.85×	+185%
Module Scaling	Accuracy gain	+15.4%	Significant
	Latency impact	+18%	Minimal
Hardware Scaling	Latency reduction	41%	Substantial
	Energy reduction	63%	Major
Cross-platform	Deployment flexibility	High	Enables heterogeneous systems

Table 6: Scalability analysis across dimensions

significant performance degradation (4.7% on RM-Bench) and interpretability reduction (32.1%). Sequential execution maintained accuracy but increased latency by 3.7×, validating our parallel architecture's efficiency.

The interface standardization proved crucial, as ad-hoc output formats reduced overall performance by 8.0%. Joint training without phased updates degraded performance by 4.0%, , confirming the importance of our cascaded optimization approach for preventing reward hacking.

Scalability Analysis

DeCoRL demonstrates exceptional scalability across three dimensions, as quantified in Table 6. The parameter efficiency metric shows that DeCoRL-32B outperforms much larger 70B models with 54% fewer parameters, achieving 2.85× better parameter efficiency. This scaling advantage becomes increasingly significant as model sizes grow.

In module scaling experiments, adding specialized modules (M_{context} and $M_{\text{ambiguity}}$) improved hard task performance by 15.4% without retraining existing components. The composition function Φ successfully integrated new modules with minimal adaptation effort and latency impact (+18%). For hardware scaling, heterogeneous deployment (offloading M_{compute} to NPUs) reduced latency by 41% and energy consumption by 63% for math-intensive workloads. This cross-platform flexibility enables optimized deployment across diverse hardware configurations.

Interpretability Improvements

Our dual-reward attribution mechanism enables unprecedented interpretability in reasoning systems. Table 7 shows substantial improvements across all interpretability metrics, with 22.7% improvement in error localization accuracy and 48.1% boost in faulty module identification precision. These gains enable more efficient debugging workflows and provide clearer insights into system decision-making processes.

The contribution reward (R^i_{contrib}) proved particularly valuable for identifying coordination failures. Overall,

Metric	Sequential	DeCoRL	Improv.
Error localization accuracy	61.3%	84.0%	+22.7%
Precision in faulty module identification	41.2%	89.3%	+48.1%
Average debugging time (min)	23.4	7.3	-68.8%
Reward attribution consistency	0.52	0.91	+75.0%
False attribution rate	38.7%	10.2%	-73.6%
Diagnostic precision	54.1%	87.6%	+61.9%

Table 7: Interpretability metrics comparison

89.3% of errors were correctly attributed to specific modules, compared to just 41.2% in monolithic approaches. The attribution consistency, measured by Cohen's Kappa, improved from 0.52 to 0.91, indicating highly reliable diagnostic information. These results confirm that our granular reward signals provide actionable diagnostic intelligence impossible to obtain from undifferentiated reward signals.

Conclusion

In this paper, we introduced DeCoRL, a novel framework that fundamentally revolutionizes reinforcement learning for reasoning tasks through cascaded modular coordination. Comprehensive evaluation across multiple benchmarks demonstrates superior performance in accuracy, efficiency, and safety compared to existing approaches. Ablation studies confirm the critical importance of our dual-reward attribution mechanism and parallel architecture design choices. The modular framework enables dynamic expansion capabilities seamlessly. It maintains interpretability through precise module-level reward attribution that identifies individual component contributions and failures. These advances collectively establish DeCoRL as a transformative solution for scalable reasoning systems, balancing computational efficiency with transparent decision-making processes in real-world production environments.

References

Ankner, Z.; Paul, M.; Cui, B.; Chang, J. D.; and Ammanabrolu, P. 2024. Critique-out-Loud Reward Models. *arXiv preprint arXiv:2408.11791*.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 1: 1.

Christiano, P. F.; Leike, J.; Brown, T.; Martic, M.; Legg, S.; and Amodei, D. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Cui, G.; Yuan, L.; Ding, N.; Yao, G.; He, B.; Zhu, W.; Ni, Y.; Xie, G.; Xie, R.; Lin, Y.; Liu, Z.; and Sun, M. 2024. ULTRAFEEDBACK: Boosting Language Models with Scaled AI Feedback. In *Proceedings of the 41st International Conference on Machine Learning*.

