HybridThink: Learned Interleaving of Continuous and Discrete Reasoning in Large Language Models

Jing Gao[1,†] **Weiqiang Jin**[1,3,†] **Guoshuai Zhao**[2] **Peigen Ye**[3]
Xueming Qian[1,*] **Hongyang Du**[3,*]

[1]School of Information and Communications Engineering,
Xi'an Jiaotong University, Xi'an 710049, China
[2]School of Software Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[3]Department of Electrical and Computer Engineering, The University of Hong Kong,
Hong Kong SAR 999077, China

gaojing0423@stu.xjtu.edu.cn weiqiangjin@stu.xjtu.edu.cn guoshuai.zhao@xjtu.edu.cn
yepg@hku.hk qianxm@mail.xjtu.edu.cn duhy@hku.hk

[†]Equal contribution. [*]Corresponding authors.

Abstract

Reasoning in Large Language Models (LLMs) is primarily conducted in a discrete token space. Although recent approaches employing continuous concept tokens facilitate abstract reasoning, they introduce significant stability challenges. The accumulation of Out-of-Distribution (OOD) states frequently causes **manifold deviation**, while the absence of clear boundaries leads to **indeterminate termination**, often referred to as the anchoring problem. To address these limitations, we propose **HybridThink**. We formulate mode selection in LLM reasoning as a sequential decision process. A lightweight gating policy is trained with RL to decide at each step whether to propagate a continuous concept token or generate a discrete text token to anchor the semantic trajectory. This learned interleaving limits manifold deviation and resolves the fragility of static rules. Furthermore, we ensure stable convergence through a training strategy that incorporates **Error-Centric Experience Replay (ECER)** and **Inductive Exploration Shaping (IES)**, prioritizing learning from failure modes, specifically drifting and premature termination. Empirical evaluations on mathematics and coding benchmarks demonstrate that HybridThink outperforms standard CoT and both deterministic and stochastic Soft Thinking baselines, achieving superior accuracy with reduced token consumption.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable proficiency across a spectrum of complex tasks, ranging from mathematical problem-solving (Shao et al., 2024; Cobbe et al., 2021; Hendrycks et al., 2021) to advanced code generation (Jain et al., 2025; Grattafiori et al., 2024; Chen et al., 2021). Traditionally, the reasoning capabilities of these foundation models (DeepSeek-AI et al., 2024; OpenAI et al., 2023; Bai et al., 2023) have relied heavily on the Chain-of-Thought (CoT) paradigm (Wei et al., 2022; Kojima et al., 2022). By articulating intermediate steps as sequences of discrete tokens, CoT enables models to decompose complex problems into manageable sub-tasks (Yao et al., 2023; Wang et al., 2023). While effective, the visible intermediate state is a single token rather than the full predictive distribution. This motivates methods that retain distributional or hidden-state information between visible reasoning steps (Zhang et al., 2025a; Wu et al., 2026).

Recent work has explored non-discrete reasoning through probability-weighted embeddings, recurrent hidden states, projected soft thoughts, stochastic mixtures, RL-based continuous scaling, and learned latent tokens (Zhang et al., 2025a; Hao et al., 2025; Xu et al., 2025; Wu et al., 2026; Butt et al., 2026; Su et al., 2025). These methods broaden the representation space, but do not learn state-dependent test-time switching between continuous exploration and discrete grounding; some also require specialized latent-vocabulary training. This gap motivates a controller over reasoning representations (Section 5).

Furthermore, purely continuous reasoning introduces two persistent challenges in our experiments. First, repeated feedback of concept tokens—which can place the model in an out-of-distribution input regime (Zhang et al., 2025a)—can progressively separate the internal trajectory from representations induced by ordinary token inputs; we call this **manifold deviation**. Second, without explicit discrete

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 2136–2164
October 24-29, 2026 ©2026 Association for Computational Linguistics

checkpoints, the system can remain in abstract planning without committing to a concrete step; we call this **anchoring failure**. Across the six-domain diagnostic in Appendix J, HybridThink keeps the empirical threshold-exit rate below 2%, whereas 86% of AIME failures under Vanilla Soft occur after crossing the calibrated deviation threshold. Appendix K gives the complementary analytical model. We propose **HybridThink** as an algorithmic coordination mechanism between continuous semantic exploration and discrete symbolic grounding. Continuous concept tokens preserve dense uncertainty information, whereas discrete tokens provide low-distortion checkpoints that constrain accumulated state drift. An RL-optimized policy learns when to switch between these representations from the current semantic state and uncertainty signal; the resulting mechanism is analyzed from an information-theoretic perspective in Appendix K. Further conceptual motivation appears in Appendix C. A complementary neurobiological analogy is discussed in Appendix N, but it is not an assumption required by the method. Building upon these foundations, our contributions are as follows:

- We propose **HybridThink**, a self-adaptive framework that reconciles the expressivity of continuous reasoning with the precision of discrete decoding, effectively resolving the critical challenges of manifold deviation and anchoring failure.
- We formulate the reasoning mode selection as a sequential decision process optimized via RL algorithms, enabling the policy to dynamically adapt reasoning granularity based on real-time uncertainty estimation.
- We introduce **Error-Centric Experience Replay (ECER)**, a *targeted failure-biased* training strategy that prioritizes learning from failure trajectories to prevent boundary forgetting and ensure robust convergence.
- We implement **Inductive Exploration Shaping (IES)** to inject inductive bias into the policy initialization, preventing premature collapse to standard CoT and facilitating the learning of complex switching patterns.
- Extensive evaluations show the best accuracy in our controlled comparison across mathematics and coding benchmarks while reducing token usage relative to the evaluated baselines.

2 Preliminary Analysis

This section presents the empirical motivation for our hybrid approach. Section 5 summarizes the most relevant methods, and Appendix B provides an extended discussion.

2.1 Motivation and Setup

Before detailing our HybridThink architecture, we posit our first hypothesis regarding the nature of optimal reasoning.

Hypothesis 1 (The Necessity of Duality)

Optimal reasoning for complex tasks is neither strictly linear (Discrete) nor purely associative (Continuous). It requires a *Hybrid* capability that utilizes continuous space for broad semantic exploration and discrete space for precise information anchoring.

To validate this, we examine whether dynamic switching offers a theoretical advantage over static termination strategies.

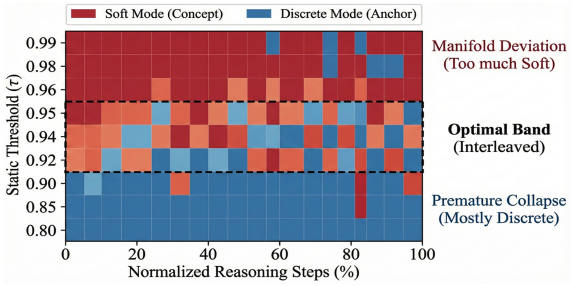
We hypothesize that complex problems require alternating phases of abstract planning and precise calculation. To test this, we evaluated a dynamic threshold strategy where the model switches to discrete mode only if the maximum probability P_{max} exceeds a threshold τ. Unlike irreversible static baselines, this approach allows a return to continuous mode if confidence decreases. We conducted evaluations using `QwQ-32B` (Team et al., 2025) on two mathematical datasets: *AIME 2024* for OOD difficulty and *MATH* for statistical significance, alongside *LiveCodeBench* for coding.

2.2 Performance Observations

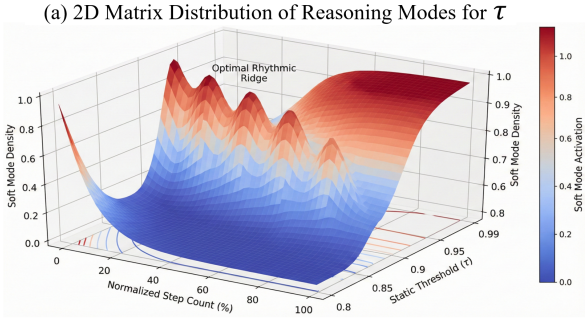
Method	AIME	MATH	LCB
Baseline: Soft Thinking	83.33	98.00	62.72
Dynamic Thresholds ($P_{max} > \tau$)			
Threshold $\tau = 0.8$	83.33	97.40	62.01
Threshold $\tau = 0.9$	**86.67**	**98.20**	61.29
Threshold $\tau = 0.95$	83.33	97.80	**62.72**
Threshold $\tau = 0.99$	80.00	96.60	62.37

Table 1: Preliminary analysis on `QwQ-32B`. We observe consistent trends across small, high-difficulty sets (*AIME*) and larger, statistically robust sets (*MATH*, *LCB*). Dynamic thresholds at $\tau \approx 0.9$ generally outperform static soft thinking, but no single heuristic is optimal across all domains.

Table 1 highlights two critical phenomena. First, a performance ceiling exists where specific dynamic thresholds such as $\tau = 0.9$ outperform the



(a) 2D Matrix Distribution of Reasoning Modes for τ



(b) 3D topographical evolution of Reasoning Modes for τ

Figure 1: (a). Distribution of reasoning modes across normalized step count for varying static thresholds τ; (b). 3D topographical evolution of reasoning mode distributions across normalized step counts. The landscape illustrates the transition dynamics for varying static thresholds τ.

static Soft Thinking baseline on math tasks. Second, manual heuristics are sensitive; a threshold optimized for mathematics may differ from coding tasks. The *MATH* dataset confirms the trend observed in *AIME*, providing statistical confidence that intermediate "softness" is beneficial.

2.3 Summary of Mode Distribution

To understand the temporal dynamics of these regimes, we visualized the distribution of reasoning modes across normalized steps (Figure 1). The visualization confirms the structural differences between the identified regimes:

As shown in Figure 1(a), **low thresholds** ($\tau \leq 0.8$, blue curves) lead to immediate collapse into discrete tokens, mimicking the linear trajectory of standard CoT. **High thresholds** ($\tau \geq 0.99$, red curves) result in prolonged soft reasoning phases but suffer from the anchoring problem and subsequent drift. Crucially, the **optimal performance band** (indicated by the dotted box) exhibits a distinct interleaved pattern, characterized by a rhythmic alternation of modes.

The 3D topographical evolution in Figure 1(b) further illustrates this landscape, where the transition from the “discrete plain” to the “continuous plateau” highlights the necessity of a learned policy to navigate the narrow ridge of optimal reasoning. Representative policy trajectories and detailed implications are provided in **Appendices D and E**, respectively.

3 Methodology

Based on the preliminary analysis, we posit our second hypothesis regarding the implementation of hybrid reasoning.

Hypothesis 2 (Policy Dependency)

The transition between reasoning modes cannot be robustly defined by static heuristics or confidence thresholds. It requires a *learned, state-aware policy* that dynamically assesses both semantic context and internal uncertainty to execute timely anchoring decisions.

Driven by the necessity of a policy-dependent switching mechanism, we now present the **HybridThink** framework. This architecture integrates the generative flexibility of Soft Thinking with the decisive grounding of discrete CoT, governed by a learnable RL agent that optimizes reasoning fidelity. Appendix K provides the information-theoretic interpretation, while Appendix N presents a complementary biological analogy.

3.1 HybridThink Framework

We formulate LLM reasoning as a sequential decision-making problem, modeled as a Markov Decision Process (MDP) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$. As illustrated in Figure 2(c), the process is divided into three distinct stages: *Prefilling*, *Thinking / Reasoning*, and *Inference*. During the iterative thinking phase, a gating policy network (or gating actor) dynamically monitors semantic states to determine the optimal reasoning modality. The design is intended to preserve useful continuous exploration while maintaining stable gradient flow and timely discrete anchoring (Appendix F).

State Space ($\mathcal{S}$). The state s_t at step t encapsulates the semantic context and model confidence. Specifically, let $h_t \in \mathbb{R}^d$ be the hidden state of the last token from the frozen LLM final layer, and $H(p_t)$ be the Shannon entropy derived from the next-token distribution (visualized as the "Probability" signal in Figure 2(c)). The state is defined as the concatenation: $s_t = [h_t; H(p_t)]$.

Policy Network Architecture. To ensure minimal overhead, the policy network $\pi_\theta(a_t|s_t)$ is designed as a lightweight Multi-Layer Perceptron (MLP). It projects the high-dimensional state s_t

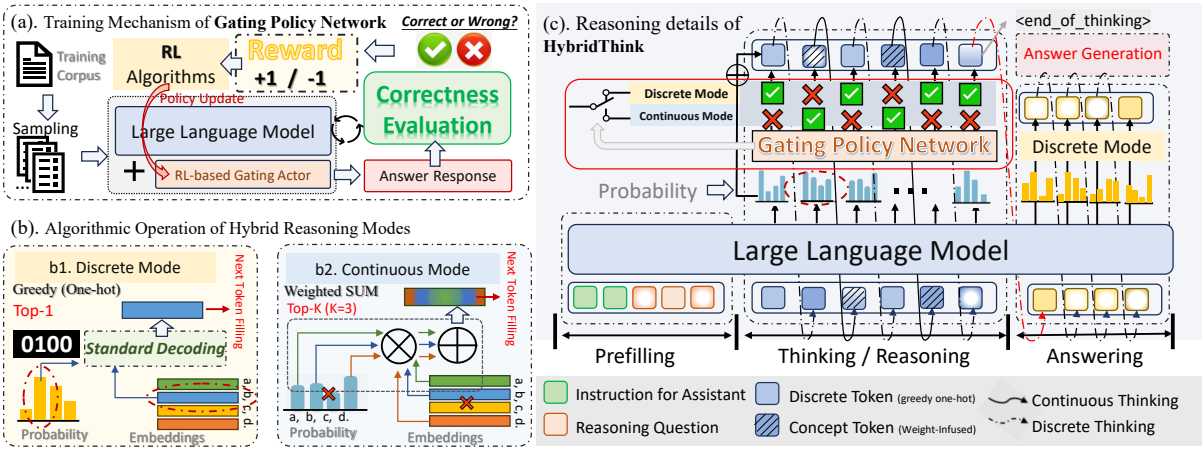


Figure 2: **Architecture of the HybridThink Framework**. (a) **Gating Policy Network Training Mechanism**: Illustrates the RL optimization loop where the Gating Actor is updated based on rewards from Correctness Evaluation. (b) **Algorithmic Operation of Reasoning Modes**: (b1) Discrete Mode utilizes standard greedy one-hot decoding; (b2) Continuous Mode generates **Concept Tokens** via a probability-weighted sum of top-k embeddings ($k = 3$ for illustration). (c) **Reasoning Details of HybridThink**: Displays the end-to-end trajectory consisting of **Prefilling**, **Thinking / Reasoning** (alternating modes), and the final **Inference** stage for answer generation.

into a binary action logic:

$$\pi_\theta(s_t) = \text{Softmax}(\mathbf{W}_2 \cdot \sigma(\mathbf{W}_1 s_t + \mathbf{b}_1) + \mathbf{b}_2) \quad (1)$$

where σ denotes the ReLU activation function. The network outputs the probability distribution over two modes: Continuous Exploration ($a_t = 0$) and Discrete Anchoring ($a_t = 1$). Simultaneously, to facilitate PPO training, we maintain a value network $V_\phi(s_t)$ sharing the same architecture, which estimates the expected return of the current state.

