Collaborative Beam Search: Enhancing LLM Reasoning via Collective Consensus

Yangyifan Xu[1,2], **Shuo Ren**[2], **Jiajun Zhang**[1,2,3]*
[1]School of Artificial Intelligence, University of Chinese Academy of Sciences
[2]Institute of Automation, Chinese Academy of Sciences [3]Wuhan AI Research
{xuyangyifan2021, shuo.ren}@ia.ac.cn, jjzhang@nlpr.ia.ac.cn

Abstract

Complex multi-step reasoning remains challenging for large language models (LLMs). While parallel inference-time scaling methods, such as step-level beam search, offer a promising solution, existing approaches typically depend on either domain-specific external verifiers, or self-evaluation which is brittle and prompt-sensitive. To address these issues, we propose Collaborative Beam Search (CBS), an iterative framework that harnesses the collective intelligence of multiple LLMs across both generation and verification stages. For generation, CBS leverages multiple LLMs to explore a broader search space, resulting in more diverse candidate steps. For verifications, CBS employs a perplexity-based collective consensus among these models, eliminating reliance on an external verifier or complex prompts. Between iterations, CBS leverages a dynamic quota allocation strategy that reassigns generation budget based on each model's past performance, striking a balance between candidate diversity and quality. Experimental results on six tasks across arithmetic, logical, and commonsense reasoning show that CBS outperforms single-model scaling and multi-model ensemble baselines by over 4 percentage points in average accuracy, demonstrating its effectiveness and general applicability.

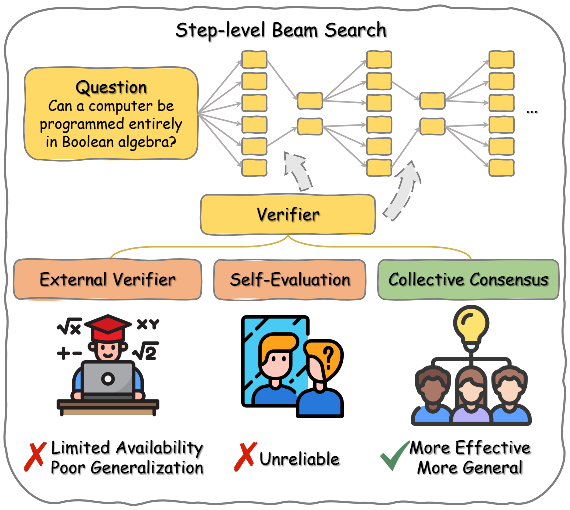

Figure 1: **Motivation of CBS.** Existing verifiers face limitations. External verifiers are largely confined to well-defined and well-explored areas like mathematical reasoning, limiting their applicability to broader reasoning tasks. Self-evaluation relies on well-designed prompts and often struggles with reliable assessment. Our approach CBS provides a more versatile and reliable verification mechanism.

1 Introduction

Improving the reasoning capabilities of large language models (LLMs), particularly for complex tasks requiring multiple reasoning steps, still faces challenges (Creswell et al., 2022; Wei et al., 2022; Chen et al., 2025a). A promising strategy to address this challenge is parallel inference-time scaling, which generates multiple candidates via sampling and then prunes bad candidates based on verification signals. By exploring of a broad space of potential reasoning paths, parallel scaling can improve the robustness and accuracy of LLM reasoning processes (Yao et al., 2023; Brown et al., 2024; Snell et al., 2025).

Step-level beam search is an effective and computationally efficient parallel scaling method (Park et al., 2024; Chen et al., 2024; Yu et al., 2024a). This approach iteratively utilize a step-level verifier to filter candidate steps generated by a single model. As illustrated in Figure 1, existing verifiers fall into two main categories: 1) External verifiers, such as process reward models (PRMs), often rely on expensive human annotations or automatic annotation via Monte Carlo Tree Search (Zheng et al., 2024). Consequently, their availability is restricted to well-defined and widely explored domains (e.g., mathematical reasoning). Furthermore, even in

* Corresponding Author

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 11398–11410
November 4-9, 2025 ©2025 Association for Computational Linguistics

