Tandem: Riding Together with Large and Small Language Models for Efficient Reasoning

Anonymous ACL submission

Abstract

Recent advancements in large language models (LLMs) have catalyzed the rise of reasoning-intensive inference paradigms, where models perform explicit step-by-step reasoning before generating final answers. While such approaches improve answer quality and interpretability, they incur substantial computational overhead due to the prolonged generation sequences. In this paper, we propose Tandem, a novel collaborative framework that synergizes large and small language models (LLMs and SLMs) to achieve high-quality reasoning with significantly reduced computational cost. Specifically, the LLM serves as a strategic coordinator, efficiently generating a compact set of critical reasoning insights. These insights are then used to guide a smaller, more efficient SLM in executing the full reasoning process and delivering the final response. To balance efficiency and reliability, Tandem introduces a cost-aware termination mechanism that adaptively determines when sufficient reasoning guidance has been accumulated, enabling early stopping of the LLM's generation. Experiments across multiple public benchmarks demonstrate that Tandem reduces computational costs by approximately 40% compared to standalone LLM reasoning, while achieving superior or competitive performance. https://anonymous.4open.science/r/Ensemble-Hub-0FD8

1 Introduction

Recent advances in large language models (LLMs) have shifted their role from simple text generators to sophisticated reasoning systems capable of solving complex problems (Ke et al., 2025). A key development in this shift is the emergence of a thinking paradigm, which extends earlier chain-of-thought (Wei et al., 2022) prompting. In this paradigm, exemplified by models such as DeepSeek-R1 (DeepSeek-AI, 2025), the internal reasoning process is explicitly externalized and structurally decoupled from final answer generation. This design substantially improves interpretability and faithfulness, and has led to state-of-the-art performance on challenging mathematical and scientific reasoning tasks.

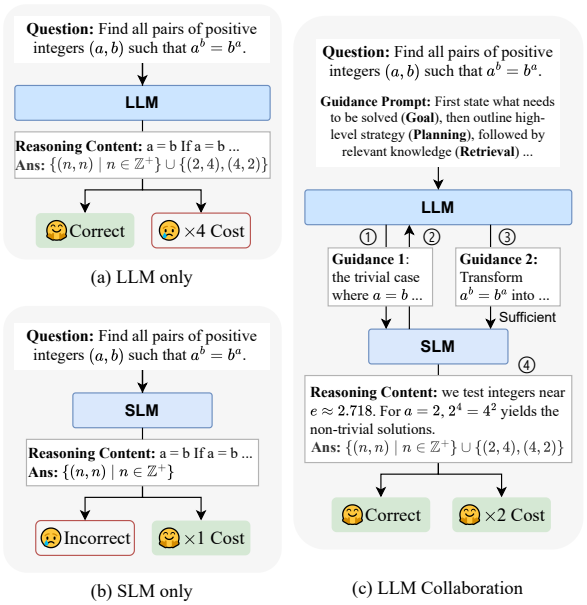

Figure 1: Comparison of reasoning-inference strategies: (a) LLM-only. (b) SLM-only. (c) LLM-SLM collaboration, where the LLM provides guidance and the SLM completes the reasoning.

However, these reasoning improvements come at a substantial computational cost. Thinking models routinely generate reasoning chains spanning thousands of tokens, typically 5–10 times longer than conventional LLM outputs (Chen et al., 2025a). This extended reasoning leads to a dramatic increase in both inference latency and operational expenses. Such overhead poses a major obstacle to real-world deployment, where applications often require real-time responses or operate under strict budget constraints (Ding et al., 2024; Ferrag et al., 2025). This raises a central question: *how can we preserve the benefits of explicit thinking*

while making it computationally efficient enough for practical deployment?

Since the inference cost of thinking LLMs is dominated by the length of generated reasoning (Han et al., 2025), prior work has attempted to reduce this overhead through reinforcement fine-tuning (RFT), encouraging the model to solve problems with shorter reasoning traces (Fang et al., 2025). However, such approaches still have limitations: (1) they require continual training of the LLM, and may degrade the model's general capabilities (Yue et al., 2025), and (2) they are inapplicable to closed-source models where only API access is available. These constraints motivate us to explore alternatives without modifying the LLM.

To address this challenge, we propose a collaborative paradigm between large and small language models (LLMs and SLMs) as a promising path toward cost-effective reasoning. As shown in Figure 1 (a) and (b), relying solely on an LLM yields correct answers but incurs high computational cost, whereas an SLM alone is efficient yet prone to errors. Our key insight is to combine the strengths of both: the LLM acts as a strategic mentor, providing high-level reasoning guidance, while the SLM serves as an agile intern, efficiently executing the detailed reasoning steps and generating the final response. This division of labor enables accurate reasoning at a fraction of the computational cost as in Figure 1 (c).

Based on this insight, we present Tandem, a novel collaborative reasoning framework that formalizes this dynamic as a mentor-intern architecture (Abell et al., 1995). In Tandem, the mentor LLM generates lightweight, early-stage thinking insights, which are then used to guide the intern SLM in constructing the full response. Our design is inherently aligned with the modular nature of human cognition as described in the ACT-R cognitive architecture (Anderson, 2007). Specifically, we decompose the LLM's reasoning into four **thinking insights**—Goal, Planning, Retrieval, and Action—which correspond to the essential stages of problem-solving: understanding objectives, outlining strategies, gathering knowledge, and executing logic (Yue, 2025). By transferring only high-level insights instead of full reasoning chains, Tandem retains the cognitive depth of the LLM while significantly reducing computational overhead by delegating the remaining reasoning tasks to the SLM. To further enhance efficiency, Tandem introduces a **cost-aware judgment mechanism** that evaluates whether the current insights are adequate for the SLM to complete the task, allowing for adaptive control over the amount of guidance provided by the LLM.

Our main contributions can be summarized in the following three aspects:

- We propose a novel collaborative reasoning paradigm inspired by the mentor–intern relationship, enabling efficient collaboration between large and small models for reasoning tasks.
- We propose the Tandem framework, which extracts key thinking insights from the LLM to guide the SLM, with a cost-aware judgment mechanism that enables the SLM to adaptively control the insight generation process.
- Experiments on MATH datasets show that a 32B–7B language model collaboration achieves 2.56% higher accuracy than the 32B LLM alone, while requiring only 59% of its computational cost.

2 Method

2.1 Framework Overview

As shown in Figure 2, Tandem establishes a collaborative reasoning process between a large language model (LLM) and a small language model (SLM). Given a question Q, the LLM generates reasoning insights through three[1] sequential stages of increasing cognitive effort, with later stages providing progressively richer content. Insights accumulated up to stage t are aggregated into an insight set $\mathcal{I}^t$. After each stage, the SLM computes uncertainty-based features from the pair $(Q, \mathcal{I}^t)$, and a dedicated classifier $\mathcal{C}$ determines whether the current guidance suffices for response completion. If the classifier predicts sufficiency, the LLM terminates generation and the SLM produces the final answer A; otherwise, the LLM proceeds to the subsequent stage for more in-depth insights.

