DRAFT-RL: Multi-Agent Chain-of-Draft Reasoning for Reinforcement Learning-Enhanced LLMs

Yuanhao Li[1], Mingshan Liu[2], Hongbo Wang[1*], Yiding Zhang[1], Yifei Ma[1], Wei Tan[3]

[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

[2]Thrust of Artificial Intelligence, The Hong Kong University of Science and Technology(GuangZhou)

[3]Faculty of Science and Engineering, University of Bristol, Bristol, United Kingdom

yh.l@bupt.edu.cn, mliu618@connect.hkust-gz.edu.cn, hbwang@bupt.edu.cn, 2024140854@bupt.edu.cn, 2024110867@bupt.edu.cn, xu25063@bristol.ac.uk

Abstract

Large Language Models (LLMs) have shown impressive capabilities in multi-step reasoning and problem-solving. Recent works introduce multi-agent reflection frameworks where multiple LLM agents critique and refine each other's outputs using reinforcement learning (RL). However, these approaches often rely on single-shot responses and lack structural diversity in reasoning exploration. In this paper, we propose DRAFT-RL, a novel framework that integrates Chain-of-Draft (CoD) reasoning into multi-agent RL training. Instead of generating single responses, each agent produces multiple drafts per query, which are then evaluated by peer agents and a learned reward model to identify the most promising trajectory. These selected drafts are used to refine future reasoning strategies through actor-critic learning. DRAFT-RL enables explicit multi-path exploration, peer-guided reflection, and reward-aligned selection, resulting in more robust and interpretable LLM agent behavior. We evaluate our method on complex reasoning tasks including code synthesis, symbolic math, and knowledge-intensive QA, demonstrating that DRAFT-RL outperforms existing reflective and RL-based agents by significant margins in both accuracy and convergence speed.

Code — https://github.com/ArthasTY/DRAFT-RL.

Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in complex reasoning tasks across domains such as mathematics, code generation, and knowledge-intensive question answering (Brown et al. 2020; Touvron et al. 2023; Chowdhery et al. 2022). These advances have enabled autonomous LLM-based agents that can interact with environments, reason about multi-step problems, and accomplish sophisticated tasks (Yao et al. 2023b; Wang et al. 2023). When combined with reinforcement learning (RL), such agents can further learn from experience and progressively improve their performance (Yuan et al. 2023; Li et al. 2024; Dou et al. 2024; Wang et al. 2024a).

Despite these promising developments, current LLM-based RL agents still face critical limitations:

- **Decision instability**: LLMs may produce inconsistent or suboptimal decisions due to stochastic generation, particularly in unfamiliar scenarios (Gudibande et al. 2023).
- **Inefficient exploration**: Standard RL agents generate only a single action per state, narrowing exploration and missing alternative solution paths (Shojaee et al. 2023).
- **Training inefficiency**: Effective policy learning requires many environment interactions, making RL training costly and slow (Liu et al. 2023).
- **Limited self-correction**: Agents often fail to detect and correct reasoning errors, causing flawed strategies to persist (Yin et al. 2024).

Recent work has attempted to address these issues through multi-agent critique frameworks (Du et al. 2023; Chan et al. 2023; Wang et al. 2024b) and RL-based alignment methods such as RLHF and RLAIF (Stiennon et al. 2020; Lee et al. 2023; Dai et al. 2024). While effective, these approaches mostly rely on single-shot responses and lack explicit mechanisms for systematically exploring and evaluating diverse reasoning paths.

In parallel, prompting methods such as Chain-of-Thought (CoT) (Wei et al. 2022) and the more concise Chain-of-Draft (CoD) (Xu et al. 2025) have demonstrated that structuring intermediate reasoning steps can substantially improve task performance. CoD, in particular, encourages concise (leq5 words) reasoning steps, enhancing clarity and modularity. However, such techniques are primarily used at inference time and have not been integrated into reinforcement learning frameworks where they could guide exploration and policy improvement.

These limitations motivate a key open question: *How can structured reasoning be effectively combined with multi-agent reinforcement learning to achieve robust exploration, collaborative evaluation, and interpretable learning?* This challenge is especially relevant in domains with multiple viable solution strategies, where evaluating diverse reasoning paths is essential.

In this paper, we propose **DRAFT-RL**, a framework that integrates CoD-style structured reasoning with multi-agent

*Corresponding author.

RL. DRAFT-RL introduces three core components:

- **Multi-Draft Generation**: Each agent produces multiple concise reasoning drafts per query, explicitly enabling exploration of diverse solution approaches.
- **Peer-Guided Evaluation**: Agents evaluate each other's drafts using predefined criteria, providing richer and more reliable feedback than single-agent assessment.
- **Reward-Aligned Selection**: A learned reward model combines peer evaluations with task rewards to select high-quality drafts and guide actor–critic learning.

Through this integration, DRAFT-RL systematically explores reasoning alternatives, identifies promising solution paths, and continually refines reasoning strategies. Unlike prior methods that rely on single-path reasoning or post-hoc candidate selection, DRAFT-RL unifies exploration, evaluation, and learning within a coherent multi-agent framework.

