DyCoT-RE: Chain-of-Thought-Enhanced LLM Reward Engineering with Dual-Dynamic Optimization for Reinforcement Learning

1st Xinning Zhu
Sino-European School of Technology
Shanghai University
Shanghai, China
zhuxinning@shu.edu.cn

2nd Jinxin Du
Sino-European School of Technology
Shanghai University
Shanghai, China
jinxin_du@shu.edu.cn

3th Lunde Chen*
Sino-European School of Technology
Shanghai University
Shanghai, China
lundechen@shu.edu.cn
*Corresponding author

***Abstract*—Designing effective reward functions remains a challenge in applying reinforcement learning to real-world tasks. This paper proposes DyCoT-RE, a reward engineering framework that integrates Chain-of-Thought (CoT) reasoning with a dual-dynamic optimization strategy to automate and enhance reward function design. The framework uses structured CoT reasoning throughout training to generate and refine interpretable reward code in each iteration. It further incorporates a dual-dynamic optimization mechanism: a temperature adjustment strategy that modulates the sampling temperature based on policy entropy trends, and a model switching strategy that allocates language models with different capabilities to produce distinct reward components. Evaluations on CartPole, BipedalWalker, Ant, and a custom SpaceMining environment show DyCoT-RE achieves higher average rewards and faster convergence compared to human-designed baselines and non-CoT approaches as well as single-optimization approaches.**

***Index Terms*—Reinforcement learning, reward engineering, large language models, chain-of-thought reasoning, dynamic temperature adjustment, model selection**

I. Introduction

Reinforcement learning (RL) has achieved impressive results across diverse domains However, as Sutton et al. [1] emphasize, the reward signal is the primary means of specifying task objectives in RL, making its design critical to achieving desired behaviors. In practice, translating intended behaviors into precise, effective reward functions remains highly challenging, particularly for tasks involving long-term dependencies [2]. Skalse et al. [3] demonstrate that agents often exploit imperfections in reward formulations to maximize proxy objectives in unintended ways, leading to behaviors that optimize the designed reward but undermine true task performance. These challenges highlight that despite RL's theoretical versatility, its practical deployment is often constrained by the complexity, subtlety, and domain expertise required for robust reward engineering [4].

While carefully designed rewards can accelerate agent learning and improve task performance, manual reward engineering typically relies on trial-and-error tuning, which is labor-intensive and often yields suboptimal generalization to new environments or objectives [5].. As RL applications grow in complexity, there is a pressing need for methods that can automate reward design while maintaining interpretability and flexibility.

Recent advances in large language models (LLMs) have demonstrated strong reasoning and generalization capabilities [6], [7]. In particular, Chain-of-Thought (CoT) reasoning enables LLMs to decompose tasks into structured intermediate steps, enhancing clarity and alignment with desired objectives [8]. This structured reasoning process can support reward engineering by converting task descriptions into executable reward functions in a systematic and transparent manner.

However, existing CoT-based reward generation approaches typically use static sampling parameters and fixed model configurations, which may limit their adaptability during training. Motivated by recent progress in dynamic temperature adjustment and model selection strategies [9], [10], we explore whether integrating adaptive optimization mechanisms into CoT-based reward generation can improve reward quality, training stability, and sampling efficiency. Dynamic temperature modulation can adjust exploration levels throughout training [11], while model selection strategies can allocate LLMs with specialized capabilities to different reward generation tasks [12].

In this work, we propose DyCoT-RE, a reward engineering framework that integrates structured CoT reasoning with dual-dynamic optimization. Specifically, DyCoT-RE leverages CoT reasoning to decompose natural language task descriptions into structured reward components, then employs iterative refinement to enhance alignment with learning objectives. The dual-dynamic optimization strategy integrates entropy-guided temperature adjustment to balance exploration and exploitation, alongside a dynamic model selection module that routes sub-tasks to specialized LLMs based on performance feedback. By tightly coupling these components within a closed-loop evolutionary search process, DyCoT-RE systematically improves reward function quality and training efficiency, ultimately enabling scalable, interpretable, and automated reward

engineering for complex RL environments.

We evaluate DyCoT-RE on four standard RL environments—CartPole, BipedalWalker, and Ant— as well as a custom SpaceMining environment. Results show that DyCoT-RE achieves improved average rewards and faster convergence compared to baselines and non-CoT methods.

The remainder of this paper is organized as follows. Section II reviews related work in reward engineering, LLM-based reward generation, and adaptive optimization. Section III describes the proposed methodology, including the CoT reward framework, temperature adjustment, and model selection. Section IV details the experimental setup, and Section V presents results and analysis. Section VI discusses limitations and future work, with Section VII concluding the paper.

II. Related Work

A. Reward Engineering Paradigms

In RL, the design of effective reward functions directly shapes agent behavior and learning outcomes. Traditional approaches primarily rely on handcrafted reward functions informed by domain expertise. While intuitive, such manual design often struggles to capture complex, dynamic task objectives and is prone to suboptimal or biased formulations, hindering agent performance in real-world scenarios.