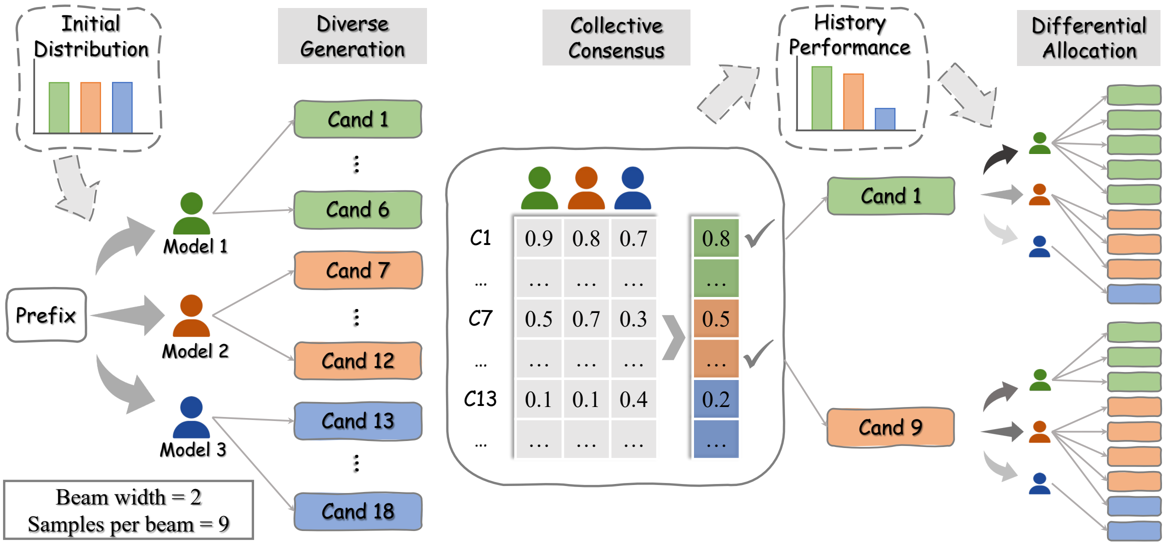

Figure 2: **The CBS framework.** CBS iteratively executes generation and verification stages. (1) Generation: we leverage multiple LLMs to generate diverse candidate steps. (2) Verification: we employ collective consensus calculated by average perplexity to facilitate external verifier-free evaluation. (3) Between iterations, we reallocate candidate quotas for the next generation stage based on the models' performance history.

domains where resources are available, verifiers exhibit limited generalization capabilities on more challenging tasks (Liu et al., 2025). 2) Prompt-based self-evaluation methods (Yao et al., 2023; Xie et al., 2023; Li et al., 2025) have the LLM prompt itself to generate critical feedback or reward scores. Nevertheless, their heavy dependence on well-designed prompts restricts their general applicability. More critically, they often struggle to reliably assess solution quality, leading to fluctuating performance (Liu et al., 2025; Wan et al., 2024). Beyond the limitations of existing verifiers, the reliance on just one model for generation offers limited candidate diversity, ultimately restricting the potential for effective scaling.

To tackle the above issues, we introduce Collaborative Beam Search (CBS), a novel framework that harnesses collective model intelligence to enhance LLM reasoning. The key insight behind CBS is the natural synergy between model ensemble and inference-time scaling: the collective power of model ensemble can enhance both the generation and verification stages iteratively within this scaling process. In the generation stage, to address the limited diversity of single-model sampling, CBS sources candidate steps from multiple LLMs. These LLMs, spanning diverse datasets, architectures, and training methodologies, exhibit distinct capabilities (Jiang et al., 2023; Xu et al., 2024). In the verification stage, to overcome the fragility of self-evaluation and the dependency on external verifiers, CBS utilizes perplexity-based collective consensus among multiple models as its reward signal, resulting in a more versatile and reliable verification. Between iterations, to address the inefficiency of "one-size-fits-all" budgeting, where all models contribute equally despite large variance in their per-task reliability, CBS incorporates a dynamic quota allocation strategy. This strategy reassigns generation budget based on each model's past performance, striking a balance between diversity and quality.

We evaluate our method on six reasoning tasks across three categories: arithmetic, logical, and commonsense reasoning. Experimental results demonstrate the superiority of our approach compared with existing single-model scaling and multi-model ensemble baselines, achieving an average improvement of more than 4 percentage points across all tasks. Further analysis elucidates the mechanism by which CBS achieves improvements through ensembling.

Our contributions can be summarized as follows:

- We propose a novel collaborative beam search method that leverages ensembling to enrich the candidate pool and perform accurate verification.

- We devise a dynamic quota allocation strategy that adjusts the generation budget based on historical performance, striking a balance between diversity and quality.

- Empirical results demonstrate the effectiveness and general applicability of our method. Further analysis elucidates how the ensemble leads to the observed performance gains.

2 Methods

We introduce CBS, a framework designed to enhance LLM reasoning by leveraging collective intelligence. The core mechanisms—diverse candidates generation (Section 2.1), collective consensus verification (Section 2.2), and differential quota allocation (Section 2.3)—are detailed in the following subsections. Pseudo-code for CBS is provided in Algorithm 1 and the case study is provided in Appendix A.

2.1 Diverse Candidates Generation

The candidate generation stage of CBS shares the core mechanism of standard beam search, which involves generating multiple potential continuations for each active hypothesis. It differs primarily in two aspects: First, CBS generates complete sentences as its intermediate steps, rather than individual tokens. Second, CBS sources these candidate steps from multiple LLMs, instead of relying on a single model. By leveraging multiple LLMs with different capabilities and internal knowledge, CBS aims to generate a more diverse set of candidate steps, thereby enabling exploration of a significantly broader solution space.

As shown in Figure 2, in the initial generation round, we allocate the candidate generation quota evenly among all participating LLMs to obtain as diverse candidate steps as possible. In the subsequent generation rounds, more sampling opportunities are assigned to LLMs that have demonstrated superior performance in the previous round. This dynamic allocation strategy will be detailed in Section 2.3.