2.2 Thinking Insight Generation

A straightforward approach to constructing the collaborative thinking-inference paradigm involves directly feeding the LLM's reasoning chains to the SLM to generate responses. However, LLMs typically produce lengthy reasoning traces that include trial-and-error explorations and verbose justifications. Delivering such unprocessed chains to the SLM incurs significant computational overhead

[1] In this paper, we illustrate Tandem through three stages, which can also be extended to include additional stages.

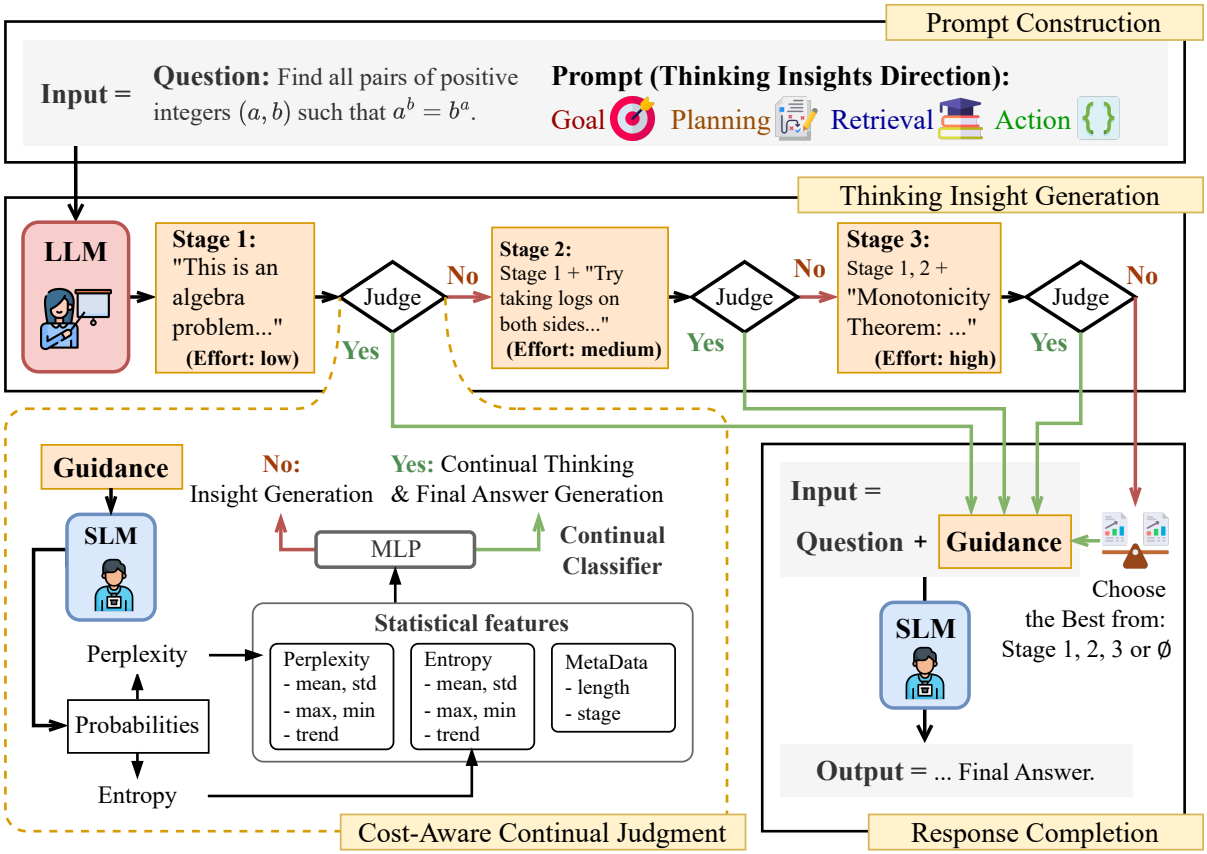

Figure 2: Workflow of the Tandem framework. The LLM generates pre-defined thinking insights (Goal, Planning, Retrieval, Action) across three thinking effort levels, while the SLM performs cost-aware continual judgment based on perplexity and entropy to decide when reasoning is sufficient and then completes the final response.

and may exceed its comprehension capacity. To mitigate this burden, we extract only the core reasoning components essential for guiding SLMs.

Inspired by the modular nature of human cognition, where cognitive processes are decomposed into distinct functional modules (Anderson, 2007), and by established pipelines for question-answering processes in LLM-based agents (Yue, 2025), we identify and design four types of thinking insights that capture the essential components of effective problem solving:

1. **Goal**: Defines the ultimate objective or question to be solved, clarifying what the model aims to achieve through reasoning.
2. **Planning**: Outlines the high-level reasoning strategy, including decomposition of subproblems and selection of solution paths.
3. **Retrieval**: Involves recalling or gathering relevant knowledge, facts, or contextual information necessary for problem solving.
4. **Action**: Executes concrete reasoning steps, calculations, or logical operations.

These components form the basis of our structured prompt p (detailed in Appendix D) that instructs the LLM to generate insights sequentially. Formally, given a question Q, the LLM $\mathcal{M}_L$ generates the thinking insights accompanied with the prompt p:

$$\mathcal{I} = \mathcal{M}_L(Q, p) \tag{1}$$

2.3 Cost-Aware Continual Judgment

Problems of varying difficulty require differing levels of reasoning guidance. To avoid wasting computational resources on simple problems while ensuring sufficient support for more complex ones, we decompose the reasoning process into stages and introduce a cost-aware judgment mechanism. This mechanism evaluates whether the current reasoning content is sufficient for the SLM to generate a reliable response, allowing the LLM to terminate reasoning early and skip unnecessary stages.

Insight Cutoff. To balance reasoning quality and computational cost, we divide the whole thinking process into several successive stages with increasing **effort levels**, e.g., three in this paper, namely

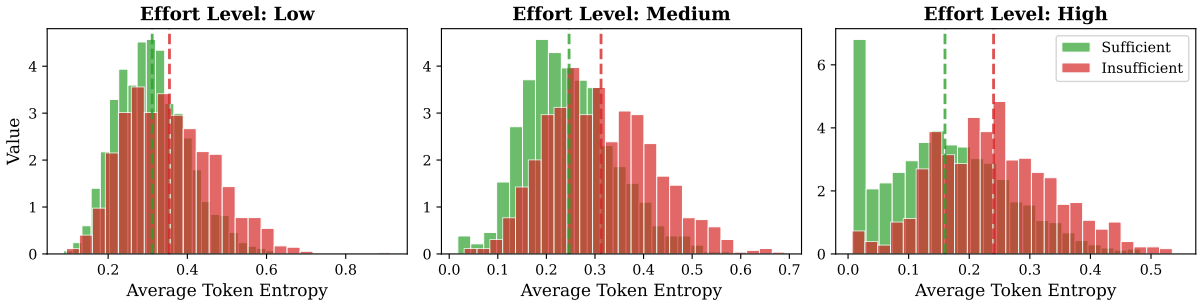

Figure 3: Entropy distribution comparison between sufficient (correct answer) and insufficient (incorrect answer) guidance across three effort levels. Higher effort levels show clearer separation.

