OptScale: Probabilistic Optimality for Inference-time Scaling

Youkang Wang[1,2*], **Jian Wang**[2*], **Rubing Chen**[1,2], **Xiao-Yong Wei**[3,1,2†]

[1] PolySmart Group [2] Department of Computing, The Hong Kong Polytechnic University
[3] College of Computer Science, Sichuan University
{yk2022.wang, rubing.chen}@connect.polyu.hk
{cs007.wei}@polyu.edu.hk

Abstract

Inference-time scaling has emerged as a powerful technique for enhancing the reasoning performance of Large Language Models (LLMs). However, existing approaches often rely on heuristic strategies for parallel sampling, lacking a principled foundation. To address this gap, we propose a probabilistic framework that formalizes the optimality of inference-time scaling under the assumption that parallel samples are independently and identically distributed (i.i.d.), and where the Best-of-N selection strategy follows a probability distribution that can be estimated. Within this framework, we derive a theoretical lower bound on the required number of samples to achieve a target performance level, providing the first principled guidance for compute-efficient scaling. Leveraging this insight, we develop OPTSCALE, a practical algorithm that dynamically determines the optimal number of sampled responses. OPTSCALE employs a language model-based predictor to estimate probabilistic prior parameters, enabling the decision of the minimal number of samples needed that satisfy predefined performance thresholds and confidence levels. Extensive experiments on representative reasoning benchmarks (including MATH-500, GSM8K, AIME, and AMC) demonstrate that OPTSCALE significantly reduces sampling overhead while remaining better or on par with state-of-the-art reasoning performance. Our work offers both a theoretical foundation and a practical solution for principled inference-time scaling, addressing a critical gap in the efficient deployment of LLMs for complex reasoning.

Code — https://github.com/Albertwyk/OptScale

Introduction

The reasoning capabilities of LLMs have become a pivotal research area, given their ability to tackle complex cognitive tasks such as mathematical reasoning, decision-making, and problem-solving (Zhang et al. 2025a; Yang et al. 2025b). Recent advances demonstrate that inference-time scaling can substantially enhance reasoning performance by enabling the model to generate diverse candidate solutions, creating a richer space for comprehensive inference through answer aggregation (Chen et al. 2025; Ding et al. 2025; Wan et al. 2024). This approach is particularly valuable for proprietary LLMs, as it can operate purely at inference time without requiring model modifications. Although effective, these techniques face a fundamental efficiency challenge: the linear relationship between candidate solutions and computational costs leads to prohibitive token consumption. This has created an urgent need for optimized scaling strategies that navigate the Pareto frontier between reasoning performance and computational efficiency, maximizing accuracy while minimizing redundant computation (Qu et al. 2025).

Parallel inference-time scaling, a dominant paradigm in this space, operates by generating N candidate responses in parallel and selecting the optimal output via learned rules (e.g., verifiers) (Zhang et al. 2025b) or consensus (e.g., majority voting) (Wang et al. 2023). Current approaches fall into three categories: verifier-based approaches that train auxiliary models to rerank candidates (Yang et al. 2025b); data-driven methods that fine-tune LLMs on high-quality reasoning traces to improve solution quality (Qu et al. 2025); and inference-time techniques that dynamically adjust the number of candidates per input (Chen et al. 2025). While these methods demonstrate practical effectiveness, they suffer from significant theoretical limitations. They rely heavily on empirical heuristics and implicit priors (such as learned verifier preferences or empirical token budgets) without proper mathematical formulation. Crucially, the field lacks a systematic derivation of these priors from first principles or a rigorous quantification of the fundamental efficiency-accuracy trade-offs inherent in parallel scaling.

This paper presents a foundational study that bridges this critical gap through three key contributions: First, we formulate a probabilistic framework that formally establishes the optimality conditions for parallel inference-time scaling under i.i.d. assumptions, where the Best-of-N selection process follows an estimable probability distribution. This theoretical formulation provides a principled understanding of scaling behavior in probabilistic terms. Second, we derive a theoretical lower bound on the number of samples required to achieve any target performance level. This result offers mathematically grounded guidance for compute-efficient scaling, representing a significant advance beyond current heuristic approaches. Third, we develop OPTSCALE, an efficient algorithm that implements these theoretical insights in practice. OPTSCALE dynamically determines the

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

optimal sample size by employing an LLM-based predictor to estimate prior parameters, ensuring minimal computational cost while satisfying predefined performance thresholds and confidence levels. Our experiments demonstrate OPTSCALE's computational efficiency: it automatically terminates sampling for simple questions while preventing excessive computation on intractable ones.

To our knowledge, this work presents the first comprehensive treatment of parallel inference-time scaling that: 1) establishes a rigorous probabilistic model of the scaling process, 2) derives fundamental limits on scaling efficiency, and 3) provides a practical algorithmic implementation that approaches these theoretical optima.

Related Work

LLM Reasoning

The ability of LLMs to perform complex reasoning tasks has garnered significant attention and seen rapid advancements (Yu et al. 2024; Xu et al. 2025). Early successes with prompting strategies like Chain-of-Thought (CoT) (Wei et al. 2022) demonstrated that LLMs could produce step-by-step reasoning traces, significantly improving performance on tasks requiring logical deduction, mathematical problem-solving, and multi-step planning (Ahn et al. 2024; Snell et al. 2024; Liang et al. 2024). This foundational work has spurred the development of more sophisticated prompting techniques, including Self-Consistency (Wan et al. 2024), which generates multiple reasoning paths and selects the most consistent answer, and Tree-of-Thought (ToT) reasoning, which allows models to explore diverse reasoning trajectories in parallel (Yao et al. 2023; Long 2023).