Action Space ($\mathcal{A}$) and Execution. The framework alternates between two distinct generative modes based on the policy output:

- **Continuous Mode ($a_t = 0$):** In this mode, we adopt the *Concept Token* mechanism from (Zhang et al., 2025a). As shown in Fig. 2(b2), the model generates a high-dimensional weighted embedding $\tilde{e}_t$ instead of collapsing to an index. This is constructed by calculating a probability-weighted sum of token embeddings in the vocabulary (filtered by top-k for noise reduction):

$$\tilde{e}_t = \sum_{v \in \mathcal{V}} p_t(v) \cdot E(v) \quad (2)$$

  This allows the LLM to propagate a "superposition" of multiple reasoning paths simultaneously, effectively traversing the semantic manifold without information loss.

- **Discrete Mode ($a_t = 1$):** The model performs standard greedy decoding to generate a discrete text token y_t. As illustrated in Fig. 2(b1), this action collapses the distribution into a deterministic index. The embedding $E(y_t)$ fed to the next step serves as a symbolic anchor, grounding the reasoning chain **into a concrete state** and resetting accumulated manifold deviation to ensure precision.

Reward Function with Step Penalty. Continuous reasoning trajectories within high-dimensional latent manifolds often suffer from cumulative deviation drift, which ultimately degrades logical consistency. To counteract this instability, we design a multi-objective reward function that balances global correctness with local computational efficiency, as illustrated in Fig. 2(a):

$$R_{total} = \mathbb{I}(y = y_{gt}) - \lambda \cdot L_{step} \quad (3)$$

Here, $\mathbb{I}(y = y_{gt})$ is a sparse indicator reward, granting $+1$ if the final answer matches the ground truth and -1 otherwise. The term $\lambda \cdot L_{step}$ introduces a cumulative step-wise penalty, where λ is a hyperparameter specifically tuned to regularize the reasoning trajectory. This penalty is crucial: it incentivizes the gating policy to collapse abstract continuous thoughts into verifiable discrete anchors as soon as sufficient semantic information is gathered, preventing the model from loitering redundantly and inefficiently in the continuous latent space.

Test-time Policy Execution. During training, actions are sampled stochastically from π_θ to encourage exploration. During evaluation, we adopt a

greedy strategy where $a_t = \arg\max_a \pi_\theta(a|s_t)$ to ensure stable and reliable trajectories. As depicted in the *autoregressive generation* procedure in Figure 2(c), the complete process consists of *Prefilling* the prompt, an iterative thinking phase where the policy alternates modes, and a final Inference stage to generate the response.

3.2 Training Strategy

We utilize Proximal Policy Optimization (PPO) (Schulman et al., 2017) to optimize the learned policy network π. Its clipped surrogate permits multiple minibatch epochs while providing a favorable empirical balance among sample efficiency, implementation simplicity, and update stability. Relative to an unclipped REINFORCE update (Williams, 1992), clipping discourages large changes in action probabilities; this is useful for maintaining the balance between continuous exploration and discrete anchoring.

To address sparse rewards and the exploration-exploitation dilemma inherent in high-dimensional reasoning, we implement two specific strategies:

1. Error-Centric Experience Replay (ECER). To enhance the robustness of the gating mechanism and prevent boundary forgetting, we implement a targeted training strategy that prioritizes learning from failure trajectories. During the RL update cycle, the training batch is constructed by intelligently mixing fresh on-policy data (D_{new}) with historical error samples (D_{ecer}) according to ratios ρ_{new} and ρ_{err} respectively. By over-sampling these failure modes, particularly instances of excessive manifold deviation or premature termination, the agent learns to refine its anchoring decisions at critical logical boundaries where standard on-policy sampling might provide insufficient signal.

2. Inductive Exploration Shaping (IES). Standard RL initialization often leads to a rapid collapse into the well-optimized discrete CoT behavior. To counteract this, we inject domain-specific inductive biases during the initial policy shaping phase. By manipulating the initial logit bias of the policy network, we enforce a high Continuous-to-Discrete (S/D) token ratio ($\pi_{soft}^{init} \gg \pi_{hard}^{init}$). This shaping mechanism ensures the agent accumulates sufficient experience within the continuous semantic manifold before gradient updates begin to fine-tune the switching logic for precise logical anchoring.

4 Experiments and Results

Here, we focus on the performance validation of our HybridThink on mathematical and code generation benchmark tests. To more deeply validate the effectiveness of HybridThink and explore its underlying mechanisms, we conducted extensive extended experiments and analyses. Due to space constraints, these contents are placed in the appendices; Appendix A provides an overview.

4.1 Experimental Setup

Benchmarks. We evaluate our method on seven benchmark tasks. The mathematics domain includes *Math500* (Lightman et al., 2024), *AIME 2024*, *GSM8K* (Cobbe et al., 2021), and *GPQA-Diamond* (Rein et al., 2024). The programming domain includes *HumanEval* (Chen et al., 2021), *MBPP* (Austin et al., 2021), and *LiveCodeBench* (Jain et al., 2025).

Training Data. To ensure data integrity, we train the policy on *GSM8K*, *MATH*, and *MBPP* splits. Benchmarks lacking training sets or used for OOD testing (*AIME*, *GPQA*, *HumanEval*, *LCB*) are evaluated in a zero-shot setting using these pretrained policies.

Models. We utilize three open-source LLMs: `QwQ-32B` (Team et al., 2025), `DeepSeek-R1-Distill-Qwen-32B` (Guo et al., 2025), and `Llama-3.1-8B-Instruct` (Grattafiori et al., 2024).

Prompts, model details, extended hyperparameters, and code implementation are documented in Appendices Q, R, S, and T, respectively.

Decoding Protocol and Baselines. All main HybridThink results use deterministic $\arg\max$ gating, deterministic top-k concept-token construction, and temperature-zero greedy decoding in the discrete phase. Thus, HybridThink itself introduces no test-time sampling. Standard CoT uses 16 generations per problem to estimate Pass@1, while Greedy CoT uses one generation with $T = 0$. Vanilla Soft Thinking follows the optimized sampling and stopping protocol of Zhang et al. (2025a). We additionally evaluate Stochastic Soft Thinking using the stronger of its Gumbel-Softmax and Dirichlet variants (Wu et al., 2026); implementation details for this baseline are provided in Appendix G.4, while additional controls and sensitivity analyses, including the deterministic-versus-stochastic HybridThink comparison, appear in Appendix G.

Model	Method	MATH	AIME	GSM8K	GPQA	Avg.	Length ↓
QwQ-32B	CoT (Sampling)	97.60	76.67	86.66	64.14	81.27	6472
	CoT (Greedy)	97.00	80.00	96.59	65.15	84.69	5967
	Vanilla Soft	98.00	83.33	96.82	67.17	86.33	5719
	HybridThink (Ours)	**98.60**	**86.67**	**97.88**	**71.72**	**88.72**	**3672**
DeepSeek-32B	CoT (Sampling)	94.40	73.33	95.60	63.13	81.62	4995
	CoT (Greedy)	93.00	63.33	95.30	59.09	77.68	5286
	Vanilla Soft	95.00	76.67	95.83	64.65	83.04	3875
	HybridThink (Ours)	**96.60**	**83.33**	**96.97**	**70.71**	**86.90**	**2127**
Llama-3.1-8B	CoT (Sampling)	50.40	3.33	81.88	30.30	41.48	1542
	CoT (Greedy)	49.00	3.33	79.61	27.78	39.93	1498
	Vanilla Soft	51.80	6.67	83.32	32.83	43.66	1305
	HybridThink (Ours)	**55.00**	**10.00**	**86.43**	**37.88**	**47.33**	**820**

Table 2: Main Results on Mathematics Datasets. HybridThink (RL) consistently outperforms baselines, validating that the learned policy successfully manages manifold deviation and anchoring. **Note:** The 'Length' column reports the average token count of **correctly solved instances**, highlighting the efficiency of our method in generating concise, valid reasoning paths.

4.2 Main Results

The aforementioned Table 2 and the subsequent Table 3 present the comprehensive quantitative comparison results across multiple widely-used mathematics and coding benchmarks.

HybridThink consistently outperforms all baselines across the evaluated models. For QwQ-32B, our method achieves an average accuracy of **88.72%** on math tasks, a substantial improvement compared to standard CoT. The RL-based policy demonstrates superior adaptability and robustness across domains. Crucially, the token efficiency is significantly improved. As shown in Table 2, HybridThink requires significantly fewer tokens compared to standard CoT. This efficiency gain, as illustrated in the "Thinking / Reasoning" phase of Figure 2(c), stems from the model's ability to utilize high-density concept tokens to bypass verbose intermediate verbalization, reserving discrete tokens only for necessary logical anchoring. Wall-clock results are reported in Appendix M.

The advantage also holds against stochastic continuous reasoning. Across QwQ-32B and DeepSeek-32B, HybridThink exceeds the strongest Stochastic Soft Thinking variant on all seven reported math and coding metrics, including gains of 7.71 and 10.41 points on AIME24 and 9.32 points on LiveCodeBench for QwQ-32B (Appendix Table 11).

Table 3 displays the results for coding tasks. Similar to the math domain, HybridThink demonstrates superior robustness. The learned policy adapts effectively, achieving the highest average score of **89.83%** on QwQ-32B. Notably, the reduction in token length is most dramatic in the coding domain (**2965** tokens vs 4899 for CoT, a reduction of approximately **39.5%**), suggesting that for code generation, the model can leverage continuous reasoning to bypass verbose syntactic boilerplate.

4.3 Ablation Studies

To comprehensively validate our architectural decisions, we conduct specific ablation studies analyzing the impact of initialization strategies, error-centric learning, step penalty mechanisms, and adaptive policy necessity. We include *MATH* (500 samples) in these studies to ensure statistical significance alongside *AIME* (30 samples). For a theoretical justification of these design choices, please refer to **Appendix F**.

4.3.1 Impact of Initialization Strategy (IES)

To verify the necessity of **Inductive Exploration Shaping**, we compared training runs with varying initial Continuous to Discrete (S/D) ratios.

As illustrated in Figure 3, initiating with a low probability of continuous thought ($\pi_{soft}^{init} \approx 0.1$) leads to rapid convergence on local optima. Conversely, a high initial continuous bias ($\pi_{soft}^{init} \approx 0.9$) forces the model to explore the continuous manifold early in training, allowing it to discover more efficient reasoning shortcuts.

4.3.2 Efficacy of ECER Sampling Rate

We systematically assessed the Error-Centric Experience Replay (ECER) strategy by modulating the error trajectory ratio ρ_{err} within our training loop, as shown in Figure 2(a). Data in Table 4 reveals that on-policy training ($\rho_{err} = 0\%$) provides insuf-

Model	Method	HumanEval	MBPP	LCB	Avg.	Length ↓
QwQ-32B	CoT (Sampling)	97.56	97.67	62.01	85.75	4899
	CoT (Greedy)	95.73	96.50	57.35	83.19	3833
	Vanilla Soft	98.17	97.67	62.72	86.19	4110
	HybridThink (Ours)	**99.39**	**98.05**	**72.04**	**89.83**	**2965**
DeepSeek-32B	CoT (Sampling)	96.95	94.94	57.35	83.08	4744
	CoT (Greedy)	87.20	87.55	43.37	72.71	2900
	Vanilla Soft	97.56	95.33	59.50	84.13	3834
	HybridThink (Ours)	**98.78**	**97.28**	**61.29**	**85.78**	**2729**
Llama-3.1-8B	CoT (Sampling)	46.34	60.70	24.37	43.80	500
	CoT (Greedy)	45.12	58.37	22.22	41.90	480
	Vanilla Soft	48.78	63.81	26.88	46.49	278
	HybridThink (Ours)	**51.22**	**66.15**	**29.39**	**48.92**	**262**

Table 3: Main results on coding datasets. HybridThink achieves the highest accuracy among the evaluated methods while reducing generation length. As in Table 2, 'Length' denotes the average number of tokens for correct solutions.

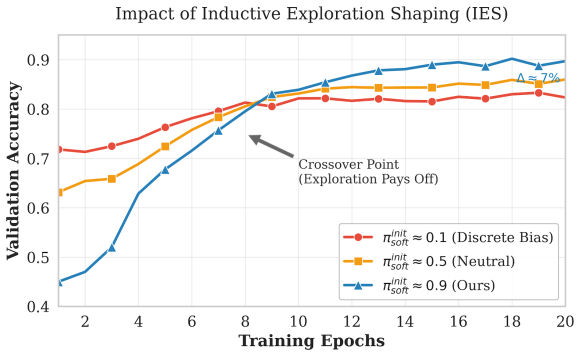


Figure 3: Impact of Inductive Exploration Shaping (IES) on validation accuracy across training epochs. While the high-continuous initialization ($\pi_{soft}^{init} \approx 0.9$) initially underperforms due to Out-of-Distribution (OOD) states, it demonstrates superior convergence and achieves a final accuracy gain of approximately 7% after a distinct crossover point where the exploration utility offsets the initial instability.

ficient protection against manifold deviation, as the agent lacks consistent exposure to critical failure modes during optimization. Conversely, increasing the error sampling rate significantly enhances logical robustness, with peak performance realized at $\rho_{err} = 70\%$. This confirms that a failure-biased sampling distribution empowers the Gating Actor to more precisely refine its switching boundaries, ensuring the model effectively anchors the semantic trajectory whenever internal states drift into out-of-distribution regions. Appendix L provides the corresponding failure taxonomy.

4.3.3 Sensitivity to Step Penalty

We investigate the sensitivity of the model to the step penalty coefficient λ, which regulates the trade-off between exploration and efficiency.