2.2 Collective Consensus Verification

In the verification stage, the CBS framework utilizes perplexity, a simple and widely used metric, to evaluate and select generated reasoning steps. Perplexity assesses the alignment of a candidate step with each LLM's internal knowledge. A lower perplexity score signifies a closer alignment, which indicates stronger model endorsement for that candidate step. The simplicity of perplexity obviates

Algorithm 1 Collaborative Beam Search

Input: Input prompt q, Beam size B, Sampled steps per stage K, Maximum step depth T
Output: Best solution sequence for q
Model: N LLMs $\mathcal{M} \leftarrow \{m_1, \dots, m_N\}$
1: Initialize prefix sequences $\mathcal{S} \leftarrow \{s_1^0, \dots, s_B^0\}$
2: **for** $i = 1$ to B **do**
3: $\quad s_i^0 \leftarrow q$
4: **end for**
5: Initialize allocation $\boldsymbol{a} \leftarrow \{a_1, \dots, a_N\}$
6: **for** $i = 1$ to N **do**
7: $\quad a_i \leftarrow K/(B * N)$
8: **end for**
9: $t \leftarrow 1$
10: **while** sequences in $\mathcal{S}$ are not complete and $t < T$ **do**
11: $\quad \mathcal{S}_{\text{candidate}} \leftarrow \{\}$
12: $\quad$ **for** each sequence $s^{(0:t-1)}$ in $\mathcal{S}$ **do**
13: $\quad\quad$ **for** $i = 1$ to N **do**
14: $\quad\quad\quad$ **for** $j = 1$ to a_i **do**
15: $\quad\quad\quad\quad s'_{\text{extended}} \leftarrow \textsc{Generation}(m_i, s^{(0:t-1)})$
16: $\quad\quad\quad\quad \mathcal{S}_{\text{candidate}} \leftarrow \mathcal{S}_{\text{candidate}} + s'_{\text{extended}}$
17: $\quad\quad\quad$ **end for**
18: $\quad\quad$ **end for**
19: $\quad$ **end for**
20: $\quad \boldsymbol{c} \leftarrow \textsc{Verification}(\mathcal{S}_{\text{candidate}}, \mathcal{M})$
21: $\quad \mathcal{S}_{\text{beam}} \leftarrow \textsc{Selection}(\mathcal{S}_{\text{candidate}}, \boldsymbol{c}, B)$
22: $\quad \boldsymbol{a} \leftarrow \textsc{Allocation}(\mathcal{S}_{\text{beam}}, K/B)$
23: $\quad \mathcal{S} \leftarrow \mathcal{S}_{\text{beam}}$
24: $\quad t \leftarrow t + 1$
25: **end while**
return sequence with highest score in $\mathcal{S}$

the need for external verifiers or intricately designed self-evaluation prompts, thereby providing our CBS framework with enhanced generalization capabilities.

Considering a set of N candidate LLMs (denoted as $\mathcal{M}$), and a set of K candidate steps (denoted as $\mathcal{S}$) generated at the current reasoning stage. We first have each candidate LLM ($m_i \in \mathcal{M}$) independently calculate the perplexity score for all candidate steps:

$$\text{PPL}_i(s_k^t) = \exp\left(-\frac{1}{|s_k^t|}\sum_{j=1}^{|s_k^t|} \log p(x_j | s_k^{(0:t-1)}, x_{<j})\right)$$

where s_k^t is the k-th candidate step generated at the t-th round, and $s_k^{(1:t-1)}$ is the prefix leading to s_k^t.

Next, we define the collective consensus metric for each candidate step as the negative of its average perplexity. Building on this, a reasoning path's collective consensus metric is derived by averaging the metrics of its constituent steps (Line 20 in

Algorithm 1).

$$\begin{aligned}\boldsymbol{c}(s_k^{(1:t)}) &= \frac{1}{t}\sum_{j=1}^{t}\boldsymbol{c}(s_k^j) \\ &= \frac{1}{t}\sum_{j=1}^{t}\frac{1}{N}\sum_{m_i\in\mathcal{M}} -\mathrm{PPL}_i(s_k^j)\end{aligned}$$

The top-B paths with the highest collective consensus form the beam for the next generation round $\mathcal{S}_{beam}$ (Line 21 in Algorithm 1).

2.3 Differential Quota Allocation

To leverage the varying strengths of participating LLMs, CBS employs a differential quota allocation strategy. Based on observed performance history, this strategy dynamically adjusts the number of candidate steps each LLM will generate in subsequent rounds. First, we quantify the performance of each model m_i in the current round by counting the number of its generated candidate steps included in selected beam $\mathcal{S}_{beam}$. Notably, if multiple models generate the same step, and that step is included in $\mathcal{S}_{beam}$, each contributing model is counted. Next, these counts are transformed into a probability distribution using a temperature-controlled softmax. We sample from this distribution to determine each model's candidate quota $\boldsymbol{a}$ for the next generation round (Line 22 in Algorithm 1).

$$a_i \propto \exp\left(\mathrm{Count}(\mathcal{S}_{beam}, i))/\tau\right)$$

Lower values of temperature τ make the selection more biased towards top-performing models, while higher values lead to more uniform selection probabilities, encouraging diversity. By adjusting the temperature, we strike a balance between candidate diversity and quality.

3 Experimental Settings

3.1 Tasks and Datasets

To demonstrate the versatility of our method, we choose benchmarks from three reasoning genres: arithmetic reasoning (GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021)), logical reasoning (PrOntoQA (Saparov and He, 2023) and ProofWriter (Tafjord et al., 2021)) and commonsense reasoning (StrategyQA (Geva et al., 2021) and Date Understanding from BIG-Bench-Hard (Suzgun et al., 2023)). Task details are provided in Appendix B.

Method	Multiple LLM Ensemble	External Verifier	Prompt-based Evaluation
PANEL			✓
LLM-Blender	✓	✓	
MOA	✓		✓
SweetSpan	✓		
LE-MCTS	✓	✓	
CBS-PRM	✓	✓	

Table 1: Summary of representative recent methods.