Parallel Inference-time Scaling

Inference-time scaling (Ke et al. 2025) has emerged as a key strategy to enhance LLM reasoning by enlarging computational resources during inference. In this paper, we primarily focus on parallel inference-time scaling, which involves generating multiple candidate solutions in parallel and selecting the best one (Snell et al. 2024).

Verifier-based Scaling Methods. To improve inference-time performance, many studies use verifiers, also known as process reward models (PRMs), to assess the quality of candidate solutions generated by a primary LLM (Setlur et al. 2024; Qu et al. 2025). These verifiers predict the correctness of a solution or assign it a quality score, guiding the selection of the final answer by re-ranking the N outputs. For example, some methods train verifiers using human feedback or generated labels (Bai et al. 2022). Recent work examines how the interaction between generators, PRMs, and problem difficulty affects optimal test-time scaling (Zhang et al. 2025b). However, relying on additional models increases computational overhead, and the strategies for determining N often lack a theoretical foundation.

Dynamic Scaling Methods. Recognizing the limitations of fixed N, several methods explore dynamic approaches to adjust computation at inference time. These techniques adapt the number of generated candidates or computational resources based on the input query or the ongoing generation process (Ding et al. 2025; Wan et al. 2024). For example, Chen et al. (2025) allocates more computation to complex problems or where initial candidate solutions show high uncertainty or disagreement. Approaches like dynamic decomposition break down solution paths into manageable steps, dedicating more resources to challenging sub-problems. The goal is to avoid unnecessary computation on simpler instances while providing sufficient resources for harder ones. While these dynamic methods improve efficiency, they often rely on heuristic rules or learned strategies without a clear theoretical framework for the trade-off between efficiency and accuracy. These dynamic methods represent a step towards more efficient scaling.

However, the rules governing the dynamic allocation are often heuristic or learned without an explicit efficiency-accuracy trade-off. Our work distinguishes itself by providing a principled probabilistic formulation of the parallel scaling. This allows us to derive a provably optimal strategy for dynamically allocating compute to maximize accuracy per token and to quantify the theoretical upper bound for efficiency, addressing the lack of systematic, theoretically grounded priors in current dynamic scaling approaches.

Scaling with Probabilistic Optimality

In this section, we formulate a theoretical framework for inference-time scaling from a probabilistic optimality perspective. Based on this framework, we present OPTSCALE, an implementation that effectively achieves the goal of compute-efficient inference-time scaling.

Theoretical Framework

Preliminaries. Given an input question q processed by an LLM $\mathcal{M}$, which generates N candidate answers $\{a_i\}_{i=1}^{N}$. Each answer a_i consists of T_i reasoning steps $\{a_{i,t}\}_{t=1}^{T_i}$. A verifier $\mathcal{V}$ (e.g., a Process Reward Model, PRM) assigns normalized scores $\{s_{i,t} \in [0,1]\}_{t=1}^{T_i}$ to each step. Defining $\mathcal{A}(\cdot)$ as an aggregation operator (typically averaging multi-step scores or using the final step score), we compute the overall verification score for answer a_i as:

$$s_i = \mathcal{A}\big[\, \mathcal{V}(a_{i,t} \,|\, q, \mathcal{M}, a_{i,<t}) \,\big] \tag{1}$$

The optimal answer a^* is selected through:

$$a^* = \underset{i \in \{1,\ldots,N\}}{\arg\max}\ s_i \tag{2}$$

Previous research on inference-time reasoning has primarily differed in their choices of the language model $\mathcal{M}$, verifier $\mathcal{R}$, and aggregator $\mathcal{A}$. In contrast, our work makes a fundamental shift in perspective by investigating the probabilistic distributions that govern these components. It is a crucial distinction that sets our approach apart from prior studies.

Verifier Score Distribution. Let us assume that the verifier scores $\{s_i\}$ for a fixed question q are samples for a random variable S that is conform to a continuous probability distribution with probability density function (PDF) $f_S(\cdot)$

and cumulative distribution function (CDF) $F_S(\cdot)$ as:

$$S \sim f_S(s|\theta, q),\ F_S(s) = \mathbb{P}(S < s) = \int_{-\infty}^{s} f_S(x)dx, \quad (3)$$

where θ represents the parameters in $\mathcal{M}$, $\mathcal{R}$, and hyperparameters for generation (e.g., the temperature t).

Distribution of the Maximum Verifier Score. Let $Y = \max\{s_1, s_2, ..., s_N\}$ denote the maximum verification score among N candidates, which serves as the core selection criterion in Eq. (2). For any observed score s, the optimal decision on whether to sample more candidates can be determined when we know the exceedance probability $\mathbb{P}(Y \leq s)$. This represents the likelihood that s is the maximal score in the complete population of possible verifications. The probability can be derived by considering the joint event where $s_i \leq s$ holds for all N independent draws, yielding:

$$\mathbb{P}(Y \leq s) = \prod_{i=1}^{N} \mathbb{P}(s_i \leq s) = [F_S(s)]^N, \quad (4)$$

which also defines the CDF of Y as $F_Y(s) = \mathbb{P}(Y \leq s)$. The PDF of Y is:

$$f_Y(s) = \frac{d}{ds} F_Y(s) = N[F_S(s)]^{N-1} f_S(s). \quad (5)$$

Note that both the PDF and the CDF of Y are functions with respect to N. Figure 3 shows a few examples of such PDF functions in different values of N.

