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

Xinning Zhu[a,1], Jinxin Du[a], Longfei Huang[a], Lunde Chen[a,*]

[a]*Sino-European School of Technology, Shanghai University, Shanghai, China*

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.

Keywords: Reinforcement learning, reward engineering, large language models, chain-of-thought reasoning, dynamic temperature adjustment, model selection

*Corresponding author
Email: lundechen@shu.edu.cn

1. 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, CoT reasoning enables LLMs to decompose tasks into structured intermediate steps, enhancing clarity and alignment with desired objectives. 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. Recent studies have emphasized the need for dynamic adaptation in LLM-based systems to improve sample efficiency, stability, and alignment with task objectives. For example, Nguyen et al. [8] demonstrate

that min-p sampling enhances creativity while preserving coherence in narrative generation tasks, while Peeperkorn et al. [9] analyze temperature as a direct modulator of LLM creativity. Moreover, Fedus et al. [10] and Du et al. [11] show that mixture-of-experts architectures can scale model capacity efficiently via adaptive routing, motivating our exploration of combining temperature modulation with expert model selection for reward engineering.

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 the standard RL environments CartPole, BipedalWalker, and Ant to demonstrate its effectiveness. However, recent studies have raised concerns that large language models may carry prior knowledge from pretraining data about these standard environments, leading to potential prompt leakage and evaluation biases [12, 13, 14]. To address this issue, we additionally design a custom SpaceMining[1] environment to assess DyCoT-RE's true generalization capabilities on tasks free from such pretrained knowledge. Experimental results show that DyCoT-RE consistently achieves higher average rewards and faster convergence compared to baseline 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

[1]Project website: `https://lola-jo.github.io/space_mining/`.

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.

2. Related Work

2.1. 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. [15] 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. [16] proposed intrinsic rewards to incentivize novel state visits, later extended by Burda et al. [17], 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 [18], Text2Reward [19] and CARD [20] harness LLMs to automatically generate, verify, and refine reward code from natural language instructions.

Beyond static code generation, feedback-driven optimization approaches have emerged. ReMiss [21] utilizes adversarial prompt generation to identify and mitigate reward misspecification vulnerabilities, enhancing LLM safety and reliability. Self-Play Preference Optimization (SPPO) [22] employs self-play to uncover Nash-equilibrium strategies that capture complex, non-transitive human preferences, advancing preference learning's applicability in RL. Additionally, PRMBench [23] 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 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.

2.2. 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. [24] demonstrated such prompts significantly improve performance in arithmetic and commonsense tasks. Building upon this, few-shot CoT [25] introduced demonstrations of stepwise solutions to guide model reasoning, while self-consistency decoding [26] aggregated multiple sampled reasoning paths to enhance answer robustness.

Further developments combined CoT with reinforcement learning optimization. DeepSeek [27] introduced a self-evolution mechanism to improve reasoning trajectories without supervised fine-tuning. Automatic prompt optimization methods [28] reduce manual engineering efforts by refining prompts based on data-driven insights.

Recent work such as PCGRLLM [29] explored CoT-based LLM reward design for procedural content generation in RL, demonstrating feasibility in structured game environments. In parallel, Zhu et al. [30] 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 reasoning with dynamic optimization holds promise for enhancing the interpretability and adaptability of RL systems.

2.3. 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 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. [31] proposed AdapT, an adaptive temperature sampling strategy for code generation tasks, which dynamically adjusts decoding temperature based on token-level difficulty to improve generation quality. Zhang et al. [32] developed Entropy-based Dynamic Temperature (EDT) sampling to regulate output entropy and diversity in natural language generation, while Cecere et al. [33] introduced Monte Carlo Temperature as a robust sampling strategy to enhance uncertainty quantification under distribution shifts. Chang et al. [34] l everaged KL-divergence-guided temperature sampling to modulate exploration adaptively. Additionally, Peeperkorn et al. [9] analyzed temperature as a creativity modulator in LLMs, whereas Evstafev [35] discussed potential limitations of temperature-based stochasticity in structured data generation. Nguyen et al. [8] proposed min-p sampling to balance creativity and coherence, achieving improved narrative generation performance.

For model selection, recent work has focused on choosing optimal model configurations or expert modules to maximize task performance within computational constraints. Switch Transformers [10] introduced sparse activation mechanisms, enabling efficient expert selection at trillion-parameter scale, while GLaM [11] leveraged Mixture-of-Experts (MoE) architectures to dynamically scale model capacity. Zhou et al. [36] proposed expert choice routing to improve mixture-of-experts efficiency, and Li et al. [37] introduced preference-conditioned dynamic routing for cost-efficient LLM generation. Hu et al. [38] presented Dynamic Ensemble Reasoning to integrate outputs from multiple specialized LLMs, enhancing system robustness. Nakaishi et al. [39] further revealed phase transition behaviors in LLM sampling regimes, informing temperature scaling strategies. Furthermore, Li et al. [40] revisited self-consistency decoding from a distributional alignment perspective, offering insights relevant to expert aggregation and answer aggregation stability.

