When More Tokens Hurt: Saturation Effects in Test-Time Compute Scaling

Anonymous ACL submission

Abstract

Recent work has shown that large language models can improve mathematical reasoning performance by allocating additional tokens at test time. However, the relationship between model scale and optimal token budgets remains unexplored. We conduct a systematic study of test-time compute scaling across four model sizes (0.5B to 7B parameters) on the GSM8K mathematical reasoning benchmark, evaluating performance at seven token budgets from 32 to 2048 tokens. We find three key results: (1) all models exhibit a performance cliff consistent across scales at the 128 to 256 token transition, with accuracy gains ranging from 8% to 51%, (2) larger models saturate at lower token budgets while achieving higher accuracy, the 7B model peaks at 512 tokens (86.8%) while the 0.5B model continues improving through 1024 tokens (18.7%), and (3) models can perform worse with excessive token budgets, with the 1.5B model losing 2.4% accuracy when increasing from 512 to 1024 tokens. These findings suggest that optimal token allocation strategies must account for model scale, and that practitioners should avoid over-allocating compute budgets at inference time.

1 Introduction

The ability of large language models (LLMs) to solve complex reasoning tasks has improved dramatically with scaling (Kaplan et al., 2020; Hoffmann et al., 2022). Recently, researchers have discovered a complementary scaling dimension: *test-time compute*, where models are given additional computational budget during inference to improve their outputs (Snell et al., 2024; Brown et al., 2024). This paradigm has shown particular promise for mathematical reasoning, where chain-of-thought prompting (Wei et al., 2022) allows models to "think longer" by generating more intermediate reasoning tokens.

Recent work on token-budget-aware reasoning (Han et al., 2025) demonstrated that LLMs can effectively compress their reasoning when given explicit token budgets, achieving substantial cost savings with minimal accuracy loss. However, this work primarily focused on token efficiency within individual models, leaving a critical question unexplored: *how do optimal token budgets vary across model scales?*

Understanding this relationship is crucial for practical deployment. If smaller models require proportionally more tokens to achieve comparable performance, practitioners face a fundamental tradeoff: should they deploy larger models with conservative token budgets, or smaller models with generous budgets? Moreover, if token budgets exhibit diminishing returns or saturation effects, over-allocating compute could waste resources, or potentially harm performance.

We address these questions through a systematic study of test-time compute scaling across four Qwen model sizes (0.5B, 1.5B, 3B, and 7B parameters) on the GSM8K mathematical reasoning benchmark (Cobbe et al., 2021). We evaluate each model at seven token budgets ranging from 32 to 2048 tokens, generating 36,932 total model outputs. Our analysis reveals three key findings:

- **Performance cliff**: All models show their largest accuracy gains at the 128→256 token transition, suggesting a fundamental threshold for mathematical reasoning regardless of model scale.
- **Inverse scaling of saturation points**: Larger models saturate at lower token budgets while achieving higher peak accuracy. The 7B model reaches 86.8% at 512 tokens, while the 0.5B model achieves only 18.7% even at 1024 tokens.
- **Performance degradation from over-allocation**: Beyond saturation points, additional tokens can reduce accuracy. The 1.5B model loses 2.4% when increasing

from 512 to 1024 tokens, suggesting that excessive compute budgets introduce errors or hallucinations.

These findings have immediate practical implications for LLM deployment: optimal token allocation must be tailored to model scale, and conservative budgets often outperform generous ones. Our work extends recent research on test-time compute efficiency (Han et al., 2025; Snell et al., 2024) by characterizing saturation phenomena across model scales and demonstrating when more compute actively hurts performance.

2 Related Work

Scaling Laws for Language Models The performance of language models has been shown to follow predictable scaling laws with respect to model size, dataset size, and training compute (Kaplan et al., 2020; Hoffmann et al., 2022). Recent work has extended this framework to test-time compute, demonstrating that models can improve their outputs by using additional inference-time computation (Snell et al., 2024). Snell et al. (2024) showed that smaller models with optimized test-time compute can outperform larger models in FLOPs-matched evaluations, suggesting that the traditional focus on pre-training scale may overlook important efficiency opportunities.