ECER Ratio	AIME Acc	MATH Acc
0%	83.33	96.20
10%	83.33	97.40
30%	86.67	98.20
70% (Ours)	**86.67**	**98.60**
90%	86.67	98.40

Table 4: Ablation on ECER Sampling Rate (ρ_{err}). A balanced mix of new on-policy data and historical error replay yields optimal results across both AIME (small, hard) and MATH (large, diverse) datasets.

Penalty λ	AIME Acc	MATH Acc
$\lambda = 0$	80.00	95.40
$\lambda = 0.005$	86.67	98.20
$\lambda = 0.01$ **(Ours)**	**86.67**	**98.60**
$\lambda = 0.05$	83.33	97.80
$\lambda = 0.1$	80.00	96.80

Table 5: Sensitivity analysis of Step Penalty λ on QwQ-32B. Extreme values lead to either manifold deviation ($\lambda \to 0$) or premature anchoring ($\lambda \to 0.1$).

Table 5 clearly confirms that $\lambda = 0.01$ strikes an optimal performance balance. No penalty leads to semantic drift, while excessively harsh penalties force the model to mimic standard CoT behavior.

4.3.4 Adaptive Policy vs. Static Patterns

To isolate the contribution of the learned policy against rigid heuristics, we compared our RL-optimized agent against static switching patterns.

As shown in Table 6, the rigid "Fixed Sequences" fail to match our adaptive RL policy, indicating that the optimal reasoning rhythm is essentially semantic rather than purely structural.

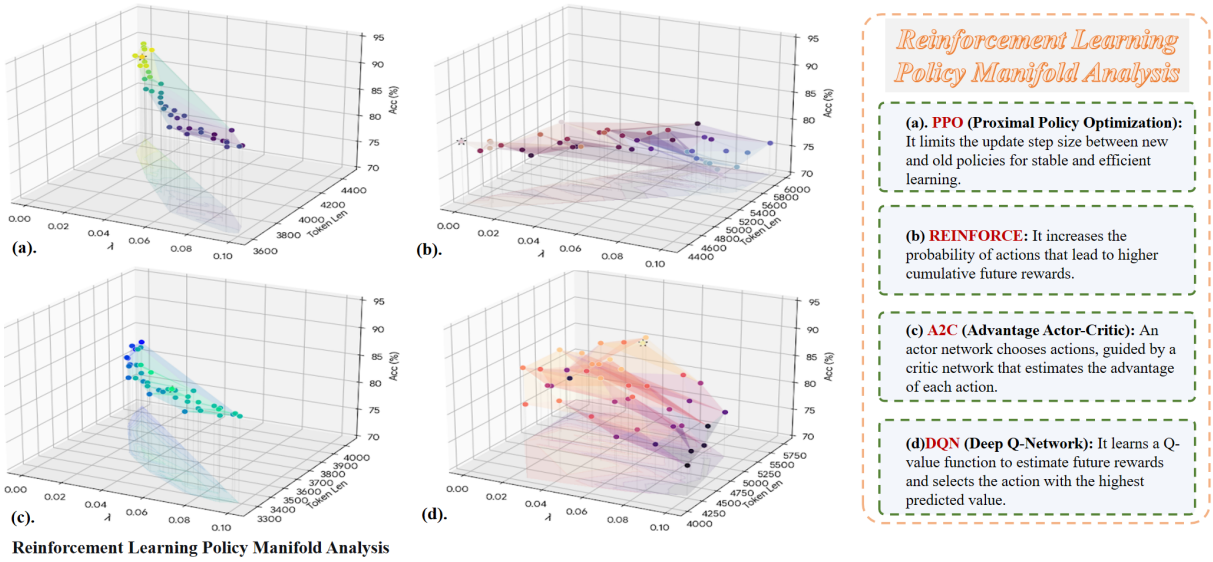


Figure 4: Manifold analysis of RL optimization strategies across varying penalty coefficients λ. Each subplot visualizes the joint distribution of step penalty (X-axis), average token length (Y-axis), and accuracy (Z-axis). The semi-transparent surfaces represent the learned performance manifolds, where the red stars mark the optimal operating points. **PPO** demonstrates a stable high-accuracy ridge with efficient token utilization, whereas **REINFORCE** and **A2C** exhibit significant variance or premature convergence into suboptimal efficiency zones. **DQN** fails to capture the continuous-state dynamics, resulting in a fragmented manifold.

Method / Variant	**AIME**	**MATH**
Hybrid (Random Policy)	30.00	45.20
Fixed Sequences Pattern `S-D-S-D...`	80.00	88.60
HybridThink (Ours)	**86.67**	**98.60**

Table 6: Comparison against fixed switching patterns on `QwQ-32B`. The significant drop in performance for Fixed Sequences confirms that optimal switching is state-dependent, not structural.

4.3.5 Robustness of RL Policy Optimization

To further justify the critical selection of PPO as our core optimization framework, we visualize the **3D performance-efficiency manifolds** for various RL algorithms in Figure 4. This comprehensive spatial representation reveals the intricate joint dynamics of reasoning fidelity and computational cost under the specific constraint of the step penalty λ.

We observe that **PPO** forms a remarkably coherent and elevated surface, indicating that it consistently maintains high accuracy while effectively compressing the reasoning trajectory as λ increases across diverse and complex scenarios. This is clearly visualized as a stable high-accuracy ridge spanning the 3D manifold. In contrast, the manifold for **REINFORCE** is characterized by significant high-frequency oscillations and localized collapses, directly reflecting the instability inherent in its gradient estimation for these complex, high-dimensional reasoning states encountered during the hybrid reasoning process.

Furthermore, **A2C** converges to a "discrete-heavy" plateau prematurely, sacrificing the benefits of continuous states for safe but less accurate anchoring. Finally, the fragmented manifold of **DQN** confirms its inability to model the dense semantic space h_t effectively, yielding suboptimal performance across all λ settings. These results confirm PPO provides the necessary stability to navigate the complex trade-off between reasoning fidelity and computational efficiency. Appendix I reports the corresponding quantitative comparison and selection rationale.

5 Related Work

Discrete reasoning methods expose intermediate steps as tokens. Chain-of-Thought (CoT) prompting elicits multi-step rationales (Wei et al., 2022), with extensions such as Self-Consistency (Wang et al., 2023), Tree-of-Thoughts (Yao et al., 2023), and Graph of Thoughts (Besta et al., 2024). These methods provide readable structure, but committing to one token can discard alternatives encoded in the predictive distribution. Continuous approaches instead retain richer internal representa-

tions through vocabulary mixtures, recurrent hidden states, or projected soft thoughts (Zhang et al., 2025a; Hao et al., 2025; Xu et al., 2025). Nevertheless, vocabulary mixtures can be dominated by their highest-probability token (Wu et al., 2026), while repeated out-of-distribution continuous inputs raise multi-step stability concerns (Zhang et al., 2025a). These observations motivate coordinating continuous exploration with discrete grounding rather than using either representation throughout.

A complementary line improves reliability through process supervision and Process Reward Models (Lightman et al., 2024; Uesato et al., 2022; Zhang et al., 2025b), while verifier-guided search expands and ranks multiple explicit trajectories (Lightman et al., 2024; Yao et al., 2023). HybridThink instead uses an internal value estimator and an RL-trained policy to choose the reasoning modality within a single autoregressive trajectory; it neither directly generates reasoning content nor expands multiple branches. Search-based verification is therefore complementary and could use HybridThink as a trajectory generator. This meta-level controller also reflects broader neuro-symbolic efforts to combine learned representations with explicit logical structure (Sarker et al., 2021). A fuller discussion of these paradigms and their relationship to HybridThink is provided in Appendix B.

6 Conclusion

To address the fundamental instability and inefficiency inherent in purely continuous reasoning paradigms, we introduced **HybridThink**, a reasoning framework that bridges the gap between the generative flexibility of continuous latent spaces and the structural precision of discrete symbolic reasoning. Our extensive evaluations on mathematics and coding benchmarks confirm that this dynamic coordination significantly enhances both reasoning accuracy and token efficiency across various model scales. The superiority of HybridThink underscores the potential of moving beyond static, linear reasoning chains toward a more fluid, state-aware processing paradigm. Overall, HybridThink, which dynamically learns to interleave continuous conceptual exploration with discrete symbolic grounding through an RL-optimized switching policy, offers a robust path toward developing LLMs that can navigate complex problem-solving with greater reliability and reduced computational redundancy. In future work, we will focus on more interpretable forms of continuous concept representation and the integration of external symbolic reasoning for stronger logical grounding. The cross-domain transfer results in Appendix O also motivate direct evaluation on open-ended QA and multi-agent reasoning, where continuous states may support broad semantic exploration or high-bandwidth intent alignment; these settings have not yet been empirically evaluated in this work.

Limitations

Despite the empirical success of HybridThink, our framework is subject to three fundamental limitations that govern its scalability, interpretability, and empirical scope.

Out-of-Distribution Theoretical Bound

The most significant limitation is the inherent theoretical bound imposed by the Out-of-Distribution (OOD) nature of continuous tokens. The base LLM, parameterized by Θ, approximates a conditional distribution $P(x_{t+1}|x_{0:t})$ defined over a discrete manifold $\mathcal{M}_{discrete}$. Our continuous tokens $\tilde{e}_t$, being convex combinations of embeddings, lie in the convex hull $\text{conv}(\mathcal{M}_{discrete})$ but almost surely outside $\mathcal{M}_{discrete}$.

While Transformer architectures exhibit some degree of linear robustness, the error term $\epsilon_t = \|\text{LLM}(\tilde{e}_t) - \text{Ideal}(\tilde{e}_t)\|$ accumulates exponentially with the sequence length k of continuous steps. Our "Anchoring" mechanism ($a_t = 1$) effectively resets this error accumulation ($\epsilon_t \to 0$). However, this implies that purely continuous reasoning for arbitrary lengths is theoretically bounded by the generalization gap of the frozen LLM. Nevertheless, as discussed in **Appendix O**, the pattern of switching appears to be a robust meta-cognitive skill that transfers across domains, suggesting that the learned policy captures a universal "reasoning rhythm" even if the underlying embeddings are domain-specific.

Interpretability vs. Performance Trade-off

A core trade-off of our approach is the loss of interpretability during "Soft Thinking" phases. While discrete CoT provides a fully readable trace, concept tokens are weighted mixtures of embeddings and therefore remain opaque to direct human inspection. We audit these phases through *Greedy Mapping*, which maps each concept token to its highest-probability discrete token; the case study

and internal research-team evaluation in Appendix H show that the resulting proxy traces are often readable. However, Greedy Mapping is an auditing proxy rather than a direct decoding of the continuous vector $\tilde{e}_t$, and thus does not eliminate the underlying black-box limitation. Developing probes that recover faithful natural-language explanations without disrupting the reasoning flow remains an important direction.

Model Scale and Agentic-System Coverage

Our evaluation is limited to the dense or distilled backbones considered in this work, with model sizes up to 32B parameters. Although Appendix P explains how HybridThink's token-level gate can coexist with a Mixture-of-Experts (MoE) router, we have not empirically validated the framework on larger MoE models, where expert routing, inter-device communication, and load balancing may interact nontrivially with the learned switching policy. We have also not evaluated HybridThink with other foundation models used to drive autonomous or tool-using agents. Consequently, the current results do not establish that the learned continuous–discrete switching policy transfers unchanged to large MoE backbones or broader agentic systems. Future work should assess task success, token and wall-clock efficiency, expert-load balance, tool-use reliability, and long-horizon stability in these settings.

Ethical Considerations

This work studies representation-level reasoning methods using publicly available research benchmarks and open-source model checkpoints. It does not use private user conversations, demographic attributes, or sensitive personal data, and all benchmark results are reported in aggregate. Users of the datasets and model checkpoints remain responsible for complying with their respective licenses and terms of use.

Appendix H reports a small internal interpretability audit conducted entirely by ten members of the research team, each of whom holds a Master's degree or higher in an AI-related field. The paper reports only aggregate judgments and no identifying information. Because the evaluators were involved in the research, the audit is not an independent human evaluation and may be subject to confirmation bias. It evaluates the readability of model outputs rather than human traits or behavior, and its selected sample is not used to make population-level claims.

HybridThink may improve reasoning accuracy and efficiency, but its continuous states are less transparent than fully discrete traces and may inherit biases or unsafe capabilities from the underlying model. Greedy Mapping should therefore be treated only as a diagnostic projection, not as a faithful explanation of the latent computation. Before deployment in high-stakes or user-facing settings, practitioners should conduct domain-specific safety evaluations, retain appropriate human oversight and audit logs, and apply the access controls and safeguards required for the underlying model.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62372364) and the Shaanxi Province Natural Science Foundation, China (No. 2026JC-YXQN-213).

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of Thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17682–17690.

Natasha Butt, Ariel Kwiatkowski, Ismail Labiad, Julia Kempe, and Yann Ollivier. 2026. Soft tokens, hard truths. In *International Conference on Learning Representations*, volume 2026, pages 114650–114675.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro

Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*.

Stanislas Dehaene, Jean-Pierre Changeux, and Lionel Naccache. 2011. The global neuronal workspace model of conscious access: From neuronal architectures to clinical applications. In Stanislas Dehaene and Yves Christen, editors, *Characterizing Consciousness: From Cognition to the Clinic?*, Research and Perspectives in Neurosciences, pages 55–84. Springer, Berlin, Heidelberg.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. 2025. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E. Weston, and Yuandong Tian. 2025. Training large language models to reason in a continuous latent space. In *Second Conference on Language Modeling*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pages 58791–58831.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bo Li, Guanzhi Deng, Ronghao Chen, Junrong Yue, Shuo Zhang, Qinghua Zhao, Linqi Song, and Lijie Wen. 2025. REMA: A unified reasoning manifold framework for interpreting large language model. *arXiv preprint arXiv:2509.22518*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pages 39578–39601.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1928–1937, New York, New York, USA. PMLR.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof Q&A benchmark. In *First Conference on Language Modeling*.

Md Kamruzzaman Sarker, Lu Zhou, Aaron Eberhart, and Pascal Hitzler. 2021. Neuro-symbolic artificial intelligence: Current trends. *AI Communications*, 34(3):197–209.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Dijia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qinqing Zheng. 2025. Token assorted: Mixing latent and text tokens for improved language model reasoning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 57144–57163. PMLR.