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

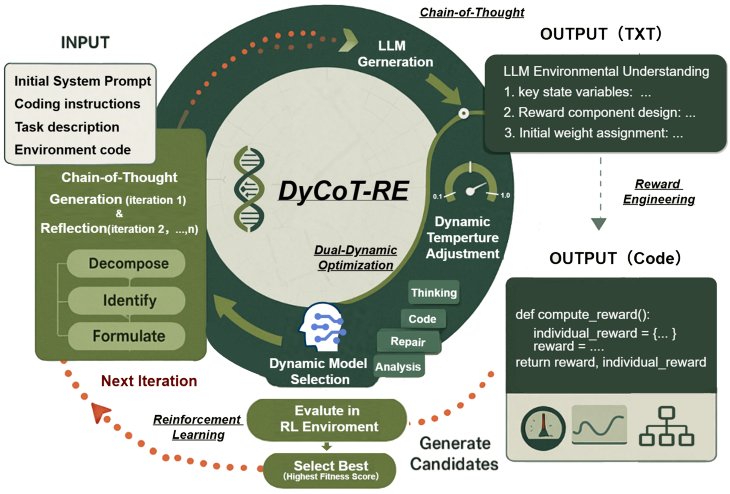

Figure 1: DyCoT-RE framework integrating CoT reasoning, dynamic temperature adjustment, and model selection in an evolutionary optimization loop.

focus on text generation diversity and calibration, whereas model selection research emphasizes computational efficiency and 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.

3. 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.

3.1. Framework Overview

Figure 1 presents an overview of DyCoT-RE, which integrates three key components: structured Chain-of-Thought (CoT) reward decomposition, dynamic temperature adjustment, and dynamic model selection.

The framework receives four inputs: natural language task descriptions specifying agent behaviors, environment interfaces defining state-action spaces, system-level prompts for reward design constraints, and coding instructions for implementation.

At its core, employs four types of LLMs: Thinking, Code, Repair, and Analysis LLMs, hereafter denoted as $\mathcal{M}_{\text{think}}$, $\mathcal{M}_{\text{code}}$, $\mathcal{M}_{\text{repair}}$, and $\mathcal{M}_{\text{analysis}}$ respectively.

These models operate in an interconnected closed-loop pipeline to produce reward functions

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

which are evaluated in the RL environment. The performance metrics guide subsequent optimization iterations.

Overall, DyCoT-RE establishes an adaptive and interpretable reward engineering framework by integrating structured reasoning with dual-dynamic optimization strategies.

3.2. Chain-of-Thought Reasoning for Reward Engineering

Let d denote the task description and (s,a) the state-action pair. As defined in Eq. (1), the reward function $r_i(s,a)$ is the sub-reward for subgoal i, generated via CoT parsing:

$$r_i(s,a) = \text{CoT}(I_i), \tag{2}$$

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

$$r_1(s, a) = -|\theta_{\text{tilt}}(s)|. \tag{3}$$

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:

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta J(\theta), \tag{4}$$

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

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t) \right]. \tag{5}$$

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

3.3. Dual-Dynamic Optimization Strategy

Figure 2 illustrates the coupled feedback architecture of DyCoT-RE, where temperature adjustment and model selection operate in synergy to enhance reward function generation and RL performance.

3.3.1. Dynamic Temperature Adjustment

Temperature adjustment modulates the sampling temperature T based on policy entropy H, confidence C, and performance R. Entropy reflects generative diversity, confidence measures output stability, and performance evaluates reward improvement relative to historical best.

The update rule is formulated as:

$$T_{k+1} = \text{clip}\left(T_k + \alpha \frac{\partial T}{\partial H_k} \Delta t \right), \tag{6}$$

where α is a learning rate, and the gradient $\frac{\partial T}{\partial H_k}$ reflects entropy-driven adjustment. An alternative implementation combines smoothing and multiplicative adjustment:

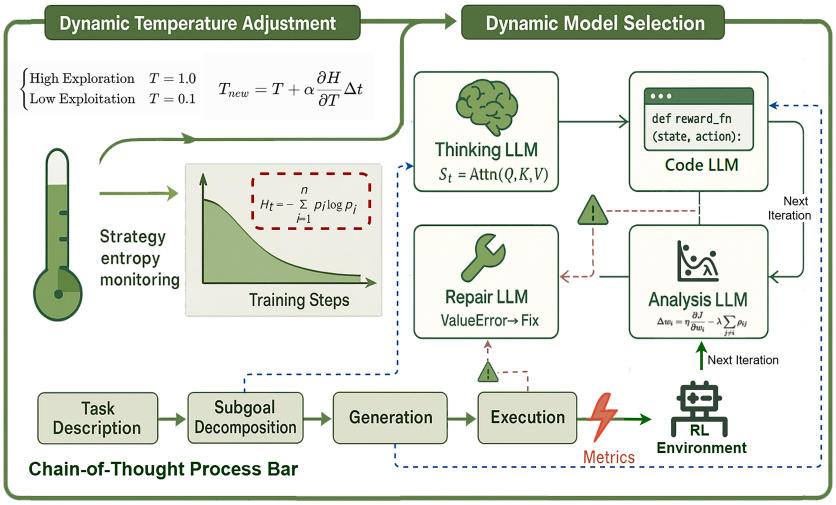

Figure 2: DyCoT-RE control flow integrating temperature modulation and model selection within CoT iterative reasoning.

$$T_{k+1} = \text{clip}\left(\alpha T_k + (1-\alpha) T_k f(H_k, C_k, R_k)\right). \tag{7}$$

Here, $f(H, C, R)$ integrates:

$$f(H, C, R) = f_R(R) f_H(H) f_C(C), \tag{8}$$

where f_R ensures performance protection, f_H regulates entropy bounds, and f_C maintains confidence stability.