Chain-of-Thought Reasoning Chain-of-thought (CoT) prompting (Wei et al., 2022) enables language models to solve complex reasoning tasks by generating intermediate steps. Subsequent work has explored various refinements, including self-consistency (Wang et al., 2023), least-to-most prompting (Zhou et al., 2022), and tree-of-thoughts (Yao et al., 2023). These methods implicitly allocate variable amounts of test-time compute but typically do not explicitly control or optimize token budgets.

Token-Budget-Aware Reasoning Most directly related to our work, Han et al. (2025) introduced TALE, a framework for token-budget-aware LLM reasoning. They demonstrated that models can effectively compress their reasoning when given explicit token budgets, achieving 68.6% token reduction with minimal accuracy loss. Wang et al. (2024) studied budget-aware evaluation of reasoning strategies, finding that simple baselines like self-consistency often outperform more complex methods when compute budgets are controlled. However, neither work systematically studied how optimal budgets vary across model scales.

Recent work on "thinking-optimal scaling" (Yang et al., 2025) found that excessive reasoning length can impair performance, particularly on easier tasks. Our findings complement this work by characterizing saturation phenomena across multiple model sizes and demonstrating that the relationship between model scale and optimal token budget is non-trivial. While Yang et al. (2025) focused on training data curation, we examine saturation effects in pre-trained models during inference.

Mathematical Reasoning Benchmarks GSM8K (Cobbe et al., 2021) is a widely-used benchmark of grade-school math word problems designed to test multi-step reasoning. It has become a standard evaluation for studying chain-of-thought capabilities and test-time compute scaling (Wei et al., 2022; Han et al., 2025; Snell et al., 2024). Our choice to focus on GSM8K enables direct comparison with prior work while studying a task where reasoning token allocation has clear interpretability.

3 Experimental Setup

3.1 Models and Configuration

We evaluate four models from the Qwen2.5-Instruct family (Team, 2024), spanning a 14× range in parameter count: Qwen2.5-0.5B, Qwen2.5-1.5B, Qwen2.5-3B, and Qwen2.5-7B. All models are quantized to 4-bit precision using bitsandbytes (Dettmers et al., 2022) to enable efficient inference on consumer hardware. We use the models' default instruction-following format with chain-of-thought prompting.

3.2 Token Budget Configuration

We evaluate seven token budgets (32, 64, 128, 256, 512, 1024, 2048) that limit the maximum length of each model's generated reasoning chain. All generations use greedy decoding (temperature=0) for deterministic outputs with the prompt: *"Let's solve this step by step."*

3.3 Dataset and Evaluation

We use the validation split of the GSM8K dataset (Cobbe et al., 2021), which contains 1,319 grade-school math word problems. Each problem requires multi-step arithmetic reasoning and has a numeric answer.

For evaluation, we extract the final numeric answer from each model's generated output using regular expressions to identify the last number in the response. We compute accuracy as exact match

between the extracted answer and the ground truth. This yields 36,932 total model outputs (4 models × 7 budgets × 1,319 problems).

3.4 Metrics

We analyze the following metrics:

Accuracy vs Budget: For each model and token budget, we compute the percentage of problems solved correctly.

Saturation Point: The minimum token budget at which a model reaches within 1% of its maximum accuracy across all budgets.

Performance Cliff: The budget transition with the largest absolute accuracy gain for each model.

Parameter Efficiency: Accuracy divided by model size (in billions of parameters), measured at specific token budgets to compare efficiency across model scales.

4 Results

We present our findings in three parts: overall accuracy trends across model scales and token budgets, saturation phenomena, and efficiency analysis.

4.1 Performance Cliff at 256 Tokens

Figure 1 shows accuracy as a function of token budget for all four models. Interestingly, all models exhibit their largest performance gain at the same transition: from 128 to 256 tokens. The magnitude of this jump varies by model size, Qwen-0.5B gains +8.0%, Qwen-1.5B gains +23.3%, Qwen-3B gains +44.4%, and Qwen-7B gains +50.9%, but the location of the cliff is consistent.

