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

Xinning Zhu[a], 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 structured 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 suggest that although reinforcement learning offers strong theoretical flexibility, its real-world applications are frequently limited by the complexity and domain-specific knowledge required to design effective reward functions. [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 facilitates reward engineering by translating task descriptions into reward functions with explicitly defined components and objectives.

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 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.These findings motivate the integration of temperature modulation with expert model selection as a promising direction for improving reward engineering in LLM-driven systems.

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 coupling these components within a closed-loop evolutionary search process, DyCoT-RE enhances reward function design and improves training efficiency, supporting scalable and automated reward specification in complex reinforcement learning tasks.

We evaluate DyCoT-RE on the standard RL environments CartPole, Bipedal-Walker, and Ant to assess its performance across different control tasks. While these environments are widely used, recent studies have raised concerns that large language models may possess latent knowledge about them from pretraining corpora, potentially leading to prompt leakage and evaluation bias[12, 13, 14]. To address this concern and test DyCoT-RE's true generalization capabilities, we additionally introduce a custom-designed environment, SpaceMining[1], which is explicitly excluded from pretraining. Experimental results show that DyCoT-RE consistently achieves higher average rewards and faster convergence than baseline and non-CoT methods across all evaluated environments.

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

The remainder of this paper is organized as follows. Section II reviews related work in reward engineering and CoT-based LLM reasoning. Section III introduces the DyCoT-RE framework, including CoT-based reward construction and the dual-dynamic optimization strategy. Section IV presents experimental evaluations, including setup, benchmark comparisons, ablations, and interpretability analysis. Section V discusses the limitations and potential future directions. Section VI concludes 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 the growing use of LLMs for automated reward engineering in RL. EUREKA [18] treats LLMs as autonomous agents that iteratively generate, mutate, and select reward code via performance-based tournaments. Text2Reward [19] focuses on direct reward code generation from natural language, while CARD [20] emphasizes iterative code refinement guided by LLM feedback.

Beyond static reward generation, recent feedback-driven approaches such as adversarial prompt detection [Xie et al., 2024], self-play alignment [Wu et al., 2024], and benchmarking frameworks [Song et al., 2025] have advanced reward modeling quality. 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 reinforcement 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 have demonstrated the effectiveness of CoT reasoning in language models. DeepSeek [27] showed that structured reasoning capabilities can emerge through self-evolution mechanisms, achieving complex chain-of-thought reasoning 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 sampling strategy for code generation tasks, which dynamically adjusts decoding temperature based on

token-level difficulty to improve generation quality. Zhang et al. [32] introduced Entropy-based Dynamic Temperature (EDT) sampling, which regulates output entropy and diversity in natural language generation. Cecere et al. [33] proposed Monte Carlo Temperature, a robust sampling method to enhance uncertainty quantification under distribution shifts. Chang et al. [34] leveraged KL-divergence-guided temperature control to modulate exploration adaptively.

Beyond static sampling strategies, recent work has begun addressing the broader challenge of adapting reward generation mechanisms to evolving training dynamics. Evstafev [35] highlighted limitations of stochastic sampling in tasks requiring structured outputs, motivating the need for more context-aware adaptation techniques. Nakaishi et al. [39] further revealed phase-transition behaviors in LLM sampling regimes, informing principled temperature scaling strategies.

For model selection, various strategies have been proposed to enhance efficiency, robustness, and alignment under resource constraints. Zhou et al. [36] introduced expert choice routing to improve Mixture-of-Experts (MoE) efficiency, while Li et al. [37] developed preference-conditioned dynamic routing for cost-effective LLM generation. Hu et al. [38] presented Dynamic Ensemble Reasoning to integrate outputs from multiple specialized LLMs, enhancing robustness. Li et al. [40] revisited self-consistency decoding from a distributional alignment perspective, offering insights into stable expert aggregation.

Despite recent advances, temperature adaptation has mainly aimed at enhancing generative diversity and calibration, while model selection focuses on efficiency and specialization. Extending these techniques to reward generation offers a promising direction 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.

