S-BGM: Stable Self-Evolving LLMs via Cognitive Bipartite Graph Modeling

Songyang Liu , Chaozhuo Li * , Rui Zhong , Zejian Chen , Chenxu Wang , Litian Zhang , Qiwei Ye , Zheng Liu , Yiming Hei

Posted Date: 22 May 2026

doi: 10.20944/preprints202605.1514.v1

Keywords: Large Language Models; self-evolution; curriculum learning

 doi:10.20944/preprints202605.1514.v1

Article

S-BGM: Stable Self-Evolving LLMs via Cognitive Bipartite Graph Modeling

Songyang Liu [1], Chaozhuo Li [1,*], Rui Zhong [1], Zejian Chen [1], Chenxu Wang [1], Litian Zhang [1], Qiwei Ye [2], Zheng Liu [2] and Yiming Hei [3]

[1] Beijing University of Posts and Telecommunications, China
[2] Beijing Academy of Artificial Intelligence, China
[3] China Academy of Information and Communications Technology, China
* Correspondence: lichaozhuo@bupt.edu.cn

Abstract

Self-evolving LLMs that iteratively train a problem-generating Questioner and a problem-solving Solver present a promising path toward reducing human supervision. Existing approaches optimize each role independently with short-term, separate rewards, neglecting the co-evolutionary dynamics of the Questioner–Solver system as a whole. This leads to model collapse, where the Solver's performance plateaus or degrades after only a few iterations. To address this, we propose S-BGM, a framework for stable self-evolving LLMs based on bipartite graph modeling. S-BGM models Questioner–Solver interactions as a Cognitive Bipartite Graph, where the two partitions discretize the question space into semantic clusters and the response space into uncertainty intervals. The long-term interactions are encoded into the graph topology, facilitating the tracking of holistic system status. A novel structural entropy is further integrated into both roles' training objectives to measure system-level stability, preventing excessive concentration on the few global interaction structures while preserving local reward optimization. To keep this stability signal dynamically aligned with ongoing co-evolution, the Cognitive Bipartite Graph is updated via a sliding window mechanism that reinforces recent interactions and discards outdated ones. Extensive experiments on ten datasets demonstrate the superiority of S-BGM.

Keywords: Large Language Models; self-evolution; curriculum learning

1. Introduction

To alleviate the reliance on human supervision, self-evolving frameworks have emerged as a promising paradigm in which LLMs autonomously generate training data and derive reward signals to guide their own optimization [1–4]. A typical self-evolving framework generally comprises two roles: a Questioner responsible for generating training questions and a Solver tasked with answering them [5]. These two components are trained in an alternating fashion to facilitate co-evolution, as illustrated in Figure 1(a). Existing self-evolving models are often subject to a critical limitation: as training iterations progress, the Solver's performance initially improves rapidly but soon plateaus and may even degrade, a phenomenon commonly referred to as **model collapse** [6,7]. Prior work has sought to address this issue from several perspectives, including introducing external data [8,9], designing reliable verification signals [10,11], and improving the diversity of training data [12,13].

However, existing approaches for mitigating model collapse generally concentrate on a single role (the Questioner or the Solver), overlooking the evolving dynamics of the holistic Questioner-Solver system. This narrow focus addresses only the symptoms rather than the underlying cause, leading to the challenges of the **short-term** and **separate** training reward. In the general self-evolving paradigm, each role is optimized only according to the immediate reward in the current iteration, without considering how this update will affect the long-term evolution of the coupled system. For

 doi:10.20944/preprints202605.1514.v1

instance, upon discovering that the current Solver is uncertain about calculus problems, the Questioner may generate more difficult calculus questions in order to maximize its short-term reward. This behavior may far exceed the Solver's capability upper bound, leading to model collapse. In addition, the Questioner and Solver are rewarded as separate components instead of a whole. Optimized against fixed counterparts, the Questioner exploits momentary uncertainty instead of exploring broadly, and the Solver fits a static distribution, producing rote answers without diverse reasoning [14]. Consequently, as shown in Figure 1(b) and (c), the edges in the initial state denote broad exploration across mathematical domains, which collapses into concentrated edges in model collapse.

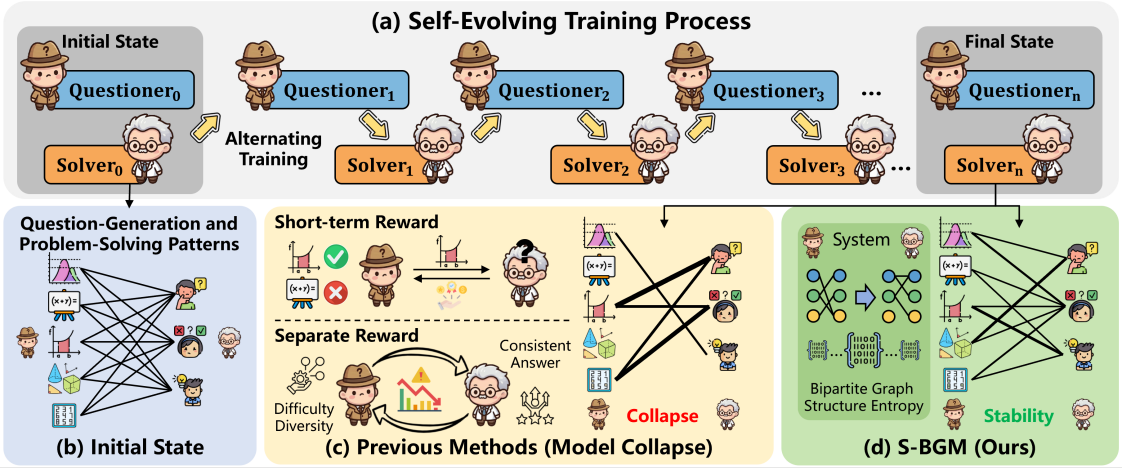

Figure 1. The self-evolving framework and the comparison between prior works and ours.

In this paper, we argue that self-evolving training should incorporate dynamic signals to depict stability at the system level, rather than focusing solely on role-level rewards. A straightforward strategy is to store and utilize all historical checkpoints in a general fashion [15]; however, this leads to prohibitive resource costs and yields disconnected rewards. Differently, we propose modeling the dynamics of the self-evolving system through a bipartite graph, where the two disjoint vertex sets represent the semantic distribution of questions generated by the Questioner and the problem-solving ability of the Solver. Edges across the two node sets depict how the Questioner and the Solver interact globally, allowing the coupled behavior of the two roles to be analyzed as **a unified objective**. When a question is generated and a response is produced for that question, the corresponding nodes are connected based on this interaction, with the edge weight recording the interaction frequency. During self-evolving training, new interactions strengthen or create edges, while less frequent interactions gradually weaken or remove edges, allowing the graph topology to track the **long-term evolution** of the coupled system. From this perspective, model collapse can be identified as a topological concentration phenomenon in this bipartite interaction graph. As shown in Figure 1(b,c), the initially broad interaction topology gradually collapses into a small subset of states, reflecting repetitive question generation and increasingly narrow Solver behavior. A stable self-evolving system in Figure 1(d) instead maintains a distributed interaction topology, where diverse question patterns continue to induce varied Solver behaviors throughout the co-evolution process.

Building on this insight, we propose **S-BGM**, a framework for **S**table self-evolving LLMs based on **B**ipartite **G**raph **M**odeling. First, we formalize self-evolving as a coupled Questioner-Solver system from a mathematical bipartite graph view. After that, a Cognitive Bipartite Graph is constructed by discretizing the Questioner's question space into semantic clusters and the Solver's response space into uncertainty intervals, such that each interaction induces an edge across the two partitions. To quantify system stability from a holistic perspective, we propose the structural entropy to measure the dispersion of interaction weights over the bipartite graph topology, thereby reflecting whether the Questioner-Solver system remains broadly coupled or collapses into a concentrated interaction pattern. Such entropy is added into the training signal of the Questioner and Solver to encourage

 doi:10.20944/preprints202605.1514.v1

each role to optimize its local reward while avoiding excessive concentration of the global interaction structure. The Cognitive Bipartite Graph is updated by a sliding window that ensures newly observed interactions strengthen the corresponding edges, while outdated interactions are removed. Thus, the graph continuously tracks the evolution of the coupled system for stabilizing self-evolving training. Our main contributions are summarized as follows:

- As far as we know, we are the first to propose model self-evolving systems as a Cognitive Bipartite Graph to facilitate long-term interactions in a holistic system view.
- We propose S-BGM, a novel self-evolving paradigm integrated with the proposed structural entropy loss and dynamic graph topology update scheme, providing system-level stability signals instead of traditional role-level rewards.
- Comprehensive evaluations across mathematical and general reasoning benchmarks show that S-BGM reliably enhances end-task performance, maintains curriculum learning dynamics, and enables more stable long-horizon self-evolution than representative baselines.

2. Preliminary Knowledge and Analysis

2.1. Cognitive Bipartite Graph Modeling for Self-evolving System

A typical self-evolving system is composed of two roles: a questioner model $\mathcal{M}_Q$ and a solver model $\mathcal{M}_S$. To characterize the interaction dynamics between two roles, the system is modeled as a **Cognitive Bipartite Graph** $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, where the vertex set consists of two disjoint partitions:

- $\mathcal{V}_Q = \{v_1, \ldots, v_m\}$ (*Question-Side Semantic States of* $\mathcal{M}_Q$): Each node $v_i \in \mathcal{V}_Q$ represents a state in the question-generation space of $\mathcal{M}_Q$, characterizing the semantic distribution of questions produced by the Questioner.
- $\mathcal{V}_S = \{u_1, \ldots, u_n\}$ (*Solver-Side Problem-Solving States of* $\mathcal{M}_S$): Each node $u_j \in \mathcal{V}_S$ represents a behavioral state in the response space of $\mathcal{M}_S$, characterizing how the Solver responds to generated questions.

Each edge $(v_i, u_j) \in \mathcal{E}$ represents an observed interaction between the Questioner and the Solver, where a question-side state v_i is associated with a response-side state u_j. The edge weight w_{ij} quantifies how frequently interactions between the question-side state v_i and the response-side state u_j are observed during self-evolving training. Under this abstraction, the evolving state of the system is fully represented by the bi-adjacency matrix $\mathbf{B} \in \mathbb{R}^{m \times n}$ throughout the training process, where $\mathbf{B}_{ij} = w_{ij}$ for observed interactions $(v_i, u_j) \in \mathcal{E}$ and $\mathbf{B}_{ij} = 0$ otherwise.

A straightforward strategy is to treat each question or answer as an independent node. However, this causes the graph to grow with the number of samples, resulting in a sparse and unreliable structure. To address this, microscopic interactions are discretized by projecting them onto a finite set of macroscopic cognitive states. On the $\mathcal{M}_Q$ side, **semantic-based discretization** is applied: the high-dimensional embedding space of generated questions is partitioned into m representative semantic regions, and each question q is mapped to a vertex $v_i \in \mathcal{V}_Q$ by nearest-centroid proximity. On the $\mathcal{M}_S$ side, **uncertainty-based discretization** is applied: the consensus ratio across multiple Solver samples serves as a proxy for response uncertainty and is discretized into n intervals, each corresponding to a vertex $u_j \in \mathcal{V}_S$ that represents a distinct confidence level. Edge weights $\mathcal{W}$ reflect the frequency of recent interactions between these macroscopic states, so the bi-adjacency matrix $\mathbf{B}$ compactly represents the current system interaction status.

2.2. Understanding Model Collapse via Cognitive Bipartite Graph

Our research begins with an examination of the correlation between model collapse and the evolving topology of the cognitive bipartite graph. This study conducts five iterations of self-evolution within the R-Zero framework [5] on Qwen3-4B-Base [16]. Then, our study validate the Solver's performance on the popular mathematical datasets used in Section 4.1. As illustrated in the left panel of Figure 2, R-Zero exhibits a typical model collapse, where the performance initially improves rapidly but soon plateaus and finally decreases. To understand model collapse from the perspective of

 doi:10.20944/preprints202605.1514.v1

the interaction graph $\mathcal{G}$, we analyze the normalized weighted-degree ratios of nodes at iterations 0 and 5. In the middle and right panels of Figure 2, the x-axis sorts nodes by descending normalized weighted-degree ratio, while the y-axis shows each node's proportion of the total degree mass. After five iterations, both the Solver side $\mathcal{V}_S$ and the Questioner side $\mathcal{V}_Q$ show clear topological concentration. On the Solver side, the distribution becomes more skewed, indicating that interactions increasingly focus on a small number of response states. This suggests that the Solver becomes increasingly overconfident during training. On the Questioner side, the curve also becomes more concentrated: only a few nodes carry most of the weight, while most nodes are rarely visited. This indicates that the Questioner gradually focuses on a limited set of question types, reducing the diversity of the generated curriculum. These results show that model collapse can be measured as topological concentration in the Cognitive Bipartite Graph, motivating a topology-aware global signal to regulate self-evolving training. Please refer to Appendix E for the detailed theoretical proof.