3.3.2. Dynamic Model Selection

Dynamic model selection adaptively routes sub-tasks to specialized LLMs, leveraging their complementary strengths across the reward engineering pipeline.

The four LLM classes include: $\mathcal{M}_{\text{think}}$ for task understanding and semantic decomposition, $\mathcal{M}_{\text{code}}$ for reward function synthesis, $\mathcal{M}_{\text{repair}}$ for code correction, and $\mathcal{M}_{\text{analysis}}$ for performance evaluation and sub-reward weight updates.

Formally, the decomposition and generation process can be expressed as:

$$g_i = \mathcal{M}_{\text{think}}(d, E), \tag{9}$$

$$w_i = \mathcal{M}_{\text{analysis}}(R_i), \tag{10}$$

$$r_i(s, a) = \mathcal{M}_{\text{code}}(g_i, w_i), \tag{11}$$

where d is the task description and E the environment specification. Repair LLM intervenes when $\mathcal{M}_{\text{code}}$ outputs execution errors during evaluation.

Model selection at iteration k follows:

$$M_{k+1} = \begin{cases} \arg\max_{m \in \mathcal{M}_s} \text{Perf}(m), & 1 - \epsilon, \\ \text{Random}(\mathcal{M}_s \setminus \{M_k\}), & \epsilon, \end{cases} \tag{12}$$

where $\mathcal{M}_s$ is the model pool for stage s, $\text{Perf}(m)$ the performance score, and ϵ the exploration rate ensuring selection diversity.

At each iteration, the next model M_{k+1} is selected by choosing the highest-performing model from the pool $\mathcal{M}_s$ with probability $1-\epsilon$, or randomly selecting another model with probability ϵ. This pool includes the Repair LLM, which is triggered when error correction is needed during reward code evaluation.

3.3.3. Joint Adaptive Optimization

Temperature adjustment and model selection form a joint adaptive optimization loop. Temperature T influences the sampling distribution of reward candidates:

$$r_i(s, a) \sim p(r_i | g_i, T), \tag{13}$$

where $g_i = \mathcal{M}_{\text{think}}(d, E)$ denotes the subgoal generated by the Thinking LLM given task description d and environment E. This sampling process modulates policy entropy and, consequently, cumulative rewards.

Concurrently, model selection determines the semantic decomposition quality, code correctness, error repair, and performance evaluation, thereby shaping the expressivity and effectiveness of reward functions.

The combined objective of temperature adjustment and model selection is to maximize:

$$J(\theta; T, M) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t R_{T,M}(s_t, a_t)\right], \tag{14}$$

where $R_{T,M}(s, a)$ is the reward function generated under temperature T and model configuration M.

This joint optimization is formalized as:

$$(T^*, M^*) = \arg\max_{T,M} J(\theta; T, M). \tag{15}$$

Overall, this interconnected loop iteratively adapts both sampling temperature and model selection strategies to maximize the expected RL objective as defined in Eq. (5), ensuring stable, diverse, and effective reward engineering throughout the training process.

4. Experiments

4.1. Experimental Setup

DyCoT-RE is evaluated on four reinforcement learning environments, which span diverse task difficulties, action spaces, and generalization challenges.

Standard benchmarks include CartPole, BipedalWalker, and Ant, covering discrete control tasks and high-dimensional continuous locomotion.

To assess generalization beyond pretrained benchmark knowledge, we introduce **SpaceMining**, a custom-designed environment characterized by multi-objective resource collection, sparse and delayed rewards, and dynamic energy constraints. Unlike standard benchmarks with canonical reward patterns, SpaceMining lacks prior reward templates and was constructed specifically to test whether LLM-based reward generation—such as DyCoT-RE—can succeed

through pure task and code-level reasoning. A detailed analysis of DyCoT-RE's performance and reward decomposition design in SpaceMining is provided in Section 4.4.

SpaceMining is a custom environment introduced in this work to evaluate generalization beyond pretrained corpora. Unlike standard benchmarks, it lacks any preexisting reward functions or prior task-specific heuristics. The agent observes partial information about its position, energy, and surrounding asteroids (total 53D), and must produce continuous thrust and mining actions (3D) to collect resources efficiently. This setup explicitly tests whether LLM-based reward design can succeed through pure task and code understanding, without prior dataset bias.

4.1.1. Baselines and Comparison Strategies

To contextualize DyCoT-RE's performance, we evaluate against two primary baselines:

(1) Human-designed rewards Manually engineered reward functions from standard Gymnasium implementations (CartPole, BipedalWalker, Ant) and custom heuristics for SpaceMining. These represent conventional RL practice with expert prior knowledge.