Below 128 tokens, all models perform near chance level (<10% accuracy), suggesting this represents a minimum viable budget for multi-step mathematical reasoning. The universal nature of this threshold across model scales suggests that it reflects a fundamental requirement of the task rather than a model-specific characteristic.

4.2 Saturation Points Scale Inversely with Model Size

Although all models show the performance cliff at 256 tokens, their saturation behavior differs dramatically. Table 1 summarizes the saturation analysis. The 7B model reaches its peak accuracy of 86.8% at just 512 tokens and shows no improvement with additional budget. In contrast, the 0.5B model continues to improve through 1024 tokens but achieves only 18.7% accuracy at its peak.

Critically, larger models not only achieve higher maximum accuracy but also saturate at *lower* token budgets. The 7B model peaks at 512 tokens, while the 1.5B model peaks at 512 tokens, and the 0.5B model requires 1024 tokens. The 3B model shows an intermediate pattern, continuing to improve slightly through 1024 tokens (79.2%).

Model	Max Acc.	Sat. Budget	Acc. @ 2048
Qwen-0.5B	18.7%	1024	18.1%
Qwen-1.5B	49.6%	512	48.1%
Qwen-3B	79.2%	1024	78.9%
Qwen-7B	86.8%	512	86.1%

Table 1: Saturation analysis across model scales. Larger models achieve higher peak accuracy at lower token budgets. Accuracy at 2048 tokens shows degradation or stagnation for all models.

Figure 1: Accuracy vs token budget across model scales. All models exhibit a performance cliff at the 128→256 token transition, but larger models achieve higher peak accuracy and saturate at lower budgets.

4.3 Performance Degradation from Over-Allocation

Perhaps most surprisingly, three of the four models show performance *degradation* when budgets are given beyond their saturation point (Table 1). The 1.5B model exhibits the largest drop, losing 2.4% accuracy when increasing from 512 to 1024 tokens (49.6% → 47.2%). The 0.5B model loses 0.6% from 1024 to 2048 tokens, and the 3B model loses 0.3%. The 7B model shows minimal degradation, dropping only 0.7% from 512 to 1024 tokens before stabilizing, compared to the larger drops seen in smaller models.

This degradation suggests that excessive token budgets allow models to introduce errors through hallucination, incorrect reasoning steps, or premature conclusions that are later contradicted. The effect is most pronounced in smaller models, which may lack the capacity to maintain consistency over longer reasoning chains.

4.4 Parameter Efficiency Analysis

Figure 2 shows accuracy per billion parameters at different token budgets. At 256 tokens, just past the

performance cliff, smaller models are surprisingly efficient: the 0.5B model achieves 0.32 accuracy per billion parameters, compared to 0.09 for the 7B model. However, this efficiency is misleading: the 0.5B model's absolute accuracy of 16.0% is far below practical utility.

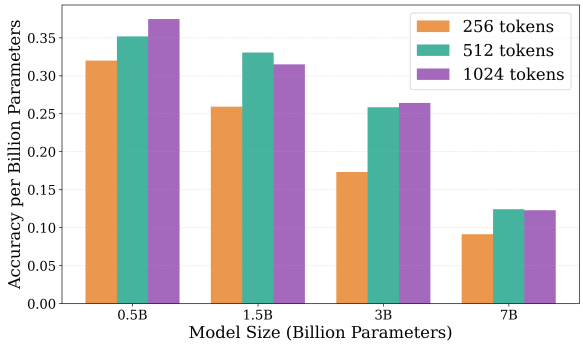

Figure 2: Accuracy per billion parameters at different token budgets. Smaller models show higher parameter efficiency but reach lower absolute accuracy.

At 512 tokens, the efficiency gap narrows (0.35 vs 0.12 accuracy/billion parameters) as larger models approach saturation while smaller models continue scaling. This suggests a fundamental trade-off: smaller models extract more performance per parameter at any given budget, but larger models reach higher absolute performance with less total compute (fewer tokens needed).