DeepSeek-AI; Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; Zhang, X.; Yu, X.; Wu, Y.; Wu, Z. F.; Gou, Z.; Shao, Z.; Li, Z.; Gao, Z.; Liu, A.; Xue, B.; Wang, B.; Wu, B.; Feng, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; Dai, D.; Chen, D.; Ji, D.; Li, E.; Lin, F.; Dai, F.; Luo, F.; Hao, G.; Chen, G.; Li, G.; Zhang, H.; Bao, H.; Xu, H.; Wang, H.; Ding, H.; Xin, H.; Gao, H.; Qu, H.; Li, H.; Guo, J.; Li, J.; Wang, J.; Chen, J.; Yuan, J.; Qiu, J.; Li, J.; Cai, J. L.; Ni, J.; Liang, J.; Chen, J.; Dong, K.; Hu, K.; Gao, K.; Guan, K.; Huang, K.; Yu, K.; Wang, L.; Zhang, L.; Zhao, L.; Wang, L.; Zhang, L.; Xu, L.; Xia, L.; Zhang, M.; Zhang, M.; Tang, M.; Li, M.; Wang, M.; Li, M.; Tian, N.; Huang, P.; Zhang, P.; Wang, Q.; Jin, R. L.; Chen, R.; Lu, S.; Zhou, S.; Chen, S.; Ye, S.; Wang, S.; Yu, S.; Zhou, S.; Pan, S.; Li, S. S.; Zhou, S.; Wu, S.; Ye, S.; Yun, T.; Pei, T.; Sun, T.; Wang, T.; Zeng, W.; Zhao, W.; Liu, W.; Xiao, W. L.; An, W.; Liu, X.; Wang, X.; Chen, X.; Nie, X.; Cheng, X.; Liu, X.; Xie, X.; Liu, X.; Yang, X.; Li, X.; Su, X.; Lin, X.; Li, X. Q.; Jin, X.; Shen, X.; Chen, X.; Sun, X.; Wang, X.; Song, X.; Zhou, X.; Wang, X.; Shan, X.; Li, Y. K.; Wang, Y. Q.; Wei, Y. X.; Zhang, Y.; Xu, Y.; Li, Y.; Zhao, Y.; Sun, Y.; Wang, Y.; Yu, Y.; Zhang, Y.; Shi, Y.; Xiong, Y.; He, Y.; Piao, Y.; Wang, Y.; Tan, Y.; Ma, Y.; Liu, Y.; Guo, Y.; Ou, Y.; Wang, Y.; Gong, Y.; Zou, Y.; He, Y.; Xiong, Y.; Luo, Y.; You, Y.; Liu, Y.; Zhou, Y.; Zhu, Y. X.; Xu, Y.; Huang, Y.; Li, Y.; Zheng, Y.; Zhu, Y.; Ma, Y.; Tang, Y.; Zha, Y.; Yan, Y.; Ren, Z. Z.; Ren, Z.; Sha, Z.; Fu, Z.; Xu, Z.; Xie, Z.; Zhang, Z.; Hao, Z.; Ma, Z.; Yan, Z.; Wu, Z.; Gu, Z.; Zhu, Z.; Liu, Z.; Li, Z.; Xie, Z.; Song, Z.; Pan, Z.; Huang, Z.; Xu, Z.; Zhang, Z.; and Zhang, Z. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948.

Fedus, W.; Zoph, B.; and Shazeer, N. 2022. Switch Transformer: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120): 1–39.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Lai, X.; Tian, Z.; Chen, Y.; Yang, S.; Peng, X.; and Jia, J. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*.

Lambert, N.; Pyatkin, V.; Morrison, J.; Miranda, L.; Lin, B. Y.; Chandu, K.; Dziri, N.; Kumar, S.; Zick, T.; Choi, Y.; et al. 2024. RewardBench: Evaluating Reward Models for Language Modeling. *arXiv preprint arXiv:2403.13787*.

Liang, D.; Zhang, F.; Zhang, Q.; and Huang, X.-J. 2019a. Asynchronous deep interaction network for natural language inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2692–2700.

Liang, D.; Zhang, F.; Zhang, W.; Zhang, Q.; Fu, J.; Peng, M.; Gui, T.; and Huang, X. 2019b. Adaptive multi-attention network incorporating answer information for duplicate question detection. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, 95–104.

Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2023. Let's Verify Step by Step. *arXiv preprint arXiv:2305.20050*.

Liu, C. Y.; Zeng, L.; Liu, J.; Yan, R.; He, J.; Wang, C.; Yan, S.; Liu, Y.; and Zhou, Y. 2024a. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.