(2) Eureka A LLM-based framework for automated reward synthesis, using code generation and symbolic verification. We adapt its pipeline to Gymnasium-based evaluations for fair comparison.

Ablation Comparisons To evaluate individual contributions within DyCoT-RE, we include the following controlled variants:

- **Zero-shot reward generation**: Reward generation using direct prompts without CoT reasoning.

- **Fixed temperature**: Sampling temperature is held constant throughout training, with results reported across a sweep of static values. This setup contrasts with DyCoT-RE's entropy-guided temperature modulation.

- **Fixed model backend**: A single large language model is statically selected for all reward generation, without dynamic model switching. We report results

across a range of representative LLMs for fair comparison.

These comparisons allow us to isolate the impact of CoT-based reasoning and the dual-dynamic mechanisms in DyCoT-RE.

4.1.2. Implementation Details

All experiments use Proximal Policy Optimization from Stable-Baselines3, with hyperparameters summarized in Table **??**.

Training uses the Adam optimizer with a learning rate of 0.0003, batch size 64, $\lambda = 0.95$, and $\gamma = 0.999$.

Reward functions are generated by DyCoT-RE with a local LLM deployed via Ollama. Each chain-of-thought iteration samples eight candidate rewards in parallel with a temperature of 0.6 to balance diversity and coherence.

Final performance is reported as the mean over ten test episodes with different seeds.

Environment	**State Dim**	**Action Dim**	**Max Steps**	**Task Type**
CartPole-v1[2]	4	1	500	Balance
BipedalWalker-v3[3]	24	4[c]	1600	Locomotion
Ant-v5[4]	111	8[c]	1600	Multijoint Movement
SpaceMining(Custom)	53	3[c]	1200	Resource Navigation

4.2. Performance Across Environments

We compare DyCoT-RE against Human-designed and Eureka baselines across four reinforcement learning environments: CartPole, BipedalWalker, Ant, and SpaceMining, each increasing in task complexity and generalization difficulty.

Figure 3 summarizes the final average rewards. While all methods perform similarly in CartPole, DyCoT-RE begins to show clear advantages in BipedalWalker and Ant, and significantly outperforms alternatives in the custom-designed SpaceMining environment. These results suggest DyCoT-RE scales better with increasing task difficulty, especially when sparse feedback and high-dimensional state-action spaces challenge handcrafted or single-pass reward generation.

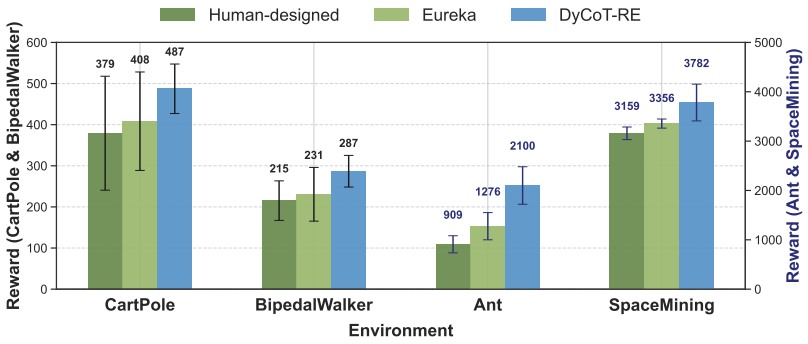

Figure 3: Final average rewards across environments: DyCoT-RE (blue), Eureka (light grass), and Human-designed (dark grass).

Figure 4 provides deeper insights into the learning dynamics. In CartPole, all methods quickly converge due to its low-dimensional discrete dynamics, though DyCoT-RE converges slightly faster. In BipedalWalker, DyCoT-RE shows a two-phase learning pattern: initial exploration volatility followed by stable growth and smoother long-term convergence. The Ant environment, characterized by high-dimensional continuous control, exposes DyCoT-RE's strengths most clearly—where baseline methods plateau early, DyCoT-RE sustains improvement well into later training stages. In SpaceMining, DyCoT-RE maintains steady learning while baselines exhibit flat or highly unstable performance, indicating its capacity to generalize in tasks absent from pretraining corpora.

Table **??** presents three complementary metrics: final reward (effectiveness), early success rate (efficiency), and late-stage gain (stability). Results confirm that DyCoT-RE matches or exceeds baselines in simple environments and substantially outperforms them as task complexity increases. In Ant and SpaceMining, DyCoT-RE not only achieves higher final scores but also sustains progress throughout training, avoiding early saturation seen in the baselines.