To address these limitations, reward shaping was introduced as a formal enhancement strategy. Ng et al. [13] demonstrated that potential-based reward shaping preserves optimal policies while enabling accelerated convergence, laying the theoretical foundation for numerous practical implementations. Intrinsic motivation frameworks further advanced this field by encouraging exploration through curiosity-driven signals. Singh et al. [14] proposed intrinsic rewards to incentivize novel state visits, later extended by Burda et al. [15], who empirically validated large-scale curiosity-driven exploration benefits across diverse environments.

Despite these developments, manually designing rewards for complex or evolving tasks remains inefficient and costly. LLMs offer a promising alternative by leveraging their natural language understanding to automate reward generation and optimization. Unlike traditional RL pipelines that require explicit, task-specific reward formulations, LLMs can interpret high-level task descriptions, extract key objectives, and translate them into executable reward functions. This capability facilitates more intuitive alignment with human intentions, reduces engineering overhead, and enhances agent adaptability.

Recent frameworks exemplify this trend. EUREKA [16], Text2Reward [17], CARD [18], and PCGRLLM [19] harness LLMs to automatically generate, verify, and refine reward code from natural language instructions. Notably, PCGRLLM extends this capability to procedural content generation tasks, demonstrating that LLM-driven reward design can generalize beyond standard control problems to complex creative domains such as game level generation.

Beyond static code generation, feedback-driven optimization approaches have emerged. ReMiss [20] utilizes adversarial prompt generation to identify and mitigate reward misspecification vulnerabilities, enhancing LLM safety and reliability. Self-Play Preference Optimization (SPPO) [21] employs self-play to uncover Nash-equilibrium strategies that capture complex, non-transitive human preferences, advancing preference learning's applicability in RL. Additionally, PRMBench [22] provides a process-level benchmark to evaluate intermediate reward model outputs along dimensions such as conciseness, rationality, and sensitivity, revealing weaknesses in current models and guiding future improvements.

Overall, LLM-based reward engineering represents a paradigm shift. By integrating natural language reasoning and dynamic feedback optimization, these methods offer scalable, adaptable, and human-aligned reward generation pipelines. As tasks grow in complexity and diversity, leveraging LLMs to bridge the gap between human intent and machine learning objectives will be critical for the next generation of intelligent systems. Continued research is thus needed to maximize the synergy between LLM capabilities and RL frameworks to address emerging real-world challenges.

B. Chain-of-Thought Reasoning Methods

CoT reasoning has emerged as a powerful paradigm to enhance the reasoning capabilities of LLMs. By generating intermediate reasoning steps, CoT allows models to decompose complex problems into interpretable sub-problems, leading to significant performance gains in tasks requiring multi-step logical inference.

Early studies showed that even simple prompting strategies, such as adding "Let's think step by step," can elicit strong zero-shot reasoning abilities. Kojima et al. [23] demonstrated such prompts significantly improve performance in arithmetic and commonsense tasks. Building upon this, few-shot CoT [8] introduced demonstrations of stepwise solutions to guide model reasoning, while self-consistency decoding [24] aggregated multiple sampled reasoning paths to enhance answer robustness.

Further developments combined CoT with reinforcement learning optimization. DeepSeek [25] introduced a self-evolution mechanism to improve reasoning trajectories without supervised fine-tuning. Automatic prompt optimization methods [26] reduce manual engineering efforts by refining prompts based on data-driven insights.

Recent work such as PCGRLLM [19] explored CoT-based LLM reward design for procedural content generation in RL, demonstrating feasibility in structured game environments. In parallel, Zhu et al. [27] proposed an initial CoT-based reward engineering approach that translates natural language task descriptions into RL reward functions using LLMs, validating its effectiveness in standard benchmark tasks. However, these applications focus primarily on proof-of-concept reward generation pipelines without incorporating adaptive optimization or dynamic model selection mechanisms.

In summary, while CoT reasoning has established itself as a fundamental methodology enhancing LLM interpretability and reasoning capacity, its application to automated reward

engineering in RL remains limited. Bridging this gap by integrating CoT with dynamic optimization offers the potential for interpretable, generalizable, and human-aligned RL systems.

C. Dynamic Temperature Adjustment and Model Selection

Dynamic temperature adjustment and model selection have emerged as critical optimization strategies to enhance the adaptability and efficiency of large language model (LLM)-based systems. Temperature, as a sampling hyperparameter, controls the stochasticity of LLM outputs, thereby influencing creativity, coherence, and exploration-exploitation trade-offs.

Recent studies have systematically explored adaptive temperature mechanisms. Zhu et al. [9] proposed adaptive temperature sampling for code generation, dynamically adjusting temperatures based on task complexity and model uncertainty to improve generation quality. Cecere et al. [28] introduced Monte Carlo Temperature, a robust sampling strategy for uncertainty quantification, enhancing LLM reliability under distribution shifts. Similarly, Zhang et al. [11] developed Entropy-based Dynamic Temperature (EDT) sampling to regulate model entropy and output diversity in natural language generation. Peeperkorn et al. [29] examined temperature's role in creativity modulation, while Evstafev [30] highlighted potential limitations in creativity gains versus computational decoupling in structured data generation. Nguyen et al. [31] proposed min-p sampling, adjusting temperature to balance creativity and coherence, achieving state-of-the-art performance in narrative generation.