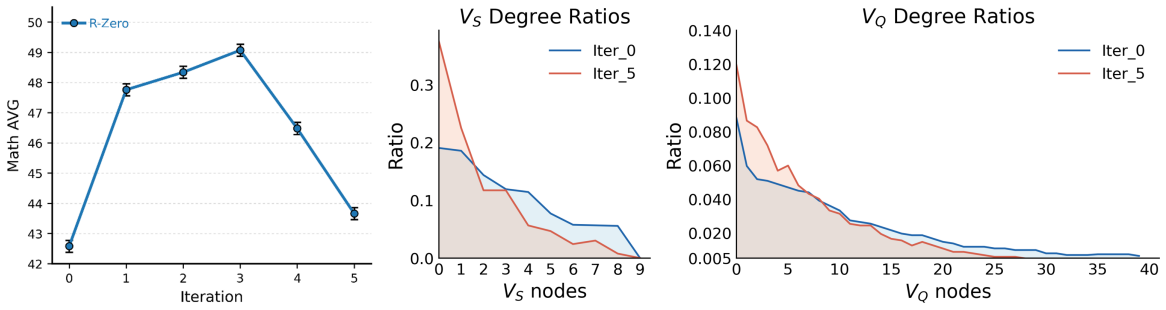

Figure 2. Empirical analysis of model collapse in R-Zero, a representative self-evolving method.

3. Methodology

Figure 3 presents the overview of our proposal, including the construction of the Cognitive Bipartite Graph, the training of the Questioner and Solver under the constraint of structural entropy.

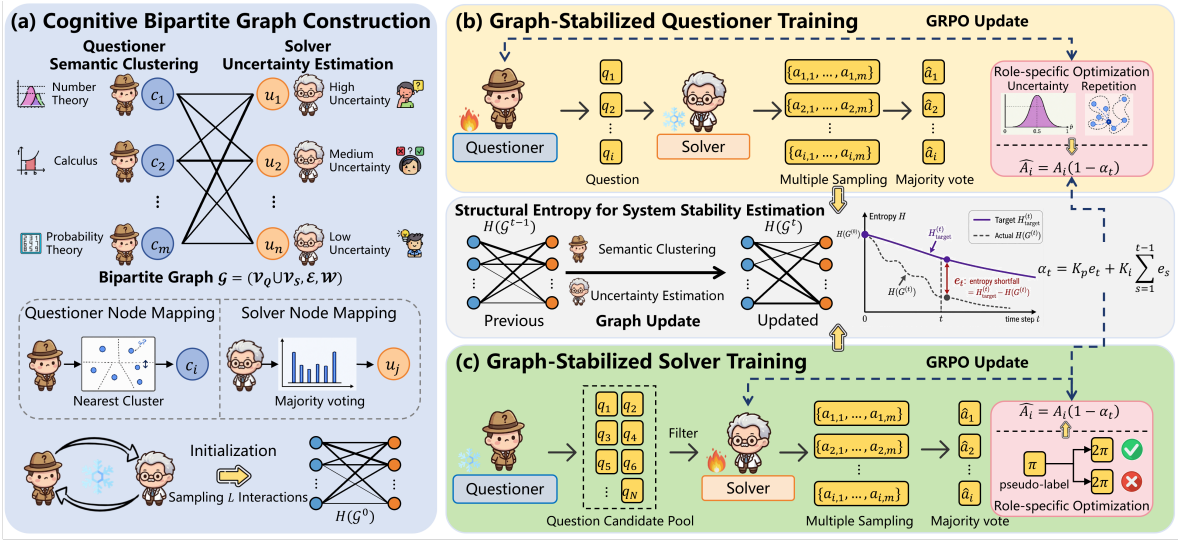

Figure 3. The framework of the proposed S-BGM model.

3.1. Cognitive Bipartite Graph Construction

Based on Section 2, the self-evolving system is modeled as a cognitive bipartite graph $\mathcal{G} = (\mathcal{V}_Q \cup \mathcal{V}_S, \mathcal{E}, \mathcal{W})$, where the two partitions characterize the evolving interactions between the two roles.

Questioner Node Mapping via Semantic Clustering. To represent the Questioner-side node set $\mathcal{V}_Q$, we discretize the continuous question space into semantic clusters, mapping semantically similar questions to the same node so that each node captures a coarse-grained question type rather than a single problem. Following previous work [13], we focus on mathematical reasoning and adopt the

 doi:10.20944/preprints202605.1514.v1

training set of the MATH dataset [17] as the reference corpus $\mathcal{D}$, which provides a fixed semantic coordinate system. Each problem in $\mathcal{D}$ is embedded by an embedding model $E(\cdot)$, and K-means clustering is then applied to obtain m cluster centers $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$, corresponding to distinct mathematical sub-topics (e.g., number theory, calculus, probability). Each generated question q is then mapped to its nearest cluster center (node) by cosine similarity: $v^*(q) = \arg\max_{v_i \in \mathcal{C}} \cos(E(q), c_i)$.

Solver Node Mapping via Uncertainty Estimation. On the Solver side $\mathcal{V}_S$, responses with similar uncertainty are assigned to the same node, capturing the Solver's confidence. For a question q, the Solver draws R responses $\{a_j\}_{j=1}^{R}$. Following [3,5,8], the consensus answer $\tilde{a}$ is obtained by majority voting, yielding the consensus ratio $r(q) = \frac{1}{R}\sum_{j=1}^{R} \mathbf{1}[a_j = \tilde{a}]$, where $\mathbf{1}$ is the indicator function. Larger $r(q)$ indicates higher confidence, smaller $r(q)$ denotes higher uncertainty. The response is then mapped to the node $u^*(q) = u_{\min(\lfloor n \cdot r(q) \rfloor + 1,\, n)}$, with n the number of Solver-side nodes, thereby grouping responses with similar consensus ratios.

Cognitive Bipartite Graph Initialization. The cognitive bipartite graph is initialized as a stable reference state for subsequently evolved models, grounded in the vanilla Questioner and Solver. A set of L interactions are further sampled, in which each question is mapped to a Questioner-side node $v_i \in \mathcal{V}_Q$ via semantic clustering, and the corresponding response is mapped to a Solver-side node $u_j \in \mathcal{V}_S$ via uncertainty estimation. Starting from an all-zero adjacency matrix $\mathbf{B}^{(0)} \in \{0\}^{m \times n}$, the entry at (v_i, u_j) is updated as $\mathbf{B}_{ij}^{(0)} \leftarrow \mathbf{B}_{ij}^{(0)} + 1$. After processing all L interactions, the resulting bipartite graph $\mathcal{G}^{(0)}$ represents the initial stable state of the system.

3.2. Graph-Stabilized Questioner Training

The Questioner $\mathcal{M}_Q$ is trained with two signals: a role-specific reward measuring each question's local usefulness, and a stability control mechanism applied to the normalized GRPO advantage, weighted by the Cognitive Bipartite Graph's structural entropy.

3.2.1. Role-Specific Optimization Reward

Following prior works [5,9], role-specific rewards are employed to encourage the Questioner to generate questions that are both informative for the Solver and sufficiently diverse. For each generated question q_i, the Solver samples R candidate answers $\{a_{i,j}\}_{j=1}^{R}$. The pseudo-label $\tilde{a}_i$ is obtained via majority voting, while the consensus ratio $\hat{p}_i$ is computed as an empirical estimate of the Solver's confidence. Given that an effective question should ideally lie close to the Solver's ability boundary, an uncertainty reward centered at $\hat{p}_i = 0.5$ is introduced: $r_{\text{unc}}(q_i) = \exp\left(-\frac{(\hat{p}_i - 0.5)^2}{2\sigma^2}\right)$, where σ controls the tolerance around the target uncertainty level. To discourage redundant question generation, a historical question embedding pool is maintained by $\mathcal{P}_{\text{emb}} = \{E(q_k)\}_{k=1}^{M}$. For each newly generated question q_i, its maximum cosine similarity s_i to the pool $\mathcal{P}_{\text{emb}}$ is computed, and the repetition penalty is subsequently defined as: $r_{\text{rep}}(q_i) = \max(0, s_i - \tau)$, where τ is the similarity threshold. For valid questions, the final role-specific optimization reward is

$$r_Q(q_i) = \max\big(0, r_{\text{unc}}(q_i) - r_{\text{rep}}(q_i)\big). \tag{1}$$

3.2.2. Structural Entropy for System Stability Estimation

Bipartite Graph Update. During self-evolving training, the sum of edge weights of in the cognitive bipartite graph is kept at size L and is updated with a sliding window to track recent interactions. For each generated interaction (i.e., edge) consisting of a Questioner-side node v_i and a Solver-side node u_j, the corresponding entry in the adjacency matrix is updated as $\mathbf{B}_{ij} \leftarrow \mathbf{B}_{ij} + 1$. At the same time, the oldest interaction $(v_{i'}, u_{j'})$ is removed by performing the update $\mathbf{B}_{i'j'} \leftarrow \mathbf{B}_{i'j'} - 1$. The resulting graph $\mathcal{G}^{(t)}$ summarizes the latest L interactions and supports the system stability estimation.

Structural Entropy Calculation. We define structural entropy to quantify the dispersion of interaction weights across the bipartite graph. It measures whether Questioner–Solver interactions are broadly distributed over the state space or concentrated on a limited subset of states. This distribution

 doi:10.20944/preprints202605.1514.v1

pattern serves as an indicator of the system's current structural status, and thereby provides a global stability signal. Inspired by [18], let d_i be the weighted degree of node i, $\mathrm{vol}(\mathcal{G}) = \sum_i d_i$ be the graph volume, and $\mathcal{T}$ be the partitioning tree of height K to represent a hierarchical clustering of the vertices. The K-dimensional structural entropy of $\mathcal{G}$ is defined as the minimum structural entropy $H^{\mathcal{T}}(\mathcal{G})$ achieved across all possible partitioning trees of height K:

$$H^{\mathcal{T}}(\mathcal{G}) = -\sum_{\alpha \in \mathcal{T}, \alpha \neq \lambda} \frac{g_\alpha}{\mathrm{vol}(\mathcal{G})} \log_2 \frac{V_\alpha}{V_{\alpha^-}}, \quad H(\mathcal{G}) = \min_{\mathcal{T}:\mathrm{height}(\mathcal{T})=K} H^{\mathcal{T}}(\mathcal{G}) \tag{2}$$

where α is a non-root node in the partitioning tree, λ is the root of $\mathcal{T}$, α^- is the parent of α, V_α is the volume of the vertex subset associated with α, and g_α is the total weight of edges leaving that subset. A high-entropy graph indicates that Questioner-Solver interactions are broadly distributed across diverse states, suggesting that the system operates in a normal manner. Conversely, a low-entropy graph suggests that interactions are concentrated on a small subset of states, denoting that the system has a tendency to collapse. More details about structural entropy are provided in Appendix F.

System Stability Estimation. Let t denote the training time step, and let $H\left(\mathcal{G}^{(t)}\right)$ denote the structural entropy at step t. A naïve strategy is to maximize $H\left(\mathcal{G}^{(t)}\right)$, which may over-regularize the system by forcing interactions to remain uniformly dispersed, which can suppress useful learning signal. Instead of pursuing maximum entropy, we anchor the objective to the initial graph structure and allow its entropy to decay gradually throughout training. The initial graph $\mathcal{G}^{(0)}$ is treated as a stable reference state, and its entropy $H\left(\mathcal{G}^{(0)}\right)$ defines a training target that decays slowly over time:

$$H^{(t)}_{\mathrm{target}} = H(\mathcal{G}^{(0)}) - \gamma \log(1+t), \quad e_t = H^{(t)}_{\mathrm{target}} - H(\mathcal{G}^{(t)}), \tag{3}$$

where $\gamma \geq 0$ controls the rate of entropy decay. The error term e_t measures the deviation between the current structural entropy and the target entropy. A positive e_t indicates that the current interaction topology is over-concentrated, suggesting a stronger need for stability regulation. A direct way to regulate the system is to react to this instantaneous error. However, relying only on e_t may overlook persistent drift. If the graph entropy remains below the target threshold for multiple steps, such one-step proportional feedback cannot fully capture the long-term cumulative instability. Therefore, we introduce a proportional-integral (PI) control mechanism that combines the current entropy error with its accumulated historical values:

$$\alpha_t = \mathrm{clip}\left(K_p e_t + K_i \sum_{s=1}^{t-1} e_s,\ 0,\ 1\right), \tag{4}$$

where $K_p = 1$ is the proportional gain for reacting to the current entropy gap, $K_i = 0.01$ is the integral gain for accumulating persistent deviations over previous steps, and $\mathrm{clip}(\cdot, 0, 1)$ constrains the coefficient to $[0, 1]$. As a result, α_t provides an adaptive scaling factor that reflects the strength of stability regulation required at training step t. Please refer to Appendix G for more details.

3.2.3. Training Objective Function

The role-specific optimization reward r_Q and the estimated stability coefficient α_t are incorporated into the standard GRPO [19] function. For each sampled group of size G, the role-specific reward r^i_Q of the i-th generated question is first normalized within the group to obtain the GRPO advantage $\hat{A}_i$, and the estimated stability coefficient α_t is then applied to the normalized advantage:

$$\hat{A}_i = \frac{r^i_Q - \mathrm{mean}(r^1_Q, \ldots, r^G_Q)}{\mathrm{std}(r^1_Q, \ldots, r^G_Q) + \epsilon_{\mathrm{norm}}}, \quad \hat{A}'_i = (1 - \alpha_t)\hat{A}_i. \tag{5}$$

 doi:10.20944/preprints202605.1514.v1

The subsequent process follows the standard GRPO. When structural entropy falls below the target threshold, indicating excessive concentration of interactions, α_t is increased to suppress the effective advantage magnitude and penalize the model update. Please refer to Appendix G.4 for more details.