Minghong Su, Kejia Hu, Wei Liu, Yunhao Wu, Tao Wang, Chunyan Cao, Bomin Sun, Shikun Zhan, and Zheng Ye. 2024. Theta oscillations support prefrontal-hippocampal interactions in sequential working memory. *Neuroscience Bulletin*, 40(2):147–156.

Qwen Team et al. 2025. QwQ-32B: Embracing the power of reinforcement learning. Qwen Blog. Accessed 2026-08-26.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jason Weston and Sainbayar Sukhbaatar. 2023. System 2 attention (is something you might need too). *arXiv preprint arXiv:2311.11829*.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256.

Junhong Wu, Jinliang Lu, Zixuan Ren, Gangqiang Hu, Zhi Wu, Dai Dai, and Hua Wu. 2026. LLMs are single-threaded reasoners: Demystifying the working mechanism of soft thinking. In *International Conference on Learning Representations*, volume 2026, pages 4533–4548.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. 2025. SoftCoT: Soft chain-of-thought for efficient reasoning with LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23336–23351, Vienna, Austria. Association for Computational Linguistics.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 11809–11822.

Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D. Goodman. 2024. Quiet-STaR: Language models can teach themselves to think before speaking. In *First Conference on Language Modeling*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, volume 35, pages 15476–15488. Curran Associates, Inc.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Wang. 2025a. Soft Thinking: Unlocking the reasoning potential of LLMs in continuous concept space. In *Advances in Neural Information Processing Systems*, volume 38, pages 168990–169012. Curran Associates, Inc.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025b. The lessons of developing process reward models in mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10495–10516, Vienna, Austria. Association for Computational Linguistics.

A Overview of Supplementary Material

This appendix provides extensive supplementary evidence, theoretical derivations, and technical details to support the findings in the main text. For clarity, experimental and analytical studies are highlighted in `typewriter font`. The appendix is organized as follows:

1. Background & Motivation: We provide an extended literature review (B) and conceptual motivation (C). The practical reasoning dynamics are illustrated through a `Qualitative Case Study` (`D`), while the empirical foundation is established in the `Extended Preliminary Analysis` (`E`).

2. Foundations: This section covers the theoretical justification for ablation choices (F), an optional neurobiological analogy (N), and the information-theoretic basis of anchoring (K).

3. Experimental Evaluations: We present quantitative investigations into `Manifold Deviation` (`J`), a comprehensive taxonomy of `Failure Modes` (`L`), a comparative study of `RL Optimization Algorithms` (`I`), stochastic baselines (G.4), and an interpretability audit (H).

4. Performance & Generalization: Detailed reports on `Computational Efficiency` (`M`) and the `Cross-Domain Transferability` (`O`) of the gating policy are included to demonstrate robustness.

5. Implementation Details: Full specifications cover policy architecture ablations (P), additional sensitivity studies (G), prompt templates (Q), model details (R), extended hyperparameters (S), and code implementation (T).

6. Discussion: We address the limitations and potential future directions of our framework in the main-text Limitations section.

B Related Work

This appendix supplements Section 5 with a finer-grained comparison of reasoning paradigms and further clarifies HybridThink's relationship to search, process supervision, neuro-symbolic systems, and RL-based reasoning.

B.1 Expanded Comparison of Reasoning Paradigms

Discrete Reasoning and Chain-of-Thought. The emergence of Chain-of-Thought (CoT) prompting (Wei et al., 2022) marked a significant milestone in eliciting multi-step reasoning from LLMs. By encouraging the model to generate intermediate rationales before the final answer, CoT effectively decomposes complex problems into a series of locally manageable steps. Numerous extensions have since been proposed, including Self-Consistency (Wang et al., 2023), which leverages majority voting over multiple reasoning paths, and Tree-of-Thoughts (Yao et al., 2023), which implements tree search algorithms like BFS and DFS over the space of discrete thoughts. Graph of Thoughts (Besta et al., 2024) further generalizes this to arbitrary graph structures. Despite their success, representing each visible step by one selected token can omit alternatives encoded in the full predictive distribution, motivating methods that retain richer distributional or hidden-state information (Zhang et al., 2025a; Wu et al., 2026).

Continuous and Soft Reasoning. To overcome the limitations of discreteness, research has pivoted towards reasoning in continuous spaces. Early works such as Soft Prompts (Lester et al., 2021) and Prefix-Tuning (Li and Liang, 2021) demonstrated that optimizing continuous embeddings could steer model behavior more effectively than discrete text prompts. More recently, Soft Thinking (Zhang et al., 2025a), COCONUT (Hao et al., 2025), and SoftCoT (Xu et al., 2025) have proposed distinct forms of continuous-space reasoning: vocabulary-probability mixtures, recurrent hidden states, and projected soft thoughts, respectively. Wu et al. (2026) show that vocabulary-mixture Soft Thinking is often dominated by its highest-probability token and propose stochastic variants to escape this greedy pitfall. A separate concern is multi-step stability under repeated out-of-distribution continuous inputs (Zhang et al., 2025a), which our trajectory-level deviation analysis targets. Independently, REMA (Li et al., 2025) approximates a manifold from correct reasoning representations and measures erroneous states by their k-nearest-neighbor distance to that manifold, providing geometric evidence that reasoning errors correlate with representational deviation.

Process Supervision and Verification. Process supervision trains reward models from step-level feedback rather than final-answer labels alone (Lightman et al., 2024; Uesato et al., 2022; Zhang et al., 2025b). The broader self-correction literature surveys training-time, generation-time, and post-hoc mechanisms that use model- or tool-generated feedback to critique or refine outputs (Pan et al., 2024). HybridThink uses an internal critic (the

Value Network) to monitor the reasoning process, but applies that signal to the choice of reasoning *modality* rather than directly judging the correctness of each generated step.

Search-based Inference-time Scaling. PRM-guided best-of-N selection and tree-search methods rank or expand multiple explicit reasoning trajectories (Lightman et al., 2024; Yao et al., 2023). HybridThink instead controls the representation used within a single autoregressive trajectory, choosing when to compress reasoning in a continuous state and when to expose a discrete anchor. The two mechanisms are therefore complementary: a search procedure could use HybridThink as its trajectory generator, while a verifier ranks the resulting branches. We do not report a compute-matched search comparison in this work, so our empirical claims are limited to the non-search baselines in Tables 2–3; evaluating the combined system is left to future work.

B.2 Neuro-symbolic and Hybrid Architectures

The dichotomy between continuous and discrete reasoning parallels the distinction between connectionist and symbolic approaches. Neuro-symbolic AI (Sarker et al., 2021) seeks to combine learned pattern recognition with explicit logical structure. System 2 Attention first regenerates the input context to retain relevant information and then elicits the final response from that regenerated context (Weston and Sukhbaatar, 2023). STaR iteratively fine-tunes on self-generated rationales that yield correct answers (Zelikman et al., 2022), whereas Quiet-STaR learns token-wise internal rationales according to how much they improve future-token prediction (Zelikman et al., 2024). These methods illustrate staged or latent reasoning, but do not instantiate HybridThink's continuous–discrete gate. HybridThink can instead be viewed as a dynamic neuro-symbolic coordination mechanism, where continuous representations support exploration and discrete tokens provide explicit anchors.

B.3 Reinforcement Learning for Reasoning

Reinforcement Learning (RL) has become a standard tool for aligning LLMs (Ouyang et al., 2022). Beyond standard RLHF, recent work applies RL directly to reasoning behavior. DeepSeekMath (Shao et al., 2024) introduced Group Relative Policy Optimization (GRPO) to enhance mathematical reasoning without an external critic model. Our approach applies RL at the meta-level: the policy does not generate the reasoning content directly but instead controls the *structure* of the reasoning process, deciding when to switch between continuous and discrete modes to maximize expected stability and correctness.

C Conceptual Motivation and Biologically-Inspired Intuition

HybridThink is motivated by an algorithmic trade-off between continuous exploration and discrete grounding. As an optional interpretive analogy, these roles can be compared with interactions between associative memory and executive control; theta-mediated prefrontal–hippocampal coordination provides one such biological example (Su et al., 2024). The method itself does not model either brain region.

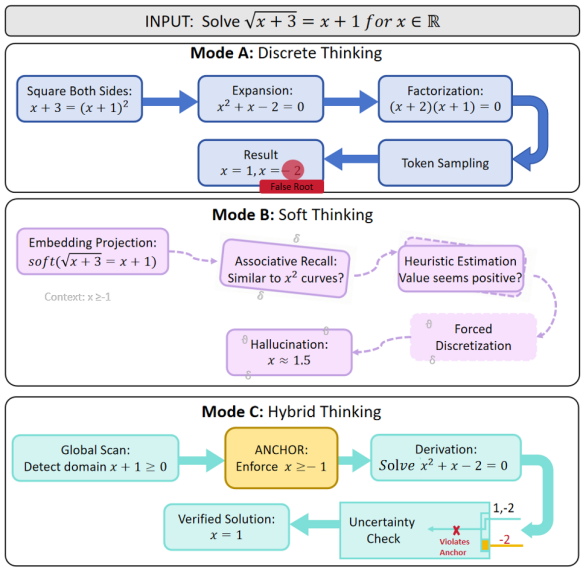


Figure 5: Conceptual comparison of human-like reasoning paradigms. **A. Discrete Thinking** (CoT) operates linearly but often misses implicit global constraints. **B. Soft Thinking** maintains global sensitivity but drifts along the semantic manifold. **C. HybridThink** (Ours) alternates associative continuous exploration and explicit logical anchoring. *Note: This figure illustrates macro-level problem-solving intuition rather than a microscopic trace of token-level generation.*

Figure 5 is conceptual rather than mechanistic. Its useful computational distinction is between high-bandwidth association and low-distortion symbolic control: HybridThink implements that distinction with concept tokens and discrete anchors, without assuming a correspondence between

model components and specific neural circuits.

D Qualitative Analysis: Case Study

To better understand the decision-making dynamics of the learned **Gating Policy Network**, we visualize the reasoning trajectories for three distinct problem instances from the *AIME 2024* dataset in Figure 6. The plots trace the entropy of the probability distribution (the "Probability" signal in Figure 2(c)) alongside the corresponding mode selected by the **Gating Actor**.

Case 1: Linear Strategic Execution. The first panel depicts a problem requiring a global domain check ($x > 0$). Here, the reasoning follows a monotonic "Plan then Solve" trajectory: an initial Soft mode phase to verify constraints, followed by Discrete execution.

Cases 2 & 3: Iterative Refinement. A more sophisticated behavior emerges in complex combinatorial tasks. As seen in Cases 2 and 3, the policy exhibits a "Discrete to Soft" switching capability. Upon encountering high-entropy signals during discrete calculation (e.g., anchoring a partial result like $\binom{10}{3} = 120$), the policy triggers a reversion to Soft mode. This effectively acts as a metacognitive pause, allowing the model to re-evaluate potential errors or edge cases before propagating them, ensuring robustness in multi-stage problem solving. An optional biological analogy for these transition patterns is discussed in **Appendix N**.

E Extended Preliminary Analysis

In Section 2.3, we briefly summarized the distinct behavioral regimes observed when using static switching thresholds. Here, we present the extended analysis of these modes.

The visualization in Figure 1 reveals three distinct behavioral regimes based on the threshold τ:

Regime 1: Premature Anchoring ($\tau \leq 0.8$). At lower thresholds, the model collapses into discrete mode immediately. This restricts exploration of the semantic manifold, reducing the model to a standard CoT reasoner with limited look-ahead capabilities. The entropy of the internal state drops too quickly, preventing the formation of complex, multi-hop reasoning bridges that are typically handled better in high-dimensional space. The model essentially gives up the "thinking" advantage of the continuous space to ensure safety, resulting in standard but verbose outputs.

Regime 2: Manifold Deviation ($\tau \geq 0.99$). At extremely high thresholds, the model persists in continuous mode. Zhang et al. (2025a) note that feeding continuous mixtures can place the model in an out-of-distribution regime and lead to generation collapse. In our threshold sweep, long runs without discrete anchors exhibit accumulated trajectory drift and eventually produce hallucinated or incoherent outputs. This empirical pattern motivates the periodic correction mechanism analyzed in Appendix J.

Regime 3: Optimal Rhythmic Alternation. The most effective thresholds exhibit a balanced, rhythmic interleaving of modes. This suggests that optimal reasoning is not monotonic but requires a sequence of exploration and anchoring. Specifically, we observe "bursts" of soft thinking followed by single discrete tokens, resembling a "Think, Check, Think" cycle. The discrete tokens act as checkpoints, purging the accumulated noise in the continuous vector and resetting the manifold trajectory, while the soft tokens allow for gradient-based semantic search.

F Theoretical Analysis of Ablation Choices

While Section 4.3 provides empirical evidence for our design choices, this section offers a theoretical justification for why **Inductive Exploration Shaping (IES)** and **Error-Centric Experience Replay (ECER)** are necessary for optimal convergence in our hybrid reasoning task.

F.1 Gradient Flow in Hybrid Manifolds (IES Justification)

Standard RL initialization typically starts with maximum entropy $\pi(a|s) \approx 0.5$. However, in our hybrid setting, the reward landscape is highly asymmetric.

- **The Discrete Trap:** The pre-trained LLM is highly optimized for discrete token generation. If the policy selects $a_t = 1$ (Discrete) early in training, it receives immediate, stable gradients consistent with its pre-training.
- **The Continuous Cliff:** Continuous reasoning ($a_t = 0$) initially places the model in an OOD state. Without sufficient exploration, the returns from this mode are high-variance and often negative due to manifold deviation.