Probabilistic Optimality. After N sampling rounds, conventional methods (e.g., Best-of-N) confront a critical trade-off: whether to continue sampling (potentially finding higher-scoring answers) or terminate (to conserve computational resources). Our solution derives the probabilistically optimal sample size N^* as the minimal value satisfying:

$$\mathbb{P}(Y \geq s_{\min}) \geq \alpha \quad (6)$$

where $s_{\min}$ denotes the quality threshold for valid solutions, α represents the required confidence level. From Eq. (4), we reformulate the probability requirement as:

$$1 - F_Y(s_{\min}) \geq \alpha \implies [F_S(s_{\min})]^N \leq 1 - \alpha. \quad (7)$$

Solving for N yields the following closed-form solution:

$$N^* \geq \left\lceil \frac{\log(1-\alpha)}{\log F_S(s_{\min})} \right\rceil, \quad (8)$$

where $\lceil \cdot \rceil$ ensures integer sample sizes.

Implementation of the Optimal Scaling

Translating the theoretical framework into practice raises a critical question: *how can we reliably estimate the verifier score distribution $f_S(s|\theta, q)$?* Let (μ, σ) parameterize this distribution, where these parameters need not imply Gaussianity. While direct estimation of (μ, σ) is non-trivial, we find that $f_S(s|\theta, q)$ can be modeled as a **truncated normal distribution** constrained to $[0, 1]$. The probability density function is given by:

$$f_S(s|\theta, q) \propto f_S(s|\mu, \sigma) = \frac{\phi(\frac{s-\mu}{\sigma})}{\sigma[\Phi(\frac{1-\mu}{\sigma}) - \Phi(\frac{0-\mu}{\sigma})]}, \quad (9)$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ denote the standard normal PDF and CDF. The optimal sample size N^* is derived via Eq. (8) using (μ, σ) estimates. To this end, we propose OPTSCALE with the following two alternatives to implement inference-time scaling with probabilistic optimality.

OPTSCALE^t: Parameter Estimation via Trainable Predictors. This approach uses offline data to learn global trends in (μ, σ). We model them as standard Gaussian random variables centered around predicted means $(\bar{\mu}, \bar{\sigma})$:

$$\mu \sim \mathcal{N}(\bar{\mu}, \sigma_\mu), \quad \sigma \sim \mathcal{N}(\bar{\sigma}, \sigma_\sigma), \quad (10)$$

where σ_μ, σ_σ control estimation error bounds. The trends $(\bar{\mu}, \bar{\sigma})$ are obtained by predictors via fine-tuning MLPs:

$$\bar{\mu} = \text{MLP}_\mu(q, \mathcal{M}), \quad \bar{\sigma} = \text{MLP}_\sigma(q, \mathcal{M}). \quad (11)$$

Given observed verifier scores $D = \{s_k\}$ at inference, we refine $(\bar{\mu}, \bar{\sigma})$ via Maximum-a-Posteriori (MAP) estimation:

$$\mu^*, \sigma^* = \arg\max_{\mu,\sigma} \Big(\sum_{k=1}^{|D|} \log f_S(s_k|\mu, \sigma) + \log f_\mu(\mu|\bar{\mu}, \sigma_\mu) + \log f_\sigma(\sigma|\bar{\sigma}, \sigma_\sigma) \Big). \quad (12)$$

where f_μ, f_σ represent prior densities of μ and σ in Eq. (10).

OPTSCALE^0: Training-free Parameter Estimation. This variant estimates parameters solely from observed data using Maximum Likelihood Estimation (MLE), reducing the need for additional training. For a new query q, we empirically initialize (μ_0, σ_0) using a heuristic strategy with uniform uncertainty. We then use bootstrapping to calculate μ and σ from previous scores iteratively. As more verifier scores $\{s_k\}$ are observed, we update the parameters by:

$$\mu^*, \sigma^* = \arg\max_{\mu,\sigma} \sum_{k=1}^{|D|} \log f_S(s_k|\mu, \sigma). \quad (13)$$

This variant does not rely on learned predictors, making it highly lightweight and efficient.

Adaptive Scaling with OPTSCALE. Both variants ultimately compute the optimal sample size N^* using their refined parameters (μ^*, σ^*) in Eq. (8), which depends on the tail distribution $F_S(s_{\min})$. Sampling continues until the current sample count $N \geq N^*$, enabling 1) early stopping for simple queries, and 2) bounded effort for more complex or ambiguous cases. This adaptive mechanism allows OPTSCALE to efficiently scale inference-time computation.

Experiments

Experimental Setup

Benchmarks. We employ the following representative reasoning benchmarks: (1) **MATH-500** (Hendrycks et al. 2021) with 500 high school competition problems across algebra and geometry, following the OpenAI evaluation split; (2) **GSM8K** (Cobbe et al. 2021) with 8.5K grade school word problems assessing basic arithmetic and textual reasoning; (3) **AIME 2024** (MAA 2024) and (4) **AIME 2025**[1],

[1]https://huggingface.co/datasets/math-ai/aime25.

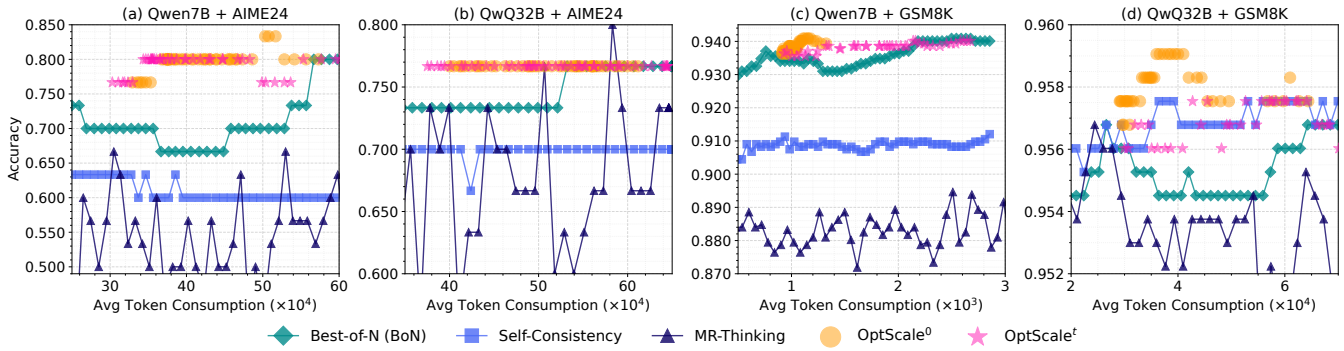

Figure 1: Scaling efficiency comparisons (accuracy vs. average token consumption): Both OPTSCALE^0 and OPTSCALE^t show consistently faster convergence and optimal accuracy-token tradeoff over the compared baseline methods.