We evaluate DRAFT-RL on three challenging domains: (1) code synthesis, (2) symbolic mathematics, and (3) knowledge-intensive question answering. DRAFT-RL consistently outperforms reflective, prompting-based, and RL-based baselines across all tasks. For example, on the MATH dataset (Hendrycks et al. 2021), DRAFT-RL achieves a 3.7% absolute improvement over the strongest baseline. The framework also converges more quickly during training and produces reasoning traces that are more interpretable and coherent.

Our key contributions are summarized as follows:

- We introduce DRAFT-RL, the first framework to integrate Chain-of-Draft reasoning with multi-agent reinforcement learning.
- We propose a peer-guided evaluation mechanism that enhances collaborative filtering of reasoning drafts.
- We develop a reward-aligned selection process that unifies peer evaluation with task-specific rewards.
- We demonstrate substantial performance gains (3.5–3.7%) across code, math, and QA benchmarks.
- We provide detailed analyses of training dynamics and emergent agent behaviors enabled by multi-draft reasoning.

The remainder of this paper is organized as follows: Section 2 reviews related work on LLM agents, reinforcement learning, structured reasoning, and multi-draft methods. Section 3 describes the DRAFT-RL framework. Section 4 outlines our experimental setup. Section 5 presents quantitative and qualitative results. Section 6 concludes with insights and future directions.

Related Work

LLM-Based Agents and Multi-Agent Systems

Recent advances in large language models have enabled sophisticated LLM-based agents capable of complex reasoning and task completion. Frameworks such as ReAct (Yao et al. 2023b), ART (Paranjape et al. 2023), and Voyager (Wang et al. 2023) use LLMs to generate action plans and execute them in various environments. Building on these single-agent frameworks, multi-agent LLM systems like ChatEval (Chan et al. 2023), CAMEL (Li et al. 2023), and AutoGen (Wu et al. 2023) enable multiple LLM agents to collaborate, critique each other's outputs, and refine solutions through iterative feedback.

While promising, these approaches rely on single-shot responses and lack structured reasoning diversity. DRAFT-RL enables multi-draft generation and collaborative evaluation for more robust reasoning.

Reinforcement Learning with LLMs

Reinforcement learning has emerged as a powerful approach for aligning LLMs with human preferences and task-specific objectives. Techniques like RLHF (Stiennon et al. 2020; Ouyang et al. 2022) use human preferences to train reward models, which then guide LLM fine-tuning. Recent work has extended this to use AI feedback instead of human feedback (RLAIF) (Lee et al. 2023; Bai et al. 2022) for greater scalability.

In the domain of code generation, approaches like CodeRL(Tang et al. 2025b; Tang, Klein, and Bissyandé 2025; Tang et al. 2024, 2025a; Le et al. 2022), StepCoder (Dou et al. 2024), and FALCON (Li et al. 2024) have shown promising results by using execution feedback to improve code quality. These methods typically train a single policy model to generate actions sequentially, whereas our work employs a multi-draft, multi-agent approach that enables more structured exploration and collaborative evaluation.

Structured Reasoning in LLMs

Structured reasoning techniques have significantly enhanced LLM performance on complex tasks. Chain-of-Thought (CoT) prompting (Wei et al. 2022) encourages LLMs to generate intermediate reasoning steps before producing final answers. Extensions like Tree-of-Thoughts (Yao et al. 2023a) and Graph-of-Thoughts (Besta et al. 2024) explore multiple reasoning paths to find optimal solutions.

Most relevant to our work is Chain-of-Draft (CoD) (Xu et al. 2025), which constrains each reasoning step to be concise ($\leq$5 words), promoting clarity and modularity. While CoD has shown impressive results on arithmetic and commonsense reasoning tasks, our work extends it to a multi-agent reinforcement learning framework, enabling more structured and diverse exploration of reasoning paths.

DRAFT-RL Framework

Problem Formulation

We consider a setting where multiple LLM agents collaborate to solve complex reasoning tasks. Each task consists of a query q and a ground truth answer a^*. The goal is to train agents to generate high-quality responses that closely match the ground truth answers.

Formally, we have N agents $\{A_1, A_2, \ldots, A_N\}$, each parameterized by θ_i. Given a query q, each agent A_i generates K drafts $\{d_i^1, d_i^2, \ldots, d_i^K\}$, where each draft d_i^k consists of a sequence of reasoning steps followed by a final answer a_i^k.