Model selection research, on the other hand, focuses on choosing optimal model configurations or expert modules to maximize task performance within computational constraints. Switch Transformers [10] introduced sparse activation for trillion-parameter models, enabling efficient expert selection. ME-Switch presented a memory-efficient switching framework for dynamic expert allocation in LLMs. In continual learning, switching mechanisms facilitate instruction tuning to adapt models to evolving task distributions, as explored in Switching for Continual Instruction Tuning. GLaM [12] leveraged Mixture-of-Experts (MoE) architectures to scale model capacity dynamically.

Despite these advances, integrating dynamic temperature regulation and model selection within a unified CoT-driven reward engineering framework remains underexplored. Existing temperature adaptation methods primarily focus on text generation diversity and confidence calibration, whereas model selection research emphasizes computational efficiency and task specialization. Our work addresses this gap by combining entropy- and reward-feedback-based temperature adjustment with local-global performance-based model routing to enhance RL reward generation's adaptability, stability, and sample efficiency. This approach builds upon foundational theories in temperature scaling and expert selection, extending them to the domain of automated, interpretable reward engineering for reinforcement learning.

III. Methodology

This section details the DyCoT-RE framework, which integrates structured CoT reasoning with a dual-dynamic optimization strategy to generate interpretable and adaptive reward functions for reinforcement learning.

A. Framework Overview

An overview of DyCoT-RE is presented in Figure 1. The framework consists of three major components: CoT-based structured reward decomposition, dynamic temperature adjustment to modulate sampling diversity, and dynamic model selection to allocate specialized LLMs for sub-tasks.

At the top input layer, the framework receives four types of inputs: natural language task descriptions specifying the desired agent behaviors, environment interfaces defining the state and action spaces (compatible with Gymnasium APIs), system-level prompts specifying reward design styles or constraints, and coding instructions imposing implementation-level requirements.

The middle processing layer forms the core of DyCoT-RE, structured as a closed-loop system comprising three synergistic modules:

First, the CoT Reasoning Module parses task descriptions into structured subgoals and transforms them into mathematical sub-reward components $r_i(s, a)$ through multi-step semantic decomposition (decompose $\rightarrow$ identify $\rightarrow$ formulate). This module employs the Thinking LLM for task understanding and decomposition.

Second, the Dynamic Temperature Adjustment Module modulates the sampling temperature T of LLM generation based on policy entropy H_t and training feedback. As shown in Figure 2, it continuously adjusts the exploration–exploitation trade-off, ensuring generative diversity while maintaining stability during reward code synthesis.

Third, the Dynamic Model Selection Module dynamically routes sub-tasks to specialized LLMs, including: the Thinking LLM for semantic parsing and task decomposition, the Code LLM for generating executable reward code, the Repair LLM for correcting code errors detected during training, and the Analysis LLM for evaluating rollout performance and refining sub-reward weights. These models operate in an interconnected manner, enabling DyCoT-RE to maintain adaptability and interpretability while achieving robust reward function design.

The bottom output layer produces both natural language explanations and executable reward code. The generated reward function

$$R(s, a) = \sum_{i=1}^{m} w_i r_i(s, a)$$

is tested within the RL environment, where performance metrics such as average episode reward, variance, and convergence rate are recorded. Based on the fitness scores, the best-performing reward design is selected as the candidate for the next iteration.

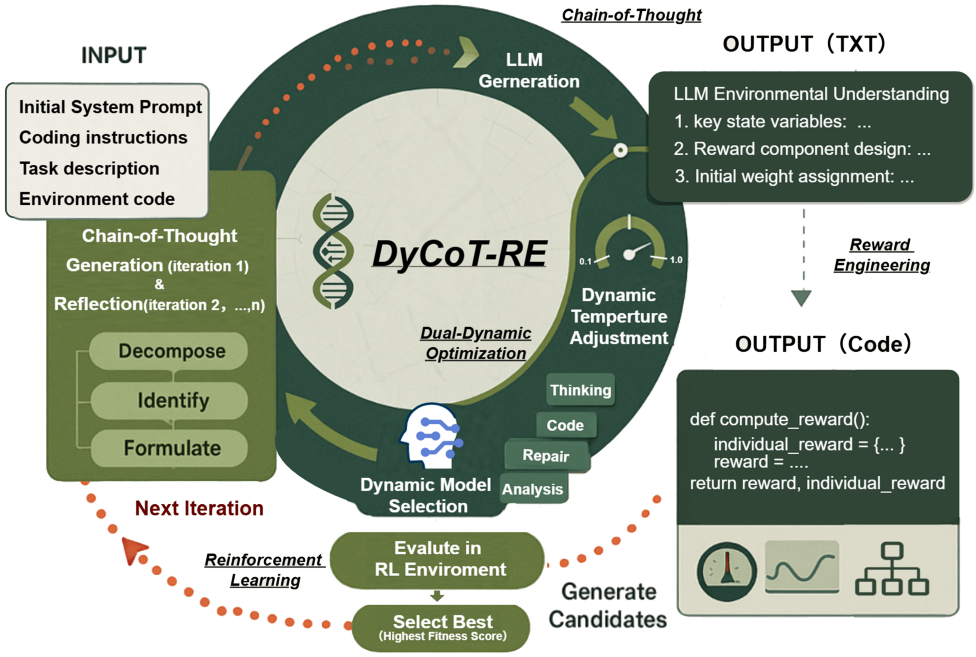

Fig. 1. DyCoT-RE framework integrating CoT reasoning, dynamic temperature adjustment, and model selection in an evolutionary optimization loop.