low, medium, and high. Each stage generates insights covering all four types (Goal, Planning, Retrieval, Action), but with progressively larger token budgets and greater depth of analysis. Let $\Delta\mathcal{I}^t$ denote the newly generated insights at stage t; the cumulative insight set is constructed as:

$$\mathcal{I}^0 = \emptyset, \quad \mathcal{I}^t = \mathcal{I}^{t-1} \oplus \Delta\mathcal{I}^t \quad (t = 1, 2, 3) \tag{2}$$

where $\oplus$ denotes sequential concatenation of insights. $\mathcal{I}^0 = \emptyset$ represents the baseline case where no reasoning insight is provided. The insight of the last stage $\mathcal{I}^3$ is equivalent to the full response $\mathcal{I}$ as defined in Equation (1). This progressive design enables adaptive allocation of reasoning guidance: simple problems may terminate early at low effort (e.g., $\mathcal{I}^1$), while complex ones proceed to higher stages for richer support.

Confidence Measurement. To assess the sufficiency of the specific reasoning stage, we use the SLM's token-level probability distribution as the confidence signal. Previous studies have found that lower perplexity indicates the SLM finds insights linguistically familiar, while lower entropy reflects higher predictive stability (Kuhn et al., 2023).

Our empirical analysis, presented in Figure 3, also reveals a consistent pattern: the SLM produces lower output entropy under guidance that leads to correct answers—compared to incorrect ones—with a more pronounced separation observed at higher reasoning effort levels. This finding motivates the use of perplexity and entropy as diagnostic indicators of reasoning sufficiency.

Concretely, for each stage t, we concatenate Q and $\mathcal{I}^t$ to form the input sequence x^t for the SLM $\mathcal{M}_S$, i.e., $x^t = Q \oplus \mathcal{I}^t$. Let x_i^t denote the i-th token in x^t; the corresponding predictive distribution at position i is computed as:

$$P_i^t = P_{\mathcal{M}_S}(\cdot \mid x_{<i}^t) \tag{3}$$

From this distribution, we compute per-token perplexity and entropy:

$$\mathrm{PPL}_i^t = \exp(-\log P_i^t(x_i^t)) \tag{4}$$

$$\mathrm{H}_i^t = -\sum_{v \in \mathcal{V}} P_i^t(v) \log P_i^t(v) \tag{5}$$

where $\mathcal{V}$ denotes the vocabulary.

Sufficiency Classification. Leveraging the distributional discrepancy between sufficient and insufficient guidance in Figure 3, we construct a classifier to retroactively assess reasoning sufficiency based on distributional statistics. Let n denote the number of tokens in the SLM input x^t. We denote the per-token PPL and entropy sequences as:

$$\mathbf{PPL}^t = \{\mathrm{PPL}_i^t\}_{i=1}^n, \quad \mathbf{H}^t = \{\mathrm{H}_i^t\}_{i=1}^n \tag{6}$$

Based on these sequences, we construct a distributional feature vector:

$$\mathbf{f}^t = \left[\phi(\mathbf{PPL}^t), \phi(\mathbf{H}^t), \Delta_{\mathrm{PPL}}, \Delta_{\mathrm{H}}, n\right] \tag{7}$$

where $\phi(\cdot)$ computes distributional statistics (mean, standard deviation, median, max, min, 25th/75th percentiles), and Δ denotes trend indicators measuring the difference between the last 20 and first 20 tokens, capturing whether confidence increases or decreases as the model processes the insight. Details are provided in Appendix A.

A classifier $\mathcal{C}$, exemplified as a multilayer perceptron (MLP), is trained on these features with binary labels indicating whether the SLM produces a correct answer under the given guidance. At inference time, the classifier outputs:

$$s^t = \mathcal{C}(\mathbf{f}^t) \in [0, 1] \tag{8}$$

where s^t denotes the sufficiency score. A binary decision is made as $y^t = \mathbf{1}[s^t > \tau^t]$, with threshold

τ^t optimized per effort level on a held-out validation set. We also explore alternative input representations (e.g., hidden states) and whether fine-tuning the SLM benefits classification in Appendix E.

2.4 Response Completion

Guided by the sufficiency assessments from the judgment mechanism, the framework dynamically determines when to halt the LLM's reasoning and transition control to the SLM, which then continues the reasoning process and generates the final answer. Specifically, at stage t, if the sufficiency decision $y^t = 1$, the SLM generates the final answer A using the question and current insights:

$$A = \mathcal{M}_S(Q \oplus \mathcal{I}^t) \tag{9}$$

If all stages yield $y^t = 0$, we fall back to selecting the configuration with the highest sufficiency score:

$$t^* = \arg\max_{0 \le t \le 3} s^t \tag{10}$$

where s^0 is computed by applying the same classifier $\mathcal{C}$ to distributional features extracted when the SLM processes Q alone. The final answer is then generated as:

$$A = \mathcal{M}_S(Q \oplus \mathcal{I}^{t^*}) \tag{11}$$

3 Experiments

To evaluate the effectiveness of Tandem, the following research questions (RQs) are investigated:

- **RQ1:** How does Tandem perform compared to a single LLM?
- **RQ2:** Does Tandem generalize to collaborations between models from different families?
- **RQ3:** What are the scaling behaviors of LLM collaboration with respect to guidance length and large–small model size combinations?
- **RQ4:** Can Tandem utilize API-accessible LLM?

3.1 Experiment Settings

Datasets. The experiments are conducted on two English mathematical reasoning benchmarks: MATH (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021). The MATH dataset contains 12.5K competition-level problems across seven subjects with five difficulty levels, split into 7.5K training and 5K test samples. The distribution

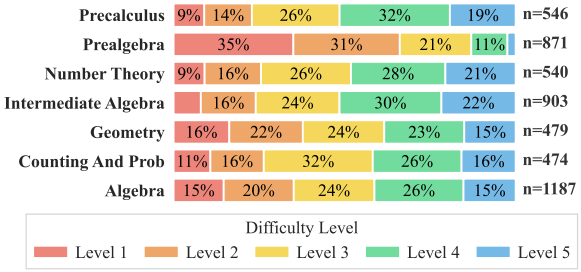

Figure 4: Sample distribution analysis of the MATH dataset (Hendrycks et al., 2021) across seven subjects and five difficulty levels. Difficulty levels range from 1 (easiest) to 5 (hardest).

of the test set is shown in Figure 4. GSM8K consists of 8.5K grade school math problems, split into 7.5K training and 1K test samples. Both datasets demand multi-step reasoning, making them suitable for evaluating Tandem.