The objective is to learn policies π_{θ_i} that maximize the expected reward:

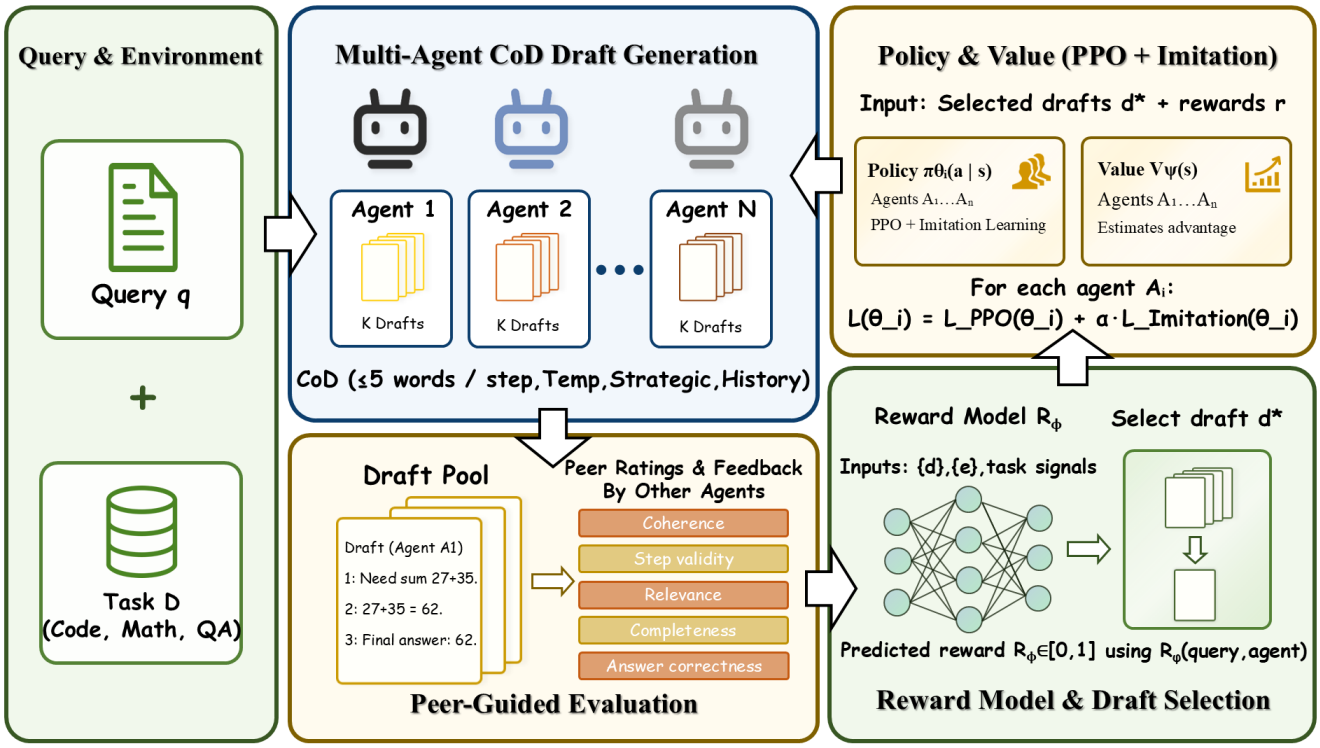

Figure 1: Overall architecture of the DRAFT-RL framework.

$$J(\theta_i) = \mathbb{E}_{q\sim\mathcal{D}, d_i^k\sim\pi_{\theta_i}(\cdot|q)}[R(d_i^k, q, a^*)] \tag{1}$$

where $\mathcal{D}$ is the distribution of queries, and R is a reward function that measures the quality of a draft with respect to the ground truth answer.

Chain-of-Draft Reasoning

Chain-of-Draft (CoD) is a structured reasoning approach that constrains each reasoning step to be concise ($\leq$5 words) while maintaining the key logical progression. This approach emphasizes brevity and modularity, focusing on essential reasoning milestones rather than verbose explanations.

Formally, each draft d_i^k generated by agent A_i consists of a sequence of reasoning steps $\{r_i^{k,1}, r_i^{k,2}, \ldots, r_i^{k,m}\}$ followed by a final answer a_i^k:

$$d_i^k = (r_i^{k,1}, r_i^{k,2}, \ldots, r_i^{k,m}, a_i^k) \tag{2}$$

The CoD constraint requires that each reasoning step contains at most 5 words:

$$\text{valid}(r_i^{k,j}) = \mathbb{I}[\text{word\_count}(r_i^{k,j}) \leq 5], \forall j \in \{1, \ldots, m\} \tag{3}$$

where $\mathbb{I}[\cdot]$ is the indicator function.

DRAFT-RL Architecture

The DRAFT-RL framework consists of three main components: multi-draft generation, peer-guided evaluation, and reward-aligned selection and learning.

Multi-Draft Generation In DRAFT-RL, each agent A_i generates K diverse drafts for a given query. To ensure diversity, we employ temperature variation (ranging from 0.2 to 0.8 across drafts), strategic prompting (guiding agents to explore different approaches), and draft history conditioning (ensuring subsequent drafts differ from previous ones). This approach enables explicit exploration of diverse reasoning paths, increasing the likelihood of discovering effective solutions.

The draft generation process is formalized as:

$$d_i^k \sim \pi_{\theta_i}(\cdot|q, \{d_i^1, \ldots, d_i^{k-1}\}, s_k) \tag{4}$$

where s_k is the strategic guidance for the k-th draft.

Peer-Guided Evaluation After generating drafts, agents evaluate each other's outputs according to reasoning coherence, step validity, relevance, completeness, and answer correctness. Each agent A_j evaluates drafts from other agents, providing scalar ratings and qualitative feedback:

$$e_j(d_i^k) = (s_j(d_i^k), f_j(d_i^k)) \tag{5}$$

where $s_j(d_i^k) \in [0, 1]$ is a scalar rating and $f_j(d_i^k)$ is qualitative feedback.