These findings demonstrate that DyCoT-RE is not just competitive with existing reward engineering strategies—it is distinctly advantageous in scenarios where predefined heuristics or single-shot reasoning fail to adapt. Its strength lies in dynamically decomposing and optimizing reward components as training

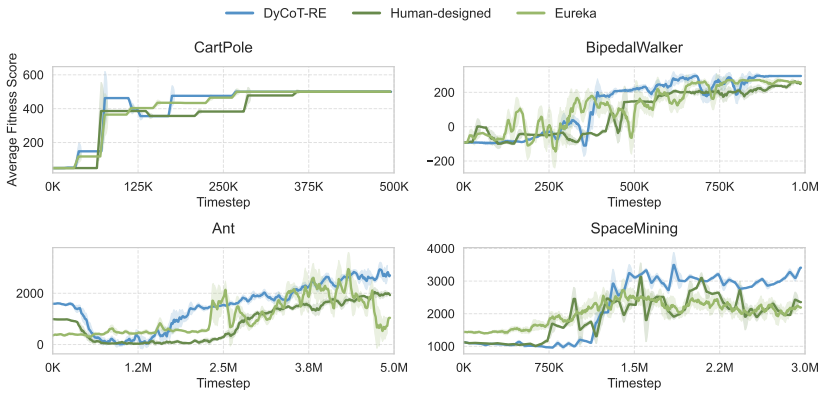

Figure 4: Learning curves across environments. DyCoT-RE exhibits consistent advantages in stability and long-term reward growth, especially in complex settings.

progresses, particularly under sparse, high-dimensional, or novel task conditions. As task complexity grows, the performance gap expands, underscoring DyCoT-RE's value as a general-purpose, scalable reward engineering solution.

4.3. Ablation Study

This section conducts an ablation study to examine the individual contribution of each component in DyCoT-RE, including CoT reasoning, temperature adjustment, and model selection, as well as their combined synergistic effects.

4.3.1. Impact of CoT Reasoning

We evaluate the impact of incorporating CoT reasoning into reward engineering by comparing DyCoT-RE's CoT-enabled variant against a zero-shot baseline lacking structured intermediate reasoning.

Figure 5 summarizes reward distributions for both methods across the four benchmark environments. A consistent pattern emerges: CoT-enabled rewards exhibit higher means, lower variance, and reduced outlier incidence compared to zero-shot. In CartPole and SpaceMining, CoT-enabled results are tightly clustered near the environment's upper performance bounds, whereas zero-shot results are widely dispersed, with a substantial proportion of runs yielding suboptimal or even near-zero rewards. This reflects CoT's ability to systematically

Table 1: Performance comparison across four RL environments. Success rate is excluded for Human baseline due to consistently successful completion under fixed reward functions. Blue: DyCoT-RE. Green: best baseline.

Env	Method	Sample Eff. ↓	Succ. Rate (%) ↑	Late Gain ↑
CartPole	DyCoT-RE	**15k**	**99**	**+3.2**
	Eureka	22k	95	+2.6
	Human	18k	–	+1.4
BipedalWalker	DyCoT-RE	**0.45M**	**98**	**+74.2**
	Eureka	0.62M	92	+29.1
	Human	0.71M	–	+18.5
Ant	DyCoT-RE	**1.2M**	**85**	**+537.4**
	Eureka	1.9M	72	+243.2
	Human	2.3M	–	+121.6
SpaceMining	DyCoT-RE	**0.8bM**	**78**	**+321.7**
	Eureka	1.8M	65	+92.8
	Human	1.2M	–	+55.9

decompose task objectives into interpretable sub-rewards, enhancing sample efficiency and stabilizing policy learning.

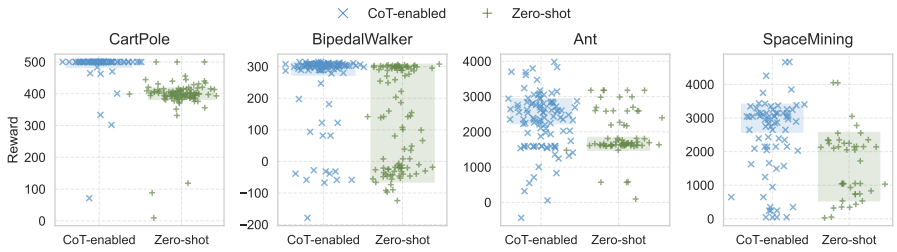

Figure 5: Final reward distributions for CoT-enabled vs zero-shot reward generation across environments.

In BipedalWalker, an illustrative deviation is observed. While CoT-enabled runs achieve rewards concentrated between 250–310 with minimal negative outcomes, zero-shot results present a bimodal distribution: one cluster achieves moderate positive rewards (∼150–250), while another cluster yields severely negative rewards (e.g. -124, -97, -73). This bimodality indicates that without

structured reasoning, reward generation often fails to encode essential balance or locomotion constraints, resulting in policies that collapse during gait training.

Taken together, these results illustrate that CoT reasoning reliably improves reward interpretability and training stability, especially in settings with sparse or ambiguous feedback.

4.3.2. Impact of Temperature Adjustment

Due to computational constraints and consistent observed trends across environments, temperature adjustment and model selection ablation were conducted on BipedalWalker as a representative continuous control task, while CoT reasoning was evaluated on all four environments to confirm its general effectiveness.

This section investigates the effect of temperature settings by sweeping $T \in [0.0, 1.0]$ on the BipedalWalker environment.

Table 2 summarizes the results, including average fitness, maximum/minimum fitness, standard deviation, and reward efficiency ratio (average fitness divided by standard deviation).