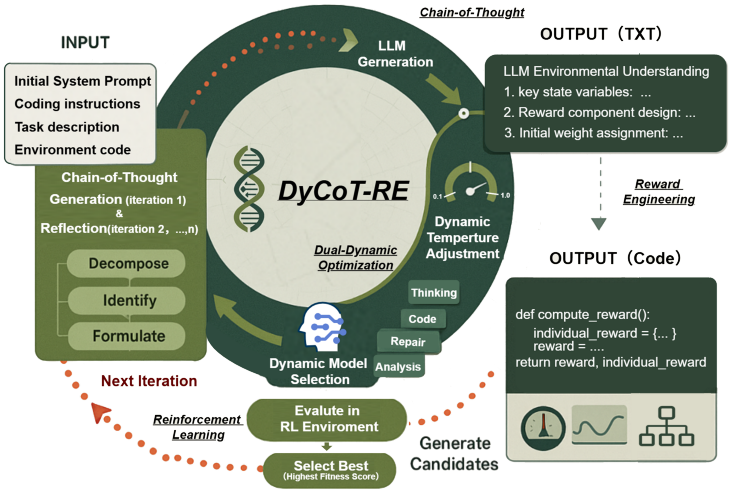

Figure 1: DyCoT-RE framework integrating CoT reasoning, dynamic temperature adjustment, and model selection in an evolutionary optimization loop.

3.1. Framework Overview

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

The framework receives four types of inputs: natural language task descriptions that specify agent behaviors, environment interfaces that define the state-action space, system-level prompts that impose design constraints on the reward, and implementation instructions that guide executable code generation.

At the core of DyCoT-RE are four specialized large language models, referred to as Thinking LLM, Code LLM, Repair LLM, and Analysis LLM. These models, denoted respectively as $\mathcal{M}$think, $\mathcal{M}$code, $\mathcal{M}$repair, and $\mathcal{M}$analysis, operate in a closed-loop pipeline that produces and refines reward functions.

Each generated reward function takes the form:

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

where $r_i(s,a)$ denotes the i-th reward component and w_i is its corresponding

weight. These components are derived through structured CoT reasoning and reflect interpretable sub-goals. The resulting reward function is deployed in the RL environment, and its performance guides the next iteration of temperature tuning and model routing.

Through the integration of structured reasoning and dual-dynamic optimization, DyCoT-RE enables adaptive and efficient reward engineering for reinforcement learning.

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:

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

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

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

The weights w_i are derived based on subgoal semantic priorities inferred by the LLM, and are iteratively adjusted by the LLM based on training performance feedback. The objective is to maximize the expected cumulative reward:

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)\right]. \tag{4}$$

To evaluate reward function quality independently of the LLM-generated rewards, we employ a fitness score metric that serves as an objective performance indicator for guiding the LLM's weight adjustment process. This CoT-based formulation ensures that each sub-reward remains semantically interpretable and traceable to its natural language origin.

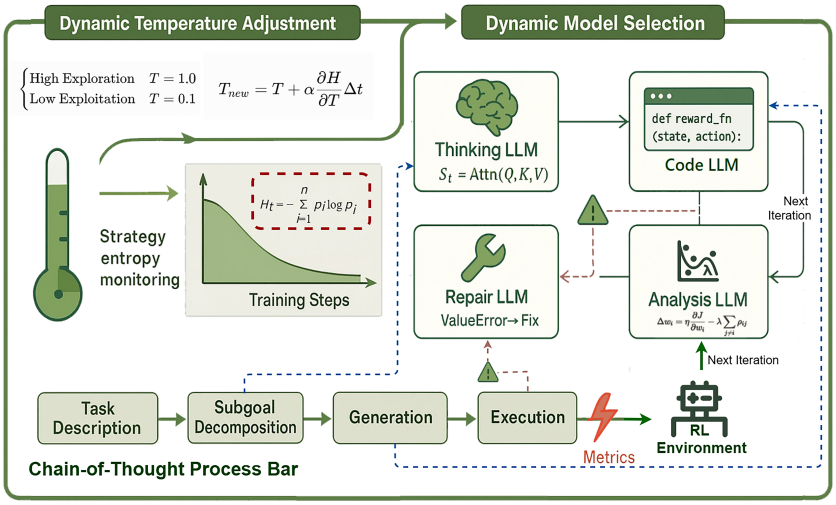

Figure 2: DyCoT-RE control flow integrating temperature modulation and model selection within CoT iterative reasoning.