Algorithm 1: DRAFT-RL Training

Initialize agent parameters $\{\theta_i\}_{i=1}^{N}$ and reward model parameters ϕ
for each training iteration **do**
  Sample batch of queries from dataset
  for each agent A_i **do**
    for each query q **do**
      Generate K diverse drafts using CoD reasoning with temperature diversity
      Ensure each reasoning step contains $\leq$5 words
    end for
  end for
  for each agent A_i **do**
    Evaluate drafts from other agents on multiple criteria
    Provide scalar ratings and qualitative feedback
  end for
  for each query q **do**
    Use reward model to predict rewards for all drafts
    Select optimal draft for each agent
    Execute selected drafts and observe rewards
  end for
  Update agent policies using PPO with imitation learning
  Update reward model based on observed rewards
end for

This peer evaluation process allows agents to identify strengths and weaknesses in each other's reasoning approaches, providing valuable feedback for improvement.

Reward-Aligned Selection and Learning To select the most promising drafts, we train a reward model R_ϕ that combines peer evaluations with task-specific metrics:

$$R_\phi(d_i^k, q, \{e_j(d_i^k)\}_{j\neq i}) \in [0,1] \tag{6}$$

For each query, we select the draft with the highest predicted reward:

$$d^* = \arg\max_{d_i^k} R_\phi(d_i^k, q, \{e_j(d_i^k)\}_{j\neq i}) \tag{7}$$

We use the selected drafts to refine agent policies through actor-critic learning, employing Proximal Policy Optimization (PPO) (Schulman et al. 2017) augmented with imitation learning from selected optimal drafts:

$$L(\theta_i) = L^{\text{PPO}}(\theta_i) + \alpha L^{\text{Imitation}}(\theta_i) \tag{8}$$

where α balances reinforcement learning and imitation learning objectives.

Training Algorithm

The DRAFT-RL training process is outlined in Algorithm 1.

This iterative process allows agents to continuously improve their reasoning strategies based on peer feedback and reward signals. The multi-draft approach enables broader exploration of the solution space, while the peer evaluation and reward-guided selection mechanisms help identify and reinforce effective reasoning patterns.·1

Experimental Setup

We evaluate DRAFT-RL across three complex reasoning domains: code synthesis, symbolic mathematics, and knowledge-intensive QA. Below, we summarize the datasets, baselines, implementation, and evaluation protocols.

Benchmarks

Code Synthesis: We use MBPP (Austin et al. 2021) and HumanEval (Chen et al. 2021), standard benchmarks with natural language prompts and functional test cases.

Symbolic Math: We test on GSM8K (Cobbe et al. 2021) (grade-school arithmetic) and MATH (Hendrycks et al. 2021) (competition-level math) to assess stepwise reasoning.

Knowledge-Intensive QA: We include HotpotQA (Yang et al. 2018) (multi-hop retrieval) and MMLU (Hendrycks et al. 2020) (broad-domain factual QA) to evaluate general knowledge reasoning.

Baselines

We compare DRAFT-RL to:

- **Prompting:** Chain-of-Thought (CoT) (Wei et al. 2022), Chain-of-Draft (CoD) (Xu et al. 2025), and Self-Consistency (Wang et al. 2022).
- **Frameworks:** ReAct (Yao et al. 2023b), Reflexion (Shinn et al. 2023), and ChatEval (Chan et al. 2023).
- **RL-based:** RLHF (Ouyang et al. 2022) and RLAIF (Lee et al. 2023).

All baselines use Claude-3.5-Sonnet as the foundation model.

Implementation Details

Agents and Model: We use 3 agents with independently fine-tuned adapters (130M parameters each) atop Claude-3.5-Sonnet. The reward model is a 12-layer transformer (100M parameters) trained per domain.

Draft Generation: Each agent generates $K = 5$ drafts per query using varied temperatures ([0.2–0.8]), strategic prompts, and history conditioning to ensure diversity. CoD constraints (=¡5 words per reasoning step) are enforced.

Evaluation and Learning: Agents score peer drafts on coherence, step validity, relevance, completeness, and answer correctness. A learned reward model integrates peer scores and task-specific signals to guide PPO + imitation learning.

Training: We use AdamW optimizer and PPO with $\epsilon = 0.2$, $\gamma = 0.99$, $\lambda = 0.95$, and imitation weight $\alpha = 0.5$. Training runs for 10 epochs with early stopping. All experiments were conducted on 64×A100 GPUs (125k GPU-hours total).