3.3. Graph-Stabilized Solver Training

After the Questioner is updated, the Solver is also trained with two objectives. First, the updated Questioner is utilized to generate a candidate pool $\mathcal{P}_{\text{candidate}} = \{q_i\}_{i=1}^{N}$ as a training set for the Solver. Following previous work [20], for each candidate question, the current Solver produces m answers, from which a majority-vote pseudo-label $\tilde{a}_i$ and a consensus ratio $\hat{p}_i$ are derived as described above. A question is retained only when its consensus ratio lies near the Solver's uncertainty boundary: $\left|\hat{p}_i - \frac{1}{2}\right| \leq \delta$, where δ is the tolerance threshold that controls the width of the retained uncertainty region. This filtering step removes questions that are already too easy for the Solver, as well as questions whose pseudo-labels are too unreliable [20,21]. The Solver $\mathcal{M}_S$ is then trained on the resulting dataset $\mathcal{D}_S = \{(q_i, \tilde{a}_i)\}$ with GRPO. For each retained question q_i, the Solver samples R candidate answers $\{a_{i,j}\}_{j=1}^{R}$ and receives a binary verifiable reward, termed the role-specific optimization reward, based on the pseudo-label obtained through majority voting: $r_S(a_{i,j}; q_i) = \mathbf{1}[a_{i,j} = \tilde{a}_i]$. Meanwhile, the cognitive bipartite graph is updated with newly generated Questioner-Solver interactions, and the stability scaling factor α_t is computed as described in Section 3.2.2. The Solver is therefore optimized with the standard pseudo-label-based correctness reward, while its normalized GRPO advantage is further scaled by the system-level stability signal.

3.4. Iterative Co-Evolution Paradigm

S-BGM forms a complete self-evolving loop by alternating between Questioner and Solver optimization. At each iteration, the Solver is kept fixed while the Questioner is optimized with the role-specific reward r_Q to generate questions that are both informative and diverse. Meanwhile, every valid Questioner-Solver interaction is mapped onto a cognitive bipartite graph, whose latest structural entropy is used to estimate the stability coefficient α_t. This coefficient modulates the normalized GRPO advantage, so the Questioner update follows standard GRPO while being regularized by the system's current stability signal. After the Questioner update, it produces a new candidate curriculum for Solver training. The Solver generates multiple responses per question; majority voting then yields pseudo-labels and consensus ratios for uncertainty-boundary filtering. The retained questions constitute the Solver training set, and the Solver is subsequently optimized with pseudo-label supervision under the same graph-based stability estimation. See Algorithm A1 for details.

4. Experiments

4.1. Experimental Setup

Datasets. Evaluation is conducted on seven popular mathematical benchmarks and three general reasoning benchmarks. The mathematical benchmarks include GSM8K [22], MATH-500 [17], AMC, Minerva Math [23], OlympiadBench [24], AIME-2024, and AIME-2025, spanning from basic to competition-level mathematics. The general reasoning benchmarks are MMLU-Pro [25], SuperGPQA [26] and BBEH [27]. Pass@1 accuracy with greedy decoding is reported for all benchmarks, with the exception of AMC and AIME, where mean@32 is used following prior work [5].

Baselines. Our method is compared against representative self-evolving methods including Absolute Zero [4], SPICE [9], R-zero [5], R-Few [8], SPIRAL [28]. Details are summarized in Appendix C.

Implementation Details. Experiments are conducted using Qwen3-4B-Base and Qwen3-8B-Base [16] as base models. Self-evolving training is run for 5 iterations using GRPO algorithm [19]. For the cognitive bipartite graph, the number of semantic clusters is $m = 100$ and the number of uncertainty intervals is $n = 10$. The embedding model is Qwen3-Embedding-0.6B. Unless otherwise specified, the structural entropy is configured with $K = 2$. More details and prompts are provided in Appendix B.

 doi:10.20944/preprints202605.1514.v1

4.2. Main Results

Table 1 shows that S-BGM achieves the best aggregate performance across both model scales. The gains are especially pronounced on mathematical reasoning benchmarks, including competition-style tasks such as AMC, Olympiad, and AIME. On general reasoning benchmarks, S-BGM remains competitive, achieving strong results on SuperGPQA and BBEH while maintaining comparable performance on MMLU-Pro. These results suggest that stabilizing the Questioner-Solver interaction benefits self-evolving training. By discouraging the interaction topology from collapsing into narrow question or response patterns, S-BGM helps preserve a more diverse and informative self-generated curriculum. This stability directly enhances mathematical reasoning and demonstrates strong generalization ability. Overall, the improvements in both Math AVG and Overall AVG indicate that graph-based stability modulation provides benefits beyond role-specific optimization.

Table 1. Comparison results of S-BGM and baselines on mathematical and general reasoning benchmarks. **Bold** denotes the best and underline denotes the second best.

Model	Math AVG	Overall AVG	Mathematical Reasoning Benchmarks							General Reasoning		
			GSM8K	MATH-500	AMC	Minerva	Olympiad	AIME-2024	AIME-2025	MMLU-Pro	SuperGPQA	BBEH
Qwen3-4B-Base												
Base Model	42.58	36.39	87.79	68.20	45.70	38.24	41.04	10.94	6.15	37.38	20.88	7.57
Absolute Zero	46.42	41.29	89.34	76.20	52.45	41.96	42.56	12.20	10.20	52.60	27.10	8.30
SPICE	50.59	45.47	92.70	78.00	57.50	51.90	42.70	12.20	**19.10**	**58.10**	30.20	12.30
SPIRAL	47.00	41.89	91.00	76.40	57.50	42.40	38.40	13.30	10.00	53.20	27.10	9.57
R-Zero	49.07	43.30	92.12	79.60	57.27	52.94	44.59	12.71	4.27	51.53	27.55	10.42
R-Few	49.06	44.08	92.60	78.00	52.40	53.20	42.80	14.50	9.90	56.20	29.40	11.80
S-BGM	**53.55**	**47.42**	**93.24**	**81.20**	**61.42**	**58.25**	**46.98**	**18.54**	15.20	56.48	**30.50**	**12.39**
Qwen3-8B-Base												
Base Model	49.18	43.31	89.08	78.00	51.95	50.00	44.74	13.85	16.67	51.80	28.33	8.63
Absolute Zero	52.68	47.20	92.00	76.60	57.89	57.90	47.80	18.40	18.20	60.50	31.89	10.80
SPICE	54.34	49.60	92.70	79.40	70.00	59.20	42.50	18.40	18.20	65.00	35.70	14.90
SPIRAL	55.48	49.85	92.57	**83.60**	65.90	56.30	49.20	**23.02**	17.80	61.35	34.77	13.98
R-Zero	54.69	48.30	94.09	82.00	61.67	60.66	48.89	16.35	19.17	58.23	31.38	10.60
R-Few	56.31	50.32	93.50	82.60	72.30	60.30	46.40	18.85	20.20	63.20	33.50	12.30
S-BGM	**58.94**	**52.84**	**94.47**	**83.60**	**73.90**	**67.32**	**49.42**	22.85	**21.05**	**65.09**	**35.72**	**14.98**

4.3. Analysis of the Correlations between Model Collapse and Structural Entropy

To examine the relationship between model collapse and structural entropy, we construct a cognitive bipartite graph and compute its structural entropy after each iteration, as shown in Figure 4. The left panel shows that R-Zero (blue curve) degrades after Iteration 3, coinciding with a sharp drop in structural entropy. In contrast, S-BGM achieves a steadily increasing Math AVG and follows a smoother entropy changing trajectory. The middle and right panels further indicate that, from Iteration 0 to 5, the normalized weighted-degree distributions of S-BGM become moderately concentrated from both the Solver and Questioner sides. On the Solver side, although the top-ranked node still becomes higher, it is relatively slight compare to Figure 2, indicating S-BGM effectively suppresses the overconfidence caused by majority voting. On the Questioner side, the distribution over most nodes remains close to the initial one, suggesting that S-BGM helps maintain diverse question-generation states.

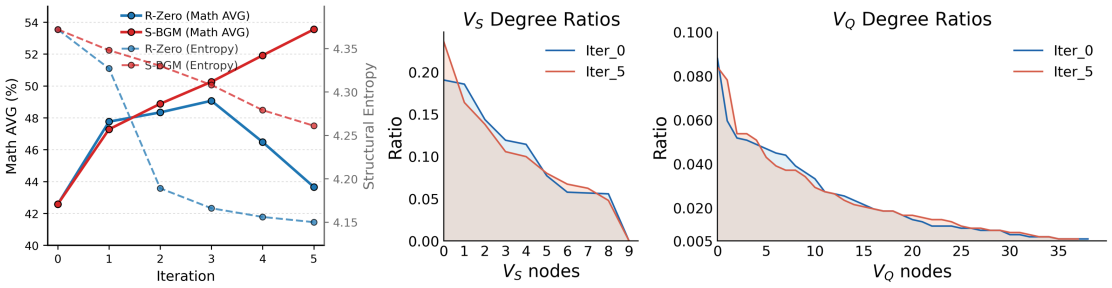

Figure 4. Structural entropy and topological concentration in model collapse.

4.4. Evolution of Question Difficulty

To examine how question difficulty evolves during S-BGM training, 200 questions are sampled from the Questioner at each iteration to construct five evaluation sets $\{\mathcal{D}_{\text{Iter } k}\}_{k=1}^{5}$, with reference labels provided by GPT-4o. Solvers from different iterations are then evaluated on these cross-iteration

 doi:10.20944/preprints202605.1514.v1

datasets. As shown in Table 2, for a fixed Solver, pass rates generally decrease on later-iteration question sets, indicating that the generated problems become progressively harder. For a fixed question set, later Solvers usually achieve higher pass rates, suggesting that the increased difficulty remains learnable rather than noisy. The matched entries stay close to 50%, showing that the Questioner continues to generate questions near the Solver's uncertainty boundary. Overall, S-BGM produces a progressive and learnable difficulty schedule that supports effective self-evolving training.

Table 2. Cross-iteration evaluation for question difficulty evolution. Each entry reports the pass rate (%) of a Solver.

Dataset	Base	Solver				
		Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
$\mathcal{D}_{\text{Iter 1}}$	**48.5**	56.5	59.0	59.5	59.0	60.5
$\mathcal{D}_{\text{Iter 2}}$	42.5	**49.0**	55.0	53.5	58.0	56.5
$\mathcal{D}_{\text{Iter 3}}$	44.0	45.5	**48.0**	50.0	51.0	54.0
$\mathcal{D}_{\text{Iter 4}}$	41.5	45.0	45.5	**49.5**	52.0	53.5
$\mathcal{D}_{\text{Iter 5}}$	38.5	41.5	40.5	45.5	**48.5**	51.5

4.5. Generalization Analysis of the Performance Gains

A critical question is whether S-BGM can sustain performance gains over multiple self-evolving iterations across different model scales. Figure 5 compares S-BGM, R-Zero, and the base models over five iterations on Qwen3-4B-Base and Qwen3-8B-Base. R-Zero improves in the early stage but later plateaus or collapses, with the degradation being most pronounced on Math AVG. On Qwen3-4B-Base, it peaks around Iteration 3 and drops sharply toward the base-model level by Iteration 5; a similar but milder decline is observed on Qwen3-8B-Base. In contrast, S-BGM maintains a consistently upward trajectory across both model scales on both Math AVG and Overall AVG. These results indicate that structural-entropy-based regularization stabilizes the multi-iteration self-evolving process. While S-BGM and R-Zero perform comparably in the early stage, their gap widens in later iterations, suggesting that S-BGM better mitigates accumulated training instability.

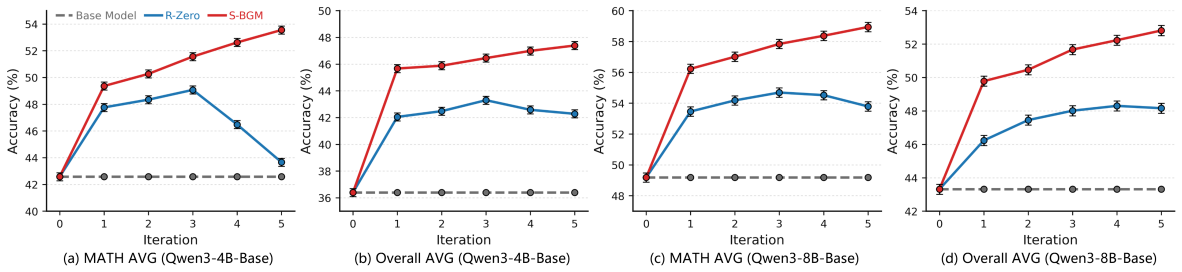

Figure 5. Performance of different iterations on various model scales.

4.6. Ablation Study

To validate the contribution of each component, we conduct ablation studies in Table 3. For role-specific rewards, removing either the uncertainty reward or the repetition penalty hurts performance, indicating that boundary-aware question generation and diversity constraints are both important. Removing uncertainty boundary filtering for Solver training also causes degradation, suggesting that overly easy or overly difficult questions provide less useful training signals. For system-level stabilization, removing the entire structural entropy module or replacing PI control with the instantaneous error e_t underperforms the full method. This confirms that global topology regulation and accumulated entropy deviations help produce a more reliable stability signal.

 doi:10.20944/preprints202605.1514.v1

Table 3. Ablation study for the proposed S-BGM model.