3.2 Candidate LLMs

We select four open-source LLMs, approximately 7B to 9B in size, as candidate models for ensemble in each task. For arithmetic reasoning tasks, we use two general domain LLMs: Yi-1.5-9B (Young et al., 2024) and InternLM-2.5-7B (Cai et al., 2024), along with two math LLMs: Rho-Math-7B (Lin et al., 2024) and DeepSeek-Math-7B (Shao et al., 2024). For other tasks, we utilize four general models: Yi-1.5-9B, InternLM-2.5-7B, Gemma-2-9B (Team et al., 2024), and Llama-3.1-8B (Grattafiori et al., 2024).[1]

These models are trained on large-scale, high-quality datasets, establishing a strong knowledge base that allows them to perform well on public benchmarks. Sourced from distinct institutions, these models exhibit inherent diversity, which provides opportunities for effective ensemble.

3.3 Baselines

To provide a comprehensive evaluation of CBS, we compare it against diverse baselines from three groups: classical single LLM inference-time scaling methods, representative recent methods from related problems, and a variant of our method. The key characteristics of the latter two groups are summarized in Table 1.

Classical Baselines We evaluate greedy decoding, self-consistency (SC) (Wang et al., 2022), Best-of-N (BoN) (Lightman et al., 2023) and step-level beam search (BS) as inference time scaling baselines for single LLMs.

PANEL Li et al. (2025) utilize self-generated natural language critiques as feedback to guide the step-level tree search. We leverage the best-performing model for each task to establish a strong baseline for comparison.

[1]The scalability of our method to ensembles of larger models is discussed in Appendix C.

Base LLM	Method	Logical Reasoning		Commonsense Reasoning		Average
		PrOntoQA	ProofWriter	StrategyQA	Date	
Yi-1.5	Greedy	74.00	54.67	71.18	81.60	70.36
	SC	77.40	62.67	**74.67**	82.80	74.55
	BON	67.00	23.83	68.56	82.00	60.35
	BS	75.60	62.83	69.00	84.80	73.06
InternLM-2.5	Greedy	64.00	49.67	72.93	87.20	68.45
	SC	68.00	59.67	72.93	<u>88.80</u>	72.35
	BON	59.20	27.5	72.05	84.40	60.79
	BS	73.60	55.67	71.62	86.80	71.92
Gemma-2	Greedy	69.80	50.00	70.31	82.00	68.03
	SC	74.80	51.00	70.74	85.20	70.44
	BON	60.60	31.17	69.00	84.80	61.39
	BS	69.80	52.50	70.74	84.40	69.36
LlaMa-3.1	Greedy	71.60	58.67	69.43	82.80	70.63
	SC	<u>80.20</u>	<u>63.83</u>	71.18	84.80	<u>75.00</u>
	BON	63.40	39.17	62.88	85.20	62.66
	BS	79.00	61.00	69.00	86.00	73.75
Top-1	PANEL	78.80	50.00	72.05	84.80	71.41
All	LLM-Blender	19.40	42.83	66.81	84.40	53.36
All	MOA	76.00	55.33	73.80	87.60	73.18
All	SweetSpan	77.00	59.17	**74.67**	88.40	74.81
All	CBS(Ours)	**83.80(+3.60)**	**67.17(+3.34)**	**74.67(+0.0)**	**92.00(+3.20)**	**79.41(+4.41)**

Table 2: Main results on logical and commonsense reasoning tasks. We highlight the best result in **bold** and the second-best result with an <u>underline</u>, respectively. LE-MCTS and CBS-PRM are excluded from these four tasks due to the unavailability of the required external verifier resources.

LLM-Blender utilize a fusion model to merge the top-ranked candidates selected by a pairwise ranker and produce an improved output.

MOA Wang et al. (2024a) construct a layered Mixture-of-Agents architecture in which each layer consists of multiple LLM agents. Each agent takes all the outputs from agents in the previous layer as auxiliary information in generating its response.

SweetSpan Xu et al. (2025) propose a span-level model ensemble method that iteratively selects the best fixed-length spans generated by multiple LLMs to construct the final output.

LE-MCTS Park et al. (2024) present a process-level model ensemble framework that incorporates Monte Carlo Tree Search for complex math reasoning.

CBS-PRM This variant replaces our verification component with an external process reward model Math-Shepherd (Wang et al., 2024b), the same one employed by LE-MCTS.

3.4 Implement Details

Unless explicitly modified, we utilize nucleus sampling decoding with a temperature of 0.6 and a top-p value of 0.9 in all experiments. Across all tasks, we apply a 3-shot chain-of-thought prompt and report accuracy as the performance metric. For our proposed method, we set the beam size $B = 4$, the sample size $K = 32$, and the softmax temperature $\tau = 0.1$. We define each sentence starting with a "Step" marker as a reasoning step. The hyperparameters for baselines are configured as described in their respective papers.

4 Experimental Results and Analysis

The main results on logical reasoning, commonsense reasoning, and arithmetic reasoning tasks are shown in Table 2 and Table 3.