Evaluation Metrics. We evaluate model performance using three complementary metrics: (1) **Accuracy**, defined as the percentage of correctly solved problems under the standard evaluation protocol of each dataset; (2) **Inference Length**, defined as the total number of tokens generated during inference. Let L_L and L_S denote the numbers of tokens generated by the LLM and SLM, respectively; the overall inference length is $L_L + L_S$; and (3) **Computational Cost**, approximated by the total TFLOPs incurred by both models. Let $|\theta_L|$ and $|\theta_S|$ denote the parameter counts of the LLM and SLM, respectively. The cost is computed as

$$\text{Cost} = \frac{2}{10^{12}} \left(|\theta_L| L_L + |\theta_S|(L_L + L_S)\right) \tag{12}$$

Implementation details. For the **LLMs**, we consider two model families: DeepSeek-R1 (DeepSeek-AI, 2025) and Qwen3 (Yang et al., 2025). We use DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Qwen-32B (denoted as DeepSeek-7B and DeepSeek-32B), together with Qwen3-8B and Qwen3-32B. All LLMs operate in the thinking mode with deterministic generation settings: temperature = 0, top-p = 1.0, and no frequency penalty. To investigate the cost-performance trade-off, we empirically evaluate three fixed thinking lengths: 100, 500, and 1,000 tokens (corresponding to low, medium, and high effort levels). The maximum output length for complete answers is set to 8,192 tokens.

For the **continual classifier**, we train it on the training splits of both datasets; for MATH, we train

Series	Model	Metric	Algebra	Counting	Geometry	Intermediate Num	Prealgebra	Precalc	Average
Single	7B	Acc.	93.09	75.95	65.34	60.13	76.85	89.55	62.45	77.14
		Len.	2,194	2,786	2,854	3,218	2,891	2,084	3,098	2,732
		Cost	**30.71**	**39.00**	**39.96**	**45.06**	**40.47**	**29.17**	**43.37**	**38.25**
	32B	Acc.	94.78	82.07	68.89	64.45	85.00	91.04	67.22	80.90
		Len.	**2,064**	**2,676**	**2,816**	**3,162**	**2,678**	**2,013**	**3,003**	**2,630**
		Cost	132.13	171.26	180.26	202.36	171.40	128.84	192.20	168.35
Baseline	7B+32B (low)	Acc.	94.27	77.43	68.27	60.24	80.74	90.36	65.38	78.74
		Len.	2,155	2,803	2,912	3,250	2,860	2,060	3,106	2,735
		Cost	36.64	45.71	47.24	51.97	46.51	35.30	49.95	44.76
	7B+32B (medium)	Acc.	93.93	79.96	69.94	63.90	83.33	91.16	67.40	80.36
		Len.	2,205	2,959	3,034	3,380	2,963	2,116	3,312	2,853
		Cost	62.90	73.49	74.48	79.39	73.53	61.53	78.43	71.96
	7B+32B (high)	Acc.	95.53	82.49	72.86	67.55	88.52	92.54	71.61	83.18
		Len.	2,178	3,057	3,190	3,580	2,980	2,119	3,406	2,930
		Cost	93.87	106.77	108.39	114.07	105.47	92.10	111.64	104.62
Tandem	7B+32B	Acc.	**96.04** (+2%)	**82.70** (+1%)	**73.07** (+7%)	**67.77** (+6%)	**88.70** (+5%)	**92.65** (+2%)	**71.98** (+8%)	**83.46** (+4%)
		Len.	2,175 (+6%)	3,057 (+15%)	3,136 (+12%)	3,549 (+13%)	2,976 (+12%)	2,118 (+6%)	3,402 (+14%)	2,916 (+11%)
		Cost	86.27 (-35%)	105.78 (-39%)	96.26 (-47%)	107.02 (-48%)	99.86 (-42%)	91.92 (-29%)	110.94 (-43%)	99.72 (-41%)

Table 1: Performance comparison of different model configurations on the MATH dataset (Hendrycks et al., 2021) using DeepSeek-7B and -32B. We report accuracy (%), average inference length (tokens), and computational cost (TFLOPs). Bold values denote the best performance and underlined values denote the second-best performance for each subject. Percentage changes are computed relative to DeepSeek-32B.

a separate classifier for each subject. A sample is labeled as *sufficient* ($y = 1$) if the SLM produces the correct answer given the current insights, and as *insufficient* ($y = 0$) otherwise. We adopt a stratified 70/30 train–validation split with random seed 42. The classifier is a two-hidden-layer MLP (64 and 32 units) with ReLU activations and dropout ($p = 0.3$), optimized with Adam (lr $= 10^{-4}$) for up to 3 epochs, using early stopping based on validation accuracy. The threshold τ_e for each effort level is determined by grid search over the range $[0.05, 0.95]$ with step size 0.05 on the training set.

3.2 Overall Performance Comparison (RQ1)

As shown in Table 1, the experiment compares the results of single model inference, fixed-length thinking baselines, and our Tandem approach across seven mathematical subjects.

Thinking Length Analysis. Overall, incorporating the 32B model's initial thinking improves the 7B model's performance across all subjects. Notably, with sufficient guidance (high effort level), the collaboration not only surpasses the 7B baseline but also exceeds the 32B model's standalone performance on most subjects. This suggests a synergistic effect where the combination of the LLM's high-level reasoning and the SLM's execution capability achieves results beyond what either model can accomplish alone.

Tandem Performance. Tandem effectively identifies beneficial thinking processes and adaptively allocates computational resources. It achieves the best accuracy across all subjects, outperforming both the 7B and 32B standalone models. Compared to the 32B mentor model, Tandem achieves 2.56 percentage points higher accuracy while requiring only 59% of the computational cost. These results demonstrate that our cost-aware judgment mechanism successfully balances performance and efficiency through dynamic resource allocation.

3.3 Cross-Family Generalization (RQ2)

To evaluate whether Tandem generalizes beyond a single model family, we test collaborations between DeepSeek and Qwen3. Table 2 presents results on both the MATH and GSM8K datasets.

Cross-family collaboration is effective. Our experiments show that cross-family collaboration

Model	MATH			GSM8K		
	Acc.	Len.	Cost	Acc.	Len.	Cost
Single Models						
① Qwen3-8B	60.86	3,197	51.15	89.61	1,991	31.86
② Qwen3-32B	69.50	3,022	193.41	94.01	1,625	104.00
③ DeepSeek-7B	76.92	2,661	**37.25**	87.11	1,124	**15.74**
④ DeepSeek-32B	80.76	2,560	163.84	94.47	**1,049**	67.14
Collaboration						
①+②	64.42	3,160	96.69	94.77	1,727	57.00
①+④	63.72	3,055	94.71	95.22	1,360	44.18
③+②	79.96	**2,520**	58.06	94.62	1,488	76.87
③+④	**83.34**	2,820	97.95	**95.45**	1,059	52.66

Table 2: Performance of cross-family collaborations on the MATH (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021) datasets. Bold and underlined values denote the best and second-best performance, respectively.