Static sweeps show that mid-range temperatures ($T = 0.4$–0.6) yield higher fitness and lower variance than extremes. DyCoT-RE's dynamic temperature regulation consistently outperforms these static settings, adapting temperature during training to maintain an effective exploration-exploitation trade-off.

4.3.3. Impact of Model Selection

Finally, we evaluate the impact of DyCoT-RE's dynamic model selection by comparing it against individual static LLM baselines on the BipedalWalker environment. Table 3 summarizes average fitness, max/min fitness, standard deviation, and reward efficiency ratio (average fitness divided by standard deviation). Only models with sufficient sample sizes ($N > 10$) are included.

DyCoT-RE achieves the highest average fitness (307.28) with exceptionally low standard deviation (6.00) and the best efficiency ratio (51.21), demonstrating

Table 2: BipedalWalker performance under different temperature settings. Top-1 temperature is highlighted in green, DyCoT-RE result in blue.

Temp	Avg Fit↑	Max/Min Fit ↑	Std↓	Eff. Ratio↑
0.0	81.93	301.15/-53.01	148.67	0.55
0.1	149.56	310.94/-33.73	155.22	0.96
0.2	242.74	313.60/-20.27	125.47	1.93
0.3	210.07	311.17/-46.69	140.49	1.50
0.4	244.19	311.94/-17.13	107.02	2.28
0.5	215.84	310.69/-14.25	131.01	1.65
0.6	248.11	312.48/-21.18	120.45	2.06
0.7	225.88	312.42/-12.08	131.43	1.72
0.8	74.33	310.42/-46.72	143.91	0.52
0.9	145.02	310.02/-31.58	148.51	0.98
1.0	192.38	310.40/-39.05	138.37	1.39
DyCoT-RE	**292.8**	315.6/87.2	55.2	5.30

its capability to consistently generate high-quality rewards across diverse subtasks.

In contrast, while `codegemma` achieves competitive performance (Avg Fit: 260.8), its higher standard deviation (115.2) results in a substantially lower efficiency ratio (2.26), indicating reduced stability and generalizability compared to DyCoT-RE.

While individual models occasionally excel on narrow tasks, DyCoT-RE's dynamic model routing provides the most generalizable performance, effectively balancing reward quality, stability, and adaptability in complex continuous control environments.

In summary, the ablation study demonstrates that each module—CoT reasoning, temperature adjustment, and model selection—contributes significantly to DyCoT-RE's superior performance, and their combination leads to synergistic gains.

Table 3: Performance comparison of different LLM backends (N=20) in BipedalWalker. ↑: higher is better; ↓: lower is better. grass: per-column best baseline. Blue: DyCoT-RE (dynamic selection).

Model (size)	Avg Fit ↑	Max Fit ↑	Std ↓	Eff. Ratio ↑
codegemma (7b)	260.8	301.8	34.2	3.46
codeqwen (7b)	32.6	307.6	-76.0	0.33
deepseek-coder (6.7b)	-4.5	295.3	-98.2	-0.05
deepseek-r1 (8b)	44.6	321.2	-99.6	0.37
gemma3 (4b)	278.0	314.8	-34.2	3.39
llama3.1 (8b)	162.2	318.4	-80.4	1.08
qwen2.5 (7b)	177.0	319.5	-97.2	1.17
qwen2.5-coder (7b)	125.6	313.7	-70.8	0.77
DyCoT-RE	**297.28**	**316.57**	**89.47**	**6.46**

4.4. SpaceMining Performance and Reward Decomposition Analysis

To evaluate DyCoT-RE in scenarios lacking prior task-specific knowledge, we constructed the SpaceMining environment: a custom-designed, out-of-distribution task involving multi-objective resource collection, sparse and delayed rewards, dynamic hazards, and energy constraints. This setting enables probing the model's ability to generate effective reward formulations solely from natural language task descriptions, independent of pretrained templates or domain priors.

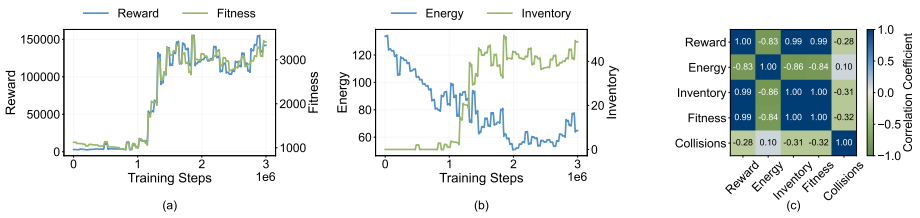

Figure 6: Training dynamics in SpaceMining: (a) reward and fitness progression; (b) energy and inventory balance; (c) correlation matrix across key metrics.

Figure 6(a) shows that both reward and fitness increase steadily with training

steps, exhibiting near-identical trends. This alignment indicates that the reward formulation not only supports short-term return maximization but also promotes long-term policy optimization. The absence of saturation or collapse suggests that DyCoT-RE facilitates sustained improvement without early convergence or reward exploitation.