4.1 CBS demonstrates superiority

Our proposed CBS consistently outperforms single LLM inference-time scaling methods and LLM ensemble methods across all types of reasoning tasks, demonstrating the effectiveness and general applicability of our approach. Notably, CBS achieves

Base LLM	Method	GSM8K	MATH	Average
Yi-1.5	Greedy	63.08	32.00	47.54
	SC	73.46	37.20	55.33
	BON	70.66	32.00	51.33
	BS	69.14	36.40	52.77
InternLM-2.5	Greedy	53.83	35.80	44.82
	SC	65.88	43.20	54.54
	BON	65.43	39.20	52.32
	BS	64.22	40.60	52.41
Rho-Math	Greedy	59.59	28.00	43.80
	SC	69.98	36.40	53.19
	BON	65.43	35.00	50.22
	BS	67.10	31.80	49.45
DS-Math	Greedy	56.41	31.60	44.01
	SC	67.55	38.00	52.78
	BON	60.50	35.40	47.95
	BS	64.44	35.20	49.82
Top-1	PANEL	65.50	35.60	50.55
All	LLM-Blender	58.83	-	-
All	MOA	63.08	33.40	48.24
All	SweetSpan	62.85	37.60	52.23
All	LE-MCTS	61.41	36.60	49.01
All	CBS-PRM	74.53	35.80	55.17
All	CBS(Ours)	**75.06(+0.53)**	**44.00(+0.80)**	**59.53(+4.20)**

Table 3: Main results on arithmetic reasoning tasks. We highlight the best result in **bold** and the second-best result with an underline, respectively. LLM-Blender is excluded from the MATH task because we find that it cannot generate properly fomatted result.

an average improvement of 4.20% on arithmetic reasoning tasks and 4.41% on logical and commonsense reasoning tasks over the second-best method. We attribute this success to CBS's effective harnessing of collective model intelligence, which broadens candidate exploration through diverse and selected LLM contributions and ensures reliable and external verifier-free evaluation via perplexity-based collective consensus.

4.2 Performance Comparison: CBS vs. External Verification

In arithmetic reasoning tasks, where PRM resources are available, our method demonstrates superior robustness and generalization compared to external verifier-based methods LE-MCTS and CBS-PRM. We observe that LE-MCTS achieves only marginal improvements on MATH, while on GSM8K, it underperforms even the greedy decoding results of the best single model. This underperformance likely stems from the detrimental effect of weak candidate models on LE-MCTS, as its performance is sensitive to the quality of its ensemble members (Park et al., 2024). In contrast, our method exhibits greater robustness by dynamically allocating more computational resources to the stronger models for a given instance. This adaptive approach allows CBS to effectively leverage the strengths of different models while mitigating the impact of weaker ones.

On the other hand, while CBS-PRM performs comparably to our method on the simpler GSM8K benchmark, its performance degrades significantly on the more challenging MATH500 dataset. This suggests that PRM exhibits limited generalization capabilities on more challenging in-domain tasks, which aligns with the observations of Liu et al. (2025). Rather than relying on external verifiers, our method uses collective consensus among the models for evaluation, demonstrating superior generalization performance.

Unlike the previous two approaches, LLM-Blender utilizes general-purpose ranking and fusion models and is applied to all tasks. However, it performs very poorly on the PrOntoQA and cannot generate properly formatted results in MATH. This indicates LLM-Blender struggles to generalize to domains with data distributions that differ from its training data.

4.3 Performance Comparison: CBS vs. Prompt-Based Self-Evaluation

As prompt-based self-evaluation methods, MOA and PANEL show limited improvement. In fact, on some tasks, they perform worse than greedy decoding. This limited self-evaluation ability of LLMs via prompting aligns with findings from previous papers (Huang et al., 2023; Stechly et al., 2023). These methods require highly specialized and complex prompts. Furthermore, the number of candidate samples supported by such approaches is constrained by the context length of the underlying LLM. For example, with PANEL, we observe that attempting to incorporate more candidates did not improve performance and even led to degradation (see our analysis in Section 4.5). In contrast, our CBS method avoids these limitations, providing a simple yet effective way to achieve consistent performance gains.

4.4 Ablation Study

To dissect the contributions of the core components within our CBS framework, we conduct a series of ablation studies. We systematically evaluate the impact of: (1) collective consensus verification, (2) diverse candidates generation, and (3) differential quota allocation. This is achieved by comparing the full CBS approach against three ablated variants.

Method	Collective Consensus	Diverse Candidates	Differential Allocation	Arithmetic Reasoning		Logical Reasoning		Commonsense Reasoning	
				GSM8K	MATH	PrOntoQA	ProofWriter	StrategyQA	Date
SS				69.14	40.60	75.60	61.00	71.62	86.80
SC	✓			74.98	43.60	76.60	62.00	72.05	91.20
MC	✓	✓		73.77	42.00	82.20	66.33	72.93	90.80
CBS	✓	✓	✓	75.06	44.00	83.80	67.17	74.67	92.00

Table 4: Ablation study on the three core components within CBS: (1) diverse candidates generation, (2) collective consensus verification, and (3) differential quota allocation.

Single-Best LLM + Self-PPL Evaluation (SS) This baseline employs the best-performing model for each task, identified by its greedy decoding performance, to conduct a standard step-level beam search. During the verification stage, we utilize only the chosen model's perplexity as the reward signal to evaluate candidate steps.

Single-Best LLM + Collective PPL Evaluation (SC) In this variant, candidates are still sourced from the single best-performing LLM for each task, identical to SS. The verification stage differs by employing the collective consensus mechanism in CBS, using the average perplexity from multiple models as the reward signal.

Multi LLM + Collective PPL Evaluation (MC) This setup uses multiple LLMs to generate candidates and applies the collective consensus mechanism for verification, resembling the complete CBS approach. The only difference is that it does not incorporate differential quota allocation; instead, each model provides an equal number of candidates in each generation round, regardless of past performance.