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:

$$T_{k+1} = \text{clip}\left(T_k + \alpha \frac{\partial T}{\partial H_k} \Delta t\right), \tag{5}$$

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:

$$T_{k+1} = \text{clip}\left(\alpha T_k + (1-\alpha) T_k f(H_k, C_k, R_k)\right). \tag{6}$$

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

$$f(H, C, R) = f_R(R) f_H(H) f_C(C), \tag{7}$$

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:

$$g_i = \mathcal{M}_{\text{think}}(d, E), \tag{8}$$

$$w_i = \mathcal{M}_{\text{analysis}}(R_i), \tag{9}$$

$$r_i(s, a) = \mathcal{M}_{\text{code}}(g_i, w_i), \tag{10}$$

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:

$$M_{k+1} = \begin{cases} \arg\max_{m \in \mathcal{M}_s} \text{Perf}(m), & 1-\epsilon, \\ \text{Random}(\mathcal{M}_s \setminus \{M_k\}), & \epsilon, \end{cases} \tag{11}$$

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

At each iteration, the next model M_{k+1} is selected from pool $\mathcal{M}_s$ using epsilon-greedy selection, where the highest-performing model is chosen with probability $1 - \epsilon$ and a random model with probability ϵ. The pool includes a Repair LLM for error correction 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:

$$r_i(s, a) \sim p(r_i|g_i, T), \tag{12}$$

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.

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

$$J(\theta; T, M) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t R_{T,M}(s_t, a_t)\right], \tag{13}$$

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

This joint optimization is formalized as:

$$(T^*, M^*) = \arg\max_{T,M} J(\theta; T, M). \tag{14}$$

Overall, this interconnected loop iteratively adapts both sampling temperature and model selection strategies to maximize the expected RL objective as defined in Eq. (4), ensuring 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 complexity levels. Standard benchmarks include CartPole, BipedalWalker, and Ant, covering discrete control tasks and high-dimensional continuous locomotion.

To assess performance on novel environments beyond pretrained knowledge, we introduce SpaceMining, a custom environment with multi-objective resource collection, sparse rewards, and dynamic energy constraints. Unlike standard environments that LLMs may encounter during pretraining, SpaceMining tests whether reward design can succeed through pure reasoning rather than prior knowledge.

4.1.1. Baselines and Comparison Strategies

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

(1) Human-designed rewards: Manually crafted reward functions from standard Gymnasium implementations for CartPole, BipedalWalker, and Ant, plus custom rules for SpaceMining.

(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.

4.1.2. 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.

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

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

4.1.3. Implementation Details

We evaluate DyCoT-RE on three environments from the Gymnasium[2] suite—CartPole-v1[3], BipedalWalker-v3[4], and Ant-v5[5]—alongside a custom-designed SpaceMining environment built to test generalization in unseen domains.

All experiments use Proximal Policy Optimization (PPO) from Stable-Baselines3 with Adam optimizer, learning rate of 3e-4, batch size 64, GAE parameter $\lambda = 0.95$, and discount factor $\gamma = 0.999$.

Reward functions are synthesized by DyCoT-RE using local LLMs served via Ollama. At each chain-of-thought iteration, eight reward candidates are sampled in parallel.

Final performance metrics are computed as the mean over ten test episodes with different random seeds.

Table 1: Task specifications for evaluated reinforcement learning environments.