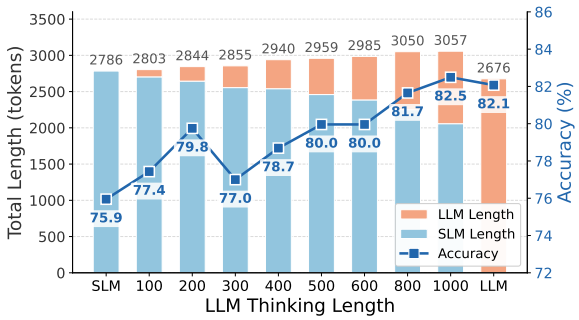

Figure 5: Accuracy and token length breakdown on the Counting & Probability subset of MATH (Hendrycks et al., 2021). The stacked bars show the composition of total length (SLM in blue, LLM in orange), while the line indicates accuracy.

yields consistent performance gains over SLM-only baselines. For instance, DeepSeek-7B paired with Qwen3-32B achieves 79.96% accuracy on MATH, surpassing both DeepSeek-7B alone (76.92%) and Qwen3-32B alone (69.50%), while reducing inference cost to less than one-third of using Qwen3-32B. This indicates that the structured insights generated by Tandem are sufficiently general to be understood and utilized across different families.

Capability alignment matters. The magnitude of improvement depends on the capability gap between models. When the SLM is relatively weak (e.g., Qwen3-8B), collaboration with DeepSeek-32B improves MATH accuracy by only 2.86% and fails to reach the LLM's standalone performance. We attribute this limitation to a comprehension bottleneck: weaker SLMs may lack the capacity to fully interpret abstract reasoning guidance, causing them to underutilize or misapply the LLM's insights. This suggests that effective collaboration requires reasonable capability alignment between mentor and intern models.

3.4 Scaling Law of LLM Collaboration (RQ3)

Guidance Length. To understand how guidance length impacts performance, we analyze the trade-off between total token length and accuracy when using DeepSeek-32B and -7B. As shown in Figure 5, accuracy generally improves as guidance length increases, eventually surpassing the pure LLM baseline. Notably, even with minimal guidance (200 tokens), the collaboration substantially outperforms the SLM-only baseline. We attribute this to the structured nature of insights: even brief guidance provides goal clarification and high-level planning that helps the SLM avoid dead-ends.

SLM	LLM	Low		Medium		High	
		Acc.	Len.	Acc.	Len.	Acc.	Len.
1.5B	–	59.92	2,898	–		–	
1.5B	7B	59.28	3,051	58.23	3,446	60.55	3,925
1.5B	14B	61.13	2,961	63.50	2,866	65.82	2,801
1.5B	32B	60.97	2,956	62.66	2,877	67.09	2,854
7B	–	75.95	2,786	–		–	
7B	14B	74.68	2,685	75.74	2,669	75.11	2,690
7B	32B	77.43	2,803	79.96	2,959	82.49	3,057
14B	–	79.96	2,613	–		–	
14B	32B	80.17	2,581	79.48	2,628	80.38	2,665
32B	–	82.07	2,675	–		–	

Table 3: Performance of LLM collaboration with different model size combinations on Counting and Probability with DeepSeek family. Rows with "–" as LLM denote single-model baselines. Low, Medium, and High correspond to the effort levels of LLM guidance.

However, the performance curve exhibits fluctuations across different guidance lengths, suggesting that the optimal amount of guidance is problem-dependent. Simple problems may be solved with brief hints, while complex ones require detailed action steps. This observation motivates our cost-aware judgment mechanism, which dynamically determines the appropriate level of guidance rather than relying on a fixed thinking budget.

Model Size. To examine the role of model size, we evaluate different combinations of LLM and SLM sizes. Table 3 reports results ranging from 1.5B to 32B across three effort levels. Rows with "–" as LLM denote single-model inference.

A key finding is that effective collaboration requires a moderate capability gap between the LLM and SLM. When the gap is too large, the weaker SLM shows only modest improvements: extremely

Model	Algebra		Counting		Geometry	
	Acc.	Cost	Acc.	Cost	Acc.	Cost
Single Models						
① DeepSeek-7B	93.09	0.18	74.26	**0.22**	65.34	**0.23**
② GPT-4o-mini	91.32	0.31	74.89	0.37	62.42	0.40
③ GPT-oss-120B	82.14	0.19	86.50	0.29	82.05	0.50
Collaboration						
①+②	94.10	0.17	86.29	0.77	77.66	0.31
①+③	**95.79**	**0.10**	**86.71**	0.23	**84.55**	0.30

Table 4: Performance and cost ($/1K samples) of collaboration with API-based LLMs on selected MATH subjects. DeepSeek-7B serves as the local SLM, while GPT-4o-mini and gpt-oss-120b are accessed via API.

small models may lack sufficient capacity to parse and execute complex reasoning patterns from much larger models, leading to partial utilization of the guidance. For instance, the 1.5B model with 32B guidance only improves from 59.92% to 67.09%, failing to match the 32B standalone performance of 82.07%. Conversely, when the gap is too small, collaboration yields marginal gains, possibly because models of similar capacity share similar reasoning styles and thus provide redundant rather than complementary insights. The optimal collaboration emerges when the SLM is capable enough to follow the LLM's guidance yet still benefits from higher-level reasoning it cannot produce independently.

3.5 API-accessible LLM Collaboration (RQ4)

A practical advantage of Tandem is its compatibility with closed-source LLMs accessible only via API, as it does not require access to model weights. To validate this, we evaluate collaborations where DeepSeek-7B serves as the local SLM, while GPT-4o-mini (OpenAI et al., 2024) and gpt-oss-120b (OpenAI et al., 2025) are accessed through API calls.

As shown in Table 4, API-based collaboration consistently improves accuracy across all subjects, with the combined system outperforming both the local SLM and the remote LLM alone. This confirms that Tandem remains effective even when the mentor is accessed via API, likely because the structured insight format transfers well regardless of communication protocol.

Notably, the collaboration also reduces inference cost. The LLM's high-level thinking insights enable the SLM to solve problems with shorter reasoning chains, eliminating the need for lengthy self-exploration. This cost reduction is particularly valuable for API-based deployment, where pricing is typically token-based and verbose outputs directly increase expenses.