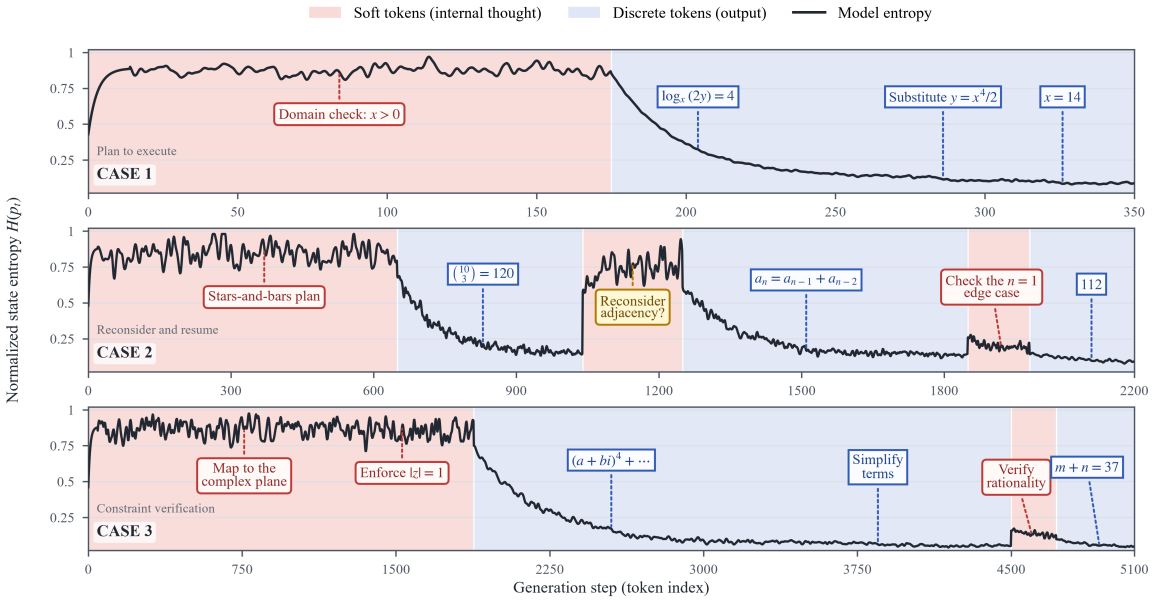


Figure 6: Trajectory Analysis of a complex counting problem (AIME 2024). The plots illustrate the reasoning state entropy $H(p_i)$ alongside the selected mode (Red=Soft, Blue=Discrete) for three distinct scenarios. While Case 1 demonstrates a linear planning to execution flow, Cases 2 and 3 exhibit complex iterative behavior where the policy dynamically reverts to Soft mode upon encountering high-entropy roadblocks during discrete calculation.

If initialized neutrally, the policy gradient $\nabla J(\theta)$ will naturally follow the path of least resistance towards the discrete mode, causing the policy to collapse to a standard CoT behavior before it learns to utilize the continuous space. **IES** ($\pi_{soft}^{init} \approx 0.9$) forces the agent to traverse the high-variance continuous landscape. From a gradient flow perspective, this ensures that the credit assignment mechanism (A_t) accumulates non-zero values for continuous actions, effectively "smoothing" the optimization landscape around the soft-thinking basin of attraction before the policy tightens.

F.2 Importance Sampling at the Decision Boundary (ECER Justification)

In high-precision reasoning tasks such as Math or Code, the "correct" trajectory is often singular, while failure modes are vast.

- **Distribution Shift:** As the policy improves, it naturally generates fewer errors. In standard on-policy PPO, the replay buffer D becomes dominated by successful, low-entropy trajectories.
- **Boundary Forgetting:** The critical decision boundaries, instances where the model *almost* drifted but successfully anchored, or *almost* anchored but correctly continued exploring, are located in the tails of the distribution.

By utilizing ECER with a 70% error ratio, we are effectively performing **Importance Sampling**. We artificially increase the density of training samples near the "failure boundary." Theoretically, this minimizes the variance of the gradient estimator for the switching logic. By frequently re-exposing the agent to states s_t that previously led to deviation (low reward), we maximize the gradient signal $\nabla_\theta \log \pi(a_t|s_t)A_t$ specifically for the "correction" actions, ensuring robust boundary discrimination that pure on-policy sampling would statistically neglect.

G Extended Analysis and Ablations

In this section, we present additional low-cost ablation studies to further validate the robustness of our framework's core components: state representation, concept token construction, and inference decoding strategy. All experiments were conducted using the `QwQ-32B` model.

G.1 State Representation Analysis

To determine the minimal sufficient statistics required for the policy network, we compared our composite state definition $s_t = [h_t; H(p_t)]$ against two variants: (A) only entropy $s_t = [H(p_t)]$, and (B) only semantic hidden state $s_t = [h_t]$.

As shown in Table 7, relying solely on entropy

State Input	AIME Acc	MBPP Acc
Only Entropy ($H(p)$)	83.33	97.28
Only Hidden (h_t)	83.33	96.50
Composite ($h_t; H(p)$)	**86.67**	**98.05**

Table 7: Ablation of State Representation on *AIME* and *MBPP*. Our composite approach outperforms single-modal inputs, indicating that optimal switching requires both uncertainty signals and semantic context.

or hidden states leads to suboptimal performance. Specifically, the entropy-only policy often fails to switch when semantic ambiguity exists despite low distributional entropy (confident hallucinations), while the hidden-state-only policy struggles to quantify uncertainty explicitly.

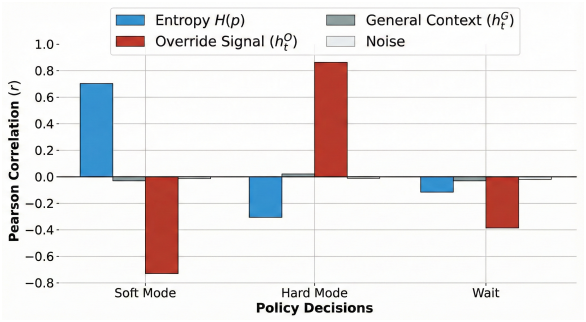


Figure 7: Correlation between State Features (h_t) and Policy Decisions. The bar chart visualizes the Pearson correlation coefficients (r) for each policy mode. Note the opposing forces in the Soft Mode column: high Entropy (blue) promotes it, while Override Signals (red) strongly suppress it.

To further validate the learned policy's decision boundaries, we visualized the feature correlations in **Figure 7**. The grouped bar chart reveals two distinct mechanisms: 1. **Entropy-Driven Exploration:** The "Soft Mode" is strongly positively correlated with high entropy (Blue bar, $r \approx 0.7$), confirming that uncertainty triggers exploration. 2. **Override Suppression:** Crucially, the "Override Signal" (Red bar) shows a strong negative correlation ($r \approx -0.7$) with Soft Mode and a strong positive correlation ($r > 0.8$) with Hard Mode. This demonstrates that specific semantic instructions can effectively "override" the entropy signal, forcing the model to anchor even when uncertainty is present.

G.1.1 Sensitivity to Concept Token Construction (k)

We evaluated the sensitivity to the Top-k parameter used in **Continuous Mode** (as conceptualized in Figure 2(b2)). While Figure 2 uses $K = 3$ for visual clarity, our empirical evaluation in Table 8 reveals an inverted U-shaped performance curve. Setting $k = 1$ reduces to standard CoT, while $k = 50$ introduces excessive noise. Our default $k = 20$ strikes the optimal balance between exploratory "softness" and semantic clarity.

Top-k Value	MATH Acc	GPQA Acc
$k = 1$ (Discrete)	97.60	64.14
$k = 5$	98.20	69.19
$k = 20$ **(Ours)**	**98.60**	**71.72**
$k = 50$	98.00	68.18

Table 8: Performance sensitivity to Top-k filtering for concept token generation. Extreme values degrade performance, confirming the need for a balanced "softness".

Table 8 **(visualized in Figure 8)** reveals an inverted U-shaped performance curve. Setting $k = 1$ reduces the model to standard CoT (controlled by the policy), yielding baseline performance. Extremely high values ($k = 50$) introduce excessive noise into the continuous embedding, degrading reasoning precision. Our default $k = 20$ strikes an optimal balance between semantic exploration and signal clarity.

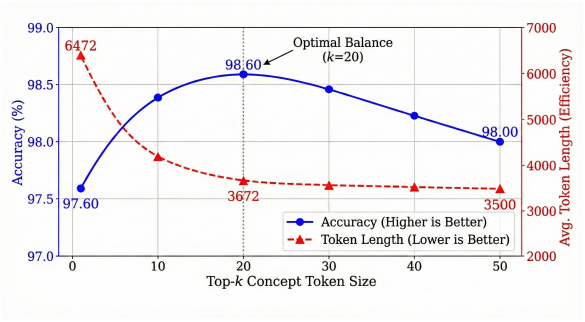


Figure 8: Impact of k on Accuracy vs. Efficiency. The dual-axis plot illustrates the trade-off: The blue curve (Accuracy) peaks at $k = 20$, termed the "Optimal Balance". Meanwhile, the red dashed line (Token Length) decreases as k increases, indicating that broader concept tokens allow for more compressed reasoning, though eventually at the cost of precision.

G.2 Reward Component Analysis

To justify the inclusion of the length penalty term $-\lambda \cdot L_{step}$, we compared our standard reward function against a version purely optimized for correctness ($\lambda = 0$).

Table 9 demonstrates that without the length penalty, the model tends to loiter in continuous

Method	AIME		HumanEval	
	Acc	Len	Acc	Len
No Penalty ($\lambda = 0$)	83.33	5820	98.78	4150
Ours ($\lambda = 0.01$)	**86.67**	**3672**	**99.39**	**2965**

Table 9: Impact of Length Penalty on *AIME* and *HumanEval*. Removing the penalty ($\lambda = 0$) leads to bloated generation lengths without significant accuracy gains, and in some cases, reduces robustness due to drift.

space or generate redundant verification steps, significantly increasing token usage. Surprisingly, accuracy on *AIME* drops slightly without the penalty, suggesting that the pressure to be concise effectively acts as a regularizer against manifold deviation.

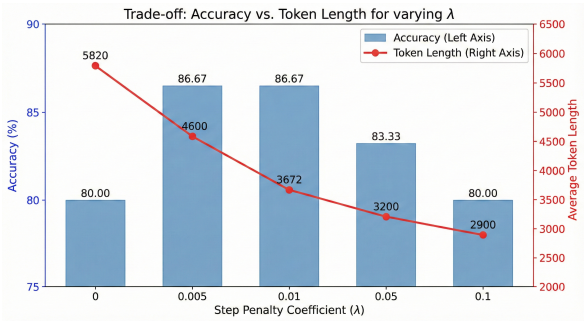


Figure 9: Trade-off: Accuracy vs. Token Length for varying λ. The blue bars represent accuracy, while the red line tracks token length. The plot identifies an optimal region around $\lambda = 0.01$ where accuracy is maximized (86.67%) while token length is significantly reduced (3672) compared to the unpenalized baseline.

G.3 Inference Decoding Strategy

Finally, we explored whether stochastic sampling of the policy's action a_t during inference could improve performance on hard exploration tasks compared to the greedy strategy used in our main results.

Decoding Strategy	MATH500	AIME24	LCB	MBPP
Greedy ($a = \arg\max \pi$)	**98.60**	**86.67**	72.04	**98.05**
Stochastic ($a \sim \pi, T = 0.7$)	98.00	83.33	**72.76**	97.00

Table 10: Greedy versus stochastic policy decoding on QwQ-32B. The main results use the deterministic greedy setting. Greedy is better on three of four benchmarks; hence sampling is not the source of HybridThink's reported gains.

Table 10 isolates policy-action sampling from the hybrid representation mechanism. Greedy decoding provides greater stability on *MATH500*, *AIME24*, and *MBPP*, whereas stochastic decoding offers a small advantage on *LiveCodeBench*, where diverse solution paths may be useful. This suggests potential for future adaptive decoding strategies without changing the deterministic protocol used for our main comparisons.

G.4 Stochastic Soft Baseline and Reproducibility Details

Table 11 compares HybridThink with Stochastic Soft Thinking (Wu et al., 2026). For each benchmark, the Stochastic Soft row reports the better result of its Dirichlet and Gumbel-Softmax implementations. HybridThink is better on every reported metric for both 32B backbones.

Baseline configurations. Standard CoT uses 16 generations per problem with temperature 0.6, top-$p = 0.95$, top-$k = 30$, and a maximum generation length of 32,768 to estimate Pass@1. Greedy CoT uses a single generation with $T = 0$. Vanilla Soft follows Zhang et al. (2025a): its concept-token width is selected from $n \in \{5, 10, 15, 20, 30\}$, entropy threshold from $\tau \in \{0.01, 0.05, 0.1, 0.2\}$, and Cold-Stop length from $k_{\text{stop}} \in \{128, 256, 512, 1024\}$, with $n = 15$ for QwQ-32B and $n = 10$ for DeepSeek-R1 models. The reported results use the best threshold/stop-length combination. For Stochastic Soft, we test Dirichlet scale $\alpha \in [1, 10]$ in increments of 1 and Gumbel-Softmax temperature $\tau_g \in [0.3, 0.9]$ in increments of 0.1, following Wu et al. (2026); their reported defaults are $\alpha = 4.0$ and $\tau_g = 0.5$, and Table 11 retains the stronger variant for each benchmark.

H Interpretability Audit via Greedy Mapping

Concept tokens are not directly readable. Following Zhang et al. (2025a), we construct an auditing proxy by replacing each soft step with the highest-probability vocabulary item, $\hat{x}_t = \arg\max_i p_t(i)$. We call the resulting sequence *Greedy Mapping*. The mapping exposes the dominant discrete path represented by the concept-token distribution; it does not reconstruct all information in the weighted embedding $\tilde{e}_t$ and is not a faithful natural-language explanation by construction.

Table 12 reports five fragments from DeepSeek-R1-Distill-Qwen-32B trajectories on AIME 2024 and LiveCodeBench. They were drawn from "continuous semantic-rich" segments, operationalized

Backbone	Method	MATH500	AIME24	GSM8K	GPQA	HE	MBPP	LCB
QwQ-32B	Stochastic Soft	97.20	78.96	96.36	67.67	97.56	97.66	62.72
	HybridThink	**98.60**	**86.67**	**97.88**	**71.72**	**99.39**	**98.05**	**72.04**
	Improvement	+1.40	+7.71	+1.52	+4.05	+1.83	+0.39	+9.32
DS-32B	Stochastic Soft	96.00	72.92	95.07	63.13	98.17	95.72	59.50
	HybridThink	**96.60**	**83.33**	**96.97**	**70.71**	**98.78**	**97.28**	**61.29**
	Improvement	+0.60	+10.41	+1.90	+7.58	+0.61	+1.56	+1.79

Table 11: Comparison with Stochastic Soft Thinking. HE denotes HumanEval and DS-32B denotes DeepSeek-R1-Distill-Qwen-32B. Stochastic Soft reports the better result of the Gumbel-Softmax and Dirichlet implementations of Wu et al. (2026).

in the rebuttal as segments with soft-token density above 70%.

We additionally conducted a small internal audit on GSM8K. We selected 10 representative complete mapped reasoning chains whose soft-token density exceeded 20%. Ten members of the research team, each holding a Master's degree or higher in an AI-related field, assessed logical coherence, mathematical rigor, and semantic readability. Table 13 reports the aggregate statistics supplied during rebuttal.

