# 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,2,*]

[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
[1]Email: zhuxinning@shu.edu.cn
[2]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[3] 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

[3]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

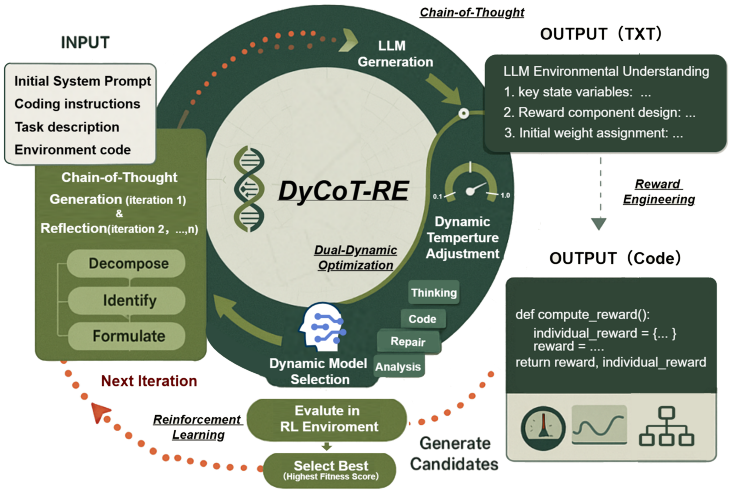


Figure 1: DyCoT-RE framework integrating CoT reasoning, dynamic temperature adjustment, and model selection in an evolutionary optimization loop.

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 $\theta$ 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 $\alpha$ 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:

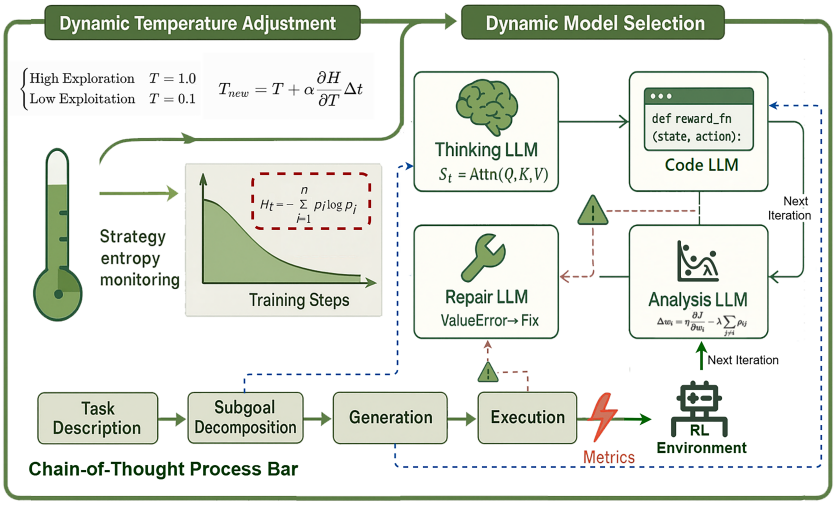


Figure 2: DyCoT-RE control flow integrating temperature modulation and model selection within CoT iterative reasoning.

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

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 $\epsilon$ 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 $\epsilon$. 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 where an agent collects resources under partial observability and dynamic physics constraints.

SpaceMining is developed as an original contribution in this work, with full code released and interactive visualizations available on the project website.

#### *4.1.1. Baselines*

We evaluate DyCoT-RE against two representative baselines to contextualize its performance:

**(1) Human-designed rewards** Standard expert-crafted reward functions using Gymnasium's native implementations for CartPole, BipedalWalker, and Ant, with manual heuristics for SpaceMining. This reflects the traditional gold standard in RL.

**(2) Eureka** A LLM-based framework that demonstrated LLMs' potential in automated reward design through code synthesis and verification. As a foundational work in this space, it provides a natural reference for assessing our incremental improvements. While Eureka focuses on direct code generation without CoT or dynamic optimization, we adapt its pipeline to our Gymnasium-based evaluation settings for consistency.

Additionally, to assess the contribution of each DyCoT-RE module, Section 4.3 conducts ablation studies comparing DyCoT-RE against its internal variants:

- Zero-shot reward generation without CoT reasoning
- DyCoT-RE without temperature adjustment
- DyCoT-RE without model selection

#### *4.1.2. Implementation Details*

All experiments use Proximal Policy Optimization from Stable-Baselines3, with hyperparameters summarized in Table 1.

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.

Table 1: Environment Specifications

| Environment | State | Action | Steps | Key Features |
|---|---|---|---|---|
| CartPole | 4D | 1D (disc) | 500 | Balance |
| BipedalWalker | 24D | 4D (cont) | 1600 | Locomotion |
| Ant | 111D | 8D (cont) | 1600 | Multijoint |
| SpaceMining | 71D | 4D (cont) | 1600 | 3D mining |

### *4.2. Performance Across Environments*

This section evaluates DyCoT-RE relative to baseline methods, focusing on its effectiveness and convergence efficiency.