Environment	Dimensions (S/A)	Steps	Task
CartPole-v1	4 / 1 (disc)	500	Balance
BipedalWalker-v3	24 / 4 (cont)	1600	Locomotion
Ant-v5	111 / 8 (cont)	1600	Multijoint
SpaceMining (Custom)	53 / 3 (cont)	1200	Navigation

4.2. Performance Across Environments

We compare DyCoT-RE against Human-designed and Eureka baselines across all evaluated environments, observing consistent performance gains. The advan-

[2] `https://gymnasium.farama.org`

[3] `https://gymnasium.farama.org/environments/classic_control/cart_pole/`

[4] `https://gymnasium.farama.org/environments/box2d/bipedal_walker/`

[5] `https://gymnasium.farama.org/environments/mujoco/ant/`

tage becomes more prominent as task complexity and generalization difficulty increase.

Figure 3 summarizes the final average rewards across all environments. DyCoT-RE consistently achieves the highest performance, followed by Eureka, with human-designed rewards showing the lowest performance. The performance gap increases with task complexity, from modest improvements in CartPole and BipedalWalker to substantial advantages in Ant and SpaceMining. These results demonstrate DyCoT-RE's effectiveness across diverse environments, particularly when facing sparse feedback and high-dimensional state-action spaces that challenge traditional reward design approaches.

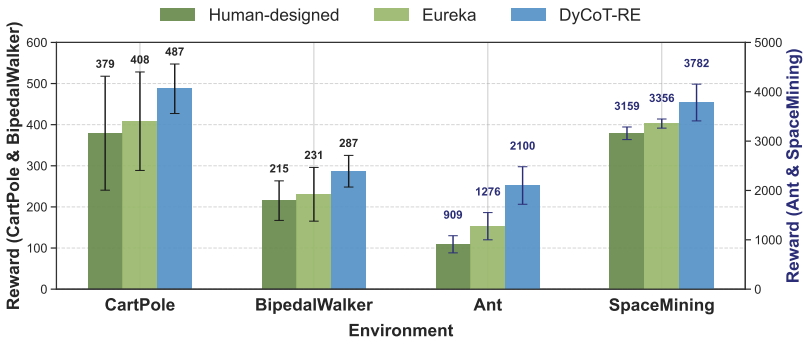

Figure 3: Final average rewards across environments: DyCoT-RE (blue), Eureka (light grass), and Human-designed (dark grass).

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.

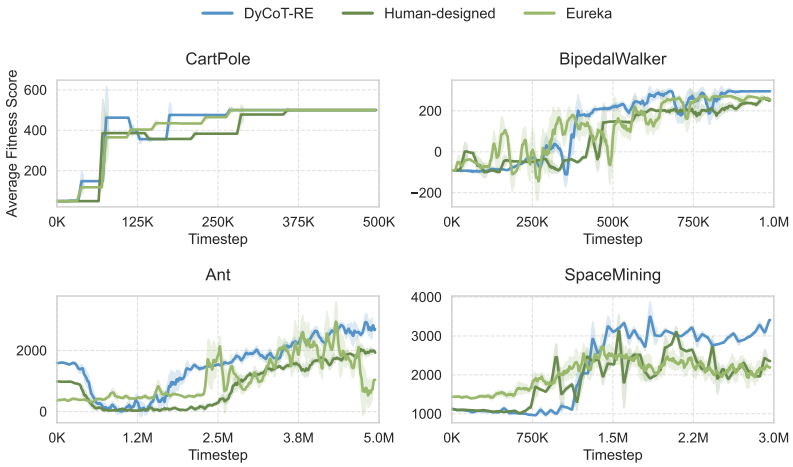

Figure 4: Learning curves across environments. DyCoT-RE exhibits consistent advantages in stability and long-term reward growth, especially in complex settings.

Table 2 compares performance across four tasks using three metrics: Sample Efficiency measures the number of timesteps required to reach a threshold reward, defined as 80% of the task-specific maximum score. Success Rate indicates the proportion of runs that eventually exceed this threshold. Late Gain quantifies the reward improvement from the early performance plateau to the final evaluation, reflecting long-term optimization potential.