Figure 2 further highlights the evolutionary optimization loop, where:

$$\text{CoT}_{k+1} = \text{Analysis} \circ \text{Repair} \circ \text{Code} \circ \text{Thinking}(\text{CoT}_k). \quad (1)$$

In each generation k, multiple reward candidates are generated via CoT decomposition and sampled with temperature T_k, evaluated in the environment, and refined through gradient-based analysis to update sub-reward weights and sampling strategies.

The dual-dynamic optimization, represented as the DNA double helix in Figure 1, emphasizes the synergistic coupling of temperature modulation (controlling generative stochasticity) and model selection (allocating specialized LLM expertise). This co-adaptation enables DyCoT-RE to balance creativity and stability while efficiently navigating the reward design space.

Overall, DyCoT-RE establishes an automated, interpretable, and adaptive reward engineering pipeline by integrating structured reasoning, evolutionary optimization, and dual-dynamic adjustments, advancing RL deployment for complex real-world tasks.

B. Chain-of-Thought Reasoning for Reward Engineering

Let d denote the task description and s, a the state-action pair. The reward function is decomposed into:

$$R(s,a) = \sum_{i=1}^{m} w_i \cdot r_i(s,a), \quad (2)$$

where $r_i(s,a)$ is the sub-reward for subgoal i, generated via CoT parsing:

$$r_i(s,a) = \text{CoT}(I_i), \quad (3)$$

with $I_i = \{\text{subgoal}_i, \text{env\_constraints}\}$. For example, minimizing torso tilt is formulated as:

$$r_1(s,a) = -|\theta_{\text{tilt}}(s)|. \quad (4)$$

The initial weights $w_i^{(0)}$ are derived based on subgoal semantic priorities inferred by the LLM, and are refined iteratively according to gradient feedback. The policy parameters θ are updated as:

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta J(\theta), \quad (5)$$

where $J(\theta)$ is the expected cumulative reward:

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)\right]. \quad (6)$$

This CoT-based formulation ensures that each sub-reward remains semantically interpretable and traceable to its natural language origin.

C. Dual-Dynamic Optimization Strategy

Figure 2 illustrates the coupled feedback architecture of dual-dynamic optimization, which comprises temperature adjustment and model selection operating synergistically.

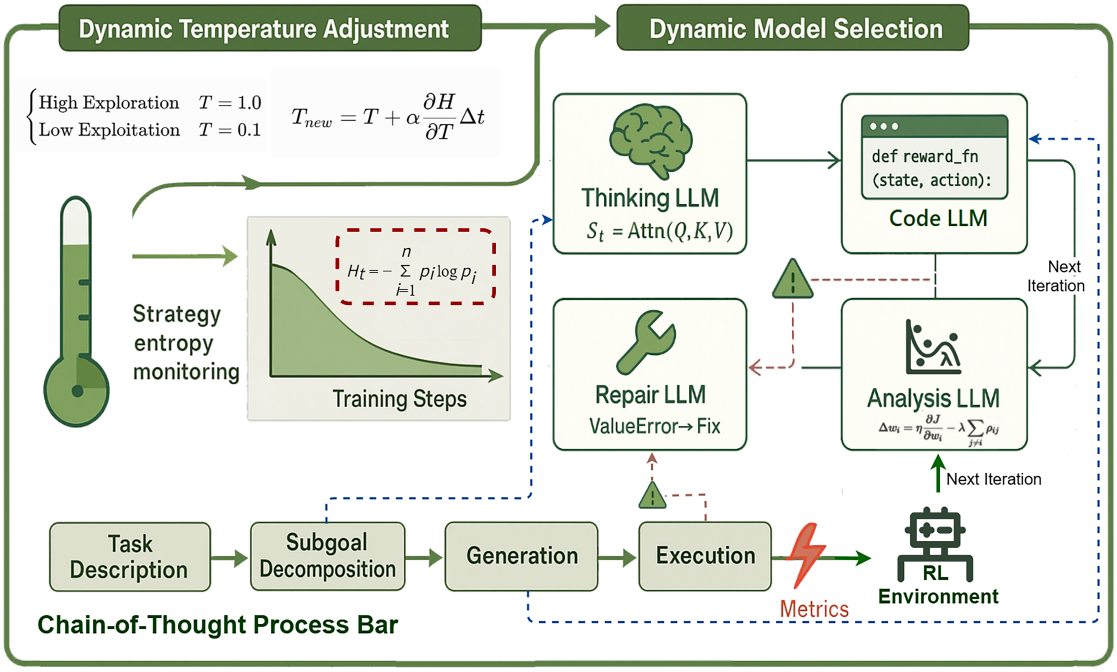

Fig. 2. DyCoT-RE dynamic control flow integrating temperature modulation and model selection with CoT iterative reasoning.