Method	Math AVG	Δ_{math}	General AVG	$\Delta_{general}$
S-BGM	**53.55**	-	**47.42**	-
Ablations on Role-specific Optimization Rewards				
⊢ w/o Uncertainty Reward $r_{unc}(q_i)$ (Questioner)	51.73	-1.82	44.21	-3.21
⊢ w/o Repetition Penalty $r_{rep}(q_i)$ (Questioner)	51.27	**-2.28**	42.43	-4.99
⊢ w/o Uncertainty Boundary Filtering (Solver)	51.94	-1.61	44.76	-2.66
Ablations on System Stability Estimation				
⊢ w/o Structural Entropy Module	51.48	-2.07	42.35	**-5.07**
⊢ w/o PI Control Mechanism	52.57	-0.98	45.38	-2.04

4.7. Hyperparameter Analysis

Figure 6 reports the sensitivity of S-BGM to the structural entropy dimension K, sliding window size L, number of semantic clusters m, and number of uncertainty intervals n. The best performance is achieved with $K = 2$, $L = 1024$, and $(m, n) = (100, 10)$, suggesting that moderate structural resolution is important for stable entropy estimation. For K, one-dimensional entropy may be insufficient to capture the coupling structure, while deeper partitions can introduce estimation noise. For L, a small window is sensitive to short-term fluctuations, whereas a large window retains outdated interactions and reduces responsiveness to recent changes. For (m, n), too few states merge distinct interaction patterns, while too many states sparsify $\mathbf{B}$ and make entropy estimation less reliable.

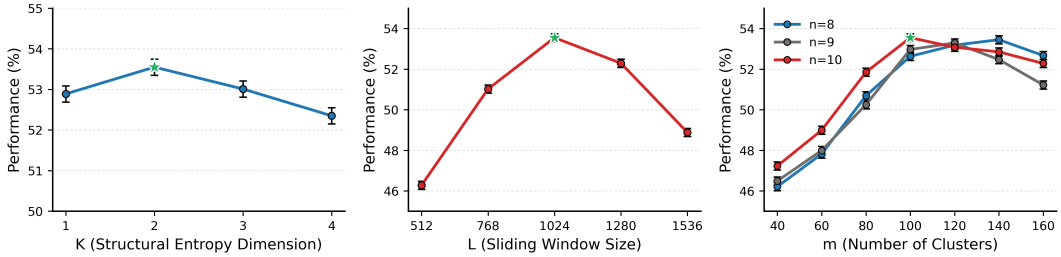

Figure 6. Hyperparameter analysis of the proposed S-BGM model.

5. Conclusions

In this work, we propose S-BGM to study model collapse in self-evolving LLMs from a system-stability perspective. S-BGM models the coupled Questioner–Solver interaction as a Cognitive Bipartite Graph and uses structural entropy to measure whether the interaction topology collapses into a narrow set of states. By incorporating this graph-level stability signal into training, S-BGM combines role-specific capability improvement with system-level stabilization. Extensive experiments show that S-BGM improves aggregate performance and stabilizes multi-iteration self-evolution.

Appendix A. Related Work

Self-Evolving in LLMs Self-evolving Large Language Models are proposed to reduce the reliance on human-annotated or externally collected data by allowing models to generate, verify, and learn from their own synthetic training data [1,2]. Recent works typically instantiate this paradigm by jointly training Questioner and Solver models, demonstrating that such self-evolving training can significantly improve model performance even without external training data [29–31]. Representative methods include R-Zero [5], which co-evolves a Questioner and Solver with uncertainty-guided question generation; Absolute Zero [4], which unifies task proposal and solving through code-executor verification; and SPIRAL [28], which uses multi-turn zero-sum self-play to construct an adaptive training curriculum. Despite these advances, self-evolving systems remain vulnerable to model collapse, motivating recent efforts to understand and stabilize their long-term evolutionary dynamics.

 doi:10.20944/preprints202605.1514.v1

Model Collapse in Self-Evolving LLMs. Model collaps e originally refers to the degeneration of generative models recursively trained on their own synthetic outputs, where the model gradually loses information about the underlying data distribution and converges to a narrow, low-diversity subset of the data [32–34]. In self-evolving LLMs, this phenomenon appears in a different but closely related form: closed-loop training systems often achieve rapid early gains, but then suffer from performance plateaus or even degradation[6,7,13]. Existing methods mainly mitigate this issue from three directions: introducing external data to break closed-loop information symmetry [8,9], improving verification signals to reduce error accumulation from majority voting or self-correction [10,11], and encouraging curriculum diversity through difficulty-aware sampling or quality-diversity search [12,14,35]. However, these approaches still suffer from short-term and role-specific reward optimization, treating the Questioner and Solver as separate components. They therefore overlook a key property of self-evolving training: the involved roles form a coupled dynamical system, where local improvements in one component do not necessarily ensure stable global evolution. This motivates us to study model collapse from a system-level stability perspective rather than only through isolated reward design.

Appendix B. Experimental Details

This section provides the key configurations used in S-BGM. All experiments are conducted on 8× NVIDIA H100 GPUs with BF16 precision and FlashAttention 2 for acceleration.

Appendix B.1. Training Hyperparameters

We detail the hyperparameters introduced in the training procedure and their roles.

- $L = 1024$: The length of the sliding window used to maintained the cognitive bipartite graph.
- $\sigma = 0.01$: Bandwidth of the Gaussian uncertainty reward r_{unc} (see Section 3.2). It controls the decay intensity of the reward when the empirical consensus ratio $\hat{p}_i$ deviates from the target uncertainty boundary of 0.5.
- $\tau = 0.6$: Cosine similarity threshold for the repetition penalty r_{rep} (see Eq. (1)). When the maximum similarity s_i between a newly generated question and the historical question pool $\mathcal{P}_{\text{emb}}$ exceeds this threshold, a penalty term $s_i - \tau$ is incurred to suppress the generation of redundant questions.
- $R = 10$: Number of samples used for the Solver's majority voting. For each generated question, the Solver produces R candidate answers to compute the pseudo-label $\tilde{a}_i$ and the consensus ratio $\hat{p}_i$, balancing pseudo-label reliability with computational sampling overhead.
- $M = 2000$: Capacity of the historical question embedding pool $\mathcal{P}_{\text{emb}}$. This pool is maintained using a sliding-window mechanism to calculate r_{rep}, ensuring that the questions generated by the Questioner maintain continuous semantic diversity during the self-evolution process.
- $N = 8000$: Total number of candidate questions generated by the updated Questioner in each iteration. These questions form the candidate pool $\mathcal{P}_{\text{candidate}}$, which is subsequently filtered via consistency checks to construct the final training set $\mathcal{D}_S$ for the Solver.
- $\gamma = 1 \times 10^{-4}$: Decay rate of the target structural entropy trajectory $H^{(t)}_{\text{target}}$ (see Eq. (3)). It governs the allowable reduction in structural entropy over training steps, reflecting a tolerance for the gradual concentration of interaction patterns as the system moves from broad exploration to focused learning.
- $K_p = 1$: Proportional gain of the PI controller for structural entropy regulation (see Eq. (4)). it provides an immediate corrective response to the current entropy deviation e_t by dynamically adjusting the stability coefficient α_t to rectify trends drifting away from the target trajectory.
- $K_i = 0.01$: Integral gain of the PI controller for structural entropy regulation (see Eq. (4)). It eliminates steady-state errors by accumulating historical entropy deviations $\sum e_s$, allowing the system to capture and correct long-term systemic collapse risks that may be invisible in a single time step.

 doi:10.20944/preprints202605.1514.v1

- $\delta = 0.25$: Filtering bandwidth for the Solver's curriculum learning. Only questions satisfying $|\hat{p}_i - 0.5| \leq \delta$ are retained in the training set $\mathcal{D}_S$, aiming to exclude tasks that are either too simple for the Solver or possess highly unreliable pseudo-labels.

Appendix B.2. Questioner Training

The Questioner is optimized with GRPO using the following hyperparameters:

- **Global Batch Size**: 128
- **Learning Rate**: 1×10^{-6}
- **Weight Decay**: 1×10^{-2}
- **KL Penalty Coefficient** (λ_{KL}): 1×10^{-2}
- **Max Steps**: 5
- **Number of Rollouts**: 4
- **Rollout Temperature**: 1.0
- **Rollout Top-p**: 0.99

Appendix B.3. Solver Training

The Solver is optimized with GRPO using the following hyperparameters:

- **Global Batch Size**: 128
- **Learning Rate**: 1×10^{-6}
- **Weight Decay**: 1×10^{-2}
- **KL Penalty Coefficient** (λ_{KL}): 1×10^{-2}
- **Max Steps**: 15
- **Number of Rollouts**: 5
- **Rollout Temperature**: 1.0
- **Rollout Top-p**: 0.99

Appendix B.4. Prompt Templates

This section presents all prompts used in S-BGM training and analysis. All prompt remain identical to R-Zero [5].

Solver Prompt Template

System Message:
Please reason step by step, and put your final answer within `\boxed{}`.

User Message:
`{problem_statement}`

Note: `{problem_statement}` is a placeholder for the actual math problem.

 doi:10.20944/preprints202605.1514.v1

Questioner Prompt Template

System Message:
You are an expert competition-math problem setter. FIRST, in your private scratch-pad, think step-by-step to design a brand-new, non-trivial problem. The problem could come from any field of mathematics, including but not limited to algebra, geometry, number theory, combinatorics, prealgebra, probability, statistics, and calculus. Aim for a difficulty such that fewer than 30% of advanced high-school students could solve it. Avoid re-using textbook clichés or famous contest problems.
THEN, without revealing any of your private thoughts, output **exactly** the following two blocks:

```
<question>
{The full problem statement on one or more lines}
</question>

\boxed{final_answer}
```

Do NOT output anything else—no explanations, no extra markup.

User Message:
Generate one new, challenging reasoning question now. Remember to format the output exactly as instructed.

GPT-4o Judge Prompt

Configuration:

- **Model:** `gpt-4o`
- **Temperature:** 0.1

System Message:
You are a math answer checker.

User Message Template:
Hi, there is an answer: `{answer}`,
and the ground truth answer is: `{response}`,
please check whether the answer is correct or not, and return the **only** Yes or No.

Note: `{answer}` is a placeholder for the model-generated solution, and `{response}` is the ground-truth answer from the benchmark.

Appendix C. Baseline Details

We compare S-BGM against the following representative self-evolving training methods:

- **Base Model**: The pre-trained Qwen3-4B-Base or Qwen3-8B-Base model without any post-training. This serves as a lower-bound reference to measure the gains brought by self-evolving training.
- **R-Zero** [5]: A foundational self-play framework for reasoning LLMs. It introduces uncertainty-driven curriculum learning and repetition penalties to guide the co-evolution of Questioner and Solver. R-Zero serves as a strong baseline that our method builds upon.
- **Absolute Zero** [4]: A self-evolving framework that leverages code execution for answer verification. The model jointly generates problems and solutions, and uses execution feedback to ensure correctness, representing a tool-augmented paradigm.
- **SPICE** [9]: A self-evolving method that improves curriculum quality via structured problem generation and selection. It emphasizes diversity and informativeness in generated questions by introducing external data to better guide Solver training.

 doi:10.20944/preprints202605.1514.v1

- **SPIRAL** [28]: A self-play framework that formulates the interaction between Questioner and Solver as a zero-sum game. It alternates optimization between the two roles to gradually improve reasoning ability.
- **R-Few** [8]: A variant of self-evolving training that incorporates few-shot guidance to stabilize training and improve sample efficiency, alleviating the instability of pure self-play.

All baselines are implemented following their original papers with default hyperparameter settings.

Appendix D. Additional Experiments on Iteration Scaling and PI Control

Appendix D.1. Analysis of More Self-Evolving Iterations

To further examine the long-horizon behavior of S-BGM, we extend the self-evolving process to more iterations on Qwen3-4B-Base, as shown in the left and middle panels of Figure A1. The results show that S-BGM continues to improve substantially in the early iterations and consistently outperforms the base model. After around five iterations, the performance curve becomes relatively stable, with only marginal additional gains. This suggests that S-BGM can prevent the rapid performance collapse observed in conventional self-evolving methods, but the benefit of further iterations gradually saturates. This result is reasonable because reinforcement learning mainly serves as a mechanism to elicit and refine the latent capabilities of the base model, rather than indefinitely expanding its intrinsic capacity [36–38]. In the early stages, self-evolving RL can effectively activate underutilized reasoning behaviors through increasingly informative training signals. However, once most learnable signals from the self-generated curriculum have been absorbed, further iterations provide diminishing returns because the achievable performance is increasingly constrained by the inherent limitations of the base model. Therefore, the plateau after several iterations should not be interpreted as model collapse, but rather as the natural saturation of RL-based capability elicitation under self-generated supervision.