DyCoT-RE achieves the best performance in all tasks. It learns faster, generalizes more reliably, and continues improving beyond early convergence. In complex environments like Ant and SpaceMining, it shows especially large gains in late-stage performance, where static baselines tend to stall or regress.

These results highlight DyCoT-RE's key strength: its dynamic reasoning and adaptive reward synthesis enable both rapid early learning and sustained progress, even under sparse or high-dimensional settings.

The results indicate that DyCoT-RE provides clear advantages in scenarios where predefined reward rules or static reasoning approaches are insufficient to capture evolving task demands. Its strength lies in dynamically decomposing and optimizing reward components as training progresses, particularly under

Table 2: Cross-task performance comparison. Blue : the best value per column, green : the second-best. Success Rate is omitted for Human due to deterministic reward completion.

Task	Method	Sample Eff. ↓	Succ. Rate ↑	Late Gain ↑
CartPole	DyCoT-RE	75k	98%	450
	Eureka	115k	95%	440
	Human	286k	–	400
BipedalWalker	DyCoT-RE	393k	92%	387
	Eureka	428k	83%	366
	Human	720k	–	309
Ant	DyCoT-RE	16.19M	85%	5398
	Eureka	21.53M	70%	2412
	Human	19.92M	–	-240
SpaceMining	DyCoT-RE	1.31M	80%	2285
	Eureka	1.74M	65%	189
	Human	1.56M	–	1230

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 sub-optimal or even near-zero rewards. This reflects CoT's ability to systematically decompose task objectives into interpretable sub-rewards, enhancing sample efficiency and stabilizing policy learning.

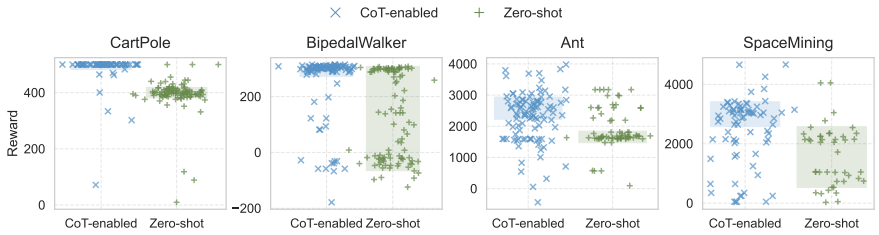

Figure 5: Final reward distributions for CoT-enabled vs zero-shot reward generation across environments.

In BipedalWalker, a notable deviation is observed. CoT-enabled runs yield a concentrated reward range between 250 and 310, with virtually no severely negative outcomes. In contrast, the zero-shot condition presents a clear bimodal distribution: one cluster achieves moderate reward levels around 150 to 250, while the other collapses into strongly negative regimes. This instability reflects the inability of zero-shot reward generation to encode essential gait balance or locomotion constraints, often resulting in policy failure.

In Ant, zero-shot results are narrowly concentrated near baseline values associated with default movement penalties. This indicates that while these reward functions do not collapse, they lack task-driving gradients for effective behavior shaping. In contrast, CoT-based rewards achieve a wider spread toward higher values, reflecting meaningful progress and more informative optimization signals.

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

Table 3: BipedalWalker performance under different temperature settings. Blue : the best value per column, green : the second-best, green : the third-best

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

Static sweeps show that mid-range temperatures 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 4 summarizes average fitness, max/min fitness, standard deviation, and reward efficiency ratio (average fitness divided by standard deviation).

DyCoT-RE achieves the highest average fitness with exceptionally low standard deviation and the best efficiency ratio, demonstrating its capability to consistently generate high-quality rewards across diverse sub-tasks.

In contrast, while codegemma achieves competitive performance, its higher standard deviation results in a substantially lower efficiency ratio, indicating reduced stability and generalizability compared to DyCoT-RE.