In Figure 6(b), the energy level remains stable throughout training, while inventory increases consistently. This indicates the agent learns an efficient cycle of energy consumption and replenishment that sustains continuous mining and delivery. The observed dynamic balance reflects successful resource strategy formation, a key success factor in SpaceMining.

Figure 6(c) quantifies correlations among reward, energy, inventory, fitness, and collisions. Reward exhibits strong positive correlation with inventory ($r = 0.99$) and fitness ($r = 0.99$), and strong negative correlation with energy ($r = -0.83$), implying that resource accumulation is the principal signal for both learning and evaluation, with energy expenditure as an implicit tradeoff. Weak correlation with collisions ($r = -0.28$) suggests the agent tolerates limited risk to maximize throughput, aligning with the environment's implicit reward-cost design.

To further investigate the mechanism evolution during DyCoT-RE's iterative reasoning process, Table 4 presents the reward decomposition components generated at selected iterations. Each configuration includes reward formulae, weight coefficients, and empirical contributions measured during the final training stage. As the iterations progress, the reward design becomes more structured: later-stage versions introduce more semantically aligned components such as energy conservation, collision penalties, and inventory-weighted delivery—capturing both efficiency and safety considerations. Table entries are color-coded to highlight DyCoT-RE's best-performing configuration.

The reward formulation matures across iterations: early designs rely on surface-level signals such as delivered amount, while later designs incorporate temporal decay, inventory balancing, energy efficiency, and penalization of unsafe behaviors. These transitions reflect DyCoT-RE's ability to iteratively refine

Table 4: Evolution of reward decomposition across DyCoT-RE iterations in SpaceMining.

Iter.	Stage	Model	Sub-reward	Weight	Contribution (%)
1	Env. Understanding	gemma3	`inventory`·0.2	0.5	41.3
			`delivered_amount`	0.5	42.7
3	Reward Design	codegemma	`delivered_amount`$\cdot e^{-d/D}$	0.4	38.2
			`energy_remaining`$/E_{\max}$	0.3	26.5
			$-\lvert\text{collision}\rvert^2$	0.2	-9.6
5	Reflection	qwen2.5	`inventory`$\cdot \log(1+x)$	0.4	36.1
			`delivered_amount`$\cdot e^{-d/D}$	0.3	35.1
			$\mathbb{I}$(new_asteroid)·0.1	0.2	15.7
			`energy_remaining`$/E_{\max}$	0.2	22.0
			$-\lvert\text{collision}\rvert^2$	-0.1	-11.0
			`inventory`·log-scaling	0.4	**38.2**
7	Reflection	DyCoT-RE	`delivered_amount`$\cdot e^{-d/D}$	0.3	**35.1**
			$\mathbb{I}$(new_asteroid)·0.1	0.2	**15.7**
			`energy_remaining`$/E_{\max}$	0.2	**22.0**
			$-\lvert\text{collision}\rvert^2$	-0.1	$\mathbf{-11.0}$

reward semantics through CoT-guided modular reasoning and adaptive model selection.

To contextualize these behavioral shifts, Table 5 summarizes associated dynamic properties for each iteration, including selected models, reward maxima, entropy, confidence, and temperature. As training progresses, reward increases substantially, while entropy and temperature decrease, indicating growing determinism and policy confidence. The iterative model switching (from gemma3 to codegemma to qwen2.5) aligns with key semantic stages, suggesting model selection plays a critical role in capturing and stabilizing complex reward logic.

Table 5: Structural dynamics of DyCoT-RE across SpaceMining iterations.

Iter.	Selected Model	Temperature ↓	Entropy ↓	Confidence ↑	Best Reward ↑
1	gemma3	0.6686	0.9899	0.8739	1486.2
3	codegemma	0.6386	0.9111	0.7397	3128.0
5	qwen2.5	0.6099	0.9061	0.6934	3494.1
7	DyCoT-RE	0.6386	0.9150	0.7444	3494.1

Together, these analyses demonstrate that DyCoT-RE not only achieves high performance in an out-of-distribution setting but also provides interpretable and progressively structured reward designs. Its iterative optimization and semantic modularity enable fine-grained alignment between task goals and reward structures, offering a compelling advantage over static or monolithic reward engineering methods in complex, novel environments.

5. Discussion

This work presents DyCoT-RE, a reward engineering framework combining Chain-of-Thought reasoning, dynamic temperature regulation, and adaptive model selection. Experiments across four RL tasks show DyCoT-RE consistently outperforms heuristic rewards and automated baselines like Eureka, especially in complex, sparse-reward environments such as Ant and SpaceMining. For simpler tasks like CartPole, gains are smaller due to already dense rewards. Ablation results highlight that CoT reasoning improves reward interpretability and stability, dynamic temperature adjustment enhances sample efficiency over static settings, and adaptive model selection reduces variance by leveraging multiple LLMs effectively.

Despite these strengths, DyCoT-RE has limitations. Its reliance on LLM inference introduces computational overhead, and performance may degrade if backend models lack domain-relevant priors. Furthermore, while Chain-of-Thought reasoning improves reward decomposition quality, its efficacy remains sensitive to prompt design and reasoning chain depth, especially in novel environments without structured knowledge templates.