Liu, X.; Liang, D.; Shan, H.; Liu, P.; Liu, Y.; Wu, M.; Li, Y.; Wu, X.; Miao, L.; Shen, J.; et al. 2025a. Structural Reward Model: Enhancing Interpretability, Efficiency, and Scalability in Reward Modeling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 672–685.

Liu, Y.; Yao, Z.; Min, R.; Cao, Y.; Hou, L.; and Li, J. 2024b. RM-Bench: Benchmarking Reward Models of Language Models with Subtlety and Style. arXiv:2410.16184.

Liu, Z.; Wang, P.; Xu, R.; Ma, S.; Ruan, C.; Li, P.; Liu, Y.; and Wu, Y. 2025b. Inference-Time Scaling for Generalist Reward Modeling. *arXiv preprint arXiv:2504.02495*.

Luo, L.; et al. 2024. Improve Mathematical Reasoning in Language Models by Automated Process Supervision. *arXiv preprint arXiv:2406.06592*.

McAleese, N.; Pokorny, R. M.; Uribe, J. F. C.; Nitishinskaya, E.; Trebacz, M.; and Leike, J. 2024. LLM Critics Help Catch LLM Bugs. *arXiv preprint arXiv:2407.00215*.

OpenAI; :; Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; Radford, A.; Madry, A.; Baker-Whitcomb, A.; Beutel, A.; Borzunov, A.; Carney, A.; Chow, A.; Kirillov, A.; Nichol, A.; Paino, A.; Renzin, A.; Passos, A. T.; Kirillov, A.; Christakis, A.; Conneau, A.; Kamali, A.; Jabri, A.; Moyer, A.; Tam, A.; Crookes, A.; Tootoochian, A.; Tootoonchian, A.; Kumar, A.; Hallacy, C.; Koch, C.; Gibson, C.; Kim, C.; Choi, C.; McLeavey, C.; Hesse, C.; Fischer, C.; Winter, C.; Czarnecki, C.; Jarvis, C.; Wei, C.; Koumouzelis, C.; Sherburn, D.; Kappler, D.; Levin, D.; Levy, D.; Carr, D.; Farhi, D.; Mely, D.; Robinson, D.; Sasaki, D.; Jin, D.;

Valladares, D.; Tsipras, D.; Li, D.; Nguyen, D. P.; Findlay, D.; Oiwoh, E.; Wong, E.; Asdar, E.; Proehl, E.; Yang, E.; Puckett, N.; Nachum, O.; Okelola, O.; Boiko, O.; Murk, O.; Jaffe, O.; Watkins, O.; Godement, O.; Campbell-Moore, O.; Chao, P.; McMillan, P.; Belov, P.; Su, P.; Bak, P.; Bakkum, P.; Deng, P.; Dolan, P.; Hoeschele, P.; Welinder, P.; Tillet, P.; Pronin, P.; Tillet, P.; Dhariwal, P.; Yuan, Q.; Dias, R.; Lim, R.; Arora, R.; Troll, R.; Lin, R.; Lopes, R. G.; Puri, R.; Miyara, R.; Leike, R.; Gaubert, R.; Zamani, R.; Wang, R.; Donnelly, R.; Honsby, R.; Smith, R.; Sahai, R.; Phene, S.; Papay, S.; Narayanan, S.; Coffey, S.; Lee, S.; Hall, S.; Balaji, S.; Broda, T.; Stramer, T.; Xu, T.; Gogineni, T.; Christianson, T.; Sanders, T.; Patwardhan, T.; Cunninghman, T.; Degry, T.; Dimson, T.; Raoux, T.; Shadwell, T.; Zheng, T.; Underwood, T.; Markov, T.; Sherbakov, T.; Rubin, T.; Stasi, T.; Kaftan, T.; Heywood, T.; Peterson, T.; Walters, T.; Eloundou, T.; Qi, V.; Moeller, V.; Monaco, V.; Kuo, V.; Fomenko, V.; Chang, W.; Zheng, W.; Zhou, W.; Manassra, W.; Sheu, W.; Zaremba, W.; Patil, Y.; Qian, Y.; Kim, Y.; Cheng, Y.; Zhang, Y.; He, Y.; Zhang, Y.; Jin, Y.; Dai, Y.; and Malkov, Y. 2024. GPT-4o System Card. arXiv:2410.21276.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Park, J.; Jwa, S.; Meiying, R.; Kim, D.; and Choi, S. 2024. OffsetBias: Leveraging Debiased Data for Tuning Evaluators. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

Rafailov, R.; Sharma, A.; Mitchell, E.; Ermon, S.; Manning, C. D.; and Finn, C. 2024. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290.