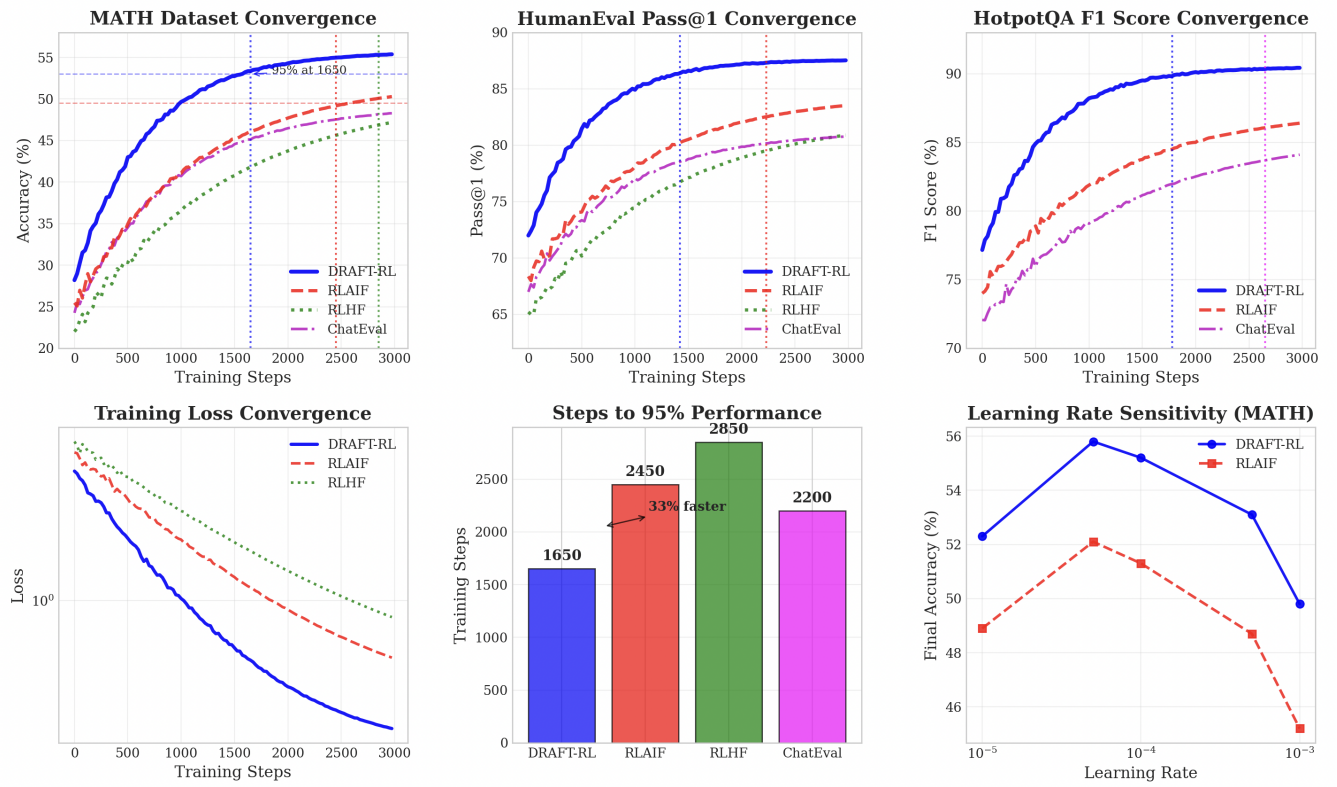

Figure 2: Training dynamics across domains, including learning curves, sample efficiency, and convergence behavior. DRAFT-RL converges 33–42% faster than RLHF/RLAIF.

Evaluation Protocols

Code Synthesis: Pass@1 via test case execution; diversity and complexity also analyzed.

Math: Accuracy based on numerical match; answers extracted via regex with tolerance.

QA: HotpotQA: EM and F1; MMLU: zero-shot and 5-shot accuracy. Domain-level breakdowns provided.

Statistical Rigor: Results averaged over 5 seeds. Significance tested via paired t-tests with Bonferroni correction ($p < 0.05$). Convergence speed measured via validation thresholds.

Results and Analysis

We present comprehensive comparisons between DRAFT-RL and strong baselines across code synthesis, symbolic mathematics, and knowledge-intensive QA. Our analysis includes quantitative metrics, ablations, convergence analysis, and qualitative insights into emergent reasoning. Extended results and ablations are provided in the supplementary appendix (Li et al. 2025).

Main Results

Code Synthesis Performance Table 1 summarizes performance on MBPP and HumanEval. DRAFT-RL achieves the highest Pass@1 on both datasets, outperforming strong reflective agents (Reflexion, ChatEval) and RL-based systems (RLHF, RLAIF). Notably, DRAFT-RL improves by +4.5% on MBPP and +3.1% on HumanEval over RLAIF, the strongest baseline.

Method	MBPP	HumanEval	Complexity	Time (s)
Prompting Baselines				
CoT	68.2	74.4	2.3	12.4
CoD	71.5	77.8	2.7	8.9
Self-Consistency	70.1	76.2	2.4	15.6
Framework Baselines				
ReAct	69.8	75.1	2.5	18.7
Reflexion	73.4	79.2	2.8	22.3
ChatEval	75.9	81.2	3.1	25.1
RL-based Baselines				
RLHF	76.8	82.7	3.0	16.2
RLAIF	78.1	84.5	3.2	14.8
DRAFT-RL	**82.6**	**87.6**	**3.6**	**19.4**

Table 1: Code synthesis results on MBPP and HumanEval (% Pass@1). All models use Claude-3.5-Sonnet as the base LM.