4 Related Works

LLM collaboration methods leverage multiple models working together to solve complex problems through division of labor and iterative discussion. These approaches can be broadly categorized by their coordination mechanisms. Debate-based methods (Chan et al., 2024) let models discuss and refine responses through multi-turn exchanges, where each model critiques and improves upon others' outputs until consensus is reached. Role-based methods (Hong et al., 2024) assign specialized functions to different models within structured workflows, such as planner, executor, and reviewer, enabling complex task decomposition. Verification-based methods (Lightman et al., 2024) employ one model to generate candidates while another validates or ranks them, improving output reliability through cross-model checking. Recent work has also explored large–small model collaboration for inference acceleration, where smaller models assist larger ones through speculative decoding (Chen et al., 2023) or draft-then-verify pipelines, reducing latency by parallelizing token generation. However, most existing approaches either route each query to a single model without leveraging complementary capabilities, or combine multiple models at the cost of substantial overhead, leaving the challenge of achieving both improved quality and cost efficiency largely unaddressed. A more comprehensive discussion of the broader landscape of LLM ensemble methods is provided in Appendix B.

5 Conclusion

In this paper, we proposed Tandem, a large–small LLM collaboration framework, which reduces computational cost while preserving the benefits of the thinking paradigm of LLMs. It separates reasoning from answer generation by letting LLMs provide structured insights and using a cost-aware mechanism to decide when these insights are sufficient for SLMs to produce answers. Experiments on two math datasets show that Tandem outperforms the mentor LLM while reducing computation by roughly 40%, suggesting that lightweight high-quality reasoning guidance can often serve as an effective substitute for full reasoning chains.

Limitations

Our work has several limitations that suggest directions for future research.

First, we evaluate Tandem exclusively on mathematical reasoning tasks (MATH and GSM8K). While these benchmarks require multi-step reasoning and are well-suited for validating our approach, the generalizability to other domains such as commonsense reasoning, code generation, or open-ended question answering remains unexplored.

Second, our framework requires training a separate MLP classifier for each dataset to predict sufficiency. Although the classifier is lightweight and training is efficient, this introduces additional overhead when adapting Tandem to new tasks.

Third, Tandem employs a fixed two-model collaboration where one LLM serves as the mentor and one SLM as the intern. This simple division may not fully exploit more sophisticated collaboration patterns, such as involving multiple models with diverse capabilities or dynamically adjusting roles based on task characteristics. Exploring finer-grained multi-model collaboration could further enhance performance.

References

Sandra K. Abell, Deborah R. Dillon, Carol J. Hopkins, William D. McInerney, and David G. O'Brien. 1995. "somebody to count on": Mentor/intern relationships in a beginning teacher internship program. *Teaching and Teacher Education*, 11(2):173–188.

John R. Anderson. 2007. *How Can the Human Mind Occur in the Physical Universe?* Oxford University Press.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. 2024. Chateval: Towards better llm-based evaluators through multi-agent debate. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *CoRR*, abs/2302.01318.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025a. Do NOT think that much for 2+3=? on the overthinking of long reasoning models. In *Forty-second International Conference on Machine Learning*.

Zhijun Chen, Jingzheng Li, Pengpeng Chen, Zhuoran Li, Kai Sun, Yuankai Luo, Qianren Mao, Dingqi Yang, Hailong Sun, and Philip S. Yu. 2025b. Harnessing multiple large language models: A survey on LLM ensemble. *CoRR*, abs/2502.18036.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid LLM: cost-efficient and quality-aware query routing. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Thinkless: LLM learns when to think. *CoRR*, abs/2505.13379.

Mohamed Amine Ferrag, Norbert Tihanyi, and Mérouane Debbah. 2025. From LLM reasoning to autonomous AI agents: A comprehensive review. *CoRR*, abs/2504.19678.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025. Token-budget-aware LLM reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24842–24855, Vienna, Austria. Association for Computational Linguistics.

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

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. Metagpt: Meta programming for A multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zhongzhan Huang, Guoming Ling, Vincent S. Liang, Yupei Lin, Yandong Chen, Shanshan Zhong, Hefeng Wu, and Liang Lin. 2025. Routereval: A comprehensive benchmark for routing llms to explore model-level scaling up in llms. *CoRR*, abs/2503.10657.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, and 1 others. 2024. Openai o1 system card. *CoRR*, abs/2412.16720.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 14165–14178. Association for Computational Linguistics.

Zixuan Ke, Fangkai Jiao, Yifei Ming, Xuan-Phi Nguyen, Austin Xu, Do Xuan Long, Minzhi Li, Chengwei Qin, Peifeng Wang, Silvio Savarese, Caiming Xiong, and Shafiq Joty. 2025. A survey of frontiers in LLM reasoning: Inference scaling, learning to reason, and agentic systems. *Trans. Mach. Learn. Res.*, 2025.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. 2024. More agents is all you need. *arXiv preprint arXiv:2402.05120*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. Routing to the expert: Efficient reward-guided ensemble of large language models. *arXiv preprint arXiv:2311.08692*.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, and oothers. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Sungjin Park, Xiao Liu, Yeyun Gong, and Edward Choi. 2024. Ensembling large language models with process reward-guided tree search for better complex reasoning. *arXiv preprint arXiv:2412.15797*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Yangyifan Xu, Jianghao Chen, Junhong Wu, and Jiajun Zhang. 2025. Hit the sweet spot! span-level ensemble for large language models. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 8314–8325. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yuxuan Yao, Han Wu, Mingyang Liu, Sichun Luo, Xiongwei Han, Jie Liu, Zhijiang Guo, and Linqi Song. 2025. Determine-then-ensemble: Necessity of top-k union for large language model ensembling. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Yao-Ching Yu, Chun-Chih Kuo, Ziqi Ye, Yu-Cheng Chang, and Yueh-Se Li. 2024. Breaking the ceiling of the LLM community by treating token generation as a classification for ensembling. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 1826–1839. Association for Computational Linguistics.

Murong Yue. 2025. A survey of large language model agents for question answering. *CoRR*, abs/2503.19213.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Jieyu Zhang, Ranjay Krishna, Ahmed H Awadallah, and Chi Wang. 2023. Ecoassistant: Using llm assistant more affordably and accurately. *arXiv preprint arXiv:2310.03046*.

Zesen Zhao, Shuowei Jin, and Z Morley Mao. 2024. Eagle: Efficient training-free router for multi-llm inference. *arXiv preprint arXiv:2409.15518*.

Wenhao Zheng, Yixiao Chen, Weitong Zhang, Souvik Kundu, Yun Li, Zhengzhong Liu, Eric P Xing, Hongyi Wang, and Huaxiu Yao. 2025. Citer: Collaborative inference for efficient large language model decoding with token-level routing. *arXiv preprint arXiv:2502.01976*.

Marija Šakota, Maxime Peyrard, and Robert West. 2024. Fly-swat or cannon? cost-effective language model choice via meta-modeling. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, page 606–615, New York, NY, USA. Association for Computing Machinery.