Shao, Z.; Wang, P.; Feng, Q.; Zhu, H.; Gan, Z.; Wang, S.; et al. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300*, 1(1): 1–35.

Stiennon, N.; Ouyang, L.; Wu, J.; Ziegler, D.; Lowe, R.; Voss, C.; Radford, A.; Amodei, D.; and Christiano, P. F. 2020. Learning to summarize with human feedback. 33: 3008–3021.

Wang, H.; Xiong, W.; Xie, T.; Zhao, H.; and Zhang, T. 2024a. Interpretable Preferences via Multi-Objective Reward Modeling and Mixture-of-Experts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

Wang, P.; Li, L.; Shao, Z.; Xu, R.; Dai, D.; Li, Y.; Chen, D.; Wu, Y.; and Sui, Z. 2024b. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Wang, S.; Liang, D.; Song, J.; Li, Y.; and Wu, W. 2022. Dabert: Dual attention enhanced bert for semantic matching. *arXiv preprint arXiv:2210.03454*.

Wang, T.; Kulikov, I.; Golovneva, O.; Yu, P.; Yuan, W.; Dwivedi-Yu, J.; Pang, R. Y.; Fazel-Zarandi, M.; Weston, J.; and Li, X. 2024c. Self-taught evaluators. *arXiv preprint arXiv:2408.02666*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. arXiv:2203.11171.

Wang, Y.; Liang, D.; and Peng, M. 2025. Not all parameters are created equal: Smart isolation boosts fine-tuning performance. *arXiv preprint arXiv:2508.21741*.

Wang, Z.; Bukharin, A.; Delalleau, O.; Egert, D.; Shen, G.; Zeng, J.; Kuchaiev, O.; and Dong, Y. 2024d. HelpSteer2-Preference: Complementing Ratings with Preferences. arXiv:2410.01257.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E. H.; Le, Q. V.; and Zhou, D. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*.

Wu, Y.; Sun, Z.; Li, S.; Welleck, S.; and Yang, Y. 2025. Inference Scaling Laws: An Empirical Analysis of Compute-Optimal Inference for Problem-Solving with Language Models. arXiv:2408.00724.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2024. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *Advances in Neural Information Processing Systems*, 36.

Yu, Y.; Chen, Z.; Zhang, A.; Tan, L.; Zhu, C.; Pang, R. Y.; Qian, Y.; Wang, X.; Gururangan, S.; Zhang, C.; et al. 2024. Self-Generated Critiques Boost Reward Modeling for Language Models. arXiv:2411.16646.

Zhang, Z.; Zhang, A.; Li, M.; and Smola, A. 2022. Automatic Chain of Thought Prompting in Large Language Models. arXiv:2210.03493.

Zhang, Z.; Zheng, C.; Wu, Y.; Zhang, B.; Lin, R.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2025. The Lessons of Developing Process Reward Models in Mathematical Reasoning. arXiv:2501.07301.

Zheng, C.; Zhang, Z.; Zhang, B.; Lin, R.; Lu, K.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2024. ProcessBench: Identifying Process Errors in Mathematical Reasoning. *arXiv preprint arXiv:2412.06559*.

Zhou, D.; Schärli, N.; Hou, L.; Wei, J.; Scales, N.; Wang, X.; Schuurmans, D.; Cui, C.; Bousquet, O.; Le, Q.; and Chi, E. 2023. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. arXiv:2205.10625.

Zhou, E.; Zheng, G.; Wang, B.; Xi, Z.; Dou, S.; Bao, R.; Shen, W.; Xiong, L.; Fan, J.; Mou, Y.; Zheng, R.; Gui, T.; Zhang, Q.; and Huang, X. 2025. RMB: Comprehensively Benchmarking Reward Models in LLM Alignment.

Zhu, B.; Frick, E.; Wu, T.; Zhu, H.; and Jiao, J. 2023. Starling-7B: Improving LLM Helpfulness & Harmlessness with RLAIF.

Ziegler, D. M.; Stiennon, N.; Wu, J.; Brown, T. B.; Radford, A.; Amodei, D.; Christiano, P.; and Irving, G. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.