Qualitative Error Reduction. We further analyzed 500 failed test cases. DRAFT-RL reduces: logic errors by 38%, syntax errors by 42%, and edge-case failures by 31%. Ex-

tended analysis of draft scaling and agent configuration trends is provided in Appendix Tables A1–A2 (Li et al. 2025). This confirms that multi-draft exploration is especially beneficial for programs requiring multi-step or non-greedy reasoning.

Mathematical Reasoning Performance Table 2 shows results on GSM8K and MATH, including fine-grained domain breakdowns. DRAFT-RL achieves consistent improvements across all mathematical domains, especially algebra (+3.9%) and geometry (+3.6%), which require structured symbolic manipulation.

Method	GSM8K	MATH	Algebra	Geometry	Calculus
Prompting Baselines					
CoT	84.3	42.7	51.2	38.4	35.9
CoD	87.1	45.9	54.8	41.7	39.2
Self-Consist.	85.7	44.2	52.9	39.8	37.5
Framework Baselines					
ReAct	83.9	43.1	50.6	39.2	36.8
Reflexion	88.4	47.3	56.1	43.9	41.7
ChatEval	89.7	49.2	58.4	45.1	43.8
RL-based Baselines					
RLHF	90.3	50.6	59.7	46.3	45.1
RLAIF	91.8	52.1	61.2	47.8	46.9
DRAFT-RL	**94.2**	**55.8**	**65.1**	**51.4**	**50.3**

Table 2: Mathematical reasoning accuracy across GSM8K and MATH. All values are %.

Error Behavior. Analysis of 300 reasoning traces shows: arithmetic errors reduced by 47%, conceptual errors by 35%, and incomplete reasoning chains by 52%. Reward-evolution dynamics supporting these results are shown in Appendix Table A4 (Li et al. 2025).

Knowledge-Intensive QA Performance Table 3 shows performance on HotpotQA and MMLU. Improvements come from better multi-hop reasoning: DRAFT-RL achieves an average hop count of 3.2 vs. 2.9 in RLAIF.

Method	EM	F1	MMLU-0S	MMLU-5S	Hops
CoT	67.2	79.4	71.8	74.3	2.1
CoD	69.8	81.7	73.5	76.1	2.3
Self-Cons.	68.5	80.3	72.7	75.2	2.2
ReAct	70.1	82.1	74.2	76.8	2.4
Reflexion	72.4	84.2	75.9	78.4	2.6
ChatEval	74.1	85.6	77.3	79.8	2.7
RLHF	75.3	86.9	78.1	80.7	2.8
RLAIF	76.7	87.4	79.2	81.5	2.9
DRAFT-RL	**79.1**	**90.5**	**82.4**	**84.7**	**3.2**

Table 3: Knowledge-intensive QA results. HotpotQA uses EM/F1; MMLU uses zero-shot and 5-shot accuracy.

A detailed breakdown of error categories beyond QA (e.g., factual vs. logical inconsistencies) is available in Appendix Table A4 (Li et al. 2025).

Comprehensive Ablation Studies

Component-wise Ablation Table 4 shows that removing Multi-Draft Generation results in the largest degradation across all tasks (–6.3% to –7.1%), confirming that diversified structured exploration is crucial.

Configuration	HumanEval	MATH	Hotpot-F1	MMLU
Full Model	**87.6**	**55.8**	**90.5**	**82.4**
w/o Drafts	80.5	48.7	84.2	76.8
w/o Peer Eval	83.8	51.3	87.1	79.2
w/o CoD	82.4	49.6	85.8	78.5
w/o Reward Model	84.1	52.2	88.3	80.1
w/o RL Training	81.7	50.4	86.4	77.9

Table 4: Component ablation across domains (% performance).

The full quantitative tables for draft scaling and agent configuration appear in Appendix Tables A1–A2 (Li et al. 2025).

Draft Quantity and Agent Configuration (Textual Integration) Increasing the number of drafts K beyond 5 yields diminishing returns: accuracy saturates at $K = 5$ while inference cost continues rising (8.2→38.5s). Using 3 agents yields the best balance between specialization and agreement. These findings align with the extended results presented in Appendix Tables A1–A2 (Li et al. 2025).

Training Dynamics and Convergence

Reward evolution trends supporting these observations are reported in Appendix Table A3 (Li et al. 2025).

Reasoning Quality and Interpretability

Table 5 shows human evaluation results: DRAFT-RL produces clearer, more complete, and more efficient reasoning chains.

Method	Clarity	Correct.	Completeness	Efficiency	Overall
CoT	3.7	3.4	3.9	3.2	3.6
Reflexion	3.9	3.8	4.1	3.5	3.8
RLAIF	4.0	4.1	4.0	3.7	3.9
DRAFT-RL	**4.3**	**4.4**	**4.2**	**4.0**	**4.2**

Table 5: Human evaluation of reasoning quality (5-point Likert scale).

A complete categorization of error types and reduction rates appears in Appendix Table A4 (Li et al. 2025).