Series	Model	Metric	Algebra		Counting		Geometry		Intermediate		Num		Prealgebra		Precalc		Average	
Non-thinking Mode																		
Single	**7B**	Acc.	93.60		76.79		67.43		67.33		85.74		89.55		67.95		80.40	
		Len.	487.0		542.2		**693.5**		1101.0		640.5		370.8		**900.4**		664.4	
		Cost	**6.82**		**7.59**		**9.71**		**15.41**		**8.97**		**5.19**		**12.61**		**9.47**	
	32B	Acc.	92.84	(-1%)	78.27	(+2%)	69.31	(+3%)	66.11	(-2%)	82.41	(-4%)	90.59	(+2%)	66.48	(-3%)	79.98	(-1%)
		Len.	**385.5**	(-21%)	452.1	(-17%)	839.8	(+22%)	1296.9	(+18%)	555.4	(-14%)	319.2	(-14%)	1105.5	(+23%)	685.3	(+4%)
		Cost	24.67	(+262%)	28.93	(+282%)	53.75	(+454%)	83.00	(+439%)	35.55	(+297%)	20.43	(+294%)	70.75	(+462%)	45.30	(+379%)
Baseline	**7B+32B (low)**	Acc.	91.83	(-2%)	75.53	(-2%)	67.43		65.34	(-3%)	80.00	(-7%)	90.36	(+1%)	67.77	(-1%)	79.00	(-2%)
		Len.	402.7	(-18%)	467.0	(-14%)	699.0	(+1%)	**1094.8**	(-1%)	578.1	(-10%)	311.8	(-16%)	966.1	(+8%)	626.8	(-6%)
		Cost	12.10	(+78%)	13.00	(+72%)	16.25	(+68%)	21.79	(+42%)	14.56	(+63%)	10.82	(+109%)	19.98	(+59%)	15.50	(+64%)
	7B+32B (medium)	Acc.	93.26	(-1%)	77.85	(+2%)	66.60	(-2%)	66.00	(-2%)	81.11	(-6%)	90.24	(+1%)	70.70	(+5%)	80.02	(-1%)
		Len.	392.3	(-20%)	**416.2**	(-24%)	774.9	(+12%)	1113.1	(+2%)	**551.7**	(-14%)	304.7	(-18%)	970.3	(+8%)	**626.5**	(-6%)
		Cost	26.43	(+288%)	27.88	(+268%)	36.04	(+272%)	44.43	(+189%)	32.43	(+262%)	21.92	(+323%)	41.07	(+226%)	32.89	(+248%)
	7B+32B (high)	Acc.	93.34	(-1%)	77.43	(+1%)	70.35	(+5%)	67.77	(+1%)	83.89	(-3%)	90.59	(+2%)	69.78	(+3%)	80.94	(+1%)
		Len.	390.1	(-20%)	450.3	(-17%)	816.0	(+18%)	1167.3	(+7%)	558.7	(-13%)	301.9	(-19%)	1039.4	(+16%)	650.8	(-3%)
		Cost	28.82	(+323%)	32.61	(+330%)	44.02	(+354%)	60.80	(+295%)	38.71	(+332%)	22.86	(+341%)	54.44	(+332%)	40.32	(+326%)
Tandem	**7B+32B**	Acc.	**94.19**	(+1%)	**81.01**	(+6%)	**70.56**	(+5%)	**69.66**	(+4%)	**85.93**	(+1%)	**90.93**	(+2%)	**73.81**	(+9%)	**82.56**	(+3%)
		Len.	474.8	(-3%)	479.3	(-12%)	809.4	(+17%)	1118.6	(+2%)	639.3	(-1%)	**301.9**	(-19%)	1055.3	(+18%)	674.6	(+2%)
		Cost	11.08	(+63%)	26.08	(+244%)	38.90	(+301%)	41.84	(+172%)	9.40	(+5%)	21.54	(+316%)	51.91	(+312%)	28.68	(+203%)

Table 5: Performance comparison under non-thinking mode of the LLMs on the MATH dataset (Hendrycks et al., 2021). We evaluate single models (7B, 32B), fixed-budget collaboration baselines (low/medium/high mentor token budgets), and Tandem. Metrics include accuracy (%), average inference length (tokens), and computational cost (TFLOPs). Bold indicates the best and underlined indicates the second-best for each subject.

A Statistical Feature Extraction Details

For the cost-aware thinking judgment mechanism described in Section 2.3, we extract a comprehensive set of statistical features from the perplexity and entropy values computed over the insight tokens. These features form a feature vector $\mathbf{f}_e$ at each effort level e.

Perplexity Features (7 dimensions). Given $\mathbf{PPL}^t = \{\mathrm{PPL}_i^t\}_{i=1}^n$, we compute:

- Mean: $\mu_{\mathrm{PPL}}^t = \frac{1}{n}\sum_{i=1}^n \mathrm{PPL}_i^t$
- Std: $\sigma_{\mathrm{PPL}}^t = \sqrt{\frac{1}{n}\sum_{i=1}^n \left(\mathrm{PPL}_i^t - \mu_{\mathrm{PPL}}^t\right)^2}$
- Median: $\mathrm{median}(\{\mathrm{PPL}_i^t\}_{i=1}^n)$
- Maximum: $\max(\{\mathrm{PPL}_i^t\}_{i=1}^n)$
- Minimum: $\min(\{\mathrm{PPL}_i^t\}_{i=1}^n)$
- 25th percentile: $Q_{25}(\{\mathrm{PPL}_i^t\}_{i=1}^n)$
- 75th percentile: $Q_{75}(\{\mathrm{PPL}_i^t\}_{i=1}^n)$

Entropy Features (7 dimensions). Given $\mathbf{H}^t = \{\mathrm{H}_i^t\}_{i=1}^n$, we compute:

- Mean: $\mu_{\mathrm{H}}^t = \frac{1}{n}\sum_{i=1}^n \mathrm{H}_i^t$
- Std: $\sigma_{\mathrm{H}}^t = \sqrt{\frac{1}{n}\sum_{i=1}^n \left(\mathrm{H}_i^t - \mu_{\mathrm{H}}^t\right)^2}$
- Median: $\mathrm{median}(\{\mathrm{H}_i^t\}_{i=1}^n)$
- Maximum: $\max(\{\mathrm{H}_i^t\}_{i=1}^n)$
- Minimum: $\min(\{\mathrm{H}_i^t\}_{i=1}^n)$
- 25th percentile: $Q_{25}(\{\mathrm{H}_i^t\}_{i=1}^n)$
- 75th percentile: $Q_{75}(\{\mathrm{H}_i^t\}_{i=1}^n)$

Trend Features (2 dimensions). Let $k = \min(20, n)$. We compute:

- Perplexity trend: $\Delta_{\mathrm{PPL}}^t = \frac{1}{k}\sum_{i=n-k+1}^n \mathrm{PPL}_i^t - \frac{1}{k}\sum_{i=1}^k \mathrm{PPL}_i^t$
- Entropy trend: $\Delta_{\mathrm{H}}^t = \frac{1}{k}\sum_{i=n-k+1}^n \mathrm{H}_i^t - \frac{1}{k}\sum_{i=1}^k \mathrm{H}_i^t$

These features capture whether the SLM becomes more or less confident as it processes x^t.