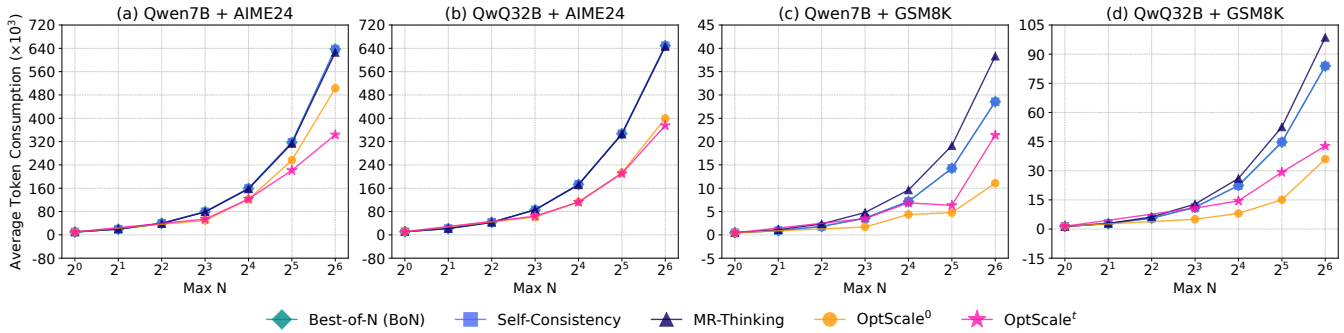

Figure 2: Token consumption of different methods when scaling across N: OPTSCALE consistently achieves reduced completion tokens when scaling to higher N over the compared baseline methods.

with each containing 30 pre-Olympiad level problems from the American Invitational Mathematics Examination, designed to test advanced mathematical reasoning; (5) **AMC**[2], a collection of intermediate-level problems from the 2023 American Mathematics Competitions. We report accuracy and tokens as evaluation metrics (Ngo et al. 2007, 2008).

Backbone Models. To assess generalizability and efficiency, we employ the following open-source backbone LLMs with different sizes and architectures: 1) **DeepSeek-R1-Distill-Qwen-7B** (Guo et al. 2025): a 7B-parameter reasoning model distilled from larger DeepSeek R1 models into a Qwen model, representing a state-of-the-art distilled model optimized for complex reasoning. 2) **Llama-3.1-8B-Instruct** (Grattafiori et al. 2024): an instruct model based on Llama architecture; 3) **QwQ-32B** (Yang et al. 2025a): a high-capacity model specialized for complex reasoning tasks. 4) **Deepseek-R1-Distill-Llama-8B**[3] (Guo et al. 2025): an 8B reasoning model distilled from larger DeepSeek R1 models into Llama. 5) **Qwen3-8B**[3] (Yang et al. 2025a), Qwen's latest reasoning model; 6) **Llama-3.2-1B-Instruct**[3] (Meta 2024), a light-weighted instruction-following model in Llama series.

Baseline Methods. We benchmark OPTSCALE against the commonly used inference-time scaling methods with different max N settings, including: 1) **Best-of-N (BoN)**: It samples N reasoning paths, then employs a verifier for scoring, and ultimately selects the best answer with the highest score; 2) **Self-Consistency (SC)** (Wang et al. 2023): It samples multiple diverse responses and aggregates the final answer based on majority voting; and 3) **Multi-round Thinking (MR-Thinking)** (Tian et al. 2025): It iteratively refines the reasoning process through multiple rounds.

Implementation Details. For scoring, we use Qwen2.5-Math-PRM-7B as our verifier. For both OPTSCALE^t and OPTSCALE^0, initial sample sizes of generated completions for enabling MLE/MAP refinement are set to be around 1/3 of the maximum allowed number of N generations. Across all experiments for OPTSCALE^0, we initialize μ_0 and σ_0 as the mean and standard deviation of the initial sampled data for each new query. To ensure a fair comparison, we re-ran all baseline methods under the same experimental conditions. For all datasets, we use a sampling temperature of 0.7 and a top-p of 0.95. All experiments are conducted on a single NVIDIA A6000 server with 8 GPUs. Further details are presented in the Appendix. We report accuracy (**Acc.**) and completion tokens (**Toks.**) as evaluation metrics.

[2]https://huggingface.co/datasets/math-ai/amc23.

[3]Main results of 1), 2), and 3) are shown in subsequent sections, while results of 4), 5), and 6) together with results of various N settings for all models are showcased in the Appendix.