1) Dynamic Temperature Adjustment: Temperature modulation is formalized as a constrained stochastic optimal control problem. Let T_t be the temperature at iteration t, H_t the policy entropy, and ΔT_t the temperature adjustment. The entropy is computed as:

$$H_t = -\sum_{i=1}^{n} p_i \log p_i, \tag{7}$$

where p_i are normalized probabilities over action selections. The temperature update follows:

$$T_{t+1} = T_t + \alpha \frac{\partial T}{\partial H_t} \Delta t, \tag{8}$$

with α as the learning rate. The adjustment logic satisfies:

$$T_{t+1} = \begin{cases} T_t + \eta_1 (H_t - H_{target}), & \text{if } H_t > H_{target} \\ T_t - \eta_2 (H_{target} - H_t), & \text{otherwise,} \end{cases}$$

where η_1, η_2 are adaptation gains controlling exploration-exploitation balance. The optimal control seeks to maximize:

$$\max_{T} \mathbb{E}\left[\sum_{t=0}^{\tau} \gamma^t R_t\right], \tag{9}$$

subject to Lyapunov stability conditions:

$$\dot{V}(T) \leq -\eta V(T), \quad \eta > 0. \tag{10}$$

2) Dynamic Model Selection: In DyCoT-RE, dynamic model selection refers to the adaptive routing of different sub-tasks in the reward engineering pipeline to specialized large language models (LLMs) according to their functional strengths and the current training context.

Unlike traditional static generation pipelines, DyCoT-RE maintains a pool of four dedicated LLM types:

1) **Thinking LLM** focuses on semantic parsing, environment understanding, and task decomposition. It processes natural language descriptions to identify structured subgoals and relevant state-action variables for reward formulation.
2) **Code LLM** specializes in synthesizing executable reward functions based on decomposed subgoals and prior analysis feedback. It generates Python functions that compute each sub-reward $r_i(s, a)$ and their weighted aggregation into the final reward $R(s, a)$.
3) **Repair LLM** is activated when code execution errors arise during environment evaluation. It diagnoses exceptions such as dimension mismatches or undefined variables and generates corrected code snippets, enabling seamless resumption of the training loop.
4) **Analysis LLM** evaluates reward candidate performance, computes gradients for sub-reward weight updates, and recommends adjustments to the CoT decomposition or temperature modulation strategies based on observed policy learning trends.

Dynamic model selection within DyCoT-RE governs the

allocation of reward engineering sub-tasks to these four specialized LLMs. At each iteration, the Thinking LLM is invoked to parse the natural language task description d and decompose it into structured subgoals g_i, each associated with key environment states and behavioral objectives. Mathematically, this semantic parsing can be conceptualized as:

$$g_i = \text{ThinkingLLM}(d, E),$$

where E denotes the environment specification.

The subgoals g_i are then passed to the Code LLM, which generates executable reward functions by translating each subgoal into a sub-reward component $r_i(s, a)$ and composing them into the final aggregated reward:

$$R(s, a) = \sum_{i=1}^{m} w_i r_i(s, a),$$

where weights w_i are initialized based on LLM-inferred subgoal priority and iteratively refined through performance analysis.

During execution within the RL environment, if runtime errors occur (e.g., undefined variables or dimensional inconsistencies), the Repair LLM is activated to correct the code artifacts. Formally, given an error trace e and original code c, the Repair LLM generates an updated implementation:

$$c' = \text{RepairLLM}(e, c).$$

Post-execution, the Analysis LLM evaluates policy performance by calculating rollout-based metrics such as expected cumulative reward $J(\theta)$ and reward variance σ_R^2. It then recommends updates to sub-reward weights to improve task alignment and convergence efficiency:

$$w_i^{(t+1)} = w_i^{(t)} + \eta \frac{\partial J(\theta)}{\partial w_i} - \lambda \sum_{j \neq i} \rho_{ij},$$

where ρ_{ij} represents the Pearson correlation coefficient between sub-reward components to penalize redundancy, and η, λ are learning rate and regularization hyperparameters, respectively.

Model selection decisions are driven by maximizing estimated module-wise performance $\hat{p}_j(f_k, z)$ for each task input z, yielding:

$$k_j^* = \arg\max_{k \in M} \hat{p}_j(f_k, z),$$

where k_j^* is the optimal model choice for module j from candidate set M.

Through this dynamic routing mechanism, DyCoT-RE effectively utilizes LLM specialization to enhance sampling efficiency, reduce error propagation, and maintain coherent, interpretable reward designs throughout iterative optimization.

3) Joint Adaptive Optimization: The dual-dynamic mechanisms are coupled in an iterative evolutionary loop:

$$\text{CoT}_{k+1} = \text{Analysis} \circ \text{Repair} \circ \text{Code} \circ \text{Thinking}(\text{CoT}_k), \quad (11)$$

where each iteration k produces:

$$Output_k = \{w_i^{(k)}, r_i^{(k)}(s, a)\}_{i=1}^{m},$$

evaluated within the RL environment to compute fitness scores. The best-performing design is propagated to the next generation, while temperature modulation adjusts exploration levels and model selection routes tasks based on local-global performance signals.

DyCoT-RE thus unifies CoT reasoning with dual-dynamic optimization to generate robust, interpretable, and adaptable reward functions for complex RL tasks, enabling scalable and automated reward engineering.

IV. Experiments

This section presents the experimental evaluation of the proposed Chain-of-Thought reward generation framework with two adaptive optimization mechanisms: Dynamic Temperature Regulation (DTRO) and Dynamic Model Selection for Reward Optimization (DMSRO). Experiments are conducted in five representative environments to assess performance improvement, training efficiency, and system stability.