Table 4: Performance comparison of different LLM backends (N=20) in BipedalWalker. Blue : the best value per column, green : the second-best

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	98.42	0.33
deepseek-coder (6.7b)	-4.5	295.3	98.19	-0.05
deepseek-r1 (8b)	44.6	321.2	122.07	0.37
gemma3 (4b)	278.0	314.8	142.04	3.39
llama3.1 (8b)	162.2	318.4	150.75	1.08
qwen2.5 (7b)	177.0	319.5	151.51	1.17
qwen2.5-coder (7b)	125.6	313.7	163.16	0.77
DyCoT-RE	297.28	316.57	89.47	6.46

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.

4.4. SpaceMining Performance and Reward Decomposition Analysis

To evaluate DyCoT-RE in an out-of-distribution scenario without task-specific priors, we constructed the SpaceMining environment—a sparse-reward, multi-objective domain involving dynamic hazards, energy constraints, and delayed feedback. This setting challenges the model to formulate effective rewards from natural language alone, testing its semantic generalization and reasoning capabilities.

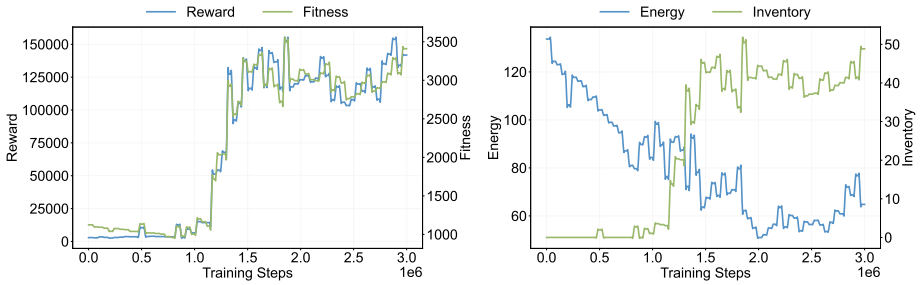

Figure 6: Training dynamics in SpaceMining. Left: reward and fitness improve steadily, showing effective long-term optimization. Right: stable energy levels and growing inventory indicate successful energy-resource balancing.

Figure 6 illustrates the learning trajectory of DyCoT-RE. From early-stage exploration to late-stage optimization, we observe concurrent increases in reward and fitness, with consistent energy regulation and inventory accumulation. These trends reflect reward structures that incentivize planning, efficiency, and safe behavior.

Figure 7 illustrates how DyCoT progressively restructures its reward architecture across eight training iterations in the SpaceMining environment. The system begins with a minimal setup focused on mining progress, distance penalties, and

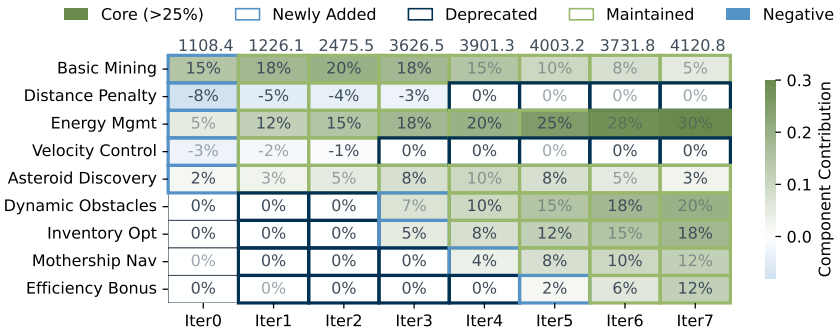

Figure 7: Dynamic Evolution of Reward Components in DyCoT-RE for SpaceMining

energy tracking. As training proceeds, DyCoT introduces new components such as obstacle avoidance in Iteration 3, inventory-based delivery efficiency in Iteration 5, and asteroid discovery incentives in Iteration 7. These additions are not arbitrary but driven by observed performance shifts. For instance, after dynamic obstacle handling is introduced, the agent's survival rate improves significantly. Conversely, some components such as velocity control lose their relevance and are phased out by Iteration 3, as their function becomes redundant once obstacle-aware navigation is in place. The energy management module, on the other hand, steadily increases in importance and becomes a core contributor throughout the later iterations. This dynamic component lifecycle—characterized by incremental expansion, adaptive weighting, and timely deprecation—demonstrates DyCoT's capacity for modular, interpretable, and performance-aligned reward optimization.