Baseline Method	MATH-500		GSM8K		AIME 2024		AIME 2025		AMC 2023	
	Acc.	Toks. (↓)	Acc.	Toks. (↓)	Acc.	Toks. (↓)	Acc.	Toks. (↓)	Acc.	Toks. (↓)
Deepseek-R1-Distill-Qwen-7B										
Best-of-N (BoN) ($N = 8$)	**94.8**	22135	**92.4**	3582	**70.0**	79367	43.3	84342	**95.0**	40511
Self-Consistency ($N = 8$)	93.4	22135	90.1	3582	60.0	79367	40.0	84342	85.0	40511
MR-Thinking ($N = 8$)	91.2	21396	88.4	4792	56.7	78432	40.0	86568	87.5	36780
OptScale^0 (Ours) ($N = 8$)	**94.8**	**11354**	**92.4**	**1687**	**70.0**	**49505**	43.3	78803	**95.0**	**29288**
OptScale^t (Ours) ($N = 8$)	**94.8**	18236	**92.4**	3492	**70.0**	53855	**46.7**	**69661**	**95.0**	30671
Best-of-N (BoN) ($N = 64$)	94.0	174693	94.0	28547	80.0	637293	**53.3**	676533	**95.0**	312241
Self-Consistency ($N = 64$)	93.4	174693	91.2	28547	60.0	637293	40.0	676533	92.5	312241
MR-Thinking ($N = 64$)	92.0	168331	88.2	38305	70.0	625850	40.0	684446	90.0	286639
OptScale^0 (Ours) ($N = 64$)	**94.6**	110001	**94.1**	**11086**	**83.3**	503002	50.0	**549344**	**95.0**	119777
OptScale^t (Ours) ($N = 64$)	**94.6**	**76284**	94.0	21386	80.0	**343491**	**53.3**	649900	**95.0**	**119282**
Llama-3.1-8B-Instruct										
Best-of-N (BoN) ($N = 8$)	63.6	9609	**88.4**	2160	**10.0**	54212	**3.3**	51510	**32.5**	15321
Self-Consistency ($N = 8$)	58.6	9609	87.5	2160	3.3	54212	0.0	51510	27.5	15321
MR-Thinking ($N = 8$)	41.6	14918	61.6	2812	0.0	46037	**3.3**	**39374**	12.5	24814
OptScale^0 (Ours) ($N = 8$)	**63.8**	**8756**	**88.4**	**1462**	**10.0**	**42859**	**3.3**	46970	**32.5**	15113
OptScale^t (Ours) ($N = 8$)	63.6	9479	**88.4**	2136	**10.0**	43212	**3.3**	46724	**32.5**	**14323**
Best-of-N (BoN) ($N = 64$)	68.8	70643	89.3	14697	**13.3**	386783	**0.0**	348361	42.5	113401
Self-Consistency ($N = 64$)	60.8	70643	89.2	14697	6.7	386783	**0.0**	348361	32.5	113401
MR-Thinking ($N = 64$)	40.8	122227	60.7	23423	0.0	**377475**	**0.0**	327074	15.0	189239
OptScale^0 (Ours) ($N = 64$)	**69.0**	64179	89.5	**5720**	**13.3**	386772	**0.0**	141904	**45.0**	**100675**
OptScale^t (Ours) ($N = 64$)	68.6	**60692**	**89.6**	12044	**13.3**	386668	**0.0**	**140707**	**45.0**	106039
Qwen/QwQ-32B										
Best-of-N (BoN) ($N = 8$)	94.6	30601	95.8	11177	**73.3**	85853	**66.7**	95235	95.0	55291
Self-Consistency ($N = 8$)	**95.2**	30601	95.6	11177	70.0	85853	**66.7**	95235	92.5	55291
MR-Thinking ($N = 8$)	94.6	30378	95.3	12832	70.0	85864	53.3	95536	87.5	47288
OptScale^0 (Ours) ($N = 8$)	95.0	**17469**	**95.9**	**5031**	**73.3**	**61449**	**66.7**	85669	**95.0**	42177
OptScale^t (Ours) ($N = 8$)	94.8	26910	95.8	10767	**73.3**	64094	**66.7**	**84766**	**95.0**	**36494**
Best-of-N (BoN) ($N = 60$)	94.8	230402	95.8	83902	**76.7**	649256	66.7	721405	97.5	420481
Self-Consistency ($N = 60$)	95.4	230402	95.8	83902	70.0	649256	63.3	721405	92.5	420481
MR-Thinking ($N = 60$)	93.8	227838	94.8	98607	73.3	646201	56.7	719654	80.0	360068
OptScale^0 (Ours) ($N = 60$)	**95.8**	107720	**95.9**	**35985**	**76.7**	399929	**70.0**	556340	**100.0**	**190633**
OptScale^t (Ours) ($N = 60$)	**95.8**	**106412**	95.8	42735	**76.7**	**375208**	**70.0**	**516346**	**100.0**	202603

Table 1: Comparison of different inference-time scaling methods on common mathematical reasoning benchmarks (with $N = 8$ and $N = 64$). "Acc." denotes accuracy (%), "Toks." indicate the total number of inference tokens.

Details on Distribution Predictor in OptScale^t. Since OptScale^t employs an auxiliary predictor for estimating the mean and standard deviation of the verifier score distribution, we introduce its training details in this section. We use Deepseek-R1-Distill-Qwen-1.5B as the backbone model to build the distribution predictor. We then freeze all layers except the last two and train the two individual MLP layers. We curate the training dataset by taking the MATH training set, which contains approximately 4,500 questions, and pre-generating 60 completions per question. This process allows us to obtain the verifier score statistics and create the <question, mean> and <question, standard deviation> pairs, accordingly. We train the predictor for 30 epochs, with a learning rate of 1e-5, weight decay of 0.01, and gradient clipping at 0.1. We set a dropout ratio of 0.2 for all MLP layers during training.

Can OptScale Ensure Optimal Scaling?

To evaluate scaling efficiency, we conducted comprehensive experiments across 5 benchmark datasets and 6 backbone models, varying the maximum number of answer candidates from 1 to 64. This experimental design required 1920 total runs per method (5 datasets $\times$ 6 backbones $\times$ 64 candidate limits), representing a significant computational investment. Due to resource constraints, we focus our comparison on the 3 most representative baseline methods: Best-of-N (BoN), Self-Consistency (SC), and Multi-Round Thinking.