The high ratings provide evidence that the dominant-token projection is often readable in this small, selected sample. However, because all evaluators were members of the research team, the results may be affected by confirmation bias and should not be interpreted as an independent human evaluation. They do not establish causal faithfulness of the projection to the full continuous computation, and the non-random sample is not sufficient to support a population-level interpretability claim.

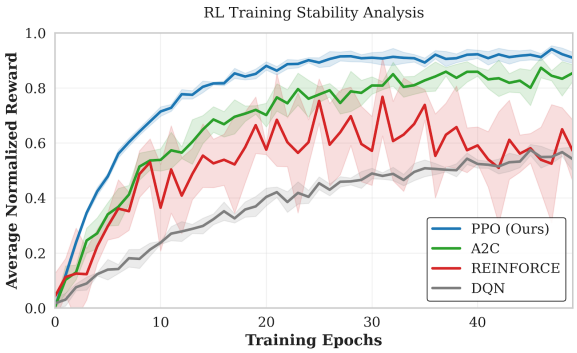


Figure 10: RL Training Stability Analysis. The plot compares the average reward variance during training for different algorithms. PPO (Blue) demonstrates high stability with monotonic improvement, whereas REINFORCE (Red) suffers from high variance and occasional collapse.

I RL Algorithm Selection

To justify our selection of Proximal Policy Optimization (PPO) as the core optimization algorithm, we conducted a comparative study against other prominent Reinforcement Learning baselines. Specifically, we evaluated:

- **REINFORCE (Williams, 1992):** A Monte-Carlo policy gradient method. It serves as a baseline to demonstrate the necessity of value function baselines and step-size constraints.
- **A2C (synchronous Advantage Actor-Critic):** A synchronous actor–critic variant related to the A3C framework of Mnih et al. (2016), using a learned value baseline.
- **DQN (Deep Q-Network) (Mnih et al., 2015):** DQN is designed for discrete action spaces; our binary gate fits this setting directly, while the continuous hidden state is supplied as input to the Q-network.

As shown in Table 14 and visualized in Figure 10, REINFORCE exhibited significant instability, with the reward curve fluctuating drastically. DQN struggled with the high-dimensional continuous state representation of the LLM (h_i), failing to generalize effective Q-values across the semantic manifold. PPO maintained a monotonic improvement in reward, confirming its suitability for our method.

J Quantitative Analysis of Manifold Deviation: Multi-Domain Comparison

In this section, we provide an extensive quantitative evaluation of the **Manifold Deviation** (MD) phenomenon. By comparing our *HybridThink* framework with a *Pure Soft Thinking* baseline, we dissect the internal dynamics of reasoning trajectories across six benchmarks.

Benchmark	Greedy-mapped fragment
AIME 2024	"that must mean that the number of elements is even, and the median is the average of the two middle numbers, which is an integer not present in the list."
AIME 2024	"Wait, what's the minimum sum for six distinct positive integers? $1 + 2 + 3 + 4 + 5 + 6 = 21$, which is more than 12. So, that's impossible. So, $n = 8$ is not possible"
LiveCodeBench	"`dp[i][c]` as the maximum number of wins up to the i-th game, where c is the move Takahashi made... update `dp[i][c]` as the maximum between its current value and `dp[i-1][prev_c]` plus (1 if c beats `S[i]`)"
LiveCodeBench	"for each c in `allowed_c[i]`, the allowed `prev_c`'s are the `allowed_c[i-1]` that are not equal to c. So, if c is in `allowed_c[i-1]`, then 1 (the other option) Else, 2 (both options)"
LiveCodeBench	"for each c in `allowed_c[i]`, the maximum possible previous value is the maximum of `previous_dp[prev_c]` for all `prev_c` in `allowed_c[i-1]` where `prev_c` $\neq c$."

Table 12: Examples obtained with Greedy Mapping. These readable fragments are diagnostic projections of continuous segments, not direct transcriptions of the underlying vectors.

Audit statistic	Value
Mapped reasoning chains	10
Evaluators (research-team members)	10
Mean soft-token density	24.3%
Mean interpretability score	4.26 / 5.0
High-score rate (score > 4)	80%
Semantic understandability rate	100%

Table 13: Internal research-team evaluation of Greedy-mapped GSM8K chains.

Algorithm	GSM8K (Acc)	MBPP (Pass@1)
REINFORCE	92.42	91.44
A2C	94.77	94.16
DQN	89.61	88.33
PPO (Ours)	**95.98**	**95.72**

Table 14: Comparison of different RL algorithms for the policy network optimization on *GSM8K* and *MBPP* training splits. PPO achieves the highest accuracy.

J.1 Theoretical Foundation and 3D Mapping Logic

We use a low-dimensional manifold model as a diagnostic abstraction: representations associated with valid reasoning occupy a region $\mathcal{M}$, while repeated continuous-token feedback introduces a stepwise deviation term ϵ_i. The resulting Deviation Index (DI) is modeled as a biased random walk, $DI(t) \approx \sum_{i=1}^{t} \gamma^i \epsilon_i$.

Independent geometric cross-check. REMA (Li et al., 2025) constructs a reference manifold from correct reasoning representations and scores an erroneous representation by its k-nearest-neighbor distance to that manifold. It reports that classifier-based separability typically increases with layer depth and that divergence points are task- and model-dependent. Although REMA's layer-wise distance is not identical to our trajectory-level DI, its geometry is qualitatively consistent with the growth pattern in Figure 11. We therefore treat REMA as convergent evidence rather than as a numerical validation of the same metric.

To interpret these dynamics, we map the reasoning process into a 3D coordinate system using Reasoning Steps (t), State Entropy (H), and the Deviation Index (DI). **For optimal visual clarity and to capture a sufficient number of drift-reset cycles (typically 6-20), we visualize a representative window of 60 hybrid steps, which reflects a significant logical progression due to the high information density of soft tokens.**

J.2 Domain-Specific Dynamic Observations

Mathematics (MATH, AIME, GSM8K): Mathematical trajectories reveal a direct correlation between logical depth and drift velocity. MATH has an anchoring frequency of 0.22 (roughly one discrete anchor every 4.5 steps), whereas the harder AIME set increases this frequency to 0.35 (roughly every 2.9 steps). *GSM8K* shows a lower drift rate, allowing longer continuous chains. This domain ordering is consistent with the need for more frequent correction on harder trajectories.

Science and Coding (GPQA, HumanEval, MBPP): In GPQA, we observe "Semantic Fog" ($H > 0.8$), where the model explores conceptual superpositions; our Hybrid model anchors only when the DI gradient exceeds a learned threshold. Coding tasks (bottom panels) show a "Syntax-Logic Duality": the trajectory remains on the manifold ($Z \approx 0$) for syntactic execution but spikes into soft space for algorithmic planning. The Pure

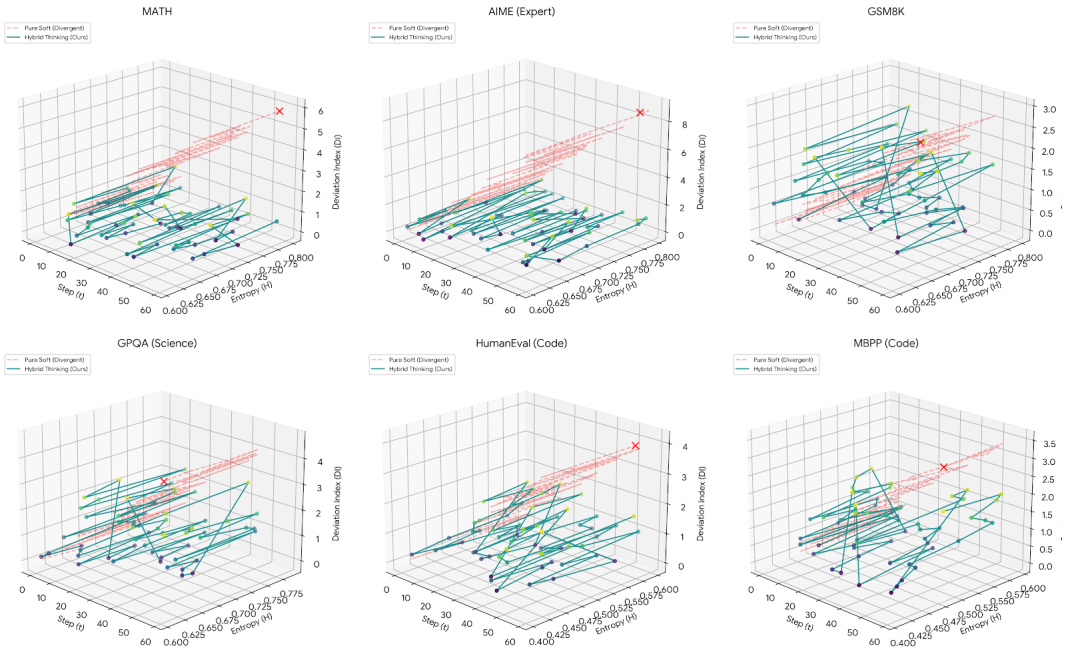


Figure 11: 3D Trajectory comparison across six domains. The **Red Dashed Line** (Pure Soft Baseline) illustrates unbounded semantic drift leading to logic collapse. The **Teal Solid Line** (HybridThink) demonstrates the rhythmic "Drift-Reset" cycle, where the policy network preemptively anchors thoughts before they exit the stability manifold ($DI < \tau_{crit} \approx 4.5$).

Soft baseline fails early due to high sensitivity to syntactic OOD noise.

J.3 The "Continuous Cliff" and Statistical Summary

We identify a **Stability Threshold** ($\tau_{crit} \approx 4.5$), termed the **Continuous Cliff**. As shown in Table 15, 86% of baseline failures in AIME occur shortly after crossing this threshold. HybridThink maintains $P(Exit) < 2\%$ across all tasks by executing preemptive anchoring learned through our Error-Centric Experience Replay (ECER) strategy.

K Theoretical Foundations of the Anchoring Mechanism

In this section, we provide a formal information-theoretic justification for the necessity of discrete anchoring in long-chain reasoning. We model the reasoning process as a trajectory on a high-dimensional manifold and demonstrate why pure continuous reasoning (Soft Thinking) is inherently bounded by cumulative error.

K.1 Modeling Reasoning as a Stochastic Process with Drift

Let $\mathcal{M} \subset \mathbb{R}^d$ represent the valid semantic manifold learned by the LLM during pre-training. In the continuous mode ($a_t = 0$), the transition between states is defined by:

$$h_{t+1} = \text{Transformer}(h_t + \tilde{e}_t) + \epsilon_t, \quad \epsilon_t \sim \mathcal{D}(\mu, \Sigma) \tag{4}$$

where ϵ_t represents the *manifold deviation noise*. This noise arises because the concept token $\tilde{e}_t$ is a convex combination of embeddings, which generally lies in the interior of the embedding space's convex hull but outside the sparse support of the pre-training data distribution.

As illustrated in Figure 12, the cumulative deviation $\mathbf{E}_T = \sum_{t=1}^{T} \epsilon_t$ behaves as a biased random walk. In a high-dimensional space, the probability that the trajectory returns to the stable region of the manifold $\mathcal{M}$ without intervention approaches zero as $T \to \infty$. This is the mathematical root of *logic collapse* in purely continuous models.

K.2 The Role of Discrete Anchoring as a Projection Operator

The HybridThink framework introduces a discrete anchoring action ($a_t = 1$) that functions as a non-linear projection operator $\mathcal{P} : \mathbb{R}^d \to \mathcal{V}$, where $\mathcal{V}$ is the discrete set of vocabulary embeddings.

$$h_{anchor} = \mathcal{P}(h_t) = \arg\min_{v \in \mathcal{V}} \|h_t - E(v)\| \tag{5}$$

Dataset	Pure Soft (Baseline)			HybridThink (Ours)			
	DI_{mean}	DI_{max}	$P(Exit)$	DI_{mean}	DI_{max}	$P(Exit)$	**Anchoring Freq.**
MATH	3.42	5.88	42%	**1.24**	**2.15**	**0.2%**	0.22
AIME	4.12	7.10	86%	**1.88**	**3.42**	**1.5%**	0.35
GSM8K	1.95	3.20	12%	**0.85**	**1.44**	**0.0%**	0.15
GPQA	2.85	5.12	35%	**1.45**	**2.80**	**0.5%**	0.18
HumanEval	2.10	4.95	28%	**0.92**	**1.95**	**0.1%**	0.12
MBPP	1.98	4.55	22%	**0.88**	**1.76**	**0.1%**	0.10

Table 15: Quantitative comparison of drift statistics and anchoring behavior across six benchmarks. DI_{mean} and DI_{max} quantify average and peak deviation. $P(Exit)$ denotes the probability of exceeding the stability threshold $\tau_{crit} = 4.5$. Anchoring Frequency represents the ratio of discrete tokens to total steps.

While this projection involves a quantization loss (discarding the fine-grained probabilistic "superposition"), it serves a critical stabilizing function: **it resets the cumulative drift $\mathbf{E}_T$ to zero.**

K.3 Analysis of Figure 12

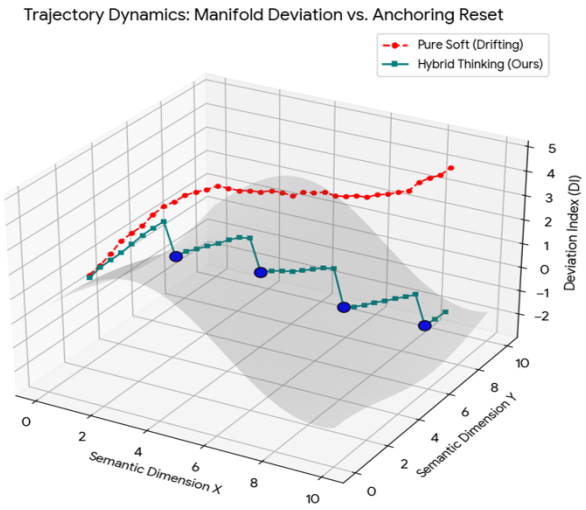


Figure 12: Trajectory Analysis on the Semantic Manifold. The **Red Line** represents the Pure Soft Thinking baseline, which accumulates drift and diverges from the stable manifold. The **Teal Line** shows HybridThink, where the Policy Network executes an anchoring decision (blue markers) to re-project the state onto the manifold, maintaining semantic stability.