4.4.1 Impact of Collective Consensus Verification

Comparing SC with SS allows us to isolate the impact of collective consensus verification. Both variants source candidates from the best-performing LLM but utilize different verification signals: SC employs collective consensus, while SS depends on self-evaluation. Our experimental results demonstrate that SC surpasses SS across all evaluation tasks. This underscores that collective consensus provides a more robust and accurate reward signal than self-evaluation. Such benefit is particularly pronounced in GSM8K, MATH, and Date Understanding.

4.4.2 Impact of Diverse Candidates Generation

Comparing MC with SC highlights the impact of diverse candidates generation. Keeping the verification method constant, we observe that sourcing candidates from multiple LLMs yields varying effects across different tasks. This presents a trade-off between the enhanced diversity afforded by incorporating weaker models and the potential degradation in candidate quality. For PrOntoQA and ProofWriter, the positive impact of diversity significantly outweighs the potential reduction in candidate quality, making diverse sourcing a primary driver for the performance improvements observed with CBS. Conversely, for tasks like GSM8K, MATH, and Date Understanding, the detrimental effect of quality degradation is more pronounced. This occurs because weaker models crowd out the candidate quota that could otherwise be allocated to more proficient models, ultimately hindering overall performance.

This observation highlights the necessity of differential quota allocation for ensuring that the pursuit of diversity does not lead to a significant compromise in quality.

4.4.3 Impact of Differential Quota Allocation

Finally, we assess the contribution of the differential quota allocation strategy by comparing CBS against MC and SC. Experimental results show that this strategy effectively improves the candidate quality from multiple sources, as evidenced by CBS outperforming MC across all tasks. Furthermore, CBS also consistently achieves better performance than SC. This suggests that underperforming models are effectively identified and their negative impact is mitigated by this allocation mechanism.

To further quantify the impact of differential allocation, we conduct a detailed analysis on the MATH dataset. For each problem, a model is considered "capable" if it successfully solves the problem via greedy decoding. While greedy decoding

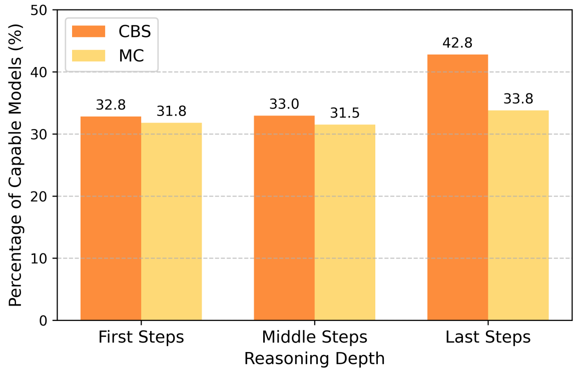

Figure 3: Effect of Differential Quota Allocation.

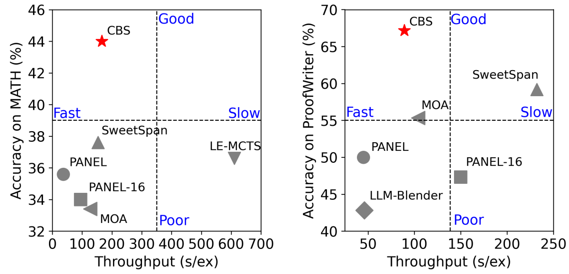

Figure 4: **Efficiency Analysis.** We compare the efficiency of our method with existing approaches based on throughput, measured in seconds per example.

correctness is an imperfect measure of a model's true ability, it serves as a reasonable proxy. We compare CBS and MC by analyzing the proportion of selected models that were capable at each problem's first, middle, and last steps. As shown in Figure 3, differential allocation increases the proportion of choosing capable models by prioritizing historically better-performing models during sampling. This effect becomes more pronounced in later steps, demonstrating that our method leverages accumulated historical information for increasingly effective allocation decisions.

4.5 Efficiency Analysis

We evaluate the efficiency of our method in comparison to existing approaches by examining throughput. Throughput is measured as the average time taken per example, reported in seconds per example (s/ex). Lower throughput values indicate better efficiency. As shown in Figure 4, our method demonstrates superior performance while maintaining a competitive time cost. To further illustrate how our method strikes a good balance between performance and efficiency, we test PANEL by increasing its candidate samples from the default of 5 to 16 (termed PANEL-16). However, we found that incorporating more candidates did not improve its performance and even led to degradation. We attribute this failure to the context length limitations of the underlying LLM, as a larger set of candidates makes it increasingly difficult for the model to make effective judgments via prompting.

5 Related Work

Our work is closely related to two fields of research: LLM ensemble and parallel inference-time scaling. This section reviews recent advancements in these fields.

5.1 Large Language Model Ensemble

Ensemble learning is a widely adopted technique to improve performance on specific tasks and ensure robust generalization by leveraging multiple complementary models (Lu et al., 2024; Chen et al., 2025b; Ren et al., 2025). Existing work explores model ensembles at different granularities. Sample-level ensemble methods (Jiang et al., 2023; Shnitzer et al., 2023; Lu et al., 2023; Jitkrittum et al., 2025; Farinhas et al., 2023) select or blend fully generated outputs, limiting dynamic correction and refinement during generation. For example, Jiang et al. (2023) rank candidate outputs from multiple LLMs using a pairwise ranking model trained on human preferences annotations, then merge the top three candidates with a fusion model fine-tuned on a mixed instruction dataset to produce an improved output. On the other hand, finer-grained approaches operate on partial outputs, ensembling at the token, word, or span level, which can mitigate error accumulation during generation. Token-level methods (Fu et al., 2023; Xu et al., 2024; Yu et al., 2024b) combine the output distribution of candidate models at each generation step. Liu et al. (2024) employ individual words as the ensembling unit, while Xu et al. (2025) leverage fixed-length spans.