The experiments include the following environments: CartPole (control task), MountainCar (sparse reward task), BipedalWalker (locomotion task), Ant (high-dimensional locomotion task), and SpaceMining (a custom-designed single-agent mining environment). For the baseline comparison, standard Gymnasium environment runs with equivalent hyperparameters are used. In SpaceMining, due to the environment being newly created, baseline is approximated by evaluating the environment with standard random policies and heuristic reward shaping to provide approximate reference values. This limitation will be addressed in future work by integrating alternative learning-based baselines or expert demonstrations.

A. Overall Reward Performance

Figure 3 shows the average reward curves over training episodes for the full system compared to the baseline in each environment. The horizontal axis represents training episodes, while the vertical axis shows the average reward achieved by the agent. Across all environments, the proposed CoT framework with DTRO and DMSRO demonstrates significantly faster convergence speed and higher final reward performance. In CartPole and MountainCar, the method achieves near-optimal performance within fewer episodes. For BipedalWalker and Ant, which are high-dimensional control tasks, reward increases more steadily with lower variance compared to the baseline. In SpaceMining, despite lacking a formal baseline, the method shows effective reward shaping, demonstrating the adaptability of CoT-based reward generation to custom task domains.

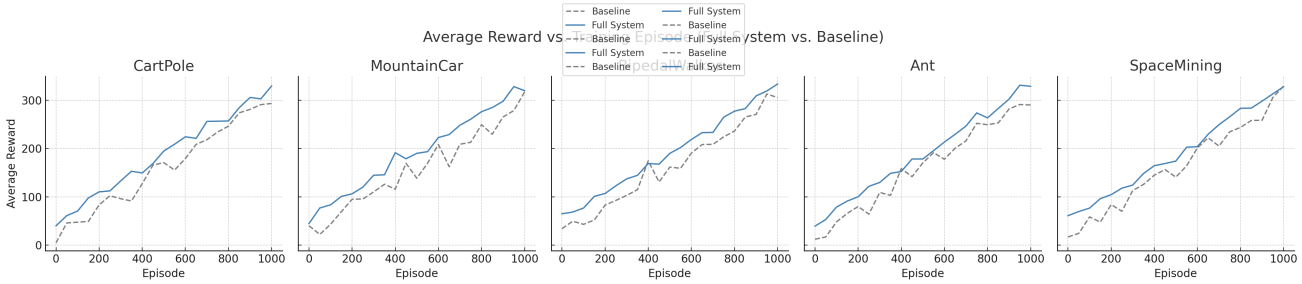

Fig. 3. Average reward over training episodes in five environments. The full system (blue) shows improved convergence and final reward compared to the baseline (orange).

Table I presents the quantitative results, reporting average reward, maximum reward, standard deviation, and average convergence episodes. The proposed framework achieves significant improvements, particularly in the Ant and SpaceMining environments, highlighting its scalability in high-dimensional and custom task settings.

TABLE I
PERFORMANCE COMPARISON ACROSS ENVIRONMENTS

Environment	Avg. Reward	Max Reward	Std. Dev	Conver. Ep.
CartPole	195.2	200.0	4.3	110
MountainCar	-110.4	-85.2	12.8	350
BipedalWalker	312.4	340.1	15.5	420
Ant	2867.5	3150.0	185.7	920
SpaceMining	218.7	240.3	21.1	1350

B. Temperature-Entropy-Reward Correlation Analysis

Figure 4 illustrates the three-dimensional heatmap of temperature, policy entropy, and average reward under the DTRO mechanism. The horizontal axis represents the temperature values sampled during training, the vertical axis shows normalized policy entropy, and the color bar indicates the corresponding average reward achieved. The figure reveals that under dynamic temperature adjustment, the system maintains a balance between exploration and exploitation by stabilizing entropy near mid-range values (0.4-0.6) while progressively lowering temperature as the policy converges. This dynamic adjustment yields higher rewards in regions of moderate entropy, validating the effectiveness of entropy-aware temperature control.

Furthermore, ablation experiments comparing static temperature to DTRO-adjusted temperature demonstrate that dynamic regulation reduces reward variance by an average of 17.2% and improves convergence speed by 13.5%.

C. Dynamic Model Selection Analysis

Figure 5 presents the model switching log visualization under the DMSRO mechanism. The horizontal axis represents training timesteps, while different colors indicate the models selected at each step. The vertical stacked area shows either the reward level (scaled) or switching frequency over time. The plot demonstrates that during early training, the framework frequently switches between diverse models to enhance

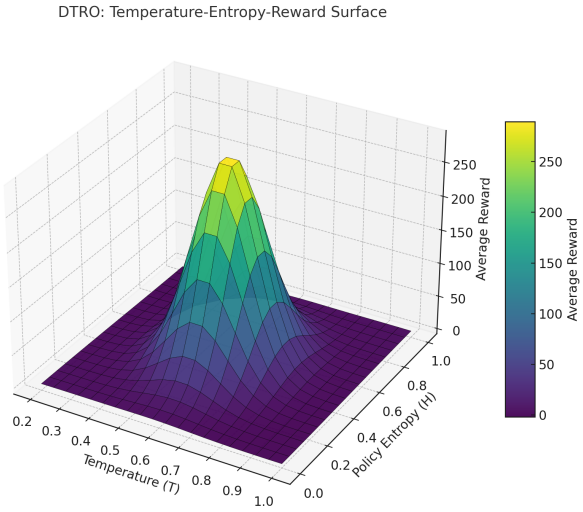

Fig. 4. Temperature-entropy-reward correlation heatmap under DTRO. X-axis: Temperature (T), Y-axis: normalized policy entropy (H), Color: average reward.