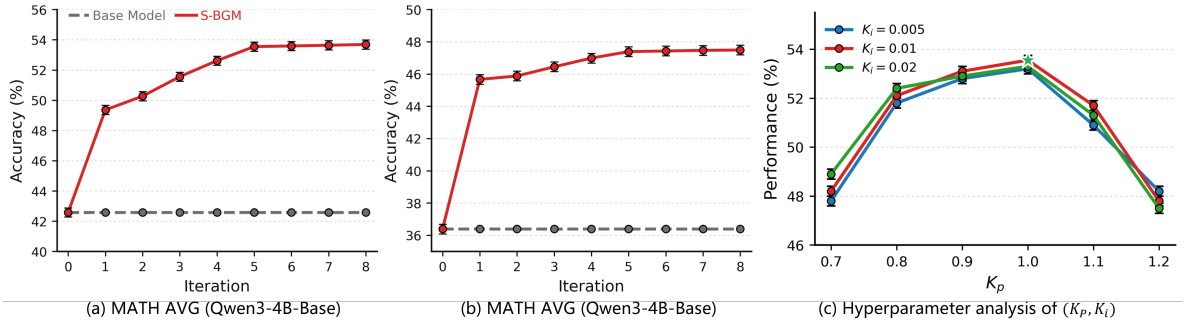

Figure A1. Additional analysis of extended self-evolving iterations and PI-control hyperparameters.

Appendix D.2. Hyperparameter Analysis of PI Control

The right panel of Figure A1 analyzes the sensitivity of S-BGM to the PI-control parameters K_p and K_i. Overall, the best performance is obtained around $K_p = 1.0$, while both smaller and larger values lead to lower performance. When K_p is too small, the controller responds weakly to entropy deviations, making stability regulation insufficient. When K_p is too large, the controller may overreact to short-term entropy fluctuations and suppress useful updates. The results are relatively stable across different values of K_i, with $K_i = 0.01$ achieving the best or near-best performance in most settings. This indicates that a moderate integral term is helpful for capturing persistent entropy deviations without introducing excessive accumulated correction. Together, these results support the effectiveness of the PI-control design and show that S-BGM performs best when the stability signal reacts sufficiently to topology drift while avoiding overly aggressive update suppression.

Appendix E. Theoretical Guarantee for Discretization

Two-stage view of the discretization. Let (σ_Q, σ_S) denote a raw interaction sample at a given training step, where $\sigma_Q \in \Sigma_Q$ is a question generated by $\mathcal{M}_Q$ in its question-generation space, and

 doi:10.20944/preprints202605.1514.v1

$\sigma_S \in \Sigma_S$ is the Solver's problem-solving behavior on that question, characterizing how $\mathcal{M}_S$ answers the generated question. Our method does not operate directly on $\Sigma_Q \times \Sigma_S$; instead, it factors through two feature maps that render each raw state numerically observable:

$$\psi_Q : \Sigma_Q \to \mathcal{Z}_Q \subset \mathbb{R}^{d_Q}, \qquad \psi_S : \Sigma_S \to \mathcal{Z}_S := [0,1], \tag{A1}$$

where ψ_Q is the semantic embedding of the generated question (underlying the clustering step) and ψ_S is the consensus ratio computed from the Solver's sampled answers, used as a proxy for its ability to correctly answer the question (underlying the uncertainty-binning step). The Questioner feature space $(\mathcal{Z}_Q, \|\cdot\|)$ and Solver feature space $(\mathcal{Z}_S, |\cdot|)$ are endowed with the sum metric $d\big((z_Q, z_S), (z'_Q, z'_S)\big) := \|z_Q - z'_Q\| + |z_S - z'_S|$.

Let π denote the joint distribution of the feature pair $(\psi_Q(\sigma_Q), \psi_S(\sigma_S))$ on $\mathcal{Z}_Q \times \mathcal{Z}_S$. The Questioner partition $\{C_i\}_{i=1}^m$ (with centroids $\{c_i\}$) lives on $\mathcal{Z}_Q$ and the Solver partition $\{I_j\}_{j=1}^n$ (with midpoints $\{s_j\}$) lives on $\mathcal{Z}_S$; the discretized distribution and bi-adjacency matrix are defined by

$$\tilde{\pi}(v_i, u_j) \;:=\; \pi(C_i \times I_j), \qquad \mathbf{B}_{ij} \;\propto\; \tilde{\pi}(v_i, u_j). \tag{A2}$$

Under this view, $\mathbf{B}$ is a summary of the feature-level joint distribution π, which is itself an observable reduction of the raw Questioner–Solver interaction.

Formal criterion. Let $\varepsilon_Q := \max_i \sup_{z \in C_i} \|z - c_i\|$ be the maximum cluster radius, $\varepsilon_S := 1/n$ the bin width, and $\varepsilon := \varepsilon_Q + \varepsilon_S$ the combined partition mesh. The discretization is said to be *rational at resolution* ε if the following three properties hold with error controlled by ε:

(P1) Expectations of Lipschitz observables under π are approximated by their bi-adjacency counterparts;
(P2) $\tilde{\pi}$ is close to π in a metric compatible with the feature-space geometry;
(P3) The Questioner–Solver dependence encoded by $\mathbf{B}$ is a consistent summary of the true feature-level coupling in π.

Assumptions.

(A1) ***Feature-level regularity.*** π admits a density p with respect to a reference product measure on $\mathcal{Z}_Q \times \mathcal{Z}_S$ that is bounded, $\|p\|_\infty \le P_{\max}$, and L_p-Lipschitz under d. This is a statement about the *feature-level* density rather than the raw interaction measure: it is automatically implied when ψ_Q, ψ_S are sufficiently smooth and the raw measure on $\Sigma_Q \times \Sigma_S$ is regular.

(A2) ***Partition resolution.*** The cluster radii ε_Q and bin width ε_S are well-defined and finite.

Theorem A1 (Rationality of discretization, non-asymptotic form)**.** *Under* (A1)–(A2)*, for any finite* $m, n \ge 1$*,*

$$\text{(P1)} \quad \sup_{\mathrm{Lip}(f) \le 1} \big|\mathbb{E}_\pi[f] - \mathbb{E}_{\tilde{\pi}}[f]\big| \;\le\; \varepsilon, \tag{A3}$$

$$\text{(P2)} \quad W_1(\pi, \tilde{\pi}) \;\le\; \varepsilon, \tag{A4}$$

$$\text{(P3)} \quad \big|\,\Delta(\pi) - \Delta(\tilde{\pi})\,\big| \;\le\; C_g\,\varepsilon, \tag{A5}$$

where $\Delta(\mu) := \|\mu - \mu^Q \otimes \mu^S\|_{\mathrm{TV}}$ *measures the coupling strength between the* $\mathcal{Z}_Q$ *and* $\mathcal{Z}_S$ *marginals of* μ*, and* C_g *is a constant depending only on* L_p *and* $P_{\max}$ *that is independent of* m, n*.*

We establish the theorem through three supporting lemmas proved in turn below.

Lemma A1 (Quantization fidelity, (P1))**.** *Under* (A1)–(A2)*, for any L-Lipschitz* $f : \mathcal{Z}_Q \times \mathcal{Z}_S \to \mathbb{R}$*,*

$$\Big|\mathbb{E}_\pi[f] - \sum_{i,j} \tilde{\pi}(v_i, u_j) f(c_i, s_j)\Big| \;\le\; L\varepsilon. \tag{A6}$$

 doi:10.20944/preprints202605.1514.v1

Proof. Write the left-hand side as

$$\sum_{i,j} \int_{C_i \times I_j} \big[f(z_Q, z_S) - f(c_i, s_j) \big] \, d\pi(z_Q, z_S).$$

On each cell $C_i \times I_j$, the L-Lipschitz condition and the definition of $\varepsilon_Q, \varepsilon_S$ give

$$|f(z_Q, z_S) - f(c_i, s_j)| \;\le\; L\big(\|z_Q - c_i\| + |z_S - s_j|\big) \;\le\; L(\varepsilon_Q + \varepsilon_S) = L\varepsilon.$$

Taking absolute values, summing over all cells, and using $\sum_{i,j} \pi(C_i \times I_j) = 1$ yields (A6). □

Lemma A2 (Topological stability, (P2)). *Under* (A1)–(A2), $W_1(\pi, \tilde{\pi}) \le \varepsilon$.

Proof. Apply the Kantorovich–Rubinstein duality to Lemma A1 with $L = 1$; the supremum over all 1-Lipschitz functions is exactly $W_1(\pi, \tilde{\pi})$. □

Remark A1. If one further refines the partitions, the standard covering estimate $\varepsilon_Q = O(m^{-1/d_Q^{\mathrm{eff}}})$ (with d_Q^{eff} the intrinsic dimension of $\psi_Q(\Sigma_Q)$) together with $\varepsilon_S = 1/n$ implies $\tilde{\pi} \Rightarrow \pi$ weakly as $m, n \to \infty$. This asymptotic regime is not required by our analysis, which is non-asymptotic in (m, n).

Lemma A3 (Coupling preservation, (P3)). *Under* (A1)–(A2), $|\Delta(\pi) - \Delta(\tilde{\pi})| \le C_g \, \varepsilon$, *where* $C_g = 2(L_p + 2P_{\max} L_p)$ *depends only on* L_p *and* $P_{\max}$ *and is independent of* m *and* n.

Proof. Let $g(z_Q, z_S) := p(z_Q, z_S) - p^Q(z_Q) p^S(z_S)$, so that $\Delta(\pi) = \frac{1}{2} \int_{\mathcal{Z}_Q \times \mathcal{Z}_S} |g|$. By (A1) and the Lipschitz property of marginalization, g is L_g-Lipschitz under d with $L_g \le L_p + 2P_{\max} L_p$.

On the discrete side, since $\tilde{\pi}(v_i, u_j) = \int_{C_i \times I_j} p$ and likewise for the marginals,

$$\tilde{g}(i, j) \;:=\; \tilde{\pi}(v_i, u_j) - \tilde{\pi}^Q(v_i) \, \tilde{\pi}^S(u_j) \;=\; \int_{C_i \times I_j} \big[p(z_Q, z_S) - \bar{p}_i^Q \, \bar{p}_j^S \big] \, dz_Q \, dz_S,$$

where $\bar{p}_i^Q$ and $\bar{p}_j^S$ are cell-averaged marginal densities. Replacing $\bar{p}_i^Q \bar{p}_j^S$ by $p^Q(z_Q) p^S(z_S)$ within $C_i \times I_j$ introduces a pointwise error bounded by $2P_{\max} L_p \, \varepsilon$ (a Lipschitz remainder controlled by (A1)). Hence

$$\tilde{g}(i, j) \;=\; \int_{C_i \times I_j} g(z_Q, z_S) \, dz_Q \, dz_S \;+\; \rho_{ij}, \qquad |\rho_{ij}| \le 2P_{\max} L_p \, \varepsilon \cdot \mathrm{vol}(C_i \times I_j), \tag{A7}$$

so that $\sum_{i,j} |\rho_{ij}| \le 2P_{\max} L_p \, \varepsilon$.

It remains to compare $\sum_{i,j} \big| \int_{C_i \times I_j} g \big|$ with $\int |g|$. By the triangle inequality, $\int_{C_i \times I_j} |g| - \big| \int_{C_i \times I_j} g \big|$ equals twice the integral of the smaller of g_+ and g_- over the cell, which is bounded by the oscillation of g on that cell:

$$\int_{C_i \times I_j} |g| \;-\; \Big| \int_{C_i \times I_j} g \Big| \;\le\; \mathrm{osc}_{C_i \times I_j}(g) \cdot \mathrm{vol}(C_i \times I_j) \;\le\; L_g \, \varepsilon \cdot \mathrm{vol}(C_i \times I_j).$$

Summing over all cells and using $\sum_{i,j} \mathrm{vol}(C_i \times I_j) = 1$,

$$\Big| \int |g| \;-\; \sum_{i,j} \Big| \int_{C_i \times I_j} g \Big| \Big| \;\le\; L_g \, \varepsilon. \tag{A8}$$

Combining (A7) and (A8) via the triangle inequality,

$$\big| \, \Delta(\pi) - \Delta(\tilde{\pi}) \, \big| \;\le\; \tfrac{1}{2} \big(L_g + 2P_{\max} L_p \big) \varepsilon \;\le\; \big(L_p + 2P_{\max} L_p \big) \varepsilon \;=:\; \tfrac{1}{2} C_g \, \varepsilon.$$

Absorbing the factor $\frac{1}{2}$ gives the stated bound with $C_g = 2(L_p + 2P_{\max} L_p)$. □

 doi:10.20944/preprints202605.1514.v1

Proof of Theorem A1. (A3) is Lemma A1 with $L = 1$; (A4) is Lemma A2; (A5) is Lemma A3. All three bounds hold for every finite $m, n \geq 1$ without any asymptotic requirement. □

Discussion: the regime $m = 100$, $n = 10$. Our implementation fixes $n = 10$, yielding $\varepsilon_S = 0.1$, and $m = 100$. The effective support of the Questioner feature marginal $\pi^Q = (\psi_Q)_\# \pi$ concentrates on a low-dimensional manifold with intrinsic dimension $d_Q^{\text{eff}} \ll \dim(\mathcal{Z}_Q)$ [39,40], so a few hundred centroids suffice to cover it at small radius ε_Q. Under this configuration ε is a moderate constant, and Theorem A1 certifies that the Cognitive Bipartite Graph faithfully reproduces the expectations, geometry, and Questioner–Solver coupling of π up to this controlled error. The Cognitive Bipartite Graph should accordingly be interpreted as a faithful—though deliberately coarse-grained—abstraction of π, whose coarsening level is chosen to balance the statistical stability of $\mathbf{B}$ against resolution.