Generalization and Transfer Learning

Table 6 reports strong cross-domain transfer: e.g., Math→Code yields a 5.8% absolute gain. Average transfer rate across domains is 69%, indicating that DRAFT-RL learns domain-general reasoning patterns.

Additional zero-shot and scalability results supporting these conclusions are provided in Appendix Tables A5–A6 (Li et al. 2025).

Train→Test	Base	Ours	Improv.	Transfer
Math→Code	68.4	74.2	+5.8	73%
Code→Math	71.7	76.9	+5.2	69%
QA→Math	69.2	73.8	+4.6	65%
Math→QA	72.1	77.3	+5.2	71%
QA→Code	66.8	71.4	+4.6	67%
Code→QA	70.3	75.1	+4.8	68%

Table 6: Cross-domain transfer performance.

Component	Time (s)	Mem. (GB)	FLOPs	%
Draft Gen.	12.8	3.2	8.4T	66%
Peer Eval.	4.1	0.8	2.1T	21%
Reward	1.9	0.4	0.9T	10%
Selection	0.6	0.1	0.2T	3%
Total	19.4	4.5	11.6T	100%

Table 7: Computation cost per query.

Computational Efficiency

Table 7 shows the per-query breakdown. Draft generation dominates (66%), but parallelization keeps wall-clock cost manageable.

For extended computational breakdowns and scaling analyses, refer to Appendix Tables A5–A6 (Li et al. 2025).

Conclusion

In this paper, we introduced DRAFT-RL, a framework that integrates Chain-of-Draft reasoning into multi-agent reinforcement learning to address key limitations of LLM-based reasoning systems. By enabling agents to generate diverse drafts, collaboratively evaluate them, and select solutions via a learned reward model, DRAFT-RL achieves 2.4–4.5% gains across code synthesis, symbolic mathematics, and knowledge-intensive QA, including a 3.7% improvement on MATH, while requiring 33–42% fewer training steps than strong RL baselines such as RLAIF and RLHF. These improvements stem from structured exploration under CoD constraints, peer-based error correction, and reward-aligned selection that induces emergent agent specialization.

References

Austin, J.; Odena, A.; Nye, M.; Bosma, M.; Michalewski, H.; Dohan, D.; Jiang, E.; Cai, C.; Terry, M.; Le, Q.; et al. 2021. Program Synthesis with Large Language Models. In *arXiv preprint arXiv:2108.07732*.

Bai, Y.; Kadavath, S.; Kundu, S.; Askell, A.; Kernion, J.; Jones, A.; Chen, A.; Goldie, A.; Mirhoseini, A.; McKinnon, C.; et al. 2022. Constitutional AI: Harmlessness from AI Feedback. *arXiv preprint arXiv:2212.08073*.

Besta, M.; Blach, N.; Kubicek, A.; Gerstenberger, R.; Podstawski, M.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Niewiadomski, H.; Nyczyk, P.; et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 17682–17690.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33: 1877–1901.

Chan, J.; Kalai, A.; Chen, J. X.; Zhang, Q. Y.; Yu, B.; Narasimhan, K.; Chang, H.; Liu, J.; Zhang, Z.; Hansen, L.; et al. 2023. ChatEval: Towards Better LLM-based Evaluators through Multi-Agent Debate. *arXiv preprint arXiv:2308.07201*.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2022. PaLM: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Dai, Y.; Wang, Z.; Kedia, N.; Freeman, C.; Kaplan, M.; Hu, E. J.; Stechlinski, P.; Wang, Z.; Dhariwal, P.; Henighan, T.; et al. 2024. Fine-tuning language models with reinforcement learning from AI critique. *arXiv preprint arXiv:2402.09972*.

Dou, S.; Liu, Y.; Jia, H.; Xiong, L.; Zhou, E.; Shen, W.; Shan, J.; Huang, C.; Wang, X.; Fan, X.; et al. 2024. Stepcoder: Improve code generation with reinforcement learning from compiler feedback. *arXiv preprint arXiv:2402.01391*.

Du, Y.; Li, S.; Torralba, A.; Tenenbaum, J. B.; and Mordatch, I. 2023. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*.

Gudibande, A.; Wallace, E.; Snell, C.; Geng, X.; Liu, H.; Abbeel, P.; Levine, S.; and Song, D. 2023. The false promise of imitating proprietary LLMs. *arXiv preprint arXiv:2305.15717*.

Hendrycks, D.; Basart, S.; Kadavath, S.; Mazeika, M.; Arora, A.; Guo, E.; Burns, C.; Puranik, S.; He, H.; Song, D.; et al. 2021. Measuring Coding Challenge Competence with APPS. In *Advances in Neural Information Processing Systems*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Le, H.; Wang, Y.; Gotmare, A. D.; Savarese, S.; and Hoi, S. C. H. 2022. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 21314–21328.

Lee, H.; Shlegeris, B.; Chan, E.; Grosse, R.; and Morris, J. X. 2023. RLAIF: Scaling Reinforcement Learning

from Human Feedback with AI Feedback. *arXiv preprint arXiv:2309.00267*.