In contrast, we propose a novel step-level ensemble method for reasoning, where variable-length complete sentences serve as the unit of ensembling. This approach ensures an uninterrupted reasoning process and demonstrates superior performance.

5.2 Parallel Inference-Time Scaling

Existing methods broadly fall into two categories: self-evaluation and external verifier-based approaches. Self-evaluation methods (Xie et al., 2023; Zhu et al., 2024; Li et al., 2025) prompt the model to generate its own feedback or reward scores. For instance, Xie et al. (2023) use self-generated answers to multiple-choice questions to guide stochastic beam search. External verifier-based methods (Yu et al., 2024a; Ma et al., 2023; Wan et al., 2024; Park et al., 2024) rely on external sources for process supervision. For example, Yu et al. (2024a) and Ma et al. (2023) enhanced heuristic search algorithms using process rewards from outcome-supervised value models and PRMs, respectively.

Both self-evaluation and external verifier-based approaches have limitations. Self-evaluation heavily rely on well-designed prompts and often struggles with reliable assessment, while external verifiers are largely confined to well-defined and well-explored areas like mathematical reasoning. In contrast, by combining model ensemble, our method provides a more reliable and generalizable verification mechanism.

6 Conclusion

In this paper, we introduce CBS, a novel framework harnessing collective model intelligence to enhance LLM reasoning. CBS expands the search space through diverse and selected LLM sources and achieves reliable, external verifier-free verification via perplexity-based collective consensus. This approach overcomes key limitations of existing methods, such as their restriction to single-model candidate generation, reliance on external verifiers, and dependence on complex prompts. Extensive experiments across arithmetic, logical, and commonsense reasoning tasks demonstrate the effectiveness and versatility of our method. By exploring the collective power of model ensembles, CBS paves the way for broader, multi-dimensional inference-time scaling, enabling expansion not only along traditional axes (e.g., sampling attempts, sequence length) but also along the model quantity dimension. Future work can explore deeper integration of model ensembles with advanced inference-time scaling techniques.

Limitations

The performance of CBS relies heavily on the diversity and quality of the candidate LLMs. While our experiments demonstrate the effectiveness of collective consensus as a reward signal and the differential quota allocation strategy in mitigating the influence of underperforming models, CBS performance can be impacted in extreme cases where the candidate LLMs exhibit substantial performance disparities or severely lack diversity (e.g., using different generations of the same model like Llama 2 and Llama 3). In such scenarios, CBS may not outperform inference-time scaling with the single best model. Approaches that use input characteristics to guide model selection before generation (Jitkrittum et al., 2025; Zhuang et al., 2024) offer a potential solution and could serve as a pre-filtering step for CBS. We leave a thorough exploration of combining these approaches for future work.

Acknowledgments

This work is supported by National Key R&D Program of China 2022ZD0160602 and the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDA04080400.

References

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. 2024. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024. Alphamath almost zero: Process supervision without process. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Jianghao Chen, Zhenlin Wei, Zhenjiang Ren, Ziyong Li, and Jiajun Zhang. 2025a. LR^2 bench: Evaluating long-chain reflective reasoning capabilities of large language models via constraint satisfaction problems. *arXiv preprint arXiv:2502.17848*.

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. *arXiv preprint arXiv: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. *arXiv preprint arXiv:2110.14168*.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2022. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *The*

Eleventh International Conference on Learning Representations.

António Farinhas, José GC de Souza, and Andre Martins. 2023. An empirical study of translation hypothesis ensembling with large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning*, pages 10421–10430. PMLR.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

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

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. *NeurIPS*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. Llm-blender: Ensembling large language models with pairwise comparison and generative fusion. In *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (ACL 2023)*.

Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Zifeng Wang, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, Aditya Krishna Menon, and Sanjiv Kumar. 2025. Universal model routing for efficient llm inference. *arXiv preprint arXiv:2502.08773*.

Yansi Li, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Qiuzhi Liu, Rui Wang, Zhuosheng Zhang, Zhaopeng Tu, Haitao Mi, et al. 2025. Dancing with critiques: Enhancing llm reasoning with stepwise natural language self-critique. *arXiv preprint arXiv:2503.17363*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, et al. 2024. Rho-1: Not all tokens are what you need. *arXiv preprint arXiv:2404.07965*.

Cong Liu, Xiaojun Quan, Yan Pan, Liang Lin, Weigang Wu, and Xu Chen. 2024. Cool-fusion: Fuse large language models without training. *arXiv preprint arXiv:2407.19807*.

Fan Liu, Wenshuo Chao, Naiqiang Tan, and Hao Liu. 2025. Bag of tricks for inference-time computation of llm reasoning. *arXiv preprint arXiv:2502.07191*.

Jinliang Lu, Ziliang Pang, Min Xiao, Yaochen Zhu, Rui Xia, and Jiajun Zhang. 2024. Merge, ensemble, and cooperate! a survey on collaborative strategies in the era of large language models. *arXiv preprint arXiv:2407.06089*.

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*.

Qianli Ma, Haotian Zhou, Tingkai Liu, Jianbo Yuan, Pengfei Liu, Yang You, and Hongxia Yang. 2023. Let's reward step by step: Step-level reward model as the navigators for reasoning. *arXiv preprint arXiv:2310.10080*.

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*.