Figure 1 shows that OptScale achieves optimal scaling. We take Deepseek-R1-Distill-Qwen-7B and QwQ-32B on GSM8K and AIME24 to evaluate both OptScale^0 and OptScale^t's scaling capacity. We examined different quality thresholds, ranging from 0.9 to 0.99, and confidence levels, from 0.9 to 0.98. Notably, all OptScale implementa-

tions along the frontier consistently outperform baselines, delivering higher accuracy with lower computational cost (token consumption). This confirms its ability to maintain scaling optimality across configurations. Figure 2 shows the scaling progress of different baseline models across the maximum allowed N over the values $\{1, 2, 4, 8, 16, 32, 64\}$. It is clearly demonstrated that OPTSCALE consistently reduces tokens compared to most baseline methods.

These results reveal several key insights: 1) OPTSCALE consistently outperforms all baseline methods across different configurations, achieving either higher accuracy with comparable token consumption or similar accuracy with substantially fewer tokens. On Qwen7B + AIME24, OPTSCALE constantly has 7% - 14% accuracy advantage over the best-performing baseline given the same token levels, while on QwQ32B + AIME24 and Qwen7B + GSM8K, OPTSCALE reaches best accuracy using 27% and 51% fewer tokens, exhibiting strong early convergence capabilities. 2) OPTSCALE's superiority is particularly apparent with larger N value. OPTSCALE's average token reduction percentage compared to the best baseline is 38%, 47%, and 54% when $N = 16, 32, 64$ respectively, indicating OPTSCALE's increased efficiency on larger N values. We also observe that token reduction is more apparent on easier benchmarks, where QwQ32B + GSM8K achieves the highest average token reduction percentage of 64%, while accuracy gains are greater on harder benchmarks (e.g., AIME24), proving our adaptive strategy's strength in complex, variable-difficulty tasks. 3) Both OPTSCALE^0 and OPTSCALE^t consistently consume significantly fewer tokens than other methods at the same accuracy levels, while OPTSCALE^0 is usually more efficient on easier benchmarks (like GSM8K) and OPTSCALE^t saves more tokens for harder benchmarks (like AIME24). 4) Through proper dynamic allocation of inference-time compute, OPTSCALE sometimes achieves accuracy levels that BoN fails to reach even when scaled indefinitely to a substantial amount of tokens, such as reaching the accuracy of 95.9% on QwQ32B + GSM8K. These advantages stem primarily from OPTSCALE's ability to evaluate question difficulty in advance and allocate resources optimally through our probabilistic framework. Further elaboration can be found in the case studies within the Appendix.

Can OPTSCALE Estimate the Verifier Score Distribution Accurately?

To evaluate our approach, we compare the estimated distributions against the ground truth. Figure 3 shows various alignment cases between our modeled distributions and empirical observations. In most cases (e.g., Questions 1, 2, 3, 4, and 8), the ground truth verifier scores follow a unimodal distribution, which is accurately captured by a truncated normal distribution over $[0, 1]$. For bimodal distributions (e.g., Questions 6 and 7), OPTSCALE maintains reasonably accurate predictions. Only in rare instances do we encounter more complex distributions: cases with multiple peaks (Question 9) or extremely narrow, high peaks (Question 5) which challenge the truncated normal approximation. However, OPTSCALE can still capture the main distribution quite well. These exceptions are also statistically insignificant and do not materially impact overall prediction quality.

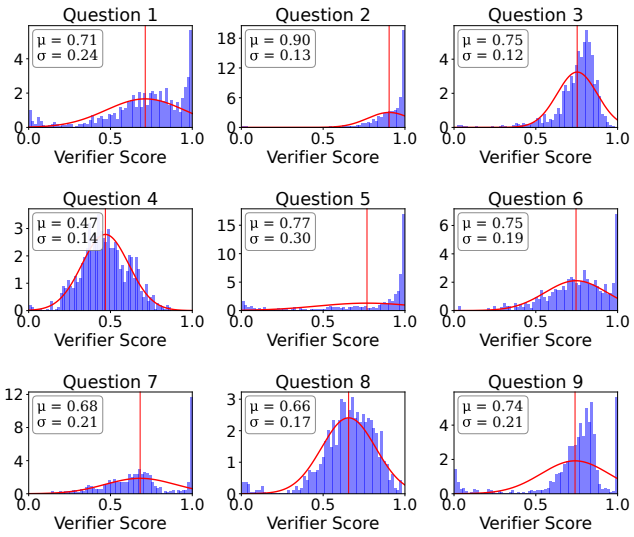

Figure 3: Samples of verifier score distribution: Real vs. Estimated. OPTSCALE accurately fits most distributions under the truncated normal distribution assumption.

How is OPTSCALE Sensitive to Its Parameters?

OPTSCALE employs two hyperparameters, namely the quality threshold $s_{\min}$ and the desired confidence level α. We repeat 66 combinations of these two parameters by varying $s_{\min}$ in the range of [0.80, 0.99] and α in [0.90, 0.99] with a step size of 0.02 and 0.01 respectively to evaluate its sensitivity. We employ Llama-3.2-1B-Instruct backbone with GSM8K benchmark for this focused study.

Impact of the Quality Threshold. Figure 4 shows the changes in accuracy and token consumption corresponding to the quality threshold $s_{\min}$ changes. As $s_{\min}$ increases, the accuracy growth gradually slows down and converges. However, token consumption continues to grow exponentially, with the token consumption multiplying by 2.5 times when the quality threshold increases from 0.8 to 0.99. Therefore, it is recommended to adopt a reasonably high threshold based on available computational budget as the cost-effectiveness of scaling declines sharply when $s_{\min}$ is close to 100%.