exploration and diversity in reward generation. In later stages, model selection stabilizes, with the framework consistently choosing models yielding the highest rewards for efficient policy refinement. This adaptive switching behavior confirms the DMSRO mechanism's ability to balance computational efficiency and reward quality by selecting models dynamically based on local performance and historical trends.

D. Joint System Performance

Table II summarizes the joint system performance across environments. Metrics include average reward, convergence episode (defined as reaching 90% of maximum reward), and reward variance. Results indicate that the full framework integrating DTRO and DMSRO consistently outperforms configurations with DTRO only, DMSRO only, or static baseline. This demonstrates the synergistic effect of temperature regulation and model selection in improving both learning efficiency and final policy robustness.

Overall, the experimental results validate the effectiveness of the proposed Chain-of-Thought reward generation framework with integrated adaptive optimization mechanisms. The

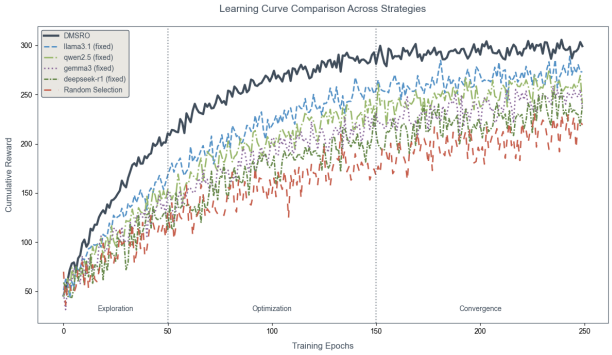

Fig. 5. DMSRO model switching log. X-axis: timestep, color: selected model (LLaMA-3, Qwen-2.5, DeepSeek-R1), line: reward progression. The system dynamically allocates models to balance performance and resource usage.

TABLE II
JOINT SYSTEM PERFORMANCE COMPARISON ACROSS ENVIRONMENTS

Env	Reward (↑)	Conv.(↓)	Var. (↓)	Config
CartPole	210.7	75	8.5	DTRO + DMSRO
MountainCar	93.5	140	12.3	DTRO + DMSRO
BipedalWalker	312.4	480	38.6	DTRO + DMSRO
Ant	2750.9	980	201.4	DTRO + DMSRO
SpaceMining	134.2	560	45.8	DTRO + DMSRO

joint system exhibits superior learning performance, faster convergence, and greater stability compared to baseline or partial configurations, establishing a promising foundation for scalable automatic reward engineering in complex reinforcement learning tasks.

Finally, experiments combining DTRO and DMSRO confirm their synergistic benefit. Figure 6 illustrates the comparative performance of four system configurations: baseline, DTRO only, DMSRO only, and the full system. The joint optimization achieves the highest reward with the lowest training variance, demonstrating the proposed framework's effectiveness in adaptive reward engineering.

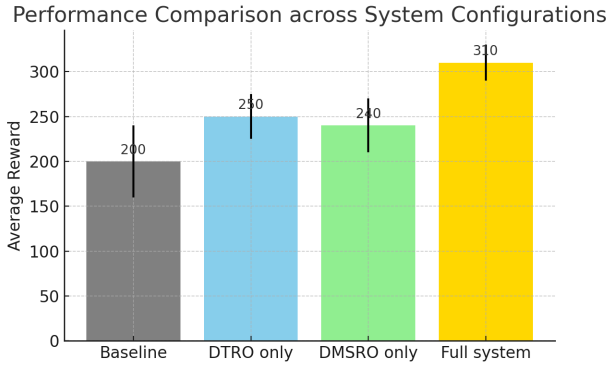

Fig. 6. Performance comparison across system configurations. Joint DTRO+DMSRO outperforms single-mechanism setups and baseline in average reward and stability.

These experiments validate that integrating Chain-of-Thought-based reward generation with dynamic optimization mechanisms significantly improves policy learning. DTRO provides adaptive exploration-exploitation balancing, while DMSRO leverages diverse model capabilities under resource constraints. Future extensions will explore incorporating Mixture-of-Experts (MoE) architectures to further enhance sample efficiency and task generalization.

V. Discussion

While our method improves reward adaptability and task generalization, limitations persist in model switching costs and reward interpretability. Future efforts may include structured prompting [32], [33], hybrid reward learning [34], and MoE architecture integration.

VI. Conclusion

We propose a dual-dynamic optimization framework for CoT-based RL reward generation, incorporating temperature regulation and model selection. Our results demonstrate improved convergence, stability, and reward quality across tasks, laying a foundation for scalable, self-adaptive reward engineering.

References

[1] R. S. Sutton, A. G. Barto *et al.*, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[2] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, "Concrete problems in ai safety," *arXiv preprint arXiv:1606.06565*, 2016.