Appendix F. Detailed Formulation and Calculation of Structural Entropy

Structural entropy measures the amount of uncertainty needed to identify the state reached by a random walk under a graph coding structure [41]. Our framework apply it to the Cognitive Bipartite Graph $\mathcal{G} = (\mathcal{V}_Q \cup \mathcal{V}_S, \mathcal{E}, \mathcal{W})$ as a system-level stability signal. High structural entropy means that interaction mass is distributed across many Questioner-side semantic states and Solver-side response states, whereas low structural entropy indicates that the interaction topology is concentrated on a small subset of states.

For computation, the bi-adjacency matrix $\mathbf{B} \in \mathbb{R}_{\geq 0}^{m \times n}$ is converted into the undirected weighted adjacency matrix

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}_{m \times m} & \mathbf{B} \\ \mathbf{B}^\top & \mathbf{0}_{n \times n} \end{bmatrix} \in \mathbb{R}_{\geq 0}^{(m+n) \times (m+n)}. \tag{A9}$$

Each entry A_{ij} denotes the interaction weight between vertices i and j. The weighted degree of node i and the graph volume are

$$d_i = \sum_j A_{ij}, \qquad \text{vol}(\mathcal{G}) = \sum_i d_i. \tag{A10}$$

For a vertex subset $S \subseteq \mathcal{V}$, its volume and boundary weight are defined as

$$\text{vol}(S) = \sum_{i \in S} d_i, \qquad g(S) = \sum_{i \in S, j \notin S} A_{ij}. \tag{A11}$$

Here $g(S)$ is the total weight of edges leaving S. Since all terms are defined through weighted volumes and volume ratios, a global normalization of edge weights is not required; the edge weights only need to be non-negative. If the graph has no edge, we set the entropy to zero.

Partitioning tree. A partitioning tree $\mathcal{T}$ is a rooted tree that represents a hierarchical partition of $\mathcal{V}$. Its root is denoted by λ and is associated with the full vertex set $S_\lambda = \mathcal{V}$. Each tree node $\alpha \in \mathcal{T}$ is associated with a non-empty vertex subset $S_\alpha \subseteq \mathcal{V}$. If α has children $\{\beta : \beta^- = \alpha\}$, where β^- denotes the parent of β, then these children form a disjoint partition of the parent subset:

$$S_\alpha = \bigcup_{\beta : \beta^- = \alpha} S_\beta, \qquad S_\beta \cap S_{\beta'} = \emptyset \quad (\beta \neq \beta'). \tag{A12}$$

Every leaf node corresponds to a singleton vertex. Thus, moving from the root to the leaves progressively refines the whole graph into modules, submodules, and finally individual vertices. The height of $\mathcal{T}$ is the maximum root-to-leaf depth; a height-K tree gives a K-level structural description of the graph. For a tree node α, we use

$$V_\alpha = \text{vol}(S_\alpha), \qquad g_\alpha = g(S_\alpha) \tag{A13}$$

to denote the volume of its associated subset and the total weight of edges leaving that subset.

 doi:10.20944/preprints202605.1514.v1

Structural entropy under a fixed tree. Given a partitioning tree $\mathcal{T}$, the structural entropy of $\mathcal{G}$ with respect to $\mathcal{T}$ is

$$H^{\mathcal{T}}(\mathcal{G}) = -\sum_{\alpha \in \mathcal{T}, \alpha \neq \lambda} \frac{g_\alpha}{\mathrm{vol}(\mathcal{G})} \log_2 \frac{V_\alpha}{V_{\alpha^-}}. \tag{A14}$$

The term $-\log_2(V_\alpha / V_{\alpha^-})$ is the code length needed to identify the child subset S_α inside its parent subset S_{α^-}, while $g_\alpha / \mathrm{vol}(\mathcal{G})$ weights this code length by how often a random walk enters or leaves the corresponding module boundary. Therefore, $H^{\mathcal{T}}(\mathcal{G})$ measures the coding uncertainty induced by the hierarchical organization encoded in $\mathcal{T}$.

***K*-dimensional structural entropy.** The K-dimensional structural entropy is obtained by choosing the height-K partitioning tree that minimizes Equation (A14):

$$H(\mathcal{G}) = \min_{\mathcal{T}:\mathrm{height}(\mathcal{T})=K} H^{\mathcal{T}}(\mathcal{G}). \tag{A15}$$

This is the same quantity denoted by $H(\mathcal{G})$ in the main text when the structural entropy dimension K is fixed. In our experiments, we set $K = 2$, so the tree captures a two-level module structure over Questioner-side and Solver-side states.

One-dimensional structural entropy. The one-dimensional structural entropy can be computed directly from the weighted degree distribution. Specifically, it is defined as

$$H(\mathcal{G}) = -\sum_{i \in \mathcal{V}} \frac{d_i}{\mathrm{vol}(\mathcal{G})} \log_2 \frac{d_i}{\mathrm{vol}(\mathcal{G})}. \tag{A16}$$

In implementation, we first compute the weighted degree of each node by summing the corresponding row of the adjacency matrix, and then substitute the normalized degree distribution into Equation (A16). Therefore, this part can be computed exactly and efficiently.

Computing multi-dimensional structural entropy. For $K > 1$, directly solving the minimization in Equation (A15) is computationally infeasible for large graphs, because the number of possible partitioning trees grows combinatorially with the number of vertices. Even when only flat partitions are considered, the search space already corresponds to all possible vertex partitions, whose number is the Bell number B_n. If the entropy of each candidate partition is computed from a dense adjacency matrix, exact computation requires approximately $O(B_n n^2)$ time. For the general K-dimensional case, the search space further expands to all height-K partitioning trees. Let $N_K(n)$ denote the number of such trees. The exact computation then requires approximately $O(N_K(n) n^2)$ time, which is infeasible for large-scale graphs.

To make the computation practical, we adopt a greedy approximation strategy. The algorithm starts from the finest partition, where each vertex is treated as an individual module. At each step, it evaluates the entropy reduction caused by merging two modules and selects the merge that yields the largest decrease in structural entropy. After merging, the module volume, cut weight, and inter-module edge weights are updated accordingly. This process is repeated until no candidate merge can further reduce the structural entropy.The resulting partitioning structure provides an approximate multi-dimensional structural entropy:

$$\widehat{H}^K(\mathcal{G}) = H^{\widehat{\mathcal{T}}}(\mathcal{G}), \tag{A17}$$

where $\widehat{\mathcal{T}}$ denotes the partitioning tree obtained by the greedy procedure. For a graph with $n = |\mathcal{V}|$ vertices, our dense-matrix implementation maintains pairwise inter-module weights and uses a priority queue to select candidate merges. Thus, the greedy approximation has an overall time complexity of approximately $O(n^2 \log n)$ and a memory complexity of $O(n^2)$, which is substantially more tractable than exhaustive enumeration. Although the complexity remains quadratic in the number of vertices, it is acceptable in our setting because the Cognitive Bipartite Graph uses $m = 100$ Questioner-side nodes and $n = 10$ Solver-side nodes, corresponding to a 100×10 bi-adjacency matrix

 doi:10.20944/preprints202605.1514.v1

and a 110×110 undirected adjacency matrix. This approximation preserves the principle of structural entropy minimization while avoiding exhaustive enumeration of all possible partitioning trees.

Appendix G. Mechanistic Analysis of PI Control and Stability Scaling

Appendix G.1. Basic Principle of Proportional-Integral Control

Proportional-integral (PI) control is a classical feedback mechanism designed to drive a controlled variable $y(t)$ toward a prescribed target y_{tar} [42,43]. At time step t, the tracking error is defined as

$$e_t = y_{\text{tar}} - y(t). \tag{A18}$$

The PI control law specifies the control input as a weighted combination of the instantaneous error and the accumulated historical error:

$$u_t = K_p e_t + K_i \sum_{s=1}^{t-1} e_s, \tag{A19}$$

where $K_p > 0$ is the proportional gain and $K_i \geq 0$ is the integral gain.

The two components play complementary roles. The proportional term $K_p e_t$ provides an immediate response to the current deviation, so larger errors induce stronger control. However, proportional control alone can leave a non-zero steady-state error under persistent disturbances. The integral term $K_i \sum_s e_s$ accumulates historical deviations; even when each instantaneous error is small, a persistent bias increases the integral response over time until the steady-state deviation is reduced. Combining the two terms therefore provides both responsiveness and improved long-horizon tracking accuracy.

Appendix G.2. Control Loop in S-BGM

In S-BGM, the structural entropy $H(\mathcal{G}^{(t)})$ of the Cognitive Bipartite Graph is used as the system-level stability signal. The target trajectory $H_{\text{target}}^{(t)}$ defines the expected entropy level at training step t:

$$H_{\text{target}}^{(t)} = H(\mathcal{G}^{(0)}) - \gamma \log(1+t), \tag{A20}$$

where $\gamma \geq 0$ controls the rate of entropy decay. The adaptive scaling factor α_t represents the strength of stability regulation. The error signal is

$$e_t = H_{\text{target}}^{(t)} - H(\mathcal{G}^{(t)}), \tag{A21}$$

where a positive e_t indicates that the current interaction topology is more concentrated than the target trajectory. The PI controller computes

$$\alpha_t = \text{clip}\left(K_p e_t + K_i \sum_{s=1}^{t-1} e_s,\ 0,\ 1 \right). \tag{A22}$$

The resulting stability coefficient scales the normalized GRPO advantage by

$$\hat{A}_i' = (1 - \alpha_t)\hat{A}_i, \tag{A23}$$

and thereby adjusts the effective update magnitude in GRPO. The intended stabilizing effect is

$$\alpha_t \uparrow \Rightarrow \ \|\Delta\theta\| \downarrow \Rightarrow \ \Delta\pi_\theta \downarrow \Rightarrow \ \Delta\mathbf{B} \downarrow \Rightarrow \ \Delta H(\mathcal{G}) \downarrow . \tag{A24}$$

We next justify this mechanism and clarify its scope. The scaling is not intended to directly maximize structural entropy or force a uniform interaction topology. It acts as a stability regulation mechanism:

 doi:10.20944/preprints202605.1514.v1

when the graph indicates excessive concentration, it suppresses large policy updates, while the role-specific rewards and subsequent sampling continue to determine which outputs are preferred.

Appendix G.3. Theoretical Analysis

Lemma A4 (Lipschitz continuity of structural entropy under smoothed edge distributions)**.** *Let* $\mathbf{B}, \mathbf{B}' \in \mathbb{R}_{\geq 0}^{m \times n}$ *be two bi-adjacency matrices maintained by a sliding window of size* L*. Since some edges may have zero weight, we define a smoothed normalized edge distribution with a fixed* $\zeta > 0$*:*

$$\tilde{p}_{ij} = \frac{B_{ij} + \zeta}{\sum_{a,b}(B_{ab} + \zeta)}, \qquad \tilde{p}'_{ij} = \frac{B'_{ij} + \zeta}{\sum_{a,b}(B'_{ab} + \zeta)}. \tag{A25}$$

Because the window contains L *interactions, each entry is uniformly lower bounded by*

$$\tilde{p}_{ij}, \tilde{p}'_{ij} \geq p_{\min} := \frac{\zeta}{L + mn\zeta}. \tag{A26}$$

Then the structural entropy computed from the smoothed graph distributions satisfies

$$\left| H(\mathcal{G}) - H(\mathcal{G}') \right| \leq C_H \|\tilde{\mathbf{p}} - \tilde{\mathbf{p}}'\|_1, \tag{A27}$$

where $C_H > 0$ *is a constant depending only on* m, n, L*, and* ζ*.*

Proof sketch. For a fixed partitioning tree $\mathcal{T}$, the structural entropy $H^{\mathcal{T}}(\mathcal{G})$ is a finite sum of terms determined by smoothed normalized edge weights and induced subset volumes. The smoothing removes the logarithmic singularity at zero: all probabilities lie in $[p_{\min}, 1]$, so each logarithmic component is Lipschitz because $|p \log p - p' \log p'|$ is bounded by a constant multiple of $|p - p'|$ on this interval. Summing over at most mn edge states yields a finite Lipschitz constant depending on $|\log p_{\min}|$. Since $H^K(\mathcal{G})$ is the minimum over a finite collection of partitioning trees of height K, and the pointwise minimum of functions sharing a common Lipschitz constant is also Lipschitz, the stated bound follows. □

Effect of stability scaling on the GRPO update. Let g_{pg} denote the gradient contribution of the GRPO policy-improvement surrogate before applying the KL penalty. Since the advantage enters this surrogate linearly, replacing $\hat{A}_i$ by $\hat{A}'_i = (1 - \alpha_t)\hat{A}_i$ gives

$$g'_{\text{pg}} = (1 - \alpha_t) g_{\text{pg}}. \tag{A28}$$

Thus the stability coefficient directly scales the update component that moves the policy toward higher-reward rollouts, while leaving the within-group ranking induced by the role-specific reward unchanged. The KL term continues to anchor the policy to $\pi_{\theta_{\text{old}}}$, so increasing α_t reduces the magnitude of destabilizing policy movement rather than changing the reward preference itself.