Impact of the Confidence Level α. Similarly, Figure 5 shows that both accuracy and token consumption increase with confidence level α. Unlike the exponential growth in completion tokens observed when increasing $s_{\min}$, token consumption rises nearly linearly with α—growing by approximately 100 tokens for every 0.01 increase in confidence. Meanwhile, the overall efficiency experiences only a modest decline as confidence grows.

Overall, we find the quality threshold $s_{\min}$ more sensitive in terms of determining both accuracy and token consumption compared to the confidence level α, making its selection much more important. During empirical experiments, we noticed that choosing a very high quality threshold (e.g., 0.99) can harm efficiency and result in suboptimal utilization of completion tokens, while a low quality threshold

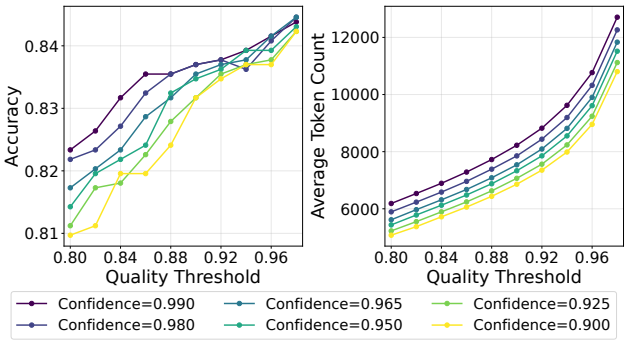

Figure 4: Sensitivity analysis of quality threshold $s_{\min}$: Model performance across target scores.

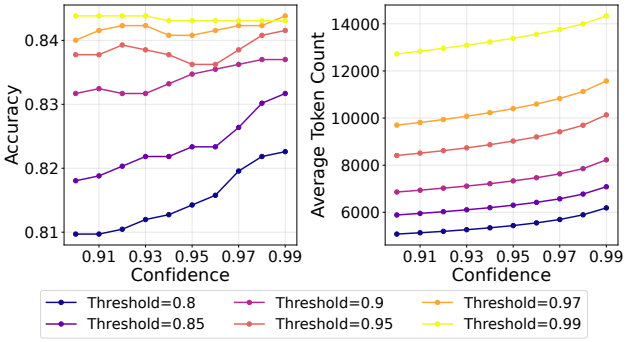

Figure 5: Sensitivity analysis of confidence level α: Model performance across target levels.

smaller than 0.9 could lead to accuracy degradation. A quality threshold $s_{\min}$ of around 0.95 to 0.96 is usually recommended, while there's much more room for selecting confidence level α: the default could be set to 0.9.

Can OptScale Achieve a New SOTA Performance across Benchmarks?

Table 1 reports the overall results of different inference-time scaling methods on commonly used mathematical reasoning benchmarks. After careful comparison, we summarize key findings below:

State-of-the-art Performance. When using the same backbone across different benchmarks, OptScale consistently achieves SOTA accuracy while drastically reducing inference token consumption. For example, using the Deepseek-R1-Distill-Qwen-7B backbone with $N = 64$, both OptScale^0 and OptScale^t achieve the highest accuracy of 94.6% on MATH-500, significantly surpassing all baselines while concurrently saving 37.0% (OptScale^0) and 56.3% (OptScale^t) than the BoN baseline.

Robustness Across Different Backbones. The performance of OptScale remains robust and highly efficient across a diverse range of backbone models. On Deepseek-R1-Distill-Qwen-7B, it achieves an average token reduction of 28.9% (OptScale^0) and 32.7% (OptScale^t) among all benchmarks. On Llama3.1-8B-Instruct, the reductions are 23.7% and 22.9%, while on QwQ-32B, such reductions even reach 45.0% and 38.5% for OptScale^0 and OptScale^t respectively. This shows that OptScale has a consistent optimal superiority across different backbones.

Overall, these results demonstrate that OptScale has strong capabilities to optimize the accuracy-efficiency trade-off across models and benchmarks. The probabilistic foundation of OptScale enables it to dynamically adjust sampling effort based on query difficulty, delivering superior efficiency without compromising reasoning quality.

OptScale^0 and OptScale^t: Which to Choose?

Both OptScale^0 and OptScale^t exhibit obvious superiority over other baseline methods. However, when it comes to comparing the performance of these two variants of OptScale internally, there's no definite conclusion which one is better. Comprehensively considering both accuracy and token consumption, OptScale^0 wins 16 out of 30 sets of comparisons in Table 1, while OptScale^t wins 14 times. Despite so, we observe a few prominent features that might guide readers in selecting the more useful variant of OptScale for their task:

1) OptScale^t seems to favor very difficult benchmarks. On the hardest benchmark AIME25, OptScale^t defeats OptScale^0 every time. This is likely because generated answers to difficult questions are more likely to have highly varying and unstable scores. OptScale^0 uses MLE to estimate optimal (μ^*, σ^*), but since the initial sampled data is unstable, MLE refinement could be very unstable. OptScale^t first evaluates questions' difficulty using the predictor and then refines the prediction using MAP. This makes it much more stable and more likely to make a well-informed prediction of optimal N. **2)** There's a minor tendency that OptScale^t seems to be better at larger N values, while OptScale^0 is better at smaller N. This probably indicates that OptScale^t is better at long-term scaling planning, while OptScale^0 is better for small-scale scaling given a limited computational budget, especially since it is completely training-free.