4.5 Implications for Deployment

Our findings reveal a clear deployment strategy: match token budgets to model scale. For the 7B model, allocating more than 512 tokens wastes compute without improving (or potentially harming) accuracy. For the 1.5B model, budgets beyond 512 tokens actively degrade performance. Practitioners should empirically determine saturation points for their specific models rather than assuming more compute is always beneficial.

The performance cliff at 256 tokens also provides a useful heuristic: any model requires at least 128-256 tokens for mathematical reasoning tasks. Budgets below this threshold are unlikely to succeed regardless of model scale.

5 Discussion and Conclusion

5.1 Key Findings and Implications

Our systematic study of test-time compute scaling across model sizes reveals three critical insights. First, the performance cliff at 256 tokens suggests that mathematical reasoning has an inherent minimum compute requirement that transcends model scale. This threshold likely reflects the number of tokens needed to decompose multi-step problems into intermediate reasoning steps.

Second, the inverse relationship between model size and saturation budgets challenges naive assumptions about compute allocation. Larger models achieve superior performance with *fewer* tokens, not more. This suggests that model capacity and test-time compute are not independent scaling dimensions, larger models can reason more efficiently within tighter computational constraints.

Third, the performance degradation beyond saturation points demonstrates a critical failure mode that practitioners must avoid. Over-allocating tokens can actively harm model performance, particularly for smaller models that may lack the capacity to maintain consistency over extended reasoning chains. This finding aligns with recent work on thinking-optimal scaling (Yang et al., 2025), which observed similar degradation effects in training data.

5.2 Comparison to Prior Work

Our results extend the TALE framework (Han et al., 2025) in several ways. While TALE demonstrated that models can compress reasoning with explicit budgets, we show that optimal compression varies systematically with model scale. TALE reported 68.6% token reduction with <5% accuracy loss; our findings suggest the achievable reduction depends critically on model size, the 7B model needs far fewer tokens than the 0.5B model to reach comparable (relative) performance.

Our work also complements budget-aware evaluation studies (Wang et al., 2024), which compared reasoning strategies within fixed compute budgets. We demonstrate that the *optimal* budget itself must be determined as a function of model scale, adding a new dimension to budget-aware analysis.

5.3 Conclusion

Test-time compute scaling is not uniform across model sizes. Our findings demonstrate that optimal token budgets must be tailored to model scale: larger models saturate faster and can degrade with excessive budgets, while smaller models require more tokens but still achieve lower peak performance. The performance cliff at 256 tokens provides a useful lower bound for mathematical reasoning tasks, while our saturation analysis offers practical guidance for deployment. Future work on test-time compute should account for model scale as a critical variable in determining optimal compute allocation strategies.

Limitations

Our study focuses on a single model family (Qwen) and task domain (mathematical reasoning). While GSM8K is a standard benchmark, saturation behaviors may differ for other reasoning tasks or model architectures. Future work should investigate whether similar patterns hold for code generation, logical reasoning, or open-ended question answering.

Moreover, we do not explore adaptive budget allocation within a single inference pass. While we identify optimal static budgets for each model, dynamic approaches that adjust budgets per-problem could further improve efficiency. Recent work on early stopping criteria (Snell et al., 2024) could be combined with our saturation analysis to develop such adaptive strategies.

Finally, our analysis is limited to greedy decoding (temperature=0). Stochastic sampling methods like self-consistency (Wang et al., 2023) may exhibit different saturation behaviors, as they aggregate multiple reasoning paths rather than generating a single chain of thought.

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*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems*, volume 35, pages 30318–30332. Curran Associates, Inc.

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.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, and 1 others. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Junlin Wang, Siddhartha Jain, Dejiao Zhang, Baishakhi Ray, Varun Kumar, and Ben Athiwaratkun. 2024. Reasoning in token economies: budget-aware evaluation of llm reasoning strategies. *arXiv preprint arXiv:2406.06461*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards thinking-optimal scaling of test-time compute for llm reasoning. *arXiv preprint arXiv:2502.18080*.

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.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and 1 others. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.