Future work could explore integrating DyCoT-RE with offline reinforcement learning to reduce online sampling costs, extending model selection mechanisms to multimodal backends for vision-language task settings, and applying human-in-the-loop evaluation to refine interpretability metrics beyond decomposition-based heuristics. Additionally, broader testing on diverse real-world tasks, such as industrial robotics and multi-agent coordination, will help assess the framework's

scalability and general applicability.

In summary, DyCoT-RE advances the automated design of reward functions by combining cognitive reasoning paradigms with dynamic optimization mechanisms, showing promise as a practical tool for scalable and interpretable reinforcement learning across task domains of varying complexity.

6. Conclusion

This paper presented DyCoT-RE, a reward engineering framework that integrates Chain-of-Thought reasoning, dynamic temperature regulation, and adaptive model selection to automate reward function design in reinforcement learning. Evaluations across a variety of tasks—from simple discrete control to complex continuous locomotion and resource collection—show that DyCoT-RE consistently outperforms human-designed heuristics and prior automated methods.

References

[1] R. S. Sutton, A. G. Barto, et al., Reinforcement learning: An introduction, Vol. 1, MIT press Cambridge, 1998.

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

[3] J. Skalse, et al., Misspecification in inverse reinforcement learning, Journal of Artificial Intelligence Research 73 (2022) 517–550.

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

[5] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, A. Dragan, Inverse reward design, Advances in neural information processing systems 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 33 (2020) 1877–1901.

[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 35 (2022) 27730–27744.

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

[9] M. Peeperkorn, T. Kouwenhoven, D. Brown, A. Jordanous, Is temperature the creativity parameter of large language models?, arXiv preprint arXiv:2405.00492 (2024).

[10] W. Fedus, B. Zoph, 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, pp. 287–311.

[11] 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.

[12] S. Huang, L. Yang, Y. Song, S. Chen, L. Cui, Z. Wan, Q. Zeng, Y. Wen, K. Shao, W. Zhang, et al., Thinkbench: Dynamic out-of-distribution evaluation for robust llm reasoning, arXiv preprint arXiv:2502.16268 (2025).

[13] Z. Qi, H. Luo, X. Huang, Z. Zhao, Y. Jiang, X. Fan, H. Lakkaraju, J. Glass, Quantifying generalization complexity for large language models, arXiv preprint arXiv:2410.01769 (2024).

[14] W. Lu, X. Zhao, J. Spisak, J. H. Lee, S. Wermter, Mental modeling of reinforcement learning agents by language models, arXiv preprint arXiv:2406.18505 (2024).

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

[16] S. Singh, R. L. Lewis, A. G. Barto, J. Sorg, Intrinsically motivated reinforcement learning: An evolutionary perspective, IEEE Transactions on Autonomous Mental Development 2 (2) (2010) 70–82.

[17] Y. Burda, H. Edwards, A. Storkey, O. Klimov, Exploration by random network distillation, arXiv preprint arXiv:1810.12894 (2018).

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

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

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

[21] Z. Xie, J. Gao, L. Li, Z. Li, Q. Liu, L. Kong, Jailbreaking as a reward misspecification problem, arXiv preprint arXiv:2406.14393 (2024).

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

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

[24] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, Y. Iwasawa, Large language models are zero-shot reasoners, Advances in neural information processing systems 35 (2022) 22199–22213.

[25] 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 35 (2022) 24824–24837.

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

[27] DeepSeek-AI, Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, arXiv preprint (2023).

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

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

[30] X. Zhu, J. Du, Q. Fu, 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.

[31] Y. Zhu, J. Li, G. Li, Y. Zhao, Z. Jin, 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, 2024, pp. 437–445.

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

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

[34] C.-C. Chang, D. Reitter, R. Aksitov, Y.-H. Sung, Kl-divergence guided temperature sampling, arXiv preprint arXiv:2306.01286 (2023).

[35] 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).

[36] Y. Zhou, T. Lei, H. Liu, N. Du, Y. Huang, V. Zhao, A. M. Dai, Q. V. Le, J. Laudon, et al., Mixture-of-experts with expert choice routing, Advances in Neural Information Processing Systems 35 (2022) 7103–7114.

[37] Y. Li, Llm bandit: Cost-efficient llm generation via preference-conditioned dynamic routing, arXiv preprint arXiv:2502.02743 (2025).

[38] J. Hu, Y. Wang, S. Zhang, K. Zhou, G. Chen, Y. Hu, B. Xiao, M. Tan, Dynamic ensemble reasoning for llm experts, arXiv preprint arXiv:2412.07448 (2024).

[39] K. Nakaishi, Y. Nishikawa, K. Hukushima, Critical phase transition in large language models, arXiv preprint arXiv:2406.05335 (2024).

[40] Y. Li, J. Zhang, S. Feng, P. Yuan, X. Wang, J. Shi, Y. Zhang, C. Tan, B. Pan, Y. Hu, et al., Revisiting self-consistency from dynamic distributional alignment perspective on answer aggregation, arXiv preprint arXiv:2502.19830 (2025).