Table 5 illustrates the evolution of key strategy-level metrics throughout all DyCoT iterations. As the architecture matures from Iteration 0 to 7, we observe a consistent decline in sampling temperature, indicating increased confidence in policy generation. Entropy fluctuates with architecture changes, notably peaking at Iteration 3 when major reward components are introduced. Confidence improves steadily, rising from 0.64 to 0.82, in parallel with fitness gains. Reward gain is computed relative to the previous iteration, with Iteration 3 marking a significant performance inflection point. Subsequent gains are more incremental,

Table 5: DyCoT Strategy Metrics Across Iterative Architectures. Blue : the best value per column, green : the second-best.

Iter	Temperature	Entropy	Confidence	Best Reward	Reward Gain (%)
0	0.800	0.912	0.642	880.2	–
1	0.700	0.946	0.705	1124.4	+27.7
2	0.669	0.928	0.748	1320.3	+17.4
3	0.639	0.986	0.778	3475.5	+163.2
4	0.600	0.934	0.791	3551.0	+2.2
5	0.556	0.914	0.807	3801.3	+7.0
6	0.534	0.962	0.815	3876.2	+2.0
7	0.517	0.972	0.824	4003.2	+3.3

reflecting architectural stabilization.

In summary, DyCoT-RE exhibits the capacity for iterative reward adaptation. It evolves from simple distance-based signals to structured reward components that support coordinated behaviors such as exploration, energy regulation, and goal-oriented progression—achieving consistent performance in environments with minimal prior knowledge.

5. Discussion

This work presents DyCoT-RE, a reward engineering framework combining Chain-of-Thought reasoning with dual-dynamic optimization strategies. Experiments across four RL tasks show DyCoT-RE consistently outperforms human-designed 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.

Results on the SpaceMining task demonstrate DyCoT-RE's ability to construct and refine structured reward components in unfamiliar, sparse-reward settings. Its progressive adaptation—from simple distance-based incentives to task-specific modules like energy balancing and obstacle avoidance—supports its utility for automated reward design in domains lacking pre-defined metrics.

Despite these strengths, DyCoT-RE has limitations. Its reliance on LLM inference introduces computational overhead compared to static reward functions. While Chain-of-Thought reasoning generally improves reward decomposition quality, the framework's performance remains dependent on the reasoning capabilities of the underlying language models. Additionally, the current approach relies on general-purpose LLMs that may not be optimized for reward engineering tasks, potentially limiting the quality and consistency of generated reward structures.

Future work could explore several promising directions. First, training specialized language models fine-tuned specifically for reward function generation may improve both efficiency and performance by better understanding RL terminology and common reward patterns. Second, integrating multi-modal inputs such as visual demonstrations or trajectory examples could enhance reward specification for complex tasks beyond natural language descriptions. Third, extending the framework to handle hierarchical reward structures and temporal reward shaping could address long-horizon planning challenges. Finally, investigating transfer learning mechanisms to leverage successful reward designs across similar environments may reduce computational costs in new domains.

DyCoT-RE presents a framework that integrates structured reasoning with adaptive optimization for automated reward design. The experimental results demonstrate its effectiveness in improving reward function quality across environments of varying complexity.

6. Conclusion

This paper introduces DyCoT-RE, a framework that addresses the challenge of automated reward function design in reinforcement learning. The key innovation lies in combining structured reasoning through Chain-of-Thought prompting with adaptive optimization mechanisms that dynamically adjust both temperature and model selection during training.

The framework's effectiveness is validated across environments ranging from simple control tasks to complex resource collection scenarios, demonstrating particular advantages in sparse-reward settings where traditional reward design proves challenging. The results highlight its potential as a scalable solution for reinforcement learning tasks requiring structured and evolving reward design.

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).