Proposition 1: Suppression of structural drift. Assume that, within a local training region, the normalized interaction distribution has bounded sensitivity to the policy-improvement update:

$$\mathbb{E}\left[\|\Delta \mathbf{p}^{(t)}\|_1 \right] \leq L_\pi \eta \|g'_{\text{pg}}\|, \tag{A29}$$

where $L_\pi > 0$ summarizes the local sensitivity of generation, semantic assignment, uncertainty binning, smoothing, and sliding-window aggregation. Combining this sensitivity bound with Lemma A4 yields

$$\mathbb{E}\left[\left| H(\mathcal{G}^{(t+1)}) - H(\mathcal{G}^{(t)}) \right| \right] \leq C_H L_\pi \eta (1 - \alpha_t) \|g_{\text{pg}}\|. \tag{A30}$$

 doi:10.20944/preprints202605.1514.v1

Equivalently, for $\rho := C_H L_\pi \eta \|g_{\text{pg}}\|$,

$$\mathbb{E}\Big[|\Delta H^{(t)}|\Big] \le \rho(1-\alpha_t). \tag{A31}$$

This result should be viewed as a drift-control statement rather than a convergence theorem: larger α_t lowers an upper bound on the amount by which a single update can perturb the graph structure. It does not by itself prove that entropy will increase. Instead, it prevents large updates from further amplifying collapse once the system enters an unhealthy regime, preserving a more stable condition under which role-specific rewards, curriculum filtering, and subsequent sampling can recover interaction diversity.

Proposition 2: Response guarantee under persistent collapse. Suppose that, over an interval $[t_0, t_0+T]$, the system persistently stays below the target trajectory so that $e_t \ge \delta > 0$ at every step. Then the integral contribution satisfies

$$K_i \sum_{s=t_0}^{t_0+T} e_s \ge K_i \delta T. \tag{A32}$$

Thus, before clipping, the stability scaling strength grows at least linearly with the duration of the collapse signal.

Proof. Since $e_s \ge \delta > 0$ for every $s \in [t_0, t_0+T]$, monotonicity of summation gives $\sum_{s=t_0}^{t_0+T} e_s \ge \delta T$. Multiplying by $K_i \ge 0$ proves the claim. □

Proposition 2 explains the role of the integral term in our setting. For slow collapse across multiple self-evolution iterations, each instantaneous deviation may be small, so pure proportional feedback would only impose limited suppression. The integral term accumulates these persistent deviations and progressively increases the stability scaling strength, slowing down further structural drift and giving the role-specific objectives and future sampling rounds an opportunity to restore diversity. This property is aligned with the long-horizon iterative training regime of S-BGM.

Appendix G.4. The Final Loss Function of GRPO

The design of $\hat{A}'_i = (1-\alpha_t)\hat{A}_i$ is important because applying a group-level coefficient directly to rewards would be canceled by GRPO's within-group normalization. By scaling the normalized advantage instead, the stability signal remains active in the gradient. Since $(1-\alpha_t)$ is non-negative, this scaling preserves the relative ranking and signs of samples within each group. Its role is therefore not to change which outputs are preferred by the role-specific reward, but to adjust the update magnitude according to the global system state. When the interaction topology is stable, α_t remains small and the update is close to standard GRPO. When the graph entropy indicates excessive concentration, α_t increases and suppresses the update, preventing the system from further amplifying unstable interaction patterns. The final loss function is consistent with the standard GRPO:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\frac{1}{G}\sum_{i=1}^{G} \min\left(\frac{\pi_\theta(x_i)}{\pi_{\theta_{\text{old}}}(x_i)}\hat{A}'_i,\ \text{clip}\left(\frac{\pi_\theta(x_i)}{\pi_{\theta_{\text{old}}}(x_i)}, 1-\epsilon, 1+\epsilon\right)\hat{A}'_i\right) + \beta\,\text{KL}(\pi_\theta \| \pi_{\theta_{\text{old}}}). \tag{A33}$$

where θ denotes the trainable parameters of the current policy model, π_θ is the current policy, and $\pi_{\theta_{\text{old}}}$ is the old policy used for importance-ratio estimation. x_i denotes the i-th sampled output in a group, which corresponds to a generated question in Questioner training or a generated response in Solver training. G is the group size, ϵ is the clipping threshold in GRPO, and β controls the strength of the KL regularization term. The term $\text{KL}(\pi_\theta \| \pi_{\theta_{\text{old}}})$ penalizes excessive deviation from the old policy, while $\hat{A}'_i$ is the stability-scaled advantage defined above.

 doi:10.20944/preprints202605.1514.v1

Appendix H. Computational Efficiency

We further examine whether the proposed stability regularization introduces noticeable computational burden. Compared with R-Zero, S-BGM only adds a lightweight graph-related preprocessing stage in each co-evolution iteration. Specifically, generated questions are first encoded by `Qwen3-Embedding-0.6B`, and then assigned to semantic clusters with retrieval over the historical embedding pool. All other major training stages, including Questioner optimization, question generation and filtering, and Solver optimization, remain unchanged. Table A1 reports the wall-clock time of each component on an 8×H100 node.

Table A1. Wall-clock overhead per co-evolution iteration.

Component	R-Zero	S-BGM
Questioner GRPO training	110 min	110 min
Embedding, clustering and retrieving	N/A	5 min
Question generation & filtering	30 min	30 min
Solver GRPO training	48 min	48 min
Total per iteration	188 min	193 min
Overhead	N/A	**+2.66%**

As shown in Table A1, the additional cost of S-BGM is limited to the embedding, clustering, and retrieval stage, which takes about 5 minutes per iteration. This overhead is small compared with the total time spent on GRPO-based Questioner and Solver training, indicating that the proposed system-stability mechanism can be incorporated into the self-evolving pipeline with only minor additional computational cost.

Appendix I. Case Study

Following prior works that primarily analyze the quality and diversity of Questioner-generated training data, we focus our case study on the generated questions as a direct manifestation of self-evolution [8,12,13]. We provide a qualitative comparison between R-Zero and S-BGM to examine how different self-evolving strategies affect the generated training questions. While R-Zero often tends to concentrate on repetitive question patterns after several iterations, S-BGM is designed to preserve a broader Questioner–Solver interaction space through cognitive bipartite graph modeling and structural entropy modulation. By comparing their generated questions, this case study provides intuitive evidence that S-BGM can alleviate curriculum collapse and maintain a more diverse and informative training curriculum.

Case Study for R-Zero. Table A2 presents representative questions generated by R-Zero after five iterations. Although these questions differ in surface details, they follow highly similar templates: most of them define a recurrence relation, ask for a divisibility condition or a modular remainder, and repeatedly involve sums of sequence terms. This indicates that the Questioner gradually concentrates on a narrow family of problem patterns, rather than continuously exploring diverse reasoning structures. Such repetition suggests a typical form of model collapse, where the generated training data may become less informative for further improving the Solver.

Case Study for S-BGM. In contrast, Table A3 shows that S-BGM generates questions covering a broader range of mathematical topics and reasoning forms. The examples include number theory, Euclidean geometry, extremal graph theory, functional equations, and probability reasoning. These questions are not simple variants of a single template, but instead require different problem-solving strategies and knowledge structures. This suggests that S-BGM better preserves the diversity of the training curriculum during self-evolution, allowing the Questioner to provide more varied and informative learning signals for the Solver.

 doi:10.20944/preprints202605.1514.v1

Table A2. Examples of questions generated by R-Zero after five iterations.

ID	Questions
A	Consider a sequence of positive integers a_n defined by the recurrence relation $a_{n+1} = a_n^2 - 2a_n + 2$ for $n \geq 1$ with the initial term $a_1 = 3$. Let S_n denote the sum of the first n terms of the sequence, i.e., $S_n = a_1 + a_2 + \cdots + a_n$. Find the smallest positive integer k such that S_k is divisible by 1000.
B	Consider a sequence of positive integers $a_1, a_2, a_3, \ldots$ defined by the recurrence relation $a_{n+1} = a_n^2 - a_n + 1$ for $n \geq 1$, with the initial term $a_1 = 3$. Let S_n denote the sum of the first n terms of this sequence. Determine the smallest positive integer k such that S_k is divisible by 1000.
C	Consider a sequence of positive integers $a_1, a_2, \ldots, a_{10}$ where each term satisfies the recurrence relation $a_{n+1} = a_n^2 - a_n + 1$ for $n \geq 1$, with the initial term $a_1 = 2$. Let S be the sum of the first 10 terms of this sequence. Find the remainder when S is divided by 1000.
D	Consider a sequence of positive integers $a_1, a_2, a_3, \ldots$ defined by the recurrence relation $a_{n+1} = a_n^2 - a_{n-1} + 1$ for all $n \geq 2$, with initial terms $a_1 = 2$ and $a_2 = 3$. Define S_k as the sum of the first k terms of this sequence. Determine the smallest positive integer k such that S_k is divisible by 1000.
E	Consider the sequence of numbers defined by $a_1 = 2$, and for $n \geq 2$, $a_n = a_{n-1}^2 - a_{n-1} + 1$. Determine the smallest integer k such that a_k is divisible by 100.

Table A3. Examples of questions generated by S-BGM after 5 iterations.

ID	Questions
A	Find the smallest positive integer n such that $n^2 - 1$ is divisible by 24 and $n^3 + 1$ is divisible by 32.
B	Let ABC be an acute-angled triangle with circumcenter O. The line parallel to BC through O intersects AC and AB at P and Q, respectively. The line through O perpendicular to PQ intersects the side BC at M. The lines AM and PQ intersect at N. Prove that $\frac{ON}{OM} \geq \frac{1}{2}$.
C	What is the minimum number of edges that must be removed from a complete graph with 10 vertices so that no cycle of length 3 remains?
D	Let $f : \mathbb{R} \to \mathbb{R}$ be a differentiable function satisfying the functional equation $f(x+y) + f(x-y) = 2f(x)\cos(y)$ for all $x, y \in \mathbb{R}$. Prove that there exists a constant $c \in \mathbb{R}$ such that $f(x) = c\cos(x)$ for all $x \in \mathbb{R}$.
E	In a magical land, there are three types of coins: gold (G), silver (S), and bronze (B). A spell has been cast such that when two different types of coins are placed together, they transform into the third type. For example, a gold and a silver coin together transform into a bronze coin, a gold and a bronze coin transform into a silver coin, and a silver and a bronze coin transform into a gold coin. You start with 1 gold, 1 silver, and 1 bronze coin. If you perform the transformation process exactly 6 times, what is the probability that the final configuration of coins will include at least one gold coin?

Appendix J. Broader Impact

This work improves the stability of self-evolving LLMs by regulating the coupled evolution between a Questioner and a Solver. By reducing dependence on large-scale human annotations, stable self-evolving training may benefit domains where expert supervision is costly, such as mathematical reasoning, scientific problem solving, and code generation. The proposed Cognitive Bipartite Graph also offers an interpretable tool for monitoring interaction dynamics and detecting collapse into narrow or repetitive patterns. However, self-evolving LLMs may reinforce errors, biases, or spurious reasoning patterns because their training data and pseudo-labels are model-generated. Majority-vote pseudo-labeling may further encourage overconfident rather than reliable responses. Therefore, S-BGM should not be regarded as a complete safety guarantee. Future applications, especially in high-stakes domains, should incorporate human oversight, domain-specific validation, safety filtering, and more reliable pseudo-labeling mechanisms.

Appendix K. Limitation and Future Work

S-BGM focuses on stable self-evolution in settings where Questioner–Solver interactions can be discretized and guided by verifiable training signals. While our experiments mainly consider mathematical reasoning, the graph-based stability signal is general in form and may be extended to other verifiable domains, such as code generation and science question answering, with domain-specific node construction. Following existing self-evolving frameworks, S-BGM adopts majority voting to construct pseudo-labels and estimate Solver uncertainty. Although this provides a simple and effective training signal, future work could further enhance pseudo-label reliability by incorporating stronger answer verification, symbolic checking, or calibrated uncertainty estimation. Finally, the current framework is suitable for tasks with verifiable answers, especially those with a unique or standardized final answer. For open-ended generation tasks where multiple outputs may be acceptable, additional evaluation protocols would be needed before applying the same self-evolving loop.

 doi:10.20944/preprints202605.1514.v1

Algorithm A1 S-BGM Training

Require: Pretrained base LLM M_0; Embedding model $E()$; Iterations T; Target entropy decay rate γ; PI gains K_p, K_i; Similarity threshold τ; Filter threshold δ.
Ensure: Trained Questioner Q_θ and Solver S_ϕ