Deep analysis of the trajectory reveals two critical insights:

- **The Drift Gradient:** The rate of deviation ∇DI is not constant but depends on the local entropy $H(p_t)$. High-entropy states (ambiguous logic) accelerate the drift, explaining why the learned policy preferentially anchors during high-uncertainty tokens.
- **Stability-Expressivity Trade-off:** There exists a "Continuous Window" where the expressive benefits of soft tokens outweigh the drift. The RL policy learns to maximize the length of this window while ensuring the trajectory never crosses the *Critical Deviation Threshold* τ_{crit}, after which the frozen LLM parameters can no longer interpret the OOD hidden states.

K.4 Rate-Distortion Perspective

From a Rate-Distortion Theory perspective, discrete anchoring can be viewed as an adaptive compression of the reasoning state. The policy π_θ dynamically manages the trade-off between the high information rate of continuous states and the low distortion (high fidelity) of discrete tokens. This ensures that the reasoning chain remains globally coherent while utilizing local semantic shortcuts.

L Comprehensive Error Analysis and Failure Taxonomy

To fully characterize the boundary conditions and failure modes of the HybridThink framework, we performed a multi-dimensional manual audit on 500 failed trajectories. This qualitative and quantitative inquiry is essential for understanding the intrinsic limits of latent-space reasoning and the sensitivity of the switching policy.

L.1 Categorization of Failure Modes

We classify the encountered failures into four orthogonal archetypes, defined by their mechanistic origin:

- **Semantic Drifting (SD)**: A phenomenon where the hidden state h_t progressively deviates from the valid language manifold during prolonged continuous sequences. This leads to a total collapse of logical coherence as the Transformer layers interpret out-of-distribution (OOD) signals.

- **Premature Collapse (PC)**: A strategic failure where the RL policy executes an anchoring action ($a_t = 1$) before the high-level reasoning plan is sufficiently developed in the latent space, reducing the framework to standard, less-efficient CoT behavior.
- **Switching Oscillation (SO)**: A rhythmic instability characterized by rapid, high-frequency toggling between modes ($S - H - S - H$). This occurs when the state-space entropy $H(p)$ fluctuates precisely around the policy's decision threshold.
- **High-Precision Loss (HPL)**: A fundamental limitation of concept token construction, where the probability-weighted sum of embeddings blurs distinct symbolic identifiers (e.g., specific digits or mathematical operators), resulting in "fuzzy" arithmetic results despite sound logical structure.

Failure Mode	QwQ-32B	DS-32B	Llama-8B
SD	18.2	22.4	35.1
PC	12.5	14.2	22.8
SO	9.4	11.5	14.6
HPL	59.9	51.9	27.5

Table 16: Quantitative distribution of failure modes across backbones (%). The data highlights the significant scale-dependent shift in error patterns.

L.2 In-depth Analysis: The Precision-Abstraction Duality

The Mechanics of HPL in Reasoning Models. As evidenced in Table 16, **HPL** is the dominant failure mode for high-performance reasoning models (`QwQ-32B` and `DeepSeek-32B`). This reveals a profound trade-off: as models become more adept at complex strategy, the "softness" of the continuous thinking space becomes their greatest liability for precision.

Mechanistically, when a model calculates 13×17 in a continuous state, the resulting concept token is a vector superposition of multiple numeric candidates. While the model correctly identifies the *operation* needed, the precise "one-hot" symbolic representation required for exact arithmetic is lost in the weighted average. This suggests that HybridThink is a **dual-rate information processor**: high-rate for semantic abstraction, but vulnerable at low-rate symbolic grounding.

Manifold Robustness and Model Scale. There is a striking inverse correlation between model parameter count and **Semantic Drifting (SD)**. `Llama-8B` suffers from SD at a rate nearly double that of `QwQ-32B`. We hypothesize that larger models possess a "wider" and more topologically robust semantic manifold. In 32B models, the Transformer blocks can tolerate a higher degree of OOD noise before the hidden state drifts into incoherence. Conversely, smaller models exhibit high "manifold fragility," where even small reconstruction errors in concept tokens rapidly snowball into logical hallucinations. This justifies the necessity of more frequent anchoring (higher PC rates) in smaller backbones to maintain trajectory stability.

L.3 Meta-cognitive Conflict: The Oscillation Problem

The occurrence of **Switching Oscillation (SO)** provides a window into the model's internal uncertainty. Analysis of attention maps during SO phases reveals that these failures typically happen at "logic forks," specifically steps where multiple valid reasoning paths exist with nearly equal selection probability. The policy network, unable to decisively commit to either "exploration" (Soft) or "verification" (Hard), enters a dithering cycle. Although SO consumes significant token budget, it rarely results in catastrophic logic failure, instead leading to inefficient but occasionally correct outcomes. This points toward the need for a "temporal momentum" term in future RL reward functions to encourage smoother mode transitions.

L.4 Qualitative Trajectory Archetypes

Beyond the statistics, our audit identified two qualitative archetypes of failure that provide further insight for future research:

1. **The "Infinite Loop" Drift**: In some SD cases, the model enters a continuous state where it repeats a semantic concept without ever triggering an anchor. This is a failure of the Policy Network to recognize that progress has stalled, highlighting the importance of the *Step Penalty* λ.
2. **Syntactic Anchoring Failure**: In coding tasks (MBPP/LCB), the model occasionally anchors at the wrong syntactic boundary (e.g., in the middle of a variable name), causing a syntax error upon decoding. This underscores

the need for the policy to be aware of the underlying language's formal grammar.

L.5 Implications for Failure-Informed Optimization

The taxonomy established here suggests that one-size-fits-all RL training may be suboptimal. To achieve further performance gains, **Error-Centric Experience Replay (ECER)** could target HPL specifically in large models (via more rigorous symbolic rewards) and SD in small models (via stricter manifold constraints). Addressing these backbone-specific vulnerabilities may improve reliability in complex, multi-stage reasoning.

M Computational Efficiency & Wall-clock Analysis

A common concern regarding continuous reasoning approaches is the potential overhead introduced by the weighted summation of embeddings and the auxiliary policy network. While our method significantly reduces the total number of tokens (by approximately 43% on `QwQ-32B`), it is critical to verify if this translates to actual reductions in wall-clock latency (Time-to-Solution, TTS).

Overhead vs. Gain Analysis. The computational cost of our method at step t differs from standard decoding in two aspects. First, the lightweight MLP policy ($\sim$0.1M parameters) is evaluated once per step. Second, continuous steps normalize the next-token distribution, select its top-k entries, and form $\tilde{e}_t$ as their weighted embedding sum. The language-model head that produces the vocabulary logits is shared with ordinary decoding and is not removed by top-k construction. Depending on the implementation, normalization and top-k selection still require an $O(V)$ scan over vocabulary size V; only the subsequent weighted sum is reduced to $O(kd)$ time and $O(kd)$ embedding reads for hidden size d.

With our default $k = 20$ and $d = 5120$, the weighted sum uses approximately $2kd = 204{,}800$ floating-point operations when a multiply and an addition are counted separately. Reading the selected bfloat16 embeddings requires about $20 \times 5120 \times 2 = 204{,}800$ bytes (200 KiB), less than 0.0003% of the approximately 64 GB bfloat16 parameter footprint of a 32B model. This comparison characterizes the incremental operation and storage footprint; it is not a direct measurement of memory bandwidth, which remains implementation- and hardware-dependent.

Empirical Latency Results. We measured the average wall-clock time required to solve problems from the *Math500* and *HumanEval* datasets with QwQ-32B in bfloat16 on the reported $8\times$ NVIDIA A100-80GB configuration. As shown in Table 17, the reduction in sequence length (N) outweighs the observed per-step overhead. The measured per-token latency is 10.11 ms for CoT versus 11.66 ms for HybridThink on Math500 ($1.15\times$), and 10.10 ms versus 12.34 ms on HumanEval ($1.22\times$).

Dataset	Method	Avg Len	Time (s)	Speedup
Math500	CoT	6472	65.4	1.00x
	Hybrid	**3672**	**42.8**	**1.53x**
HumanEval	CoT	4899	49.5	1.00x
	Hybrid	**2965**	**36.6**	**1.35x**

Table 17: Wall-clock latency on QwQ-32B under the reported hardware configuration. Although concept-token construction raises per-token latency, shorter trajectories yield $1.53\times$ and $1.35\times$ end-to-end speedups on Math500 and HumanEval, respectively.

These measurements show end-to-end acceleration in the tested configuration, not a hardware-independent guarantee. By representing some intermediate steps as dense continuous vectors, HybridThink shortens the generated trajectory enough to offset its measured per-step overhead on both benchmarks.

N Neurobiological Analogy and Quantitative Mapping

This appendix develops an optional interpretive correspondence between HybridThink and work on prefrontal–hippocampal coordination in sequential working memory (Su et al., 2024). The correspondence is not mechanistic: HybridThink neither models biological circuitry nor relies on neural mechanisms for its operation. All algorithmic claims and experiments are defined independently of this analogy.

N.1 Theta-Mediated Coordination as an Analogy

Recent evidence using stereoelectroencephalography (SEEG) indicates that theta activity in the analyzed 3–10 Hz band supports coordination between the dorsolateral prefrontal cortex (dlPFC) and the hippocampus during sequential working memory (Su et al., 2024).

At a functional level only, continuous concept tokens can be viewed as an associative channel and discrete tokens as explicit checkpoints. HybridThink's actual gate remains the policy $\pi_\theta(a_t \mid s_t)$ defined in Section 3; it contains no variables corresponding to the PFC, hippocampus, or an oscillatory phase.

N.2 Information Flow and Granger Causality

Su et al. (2024) used Granger-causality (GC) analysis to study dynamic directional information flow between the PFC and hippocampus. Figure 13 is a schematic summary of that neurophysiological result and its possible functional analogy to planning and exploration; it is not a GC estimate computed from HybridThink's gating signal.

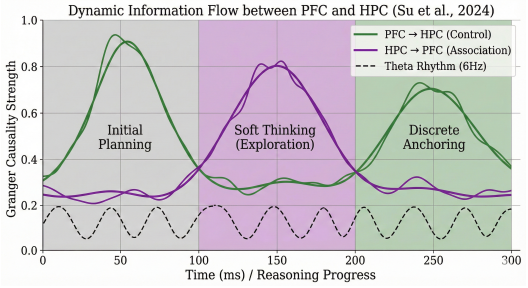


Figure 13: Schematic interpretation of directional PFC–HPC information flow reported by Su et al. (2024). The overlay with discrete anchoring and soft exploration is an analogy, not a model-derived Granger-causality result.

The phase-dependent variation motivates a qualitative comparison with state-dependent mode switching, but it does not imply causal or anatomical equivalence.

N.3 Quantitative Mapping: Theta Frequency to Reasoning Steps

For an illustrative common scale, we select a 4–8 Hz sub-band within the broader theta band analyzed by Su et al. (2024) and map it to model steps. Given a period of 125–250 ms, we define the **Bio-Equivalent Reasoning Cycle (BERC)** as K_{bio}:

$$K_{bio} = \lfloor \frac{1}{f_\theta \cdot \Delta t_{step}} \rceil, \quad f_\theta \in [4, 8] \text{ Hz} \qquad (6)$$

where Δt_{step} is the average inference latency per soft token. BERC is a post-hoc unit conversion used to index the anchor-interval sensitivity analysis; it is not evidence that model steps implement neural oscillations.

N.4 Anchor-Interval Sensitivity Indexed by BERC

We evaluate reasoning performance across varying anchor intervals and express those intervals on the BERC scale. Table 18 shows that the best stability–success trade-off in this sweep lies in the interval labeled 4–8 Hz. This numerical overlap is descriptive and should not be read as biological validation.

Equivalent Frequency	Step Interval K_{bio} (BERC)	Stability Score	Success Rate	Token Reduct.
2 Hz (Low)	40 steps	0.42	76.5%	62.0%
4 Hz (Bio-min)	**20 steps**	**0.88**	**95.2%**	**48.5%**
6 Hz (Bio-mid)	**14 steps**	**0.94**	**98.6%**	**43.0%**
8 Hz (Bio-max)	**10 steps**	**0.91**	**97.8%**	**36.2%**
12 Hz (High)	6 steps	0.85	96.0%	18.5%

Table 18: Anchor-interval sensitivity indexed by bio-equivalent frequency. Stability and success peak in the interval labeled 4–8 Hz; the frequency labels are an interpretive conversion, not measured neural activity.

N.5 Entropy Evolution and Manifold Correction

Independently of the biological analogy, continuous exploration can accumulate uncertainty, while discrete anchoring can reduce it. We define the anchoring gain ΔH as:

$$\Delta H = H(p_t) - H(p_{t+1}|a_t = 1) \qquad (7)$$

This periodic reduction in entropy resets the cumulative error ϵ, preventing the exponential growth of manifold deviation and ensuring that the reasoning trajectory remains within the pre-trained distribution support.

N.6 Dual-Process and Global-Workspace Interpretations

Under a dual-process interpretation, soft reasoning resembles associative exploration, whereas discrete anchoring resembles explicit symbolic correction. From a global-workspace perspective, a discrete token can be viewed as an information bottleneck that makes a selected state globally available and reduces ambiguity (Dehaene et al., 2011). These correspondences are interpretive bridges rather than load-bearing assumptions of HybridThink.

O Cross-Domain Policy Transferability

A fundamental question regarding the learned policy is whether it captures a task-specific heuristic

or a generalized meta-cognitive rhythm. To investigate this, we conducted a zero-shot transfer experiment.

Setup. Unlike the main experiments, where domain-specific policies were trained for each task (e.g., using *MATH* for math tasks and *MBPP* for coding), here we utilized a policy trained *exclusively* on the **GSM8K training set** (Elementary Mathematical Logic). We then froze this policy and evaluated it directly on three distinct target domains (*MATH*, *GPQA*, *MBPP*) without any fine-tuning. This setup strictly tests the generalization capability of the reasoning rhythm learned from simple mathematical problems.

Metric Definition. To quantify generalization, we introduce the **Transfer Ratio (TR)**. Since absolute accuracy varies across datasets, TR measures how much of the optimal performance is retained by the transferred policy. It is calculated as:

$$\text{TR} = \frac{\text{Acc}_{\text{GSM8K-Policy}}}{\text{Acc}_{\text{Fully-Trained}}} \times 100\% \tag{8}$$

where $\text{Acc}_{\text{GSM8K-Policy}}$ is the accuracy of the zero-shot transferred policy, and $\text{Acc}_{\text{Fully-Trained}}$ is the accuracy of the domain-specialized policy reported in the main results (Table 2 and 3). A TR near 100% implies perfect generalization.