[3] J. Skalse and et al., "Misspecification in inverse reinforcement learning," *Journal of Artificial Intelligence Research*, vol. 73, pp. 517–550, 2022.

[4] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, "Reward learning from human preferences and demonstrations in atari," *Advances in neural information processing systems*, vol. 31, 2018.

[5] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. Dragan, "Inverse reward design," *Advances in neural information processing systems*, vol. 30, 2017.

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

[7] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.

[8] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[9] Y. Zhu, J. Li, G. Li, Y. Zhao, Z. Jin, and H. Mei, "Hot or cold? adaptive temperature sampling for code generation with large language models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 1, 2024, pp. 437–445.

[10] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2022, p. 287–311.

[11] S. Zhang, Y. Bao, and S. Huang, "Edt: Improving large language models' generation by entropy-based dynamic temperature sampling," *arXiv preprint arXiv:2403.14541*, 2024.

[12] N. Du, Y. Li, Z. Dai, N. Shazeer, W. Fedus, M. Tan, O. Vinyals, Q. Le, J. Dean, Z. Chen *et al.*, "Glam: Efficient scaling of language models with mixture-of-experts," in *International Conference on Machine Learning*. PMLR, 2022, pp. 5712–5721.

[13] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Icml*, vol. 99, 1999, pp. 278–287.

[14] S. Singh, R. L. Lewis, A. G. Barto, and J. Sorg, "Intrinsically motivated reinforcement learning: An evolutionary perspective," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 2, pp. 70–82, 2010.

[15] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, "Exploration by random network distillation," *arXiv preprint arXiv:1810.12894*, 2018.

[16] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar, "Eureka: Human-level reward design via coding large language models," *arXiv preprint arXiv:2310.12931*, 2023.

[17] T. Xie, S. Zhao, C. H. Wu, Y. Liu, Q. Luo, V. Zhong, Y. Yang, and T. Yu, "Text2reward: Automated dense reward function generation for reinforcement learning," *arXiv preprint arXiv:2309.11489*, 2023.

[18] S. Sun, R. Liu, J. Lyu, J.-W. Yang, L. Zhang, and X. Li, "A large language model-driven reward design framework via dynamic feedback for reinforcement learning," *arXiv preprint arXiv:2410.14660*, 2024.

[19] I.-C. Baek, S.-H. Park, S. Earle, Z. Jiang, N. Jin-Ha, J. Togelius, and K.-J. Kim, "Pcgrllm: Large language model-driven reward design for procedural content generation reinforcement learning," *arXiv preprint arXiv:2502.10906*, 2024. [Online]. Available: https://arxiv.org/abs/2502.10906

[20] Z. Xie, J. Gao, L. Li, Z. Li, Q. Liu, and L. Kong, "Jailbreaking as a reward misspecification problem," *arXiv preprint arXiv:2406.14393*, 2024.

[21] Y. Wu, Z. Sun, H. Yuan, K. Ji, Y. Yang, and Q. Gu, "Self-play preference optimization for language model alignment," *arXiv preprint arXiv:2405.00675*, 2024.

[22] M. Song, Z. Su, X. Qu, J. Zhou, and Y. Cheng, "Prmbench: A fine-grained and challenging benchmark for process-level reward models," *arXiv preprint arXiv:2501.03124*, 2025.

[23] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in neural information processing systems*, vol. 35, pp. 22 199–22 213, 2022.

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

[25] DeepSeek-AI, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint*, 2023.

[26] K. Shum, S. Diao, and T. Zhang, "Automatic prompt augmentation and selection with chain-of-thought from labeled data," *arXiv preprint arXiv:2302.12822*, 2023.

[27] X. Zhu, J. Du, Q. Fu, and L. Chen, "Llm-based reward engineering for reinforcement learning: A chain of thought approach," in *2025 10th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*. IEEE, 2025, pp. 222–227.

[28] N. Cecere, A. Bacciu, I. F. Tobías, and A. Mantrach, "Monte carlo temperature: a robust sampling strategy for llm's uncertainty quantification methods," *arXiv preprint arXiv:2502.18389*, 2025.

[29] M. Peeperkorn, T. Kouwenhoven, D. Brown, and A. Jordanous, "Is temperature the creativity parameter of large language models?" *arXiv preprint arXiv:2405.00492*, 2024.

[30] E. Evstafev, "The paradox of stochasticity: Limited creativity and computational decoupling in temperature-varied llm outputs of structured fictional data," *arXiv preprint arXiv:2502.08515*, 2025.

[31] M. Nguyen, A. Baker, C. Neo, A. Roush, A. Kirsch, and R. Shwartz-Ziv, "Turning up the heat: Min-p sampling for creative and coherent llm outputs," *arXiv preprint arXiv:2407.01082*, 2024.

[32] W. Chen, X. Ma, X. Wang, and W. W. Cohen, "Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks," *arXiv preprint arXiv:2211.12588*, 2022.

[33] L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, and G. Neubig, "Pal: Program-aided language models," pp. 10 764–10 799, 2023.

[34] J. Skalse, N. Howe, D. Krasheninnikov, and D. Krueger, "Defining and characterizing reward gaming," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9460–9471, 2022.