Shuo Ren, Pu Jian, Zhenjiang Ren, Chunlin Leng, Can Xie, and Jiajun Zhang. 2025. Towards scientific intelligence: A survey of llm-based scientific agents. *arXiv preprint arXiv:2503.24047*.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *11th International Conference on Learning Representations, ICLR 2023*.

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

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. 2023. Large language model routing with benchmark datasets. *arXiv preprint arXiv:2309.15789*.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2025. Scaling test-time compute optimally can be more effective than scaling llm parameters. In *The Thirteenth International Conference on Learning Representations*.

Kaya Stechly, Matthew Marquez, and Subbarao Kambhampati. 2023. Gpt-4 doesn't know it's wrong: An analysis of iterative prompting for reasoning problems. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. Proofwriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Ziyu Wan, Xidong Feng, Muning Wen, Stephen Marcus Mcaleer, Ying Wen, Weinan Zhang, and Jun Wang. 2024. Alphazero-like tree-search can guide large language model decoding and training. In *International Conference on Machine Learning*, pages 49890–49920. PMLR.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. 2024a. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024b. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

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

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Xie. 2023. Self-evaluation guided beam search for reasoning. *Advances in Neural Information Processing Systems*, 36:41618–41650.

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*, pages 8314–8325.

Yangyifan Xu, Jinliang Lu, and Jiajun Zhang. 2024. Bridging the gap between different vocabularies for llm ensemble. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7133–7145.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Guoyin Wang, Heng Li, Jiangcheng Zhu, Jianqun Chen, et al. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

Fei Yu, Anningzhe Gao, and Benyou Wang. 2024a. Ovm, outcome-supervised value models for planning in mathematical reasoning. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 858–875.

Yao-Ching Yu, Chun-Chih Kuo, Ziqi Ye, Yu-Cheng Chang, and Yueh-Se Li. 2024b. Breaking the ceiling of the llm community by treating token generation as a classification for ensembling. *arXiv preprint arXiv:2406.12585*.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2024. Processbench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*.

Tinghui Zhu, Kai Zhang, Jian Xie, and Yu Su. 2024. Deductive beam search: Decoding deducible rationale for chain-of-thought reasoning. In *First Conference on Language Modeling*.

Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran. 2024. Embedllm: Learning compact representations of large language models. *arXiv preprint arXiv:2410.02223*.

A Case Study

We present an example generated by CBS using a test case from MATH in Figure 5. CBS achieves the correct solution by combining the best model-generated steps, as selected through collective consensus.

B Tasks and Datesets

Arithmetic Reasoning. GSM8K (Cobbe et al., 2021) consists of high quality linguistically diverse grade school math word problems. We used 1,319 test examples for the experiment. MATH (Hendrycks et al., 2021) is a dataset of challenging competition mathematics problems. We used the MATH500 subset for evaluation to avoid data leakage.

Logical Reasoning. ProofWriter (Tafjord et al., 2021) and PrOntoQA (Saparov and He, 2023) are widely utilized logical reasoning benchmarks using natural language. We evaluate on their respective most challenging subsets, comprising 600 and 500 samples.

Commonsense Reasoning. StrategyQA (Geva et al., 2021) is a human-curated commonsense dataset designed to test implicit multi-step reasoning. We use the validation set for test. Date Understanding from BIG-Bench-Hard (Suzgun et al., 2023) evaluates a model's ability to understand date-related information and answer time-sensitive factual questions.

C Scalability to Larger Models.

To verify the scalability of our CBS framework, we conduct an additional experiment on GSM8K using an ensemble of larger models in the 20B to 34B parameter range: InternLM2.5-20B, Mistral-24B, and Yi1.5-34B. As shown in Table 5, CBS achieves a significant improvement of 5.92% over the strongest individual model (Yi1.5-34B). This result confirms that our approach continues to provide substantial performance gains even when applied to more capable and powerful foundation models.

Method	**GSM8K**
InternLM2.5-20B	63.23
Mistral-24B	73.69
Yi1.5-34B	74.22
CBS (Ours)	**80.14**

Table 5: Performance on GSM8K with larger-scale models.

Question: For the eight counties listed below, what was the median number of students in $2005?$

```
\begin{tabular}[t]{|l|c|c|c|c|c|}
\multicolumn{6}{c}{\textbf{Number of Students per County}} \\\hline
\textbf{COUNTY}&\textbf{$2001$}&\textbf{$2002$}&\textbf{$2003$}&\textbf{$2004$}&\textbf{$2005$}\\\hline
Aiken&124&141&130&143&136\\\hline
Bamberg&17&15&15&14&11\\\hline
Barnwell&25&22&26&28&29\\\hline
Berkeley&583&557&554&553&524\\\hline
Calhoun&15&12&10&18&11\\\hline
Cherokee&19&13&18&13&19\\\hline
Chesterfield&46&18&13&22&29\\\hline
Colleton&64&49&52&46&41\\\hline
\end{tabular}
```

Yi-1.5-9B: Step1: The number of students for each county in 2005 are $136, 11, 29, 524, 11, 19, 29, 41$.

InternLM-2.5-7B: Step2: Arrange these numbers in ascending order: $11, 11, 19, 29, 29, 41, 136, 524$.

Rho-Math-7B: Step3: Since there are 8 numbers, the median is the average of the 4th and 5th numbers: $\\frac{29+29}{2}=29$.

DeepSeek-Math-7B/ InternLM-2.5-7B/ Rho-Math-7B : Step4: so the final answer is: \\boxed{29}. End of answer.

Figure 5: Example output of CBS in MATH.