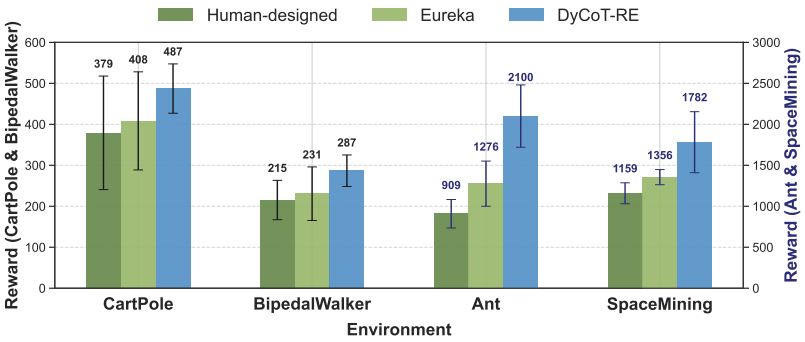


Figure 3: Final average rewards comparing Human-designed, Eureka, and DyCoT-RE across environments.

Figure 3 shows that DyCoT-RE achieves comparable or higher final rewards in all tasks. The improvement is most prominent in Ant and SpaceMining, where sparse feedback and complex task structures limit the effectiveness of handcrafted rewards.

Figure 4 illustrates training dynamics across environments. In CartPole, all methods reach high rewards quickly, but DyCoT-RE converges with fewer iterations. BipedalWalker learning is smoother under DyCoT-RE, suggesting benefits from structured reward decomposition that disentangles balance and propulsion.

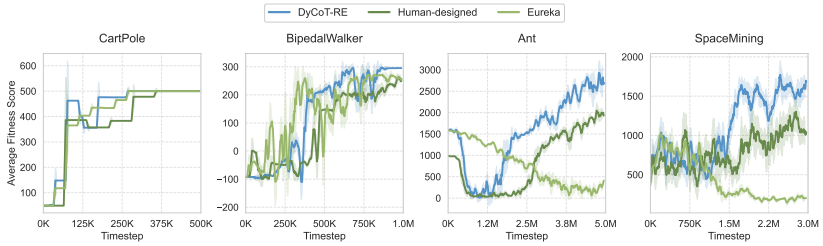


Figure 4: Learning curves for DyCoT-RE (blue), Human-designed (dark green), and Eureka (light green).

Ant shows a gradual reward increase under DyCoT-RE while baselines plateau earlier, implying improved credit assignment in high-dimensional control. In SpaceMining, DyCoT-RE maintains a steady reward increase while other methods show little progress, reflecting its adaptability to novel task domains without predefined heuristics.

Table 2: Final rewards, average convergence episodes (k), and success rates over 50 runs comparing DyCoT-RE and Eureka. Human-designed results included without convergence data.

| Environment | Method | Conv . | Success (%) |
|---|---|---|---|
| CartPole | DyCoT-RE | 2.7 | 100 |
| | Eureka | 3.8 | 95 |
| BipedalWalker | DyCoT-RE | 5.6 | 98 |
| | Eureka | 6.9 | 92 |
| Ant | DyCoT-RE | 11.5 | 85 |
| | Eureka | 14.2 | 72 |
| SpaceMining | DyCoT-RE | 7.1 | 88 |
| | Eureka | 9.6 | 65 |

Table 2 compares DyCoT-RE with Eureka and Human-designed baselines. Across all four environments, DyCoT-RE achieves the highest final rewards and fastest convergence, with success rates exceeding 85

These results indicate that DyCoT-RE's integration of Chain-of-Thought

reasoning with dynamic optimization leads to faster learning and higher rewards, particularly in tasks with sparse signals or novel objectives. Its ability to decompose complex tasks into semantically meaningful sub-rewards facilitates efficient policy learning without extensive manual reward shaping.

### *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 Chain-of-Thought (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.

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 ($\sim$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.

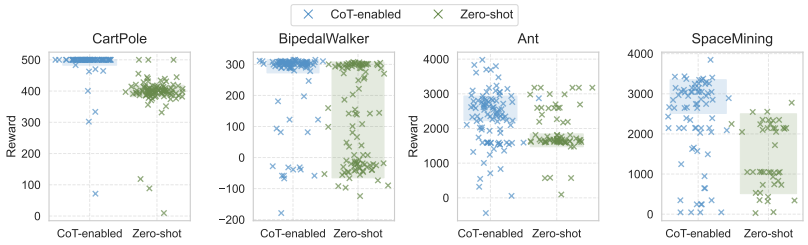


Figure 5: Final reward distributions for CoT-enabled vs zero-shot reward generation across environments.

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).

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

Table 3: 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 |

environment. Table 4 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 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.