$Q_\theta \leftarrow M_0$, $S_\phi \leftarrow M_0$
Initialize Cognitive Bipartite Graph $\mathcal{G}$ via L $Q_\theta - S_\phi$ interactions and calculate initial structural entropy $H(\mathcal{G}^{(0)})$.
Historical question embedding pool $\mathcal{P}_{embedding} \leftarrow \emptyset$, error sum $E_{sum} \leftarrow 0$, training time step $t \leftarrow 0$.
for 1 **to** T **do**
  // 1. Questioner Training
  for each training step **do**
    Sample G questions $\{q_i\}_{i=1}^{G} \sim Q_\theta$.
    Solver S_ϕ rollouts R answers $\{a_{i,j}\}$ per question; compute consensus ratio $\hat{p}_i$.
    Update Graph: Update $\mathcal{G}$ with semantic cluster and uncertainty interval mapping.
    Compute Reward:
      $r_{\text{unc}}(q_i) = \exp(-(\hat{p}_i - 0.5)^2 / 2\sigma^2)$
      $r_{\text{rep}}(q_i) = \max(0, \cos(E(q_i), \text{Embedding in } (\mathcal{P}_{embedding})) - \tau)$
      $r_Q(q_i) = \max(0, r_{\text{unc}}(q_i) - r_{\text{rep}}(q_i))$
    System Stability Modulation:
      Compute structural entropy $H(\mathcal{G}^{(t)})$ from $\mathcal{G}$.
      $H^{(t)}_{\text{target}} = H(\mathcal{G}^{(0)}) - \gamma \log(1+t)$
      $e_t \leftarrow H^{(t)}_{\text{target}} - H^K(\mathcal{G}^{(t)})$
      $\alpha_t \leftarrow \text{clip}(K_p e_t + K_i E_{sum}, 0, 1)$
      $E_{sum} \leftarrow E_{sum} + e_t$
    Modulated GRPO:
      Calculate standard advantage $\hat{A}_i$ from $\{r_Q(q_i)\}$.
      $\hat{A}'_i = \hat{A}_i \cdot (1 - \alpha_t)$
    Update θ using $\mathcal{L}_{\text{GRPO}}$ with $\hat{A}'_i$.
    $\mathcal{P}_{embedding} \leftarrow \mathcal{P}_{embedding} \cup \{E(q_i)\}$, $t \leftarrow t+1$
  end for
  // 2. Solver Training
  Generate candidate pool $\mathcal{P}_{candidate}$ using updated Q_θ.
  Filter Curriculum: $\mathcal{D}_S = \{(q_i, \tilde{a}_i) : |\hat{p}_i - 0.5| \le \delta\}$.
  for each training step **do**
    Sample R answers $\{a_{i,j}\}$ for $q_i \in \mathcal{D}_S$.
    Compute consensus ratio $\hat{p}_i$ from the sampled answers.
    Update Graph: Update $\mathcal{G}$ with semantic cluster and uncertainty interval mapping.
    Compute Reward:
      $r_S(a_{i,j}; q_i) = \mathbf{1}[a_{i,j} = \tilde{a}_i]$.
    System Stability Modulation:
      Compute structural entropy $H(\mathcal{G}^{(t)})$ from $\mathcal{G}$.
      $H^{(t)}_{\text{target}} = H(\mathcal{G}^{(0)}) - \gamma \log(1+t)$
      $e_t \leftarrow H^{(t)}_{\text{target}} - H^K(\mathcal{G}^{(t)})$
      $\alpha_t \leftarrow \text{clip}(K_p e_t + K_i E_{sum}, 0, 1)$
      $E_{sum} \leftarrow E_{sum} + e_t$
    Modulated GRPO:
      Calculate standard advantage $\hat{A}_i$ from $\{r_S\}$.
      $\hat{A}'_i = \hat{A}_i \cdot (1 - \alpha_t)$.
    Update ϕ using $\mathcal{L}_{\text{GRPO}}$ with $\hat{A}'_i$.
    $t \leftarrow t + 1$.
  end for
end for
return Final Questioner Q_θ and Solver S_ϕ.

 doi:10.20944/preprints202605.1514.v1

References

1. He, T.; Li, H.; Chen, J.; Liu, R.; Cao, Y.; Liao, L.; Zheng, Z.; Chu, Z.; Liang, J.; Liu, M.; et al. Breaking the reasoning barrier a survey on llm complex reasoning through the lens of self-evolution. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2025, 2025, pp. 7377–7417.
2. Tao, Z.; Lin, T.E.; Chen, X.; Li, H.; Wu, Y.; Li, Y.; Jin, Z.; Huang, F.; Tao, D.; Zhou, J. A survey on self-evolution of large language models. *arXiv preprint arXiv:2404.14387* **2024**.
3. Fang, W.; Liu, S.; Zhou, Y.; Zhang, K.; Zheng, T.; Chen, K.; Song, M.; Tao, D. Serl: Self-play reinforcement learning for large language models with limited data. *arXiv preprint arXiv:2505.20347* **2025**.
4. Zhao, A.; Wu, Y.; Yue, Y.; Wu, T.; Xu, Q.; Lin, M.; Wang, S.; Wu, Q.; Zheng, Z.; Huang, G. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335* **2025**.
5. Huang, C.; Yu, W.; Wang, X.; Zhang, H.; Li, Z.; Li, R.; Huang, J.; Mi, H.; Yu, D. R-zero: Self-evolving reasoning llm from zero data. *arXiv preprint arXiv:2508.05004* **2025**.
6. He, B.; Zuo, Y.; Liu, Z.; Zhao, S.; Fu, Z.; Yang, J.; Qian, C.; Zhang, K.; Fan, Y.; Cui, G.; et al. How Far Can Unsupervised RLVR Scale LLM Training? *arXiv preprint arXiv:2603.08660* **2026**.
7. Gu, Y.; Pang, L.; Ye, X.; Wang, T.; Lin, J.; Priebe, C.E.; Aue, A. SIGMA: Scalable Spectral Insights for LLM Collapse. *arXiv preprint arXiv:2601.03385* **2026**.
8. Yu, W.; Liang, Z.; Huang, C.; Panaganti, K.; Fang, T.; Mi, H.; Yu, D. Guided self-evolving llms with minimal human supervision. *arXiv preprint arXiv:2512.02472* **2025**.
9. Liu, B.; Jin, C.; Kim, S.; Yuan, W.; Zhao, W.; Kulikov, I.; Li, X.; Sukhbaatar, S.; Lanchantin, J.; Weston, J. Spice: Self-play in corpus environments improves reasoning. *arXiv preprint arXiv:2510.24684* **2025**.
10. Zhou, Y.; Liang, Z.; Liu, H.; Yu, W.; Panaganti, K.; Song, L.; Yu, D.; Zhang, X.; Mi, H.; Yu, D. Evolving language models without labels: Majority drives selection, novelty promotes variation. *arXiv preprint arXiv:2509.15194* **2025**.
11. Yu, Z.; Su, Z.; Tao, L.; Wang, H.; Singh, A.; Yu, H.; Wang, J.; Gao, H.; Yuan, W.; Weston, J.E.; et al. RESTRAIN: From Spurious Votes to Signals—Self-Training RL with Self-Penalization. In Proceedings of the The Fourteenth International Conference on Learning Representations.
12. Li, G.; He, J.; Wang, S.; Zhang, D.; Liu, R.; Zhang, R.; Yao, Z.; Fang, J.; Guo, H.; Wang, J. R-Diverse: Mitigating Diversity Illusion in Self-Play LLM Training. *arXiv preprint arXiv:2602.13103* **2026**.
13. Mishra, V. Preventing Curriculum Collapse in Self-Evolving Reasoning Systems. *arXiv preprint arXiv:2603.13309* **2026**.
14. Fan, S.; Ye, X.; Lin, Y. DARC: Decoupled Asymmetric Reasoning Curriculum for LLM Evolution. *arXiv preprint arXiv:2601.13761* **2026**.
15. Wortsman, M.; Ilharco, G.; Gadre, S.Y.; Roelofs, R.; Gontijo-Lopes, R.; Morcos, A.S.; Namkoong, H.; Farhadi, A.; Carmon, Y.; Kornblith, S.; et al. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Proceedings of the International conference on machine learning. PMLR, 2022, pp. 23965–23998.
16. Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388* **2025**.
17. Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; Steinhardt, J. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874* **2021**.
18. Zou, D.; Peng, H.; Huang, X.; Yang, R.; Li, J.; Wu, J.; Liu, C.; Yu, P.S. Se-gsl: A general and effective graph structure learning framework through structural entropy optimization. In Proceedings of the Proceedings of the ACM web conference 2023, 2023, pp. 499–510.
19. Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300* **2024**.
20. Zhang, J.; Zuo, C. Grpo-lead: A difficulty-aware reinforcement learning approach for concise mathematical reasoning in language models. In Proceedings of the Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, 2025, pp. 5642–5665.
21. Li, Z.; Chang, Y.; Zhou, Y.; Wu, X.; Liang, Z.; Sung, Y.Y.; Boyd-Graber, J.L. Semantically-aware rewards for open-ended r1 training in free-form generation. *arXiv preprint arXiv:2506.15068* **2025**.
22. Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168* **2021**.
23. Lewkowycz, A.; Andreassen, A.; Dohan, D.; Dyer, E.; Michalewski, H.; Ramasesh, V.; Slone, A.; Anil, C.; Schlag, I.; Gutman-Solo, T.; et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems* **2022**, *35*, 3843–3857.

 doi:10.20944/preprints202605.1514.v1

24. He, C.; Luo, R.; Bai, Y.; Hu, S.; Thai, Z.; Shen, J.; Hu, J.; Han, X.; Huang, Y.; Zhang, Y.; et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In Proceedings of the Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024, pp. 3828–3850.
25. Wang, Y.; Ma, X.; Zhang, G.; Ni, Y.; Chandra, A.; Guo, S.; Ren, W.; Arulraj, A.; He, X.; Jiang, Z.; et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems* **2024**, *37*, 95266–95290.
26. Du, X.; Yao, Y.; Ma, K.; Wang, B.; Zheng, T.; Zhu, K.; Liu, M.; Liang, Y.; Jin, X.; Wei, Z.; et al. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739* **2025**.
27. Kazemi, M.; Fatemi, B.; Bansal, H.; Palowitch, J.; Anastasiou, C.; Mehta, S.V.; Jain, L.K.; Aglietti, V.; Jindal, D.; Chen, Y.P.; et al. Big-bench extra hard. In Proceedings of the Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2025, pp. 26473–26501.
28. Liu, B.; Guertler, L.; Yu, S.; Liu, Z.; Qi, P.; Balcells, D.; Liu, M.; Tan, C.; Shi, W.; Lin, M.; et al. Spiral: Self-play on zero-sum games incentivizes reasoning via multi-agent multi-turn reinforcement learning. *arXiv preprint arXiv:2506.24119* **2025**.
29. Kuba, J.G.; Gu, M.; Ma, Q.; Tian, Y.; Mohan, V.; Chen, J. Language self-play for data-free training. *arXiv preprint arXiv:2509.07414* **2025**.
30. Yue, Z.; Upasani, K.; Yang, X.; Ge, S.; Nie, S.; Mao, Y.; Liu, Z.; Wang, D. Dr. Zero: Self-Evolving Search Agents without Training Data. *arXiv preprint arXiv:2601.07055* **2026**.
31. Chen, X.; Lu, J.; Kim, M.; Zhang, D.; Tang, J.; Piché, A.; Gontier, N.; Bengio, Y.; Kamalloo, E. Self-evolving curriculum for llm reasoning. *arXiv preprint arXiv:2505.14970* **2025**.
32. Tan, Z.; Li, D.; Wang, S.; Beigi, A.; Jiang, B.; Bhattacharjee, A.; Karami, M.; Li, J.; Cheng, L.; Liu, H. Large language models for data annotation and synthesis: A survey. In Proceedings of the Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, 2024, pp. 930–957.
33. Shumailov, I.; Shumaylov, Z.; Zhao, Y.; Gal, Y.; Papernot, N.; Anderson, R. The curse of recursion: Training on generated data makes models forget. *Nature* **2024**.
34. Alemohammad, S.; Casco-Rodriguez, J.; Luzi, L.; Humayun, A.I.; LeJeune, D.; Tiwary, A.; Baraniuk, R. Self-sustaining games: The rise of generative ai model collapse. *arXiv preprint arXiv:2307.01850* **2023**.
35. Pugh, J.K.; Soros, L.B.; Stanley, K.O. Quality Diversity: A New Frontier for Evolutionary Computation. *Frontiers in Robotics and AI* **2016**, *3*, 40.
36. Yue, Y.; Chen, Z.; Lu, R.; Zhao, A.; Wang, Z.; Song, S.; Huang, G. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837* **2025**.
37. Liang, X.; Li, Z.; Gong, Y.; Shen, Y.; Wu, Y.N.; Guo, Z.; Chen, W. Beyond pass@ 1: Self-play with variational problem synthesis sustains rlvr. *arXiv preprint arXiv:2508.14029* **2025**.
38. Liu, Z.; Chen, C.; Li, W.; Qi, P.; Pang, T.; Du, C.; Lee, W.S.; Lin, M. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783* **2025**.
39. Aghajanyan, A.; Gupta, S.; Zettlemoyer, L. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In Proceedings of the 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), 2021, pp. 7319–7328.
40. Valeriani, L.; Doimo, D.; Cuturello, F.; Laio, A.; Ansuini, A.; Cazzaniga, A. The geometry of hidden representations of large transformer models. *Advances in Neural Information Processing Systems* **2023**, *36*, 51234–51252.
41. Li, A.; Pan, Y. Structural information and dynamical complexity of networks. *IEEE Transactions on Information Theory* **2016**, *62*, 3290–3339.
42. Abo-Elkhair, E.; Abu-Elanien, A.E.; Mahmoud, G.M.; ElRefaie, H.B. Enhancing PI control in microgrids using machine-learning techniques. *Scientific Reports* **2025**, *15*, 38129.
43. Yang, K.; Xu, X.; Chen, Y.; Liu, W.; Lyu, J.; Lin, Z.; Ye, D.; Yang, S. EntroPIC: Towards Stable Long-Term Training of LLMs via Entropy Stabilization with Proportional-Integral Control. *arXiv arXiv:2511.15248* **2025**.