Conclusion

We introduce a principled probabilistic framework for parallel inference-time scaling in LLM reasoning, addressing the limitations of existing heuristic-based approaches. By modeling the Best-of-N selection under independently and identically distributed (i.i.d.) assumptions, we derive a theoretical lower bound on the sample size required to achieve target performance, offering the first formal guidance for compute-efficient scaling. Building on top of this, we propose OptScale, a practical algorithm that dynamically adjusts the number of samples using an LLM-based predictor. Extensive experiments across various reasoning benchmarks show that OptScale achieves superior performance with significantly reduced computational cost. Our work provides both theoretical foundations and practical implementations for more efficient and adaptive inference-time scaling.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (Grant No. 62372314). The experimental part of this work was supported by The Centre for Large AI Models (CLAIM) of The Hong Kong Polytechnic University. This work is also supported by Hong Kong Research Grants Council under the Theme-based Research Scheme (project no. T43-513/23-N), and is also supported in part by the PolyU Postdoc Matching Fund Scheme (4-W40Z). We also thank the support from PolyU Industrial Centre (IC), PolyU Graduate School (GS), Ren Da, Prof. Li Jing, Prof. Li Qing, and Marco Bettoni.

References

Ahn, J.; Verma, R.; Lou, R.; Liu, D.; Zhang, R.; and Yin, W. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

Bai, Y.; Jones, A.; Ndousse, K.; Askell, A.; Chen, A.; DasSarma, N.; Drain, D.; Fort, S.; Ganguli, D.; Henighan, T.; et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Chen, R.; Zhang, Z.; Hong, J.; Kundu, S.; and Wang, Z. 2025. SEAL: Steerable Reasoning Calibration of Large Language Models for Free. *arXiv preprint arXiv:2504.07986*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ding, Y.; Jiang, W.; Liu, S.; Jing, Y.; Guo, J.; Wang, Y.; Zhang, J.; Wang, Z.; Liu, Z.; Du, B.; et al. 2025. Dynamic parallel tree search for efficient llm reasoning. *arXiv preprint arXiv:2502.16235*.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint arXiv:2501.12948*.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Ke, Z.; Jiao, F.; Ming, Y.; Nguyen, X.-P.; Xu, A.; Long, D. X.; Li, M.; Qin, C.; Wang, P.; Savarese, S.; et al. 2025. A Survey of Frontiers in LLM Reasoning: Inference Scaling, Learning to Reason, and Agentic Systems. *arXiv preprint arXiv:2504.09037*.

Liang, Z.; Liu, Y.; Niu, T.; Zhang, X.; Zhou, Y.; and Yavuz, S. 2024. Improving llm reasoning through scaling inference computation with collaborative verification. *arXiv preprint arXiv:2410.05318*.

Long, J. 2023. Large language model guided tree-of-thought. *arXiv preprint arXiv:2305.08291*.

MAA. 2024. American Invitational Mathematics Examination - AIME. https://huggingface.co/datasets/AI-MO/aimo-validation-aime.

Meta. 2024. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

Ngo, C.-W.; Jiang, Y.-G.; Wei, X.-Y.; Wang, F.; Zhao, W.; Tan, H.-K.; and Wu, X. 2007. Experimenting vireo-374: Bag-of-visual-words and visual-based ontology for semantic video indexing and search. In *IEEE Computer Society*.

Ngo, C.-W.; Jiang, Y.-G.; Wei, X.-Y.; Zhao, W.; Wang, F.; Wu, X.; and Tan, H.-K. 2008. Beyond semantic search: What you observe may not be what you think. In *IEEE Computer Society*.

Qu, Y.; Yang, M. Y.; Setlur, A.; Tunstall, L.; Beeching, E. E.; Salakhutdinov, R.; and Kumar, A. 2025. Optimizing test-time compute via meta reinforcement fine-tuning. *arXiv preprint arXiv:2503.07572*.

Setlur, A.; Nagpal, C.; Fisch, A.; Geng, X.; Eisenstein, J.; Agarwal, R.; Agarwal, A.; Berant, J.; and Kumar, A. 2024. Rewarding progress: Scaling automated process verifiers for llm reasoning. *arXiv preprint arXiv:2410.08146*.

Snell, C.; Lee, J.; Xu, K.; and Kumar, A. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Tian, X.; Zhao, S.; Wang, H.; Chen, S.; Ji, Y.; Peng, Y.; Zhao, H.; and Li, X. 2025. Think Twice: Enhancing LLM Reasoning by Scaling Multi-round Test-time Thinking. *arXiv preprint arXiv:2503.19855*.

Wan, G.; Wu, Y.; Chen, J.; and Li, S. 2024. Dynamic self-consistency: Leveraging reasoning paths for efficient llm sampling. *arXiv preprint arXiv:2408.17017*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q. V.; Chi, E. H.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Xu, F.; Hao, Q.; Zong, Z.; Wang, J.; Zhang, Y.; Wang, J.; Lan, X.; Gong, J.; Ouyang, T.; Meng, F.; et al. 2025. Towards Large Reasoning Models: A Survey of Reinforced Reasoning with Large Language Models. *arXiv preprint arXiv:2501.09686*.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yang, W.; Ma, S.; Lin, Y.; and Wei, F. 2025b. Towards thinking-optimal scaling of test-time compute for llm reasoning. *arXiv preprint arXiv:2502.18080*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36: 11809–11822.

Yu, F.; Zhang, H.; Tiwari, P.; and Wang, B. 2024. Natural language reasoning, a survey. *ACM Computing Surveys*, 56(12): 1–39.

Zhang, Q.; Lyu, F.; Sun, Z.; Wang, L.; Zhang, W.; Guo, Z.; Wang, Y.; King, I.; Liu, X.; and Ma, C. 2025a. What, How, Where, and How Well? A Survey on Test-Time Scaling in Large Language Models. *arXiv preprint arXiv:2503.24235*.

Zhang, Z.; Zheng, C.; Wu, Y.; Zhang, B.; Lin, R.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2025b. The Lessons of Developing Process Reward Models in Mathematical Reasoning. *arXiv preprint arXiv:2501.07301*.