Table 4: Performance comparison of different LLM backends (N=20) in BipedalWalker. ↑: higher is better; ↓: lower is better. Green: 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** |

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*

The SpaceMining environment was specifically designed to evaluate DyCoT-RE in scenarios lacking prior task-specific knowledge. Unlike standard benchmarks such as CartPole or Ant, which share dynamics and reward structures commonly encountered in pretraining corpora, SpaceMining introduces a novel multi-objective resource collection task with sparse and delayed rewards, dynamic obstacles, and energy constraints. This setup aims to assess whether DyCoT-RE can generate effective reward functions purely based on task descriptions without relying on pretrained knowledge templates.

Figure 6 demonstrates DyCoT-RE's performance in the custom-designed SpaceMining environment. The reward curve exhibits an initial exploration plateau, a phase of stable increase corresponding to effective policy acquisition,

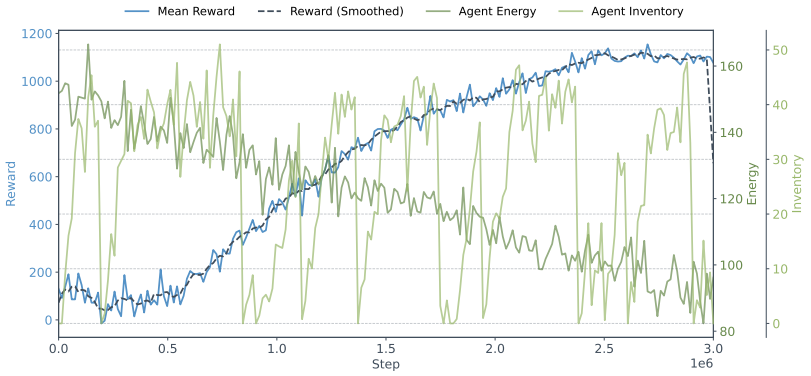


Figure 6: Training progress in SpaceMining: multi-dimensional metrics over 3M timesteps. Reward (blue), energy (dark green), and inventory (light green) reflect task-specific learning dynamics.

and a final convergence to near-maximal performance. Energy shows a smooth decreasing trend due to continuous task execution, while inventory displays cyclical fluctuations representing mining and delivery loops.

#### *4.4.1. Reward Decomposition Design and Quantitative Contributions*

Table 5 summarizes the reward decomposition designed by DyCoT-RE for SpaceMining. Each sub-reward component is listed with its computation formula, assigned weight coefficient, and empirical contribution ratio calculated over the final training stage.

Table 5: Reward decomposition components in SpaceMining with formula, coefficient, and empirical contribution.

| Sub-reward | Computation Formula | Weight | Contribution |
|---|---|---|---|
| Mining | $\log(1 + \text{mined\_amount})$ | 0.4 | 38.2% |
| Delivery | $\text{delivered\_amount} \cdot e^{-d/D}$ | 0.3 | 35.1% |
| Exploration | $\mathbb{I}(\text{new\_asteroid}) \cdot 0.1$ | 0.2 | 15.7% |
| Energy Conservation | $\text{energy\_remaining}/E_{max}$ | 0.2 | 22.0% |
| Collision Penalty | $-\|\text{collision}\|^2$ | -0.1 | -11.0% |

The decomposition captures task semantics effectively. Mining and delivery

sub-rewards dominate, aligning with primary objectives of resource extraction and transport. Exploration provides auxiliary guidance, while energy conservation ensures operational efficiency. The negative collision penalty maintains safety compliance with minimal policy interference.

DyCoT-RE's structured decomposition thus enables interpretable and modular reward design, translating high-level task requirements into quantitative objectives that facilitate effective agent training in novel, out-of-distribution environments such as SpaceMining.

## 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.

The experiments demonstrate that Chain-of-Thought reasoning improves the quality of reward decomposition, dynamic temperature adjustment enhances sample efficiency, and adaptive model selection contributes to more stable and generalizable performance. The combined effect of these components enables DyCoT-RE to better handle sparse, ambiguous, or novel task environments while maintaining stable training.

The approach requires additional computational resources and depends on the availability of capable language models. Future research will explore integration with offline reinforcement learning, expansion to multi-agent and vision-language domains, and incorporation of human feedback to better align reward design with user intent.

Overall, DyCoT-RE advances automated reward design by combining cognitive reasoning with dynamic optimization techniques, offering a pathway toward scalable and robust reinforcement learning across diverse tasks.

## References


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

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

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

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

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

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

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

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

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

[10] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2022, 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, and 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, and S. Wermter, "Mental modeling of reinforcement learning agents by language models," *arXiv preprint arXiv:2406.18505*, 2024.

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

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

[17] Y. Burda, H. Edwards, A. Storkey, and 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, and 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, and 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, and 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, and 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, and 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, and 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, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in neural information processing systems*, vol. 35, pp. 22 199–22 213, 2022.

[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*, vol. 35, pp. 24 824–24 837, 2022.

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

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

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

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

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

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

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

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

[34] C.-C. Chang, D. Reitter, R. Aksitov, and 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*, vol. 35, pp. 7103–7114, 2022.

[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, and M. Tan, "Dynamic ensemble reasoning for llm experts," *arXiv preprint arXiv:2412.07448*, 2024.

[39] K. Nakaishi, Y. Nishikawa, and 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.