Li, G.; Hammoud, H. A. A. K.; Itani, H.; Khizbullin, D.; and Ghanem, B. 2023. CAMEL: Communicative Agents for "Mind" Exploration of Large Scale Language Model Society. *arXiv preprint arXiv:2303.17760*.

Li, Y.; Liu, M.; Wang, H.; Zhang, Y.; Ma, Y.; and Tan, W. 2025. DRAFT-RL: Multi-Agent Chain-of-Draft Reasoning for Reinforcement Learning-Enhanced LLMs. arXiv:2511.20468.

Li, Z.; He, Y.; He, L.; Wang, J.; Shi, T.; Lei, B.; Li, Y.; and Chen, Q. 2024. FALCON: Feedback-driven Adaptive Long/short-term memory reinforced Coding Optimization system. *arXiv preprint arXiv:2410.21349*.

Liu, J.; Zhu, Y.; Xiao, K.; Fu, Q.; Han, X.; Yang, W.; and Ye, D. 2023. Rltf: Reinforcement learning from unit test feedback. *arXiv preprint arXiv:2307.04349*.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C. L.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training Language Models to Follow Instructions with Human Feedback. In *Advances in Neural Information Processing Systems*, volume 35, 27730–27744.

Paranjape, B.; Lundberg, S.; Singh, S.; Hajishirzi, H.; Zettlemoyer, L.; and Ribeiro, M. T. 2023. Art: Automatic multi-step reasoning and tool-use for large language models. *arXiv preprint arXiv:2303.09014*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shinn, N.; Cassano, F.; Gopinath, A.; Narasimhan, K.; and Yao, S. 2023. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*.

Shojaee, P.; Jain, A.; Tipirneni, S.; and Reddy, C. K. 2023. Execution-based code generation using deep reinforcement learning. *arXiv preprint arXiv:2301.13816*.

Stiennon, N.; Ouyang, L.; Wu, J.; Ziegler, D.; Lowe, R.; Voss, C.; Radford, A.; Amodei, D.; and Christiano, P. F. 2020. Learning to summarize from human feedback. *Advances in Neural Information Processing Systems*, 33: 3008–3021.

Tang, X.; Gao, J.; Xu, J.; Sun, T.; Song, Y.; Ezzini, S.; Ouédraogo, W. C.; Klein, J.; and Bissyandé, T. F. 2025a. SynFix: Dependency-aware program repair via Relation-Graph analysis. In *Findings of the Association for Computational Linguistics: ACL 2025*, 4878–4894.

Tang, X.; Kim, K.; Song, Y.; Lothritz, C.; Li, B.; Ezzini, S.; Tian, H.; Klein, J.; and Bissyandé, T. F. 2024. Codeagent: Autonomous communicative agents for code review. *arXiv preprint arXiv:2402.02172*.

Tang, X.; Klein, J.; and Bissyandé, T. F. 2025. Boosting Open-Source LLMs for Program Repair via Reasoning Transfer and LLM-Guided Reinforcement Learning. *arXiv preprint arXiv:2506.03921*.

Tang, X.; Olatunji, I. E.; Sun, T.; Klein, J.; and Bissyande, T. F. 2025b. Reinforcement Learning-Guided Chain-of-Draft for Token-Efficient Code Generation. *arXiv preprint arXiv:2509.25243*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozire, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, G.; Xie, Y.; Jiang, Y.; Mandlekar, A.; Xiao, C.; Zhu, Y.; Fan, L.; and Anandkumar, A. 2023. Voyager: An Open-Ended Embodied Agent with Large Language Models. *arXiv preprint arXiv:2305.16291*.

Wang, J.; Zhang, Z.; He, Y.; Song, Y.; Shi, T.; Li, Y.; Xu, H.; Wu, K.; Qian, G.; Chen, Q.; et al. 2024a. Enhancing Code LLMs with Reinforcement Learning in Code Generation. *arXiv preprint arXiv:2412.20367*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wang, Z.; Wei, J.; Yang, F.; Li, Y.; Qin, Y.; Tu, Z.; Yang, C.; Liu, Y.; Chen, K.-C.; Zhou, D.; et al. 2024b. Collaborative Reflection-Augmented Agents for Large Language Models. *arXiv preprint arXiv:2402.09*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; and Zhou, D. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, volume 35, 24824–24837.

Wu, Q.; Bansal, G.; Zhang, J.; Yang, Y.; Bursztyn, D.; Suchanek, J.; Kalai, A.; Zhu, W.; Koska, W.; Leskovec, J.; et al. 2023. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*.

Xu, S.; Xie, W.; Zhao, L.; and He, P. 2025. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2369–2380.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2023a. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36: 11809–11822.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023b. ReAct: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Yin, X.; Ni, C.; Wang, S.; Li, Z.; Zeng, L.; and Yang, X. 2024. Thinkrepair: Self-directed automated program repair. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*, 1274–1286.

Yuan, X.; Wang, X.; Wang, C.; Aggarwal, K.; Tur, G.; Hou, L.; Deng, N.; and Poon, H. 2023. Improving code generation by training with natural language feedback. *arXiv preprint arXiv:2303.16749*.