Metadata (1 dimension).

- **Length:** $n = |x^t|$, the total number of tokens in the SLM input $x^t = Q \oplus \mathcal{I}^t$.

These 18 features collectively provide a comprehensive representation of how the SLM processes the LLM-generated insights, capturing not only the overall difficulty but also the variability and temporal dynamics of the processing.

B Broader Landscape of LLM Ensemble

LLM ensemble plays an essential part in enhancing the generation performance and robustness of AI systems (Chen et al., 2025b). Existing LLM ensemble methods can be classified into three categories: 1) model selection before inference selects the best candidate LLM for a specific task or question, prioritizing smaller models whenever possible to minimize computational resources and inference time (Ding et al., 2024; Lu et al., 2023; Šakota et al., 2024; Zhao et al., 2024); 2) Output aggregation methods obtain the best answer by integrating LLM

SYSTEM

You are a reasoning assistant. Analyze the given problem and follow the structured `[Thinking Insights]` below.

INSIGHTS

1. **Goal**: ⟨*objective, constraints, and required output form*⟩
2. **Planning**: ⟨*high-level strategy; subproblem decomposition; edge cases*⟩
3. **Retrieval**: ⟨*relevant facts, formulas, or definitions; N/A if none*⟩
4. **Action**: ⟨*concrete steps and intermediate calculations*⟩

CONSTRAINTS

- Keep each component concise.
- Maintain notational consistency with the original problem.

Figure 6: Prompt template for the LLM to generate structured Thinking Insights.

outputs with different granularities. These methods can fully leverage the complementary capabilities of multiple LLMs when answering individual questions, thus offering higher potential performance (Yu et al., 2024; Yao et al., 2025; Zheng et al., 2025; Xu et al., 2025; Park et al., 2024); 3) LLM collaboration (i.e. LLM agent) employs multiple models working cooperatively through planning, reasoning and summary to reach a final answer. This approach can achieve the advantages of both previous categories through cascaded reasoning that starts with smaller models and escalates to larger ones based on output quality assessment (Li et al., 2024; Jiang et al., 2023; Zhang et al., 2023).

Model Selection (ensemble before inference) approaches select the most appropriate model before generation. For example, Fly-swat or Cannon (FORC) uses a DistilBERT model to predict which LLM performs best for each query, enabling cost-effective model selection that reduces inference costs while maintaining performance (Šakota et al., 2024). RouterEval combines LLM routing with recommender systems by collecting correctness records from thousands of models across datasets, training a router that outperforms individual strong models when selecting among about 10 candidates (Huang et al., 2025). While model selection can reduce computational costs by choosing smaller models when appropriate, training robust routers requires extensive labeled data.

Output Aggregation (ensemble during inference) methods integrate outputs of multiple LLMs during generation to leverage the knowledge of all models. These methods can be further divided into three types. Firstly, token-level methods ensemble multiple LLMs by aligning vocabularies and averages token distributions (Yu et al., 2024; Yao et al., 2025). Secondly, sentence-level approaches choose the best piece from multiple candidate text segments based on quality metrics or voting mechanisms (Xu et al., 2025). Thirdly, response-level methods generate complete responses from each model and then select the final output with ranking models, or use a separate fusion model to combine insights from all responses (Jiang et al., 2023). Although these ensemble methods can improve response quality, they incur computational overhead as all LLMs perform inference for each query. This issue becomes more severe with the emergence of reasoning-intensive models like GPT4-o1 (Jaech et al., 2024), which employ extended chain-of-thought processes that can lead to overthinking problems (Chen et al., 2025a).

C Non-thinking Mode Collaboration

Table 5 demonstrates that Tandem also generalizes effectively to non-thinking LLMs. Without explicit reasoning chains, the 7B+32B collaboration achieves the highest average accuracy of **82.56%**, surpassing both the 7B model at 80.40% and the 32B model at 79.98%, while reducing computational cost by **36.7%** relative to the 32B baseline. On Precalculus, one of the more challenging subjects, accuracy reaches **73.81%**, exceeding both single-model baselines by over 5 percentage points. These results confirm that our cost-aware collaboration framework operates independently of the reasoning paradigm, achieving superior accuracy-efficiency trade-offs through dynamic computation allocation.

D Prompt Schema Template for Thinking Insight Generation

Figure 6 summarizes the prompt schema used in our mentor–intern collaboration to elicit a compact `[Thinking Insights]` block. The mentor LLM outputs exactly four structured components (Goal, Planning, Retrieval, Action), while the intern SLM produces the final answer based on these insights, enabling consistent downstream parsing and stable guidance.

E Self-competence Awareness of SLMs

A key question in our framework is whether the SLM can reliably assess its own ability to solve a problem given the current guidance. We compare three approaches for predicting sufficiency, all using MLP classifiers with different inputs: (1) **Statistics**: perplexity and entropy features computed from the SLM's output probabilities; (2) **Hidden**: the last-layer hidden state at the final token position; and (3) **Hidden w/ LoRA**: same as Hidden, but additionally finetuning the SLM with LoRA before extracting hidden states.

As shown in Table 6, Statistics and Hidden methods achieve comparable performance, suggesting that the SLM possesses a degree of self-awareness regarding its own competence. Surprisingly, the lightweight Statistics approach, which only requires output probabilities, matches the performance of Hidden, which requires access to internal representations. This indicates that the SLM's confidence signals are already well-encoded in its output distributions.

Notably, Hidden w/ LoRA performs worse than the non-finetuned approaches, likely because finetuning on a limited dataset leads to overfitting. These findings validate our design choice of using perplexity and entropy as sufficiency indicators.

	Algebra	Counting	Geometry	Intermediate	Number	Prealgebra	Precalculus	Average
SLM	93.09	75.95	65.34	60.13	76.85	89.55	62.45	71.74
LLM	94.78	82.07	68.89	64.45	85.00	91.04	67.22	80.90
Hidden	<u>95.70</u>	**83.12**	<u>73.05</u>	**67.88**	<u>88.52</u>	**93.00**	<u>71.79</u>	<u>83.46</u>
Hidden w/ LoRA	94.09	73.02	57.36	63.09	75.60	91.45	69.57	74.88
Statistics	**96.04**	<u>82.70</u>	**73.07**	<u>67.77</u>	**88.70**	<u>92.65</u>	**71.98**	**83.47**

Table 6: Accuracy comparison across MATH subjects using different sufficiency prediction methods. All methods use MLP classifiers with different inputs: *Statistics* uses perplexity and entropy from output probabilities; *Hidden* uses last-layer hidden state at the final token; *Hidden w/ LoRA* uses hidden states from a LoRA-finetuned SLM. Bold values indicate the best performance and underlined values indicate the second-best performance.