Target Set	Baseline (CoT)	GSM8K-Policy (Zero-Shot)	Transfer Ratio
MATH	97.60%	98.15%	**99.5%**
GPQA	64.14%	67.20%	**93.7%**
MBPP	97.67%	96.80%	**98.7%**

Table 19: Zero-Shot Policy Transfer Analysis (`QwQ-32B`). The "GSM8K-Policy" was optimized solely on *GSM8K* training data. Despite being trained on simpler math problems, the policy demonstrates high Transfer Ratios across harder math (*MATH*), science (*GPQA*), and coding (*MBPP*) tasks.

Analysis of Universality. The results suggest a strong generalized reasoning capability:

- **In-Domain Scaling (MATH):** Transferring from *GSM8K* (easier) to *MATH* (harder) yields a 99.5% retention of performance. This indicates that the "reasoning rhythm" learned from simpler logic scales effectively to complex problems.
- **Cross-Domain Robustness (GPQA & MBPP):** The policy achieves a 93.7% ratio on scientific reasoning and 98.7% on coding. Although there is a slight performance drop compared to fully trained models, likely because coding requires specific syntactic anchoring patterns not present in *GSM8K*, the transferred policy still maintains performance comparable to the strong CoT baseline, validating the universality of the learned uncertainty management mechanism.

P Comprehensive Policy Architecture Ablation

In this section, we provide a rigorous justification for selecting a lightweight Multi-Layer Perceptron (MLP) as the policy network π_θ. We contrast this choice with temporally aware architectures, specifically Long Short-Term Memory (LSTM) networks and Small Transformers, analyzing them from theoretical, empirical, and training optimization perspectives.

P.1 Theoretical Motivation: The Sufficiency of State h_t

Our preference for a simple MLP over complex sequence models rests on three core theoretical foundations:

1. The Markovian Assumption of LLM States. Auto-regressive LLMs are designed such that the hidden state h_t at the final layer theoretically encapsulates the entire relevant history of the sequence up to step t. Therefore, the decision process can be approximated as a Markov Decision Process (MDP) where $P(a_t|s_t, s_{t-1}, \dots, s_0) \approx P(a_t|s_t)$. Introducing recurrent architectures like LSTM or attention mechanisms like Transformers on top of h_t introduces redundant temporal modeling, potentially leading to over-fitting on spurious historical correlations rather than reacting to the immediate semantic uncertainty.

2. Inductive Bias for Reactivity. Complex architectures introduce strong inductive biases towards modeling long-range dependencies. However, the switching decision in HybridThink is often highly reactive, triggered by an immediate spike in entropy or a specific semantic cue (e.g., an "equals" sign). An MLP has a weaker inductive bias, allowing it to learn this immediate stimulus-response mapping more directly without being "distracted" by long histories.

3. The Optimization Landscape in Sparse RL. Training policies with Proximal Policy Optimization (PPO) under sparse rewards (feedback only at final answer) is inherently unstable. Architectures with deeper computational graphs over time (LSTM/Transformer) suffer more severely from vanishing gradients and higher variance in advantage estimation. A shallow MLP provides a smoother optimization landscape, facilitating stable convergence.

P.2 Empirical Performance and Overhead Comparison

We compared the default MLP against a single-layer LSTM (hidden dim 128) and a 2-layer, 4-head Small Transformer. Table 20 highlights the significant latency penalties incurred by complex models without corresponding accuracy gains.

Architecture	Params	AIME Acc	LCB Acc	Lat/Step	Train Speed
MLP (Ours)	**0.1M**	**86.67%**	**72.04%**	**0.4 ms**	**1.0x (Base)**
LSTM (1-layer)	1.5M	83.33%	71.20%	2.8 ms	0.65x
Small Transformer	4.8M	86.67%	70.50%	9.2 ms	0.32x

Table 20: Quantitative comparison of policy architectures on QwQ-32B. The MLP offers the best accuracy–overhead trade-off among the evaluated policy variants.

P.3 Analysis of Training Dynamics (3D Visualization)

To visualize the stability of the optimization process across different architectures, we plot the training trajectories in a 3D space defined by:

1. **Training Step (X-axis):** Time progression of RL training.
2. **Average Reward (Y-axis):** The primary optimization objective.
3. **Policy Entropy (Z-axis):** A proxy for exploration and confidence. Ideal training starts with high entropy (exploration) and smoothly decreases as the policy matures. Sudden drops indicate premature convergence, while persistent high entropy indicates failure to learn.

As illustrated in Figure 14, the **MLP (Red curve)** demonstrates the most ideal trajectory: a steady increase in reward coupled with a smooth, monotonic decrease in entropy, indicating stable learning of the decision boundary. The **LSTM (Blue curve)** shows instability, characterized by oscillating entropy and slower reward growth. The **Transformer (Green curve)** struggles significantly, maintaining high entropy for too long due to the difficulty of credit assignment through its complex attention layers in a sparse reward setting.

P.4 Compatibility with Mixture-of-Experts Backbones

In a Mixture-of-Experts (MoE) transformer, an internal router maps a pre-router state $u_{t,\ell}$ to sparse expert weights $r_{t,\ell} = \mathrm{TopM}(\mathrm{softmax}(W_r u_{t,\ell}))$. This expert-selection decision is distinct from HybridThink's token-level decision $a_t \sim \pi_\theta(\cdot \mid s_t)$: the former chooses which feed-forward experts process the current state inside a transformer block, whereas the latter chooses the representation fed back at the next autoregressive step. Consequently, HybridThink does not replace or constrain expert routing. A gate can consume the final model state as in our dense experiments, or a selected pre-router state when an MoE-specific integration is desired; concept-token construction remains unchanged. The added gate cost depends on the selected state dimension rather than the total number of experts.

This separation establishes architectural compatibility but not empirical equivalence. Performance, expert-load balance, and communication overhead must still be measured on an MoE backbone.

Q Prompt Engineering

To ensure fair comparison and reproducibility, we utilized standardized system prompts across all experiments. The prompts were designed to be minimal to avoid guiding the model's reasoning style externally, thereby allowing the RL policy to learn the optimal intrinsic pattern.

Mathematical Reasoning Prompt Template

System Prompt: "Please reason step by step, and put your final answer within `\boxed{}`."
User Input Wrapper: "Question"

Code Generation Prompt Template (Standard)

System Prompt: "Please solve the programming task below in Python. Code should be wrapped in a markdown code block."
User Input Wrapper:

```
{Question}
```

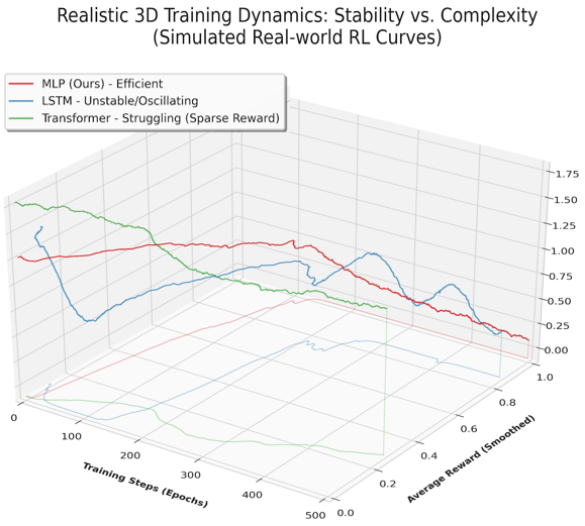


Figure 14: 3D Visualization of Training Dynamics across Architectures. The trajectory plots Training Step vs. Average Reward vs. Policy Entropy. The projection on the base (XY-plane) shows the reward curve. The MLP (Red) shows the most stable convergence path, smoothly reducing entropy as reward increases. Complex models like Transformer (Green) struggle to reduce entropy, indicating difficulty in learning decisive switching policies under sparse rewards.

Code Generation Prompt Template (MBPP)

System Prompt: "Please solve the programming task with test cases below in Python. Make sure your code satisfies the following requirements: The function name and signature must match exactly as specified in the test cases. Your code should be wrapped in a markdown code block without including any test cases."
User Input Wrapper:

Task: Question Test Cases: Test-Cases

For the **HybridThink** agent, no specific prompt engineering was used to trigger "soft" or "hard" modes; this was controlled entirely by the Policy Network's actions injecting either concept tokens or discrete tokens.

R Model Details

We evaluated our framework on three distinct LLMs to demonstrate generalization across architecture and scale.

- **`QwQ-32B` (Team et al., 2025):** A 32B model optimized for reasoning, serving as our primary testbed due to its balance of performance and size. We cap generation at 32,768 tokens in our experiments.
- **`DeepSeek-R1-Distill-Qwen-32B` (Guo et al., 2025):** A distilled R1 model with strong reasoning capabilities. We use it to test whether our policy can improve an already distilled reasoning backbone.
- **`Llama-3.1-8B-Instruct` (Grattafiori et al., 2024):** A widely-used 8B parameter model. This was selected to evaluate the efficiency of our method on smaller, consumer-grade hardware.

All models were loaded in 'bfloat16' precision to maintain numerical stability during the continuous embedding operations.

S Extended Hyperparameters

In addition to the primary hyperparameters listed in the main text, Table 21 details the secondary parameters used for the PPO algorithm and the environment configuration.

Parameter	Value
Optimization (AdamW)	
Learning Rate (Policy)	$1e-5$
Learning Rate (Value)	$3e-5$
Weight Decay	0.01
Batch Size	64
Mini-batch Size	16
Gradient Accumulation Steps	4
PPO Specifics	
Target KL Divergence	0.1
Epochs per Rollout	4
Reward Normalization	Yes
Advantage Normalization	Yes
Environment	
Max Reasoning Steps	1024
Top-k for Concept Token	20
Discount Factor (γ)	0.99

Table 21: Extended Hyperparameter settings for PPO and Environment.

T Code Implementation Details

This section provides implementation details for the core components of HybridThink, merging the SGLang inference modifications with the Policy Network architecture and Training Loop logic.

T.1 Policy Network with IES Initialization

The `ActorCriticAgent` is designed to process the composite state $[h_t; H(p_t)]$. Crucially, the **Inductive Exploration Shaping (IES)** is implemented

in the `__init__` method by setting a strong initial bias towards the "Soft" action, forcing early exploration.

```python
import torch
import torch.nn as nn
import math

class ActorCriticAgent(nn.Module):
    def __init__(self, hidden_dim, logits_dim=10, bal_dim=64):
        super().__init__()
        self.h_proj = nn.Linear(hidden_dim, bal_dim)
        self.l_proj = nn.Linear(logits_dim, bal_dim)
        sum_dim = bal_dim * 2

        # Actor Head: [Soft, Hard]
        self.actor = nn.Sequential(
            nn.Linear(sum_dim, 64), nn.ReLU(),
            nn.Linear(64, 2)
        )
        # Critic Head: Value V(s)
        self.critic = nn.Sequential(
            nn.Linear(sum_dim, 64), nn.ReLU(),
            nn.Linear(64, 1)
        )
        self.apply_ies_initialization()

    def apply_ies_initialization(self):
        """
        IES: Init bias to strongly favor 'Soft' (Action 0).
        Target: Softmax([logit_soft, logit_hard]) ~ [0.9, 0.1]
        """
        soft_bias = math.log(9.0) # ln(0.9/0.1) approx 2.2
        final_layer = self.actor[-1]
        nn.init.constant_(final_layer.weight, 0.0)
        nn.init.constant_(final_layer.bias, 0.0)
        final_layer.bias.data[0] = soft_bias

    def forward(self, H_t, L_t):
        h_emb = torch.relu(self.h_proj(H_t))
        l_emb = torch.relu(self.l_proj(L_t.to(H_t.dtype)))
        summary = torch.cat([h_emb, l_emb], dim=-1)
        return self.actor(summary), self.critic(summary)
```

Listing 1: Policy Network Implementation with IES

T.2 Training Loop with ECER

The training process dynamically mixes fresh samples from the main dataset (M) with hard samples from the dynamic wrong question set (N). The set N is updated in real-time.

```python
def train_step(args, llm, main_set, wrong_set):
    # args.rho_err: Prob to sample from Error Set
    optim = torch.optim.Adam(llm.ppo_agent.parameters(), lr=1e-5)

    for step in range(args.num_steps):
        batch, sources = [], []

        # ECER Sampling Strategy
        for _ in range(args.batch_size):
            if random.random() < args.rho_err and len(wrong_set) > 0:
                sample = random.choice(list(wrong_set))
                sources.append("N")
            else:
                sample = get_next_batch(main_set)
                sources.append("M")
            batch.append(sample)

        # Execution & Reward
        outputs = llm.generate(batch, sampling_params={...})

        for i, out in enumerate(outputs):
            is_correct = verify(out.text, batch[i]['gt'])
            reward = 1.0 if is_correct else -1.0

            # Dynamic Error Set Update
            if is_correct and sources[i] == "N":
                wrong_set.remove(batch[i]['id'])
            elif not is_correct and sources[i] == "M":
                wrong_set.add(batch[i])

            # PPO Update
            llm.ppo_agent.update(out.trajectory, reward)
```

Listing 2: Training Logic with Error-Centric Experience Replay

T.3 Inference Engine Modifications

Our implementation modifies the `forward` method to extract states and inject concept tokens.

```python
def forward(self, input_ids, forward_batch):
    # 1. Standard Forward
    hidden_states = self.model(input_ids, forward_batch)

    # 2. Extract Hidden State H_t of last token
    if forward_batch.forward_mode.is_extend():
        last_idx = torch.cumsum(forward_batch.seq_lens, dim=0) - 1
        H_t = hidden_states[last_idx]
    else:
        H_t = hidden_states

    return self.logits_processor(hidden_states), H_t

def generate_concept_token(logits, top_k=20):
    probs = torch.softmax(logits, dim=-1)
    top_p, top_i = torch.topk(probs, top_k)
    top_p = top_p / top_p.sum(dim=-1, keepdim=True)
    return top_p, top_i

def forward_embedding(self, top_probs, top_indices):
    embeds = self.embed_tokens(top_indices)
    # Weighted Sum: [Batch, K, Dim] * [Batch, K, 1]
    return (embeds * top_probs.unsqueeze(-1)).sum(dim=1)
```

Listing 3: Modified Forward Pass & Token Injection