Less is More: Improving LLM Reasoning with Minimal Test-Time Intervention

Anonymous ACL submission

Abstract

Recent progress in large language models (LLMs) has focused on *test-time scaling* to improve reasoning via increased inference computation, but often at the cost of efficiency. We revisit test-time behavior and uncover a simple yet underexplored phenomenon: reasoning uncertainty is highly localized—only a small subset of high-entropy tokens dominantly affects output correctness. Motivated by this, we propose **Minimal Test-Time Intervention (MTI)**, a training-free framework that enhances reasoning accuracy and stability with minimal overhead. MTI includes: (i) *Selective CFG intervention*, applying classifier-free guidance only at uncertain positions; and (ii) *Lightweight negative-prompt guidance*, reusing the main model's KV cache to approximate unconditional decoding efficiently. MTI yields consistent gains across general, coding, and STEM tasks—e.g., +9.28% average improvement on six benchmarks for DeepSeek-R1-7B and +11.25% on AIME2024 using Ling-mini-2.0—while remaining highly efficient. The anonymous vLLM version code can be found here.

1 Introduction

Large language models (LLMs) (Achiam et al., 2023; Team et al., 2024; Dubey et al., 2024; Guo et al., 2025a; Yang et al., 2025a; Lozhkov et al., 2024; Young et al., 2024) have advanced rapidly, with ever-increasing parameter counts, larger pretraining corpora, and improved optimization techniques leading to steady gains across benchmarks. Beyond such *training-time scaling*, recent research has increasingly focused on *test-time scaling*—allocating more computation during inference through deeper reasoning or broader search. Methods such as chain-of-thought prompting (Wei et al., 2022), self-consistency (Wang et al., 2022), and frameworks like Reflexion (Shinn et al., 2023) or Tree of Thoughts (Yao et al., 2023) demonstrate that increasing inference-time computation can enhance reasoning quality and output accuracy. However, these methods achieve better performance *by trading inference efficiency for computation*, as they require multiple reasoning trajectories, extensive sampling, or long multi-step deliberation, making them computationally expensive and sometimes impractical in real-world deployments.

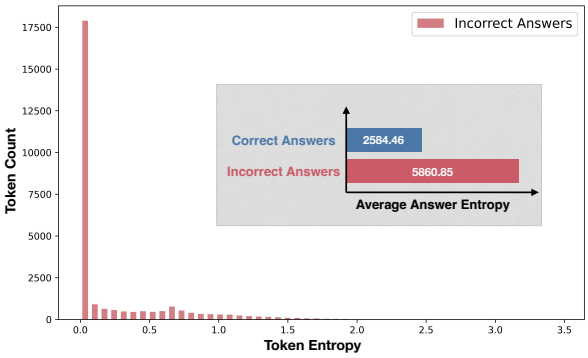


Figure 1: Incorrect answers exhibit higher average answer entropy than correct ones, primarily due to high-entropy tokens.

In this work, we challenge the prevailing assumption that improving test-time performance of large language models must inherently come at the cost of increased inference computation. To examine this, we conduct an empirical analysis on the AIME2024 benchmark, comparing reasoning trajectories that yield correct versus incorrect answers. Interestingly, as shown in the inset panel in the upper-right corner of in Fig. 1, correct responses exhibit significantly lower average answer entropy than incorrect ones, indicating that reasoning failures are associated with higher uncertainty. Furthermore, as illustrated in the main panel in Fig. 1, only a small fraction of tokens within each response have notably high entropy, yet these few tokens contribute a disproportionate share of the average answer entropy. This observation suggests that reasoning errors are not uniformly distributed

across the sequence but are instead concentrated in a few high-entropy "critical steps", where local uncertainty can propagate and destabilize subsequent predictions. We hypothesize that by selectively stabilizing these high-entropy regions, one can suppress error amplification and improve overall reasoning accuracy and consistency, all while intervening on only a minimal subset of tokens.

Building on this observation, we propose **Minimal Test-Time Intervention (MTI)**, a training-free approach that improves reasoning performance without incurring substantial inference cost. Specifically, MTI first detects high-entropy tokens by measuring the dispersion of the model's predictive distribution during generation. High entropy indicates uncertainty in token prediction, and thus a greater likelihood that local instability may propagate through the autoregressive chain. To reduce this uncertainty, MTI draws inspiration from *classifier-free guidance (CFG)* (Sanchez et al., 2023), which combines conditional and unconditional predictions to steer model outputs toward more task-aligned regions of the probability space. However, directly applying standard CFG would require maintaining a separate unconditional branch, resulting in computational and memory overhead.

To address this, MTI introduces a lightweight alternative: before generating a high-entropy token, the model temporarily augments its input with the short phrase "`OUTPUT ERROR`", which implicitly constructs an on-the-fly unconditional contrastive branch for CFG-style guidance. This branch reuses the main model's KV cache, adding only two auxiliary tokens per high-entropy position for the CFG computation. Since high-entropy tokens constitute only a small fraction of the overall sequence, the additional overhead is negligible. In practice, this design effectively shifts the model's predictive distribution toward the correct region of the probability space, enhancing reasoning stability and accuracy with negligible test-time computation.

In general, our main contributions are as follows:

- We find that high-entropy tokens are the main sources of error in LLM reasoning. Detecting them via token entropy and intervening selectively improves reasoning performance.
- We introduce a training-free test-time intervention based on CFG that selectively applies guidance to high-entropy tokens, yielding better answers with minimal overhead. We further implement negative-prompt injection via KV-cache reuse, which shrinks the unconditional branch's KV-cache allocation and more faithfully approximates the unconditional distribution used by CFG.
- We achieve robust performance across general, coding, and STEM benchmarks, showing consistent improvements over direct inference across the latest LLMs of varying scales. Specifically, our method yields a 9.28% average gain on six comprehensive tasks with DeepSeek-R1-7B and a +11.25% improvement on the AIME2024 benchmark with Ling-mini-2.0, demonstrating its broad effectiveness across different tasks.

2 Related Works

Test-time optimizations for LLMs. Test-time optimizations for LLMs. Recent research focuses on test-time or decoding-time optimizations to enhance LLM reasoning, robustness, or efficiency without retraining (Chen et al., 2025a). One line of research improves reasoning capabilities through prompt engineering and advanced sampling strategies, such as Chain-of-Thought (CoT) (Wei et al., 2022), self-consistency (Wang et al., 2022). Building on these, search and evaluation frameworks have been proposed, including Reflexion (Shinn et al., 2023), Tree of Thoughts (Yao et al., 2023), DeepConf (Fu et al., 2025), and EGB (Li et al., 2025b). Despite their effectiveness in enhancing semantic robustness, these methods often incur significant sampling and memory overhead (Sanchez et al., 2023). Furthermore, hallucination mitigation strategies such as Contrastive Decoding (Li et al., 2023; Lee et al., 2025b) introduce additional complexity by incorporating an auxiliary small model. Another direction targets computational efficiency via speculative decoding (Leviathan et al., 2023; Chen et al., 2025b; Yan et al., 2024; Zhang et al., 2023) or memory optimizations like FlashAttention (Dao et al., 2022). Unlike these computationally intensive approaches, our method selectively intervenes only on high-entropy tokens, achieving significant performance improvements with negligible overhead. Furthermore, our framework is highly flexible, serving as a plug-and-play decoding strategy that can be seamlessly combined with existing reasoning techniques. We further discuss its connection with other decoding strategies in Sec. A.7.

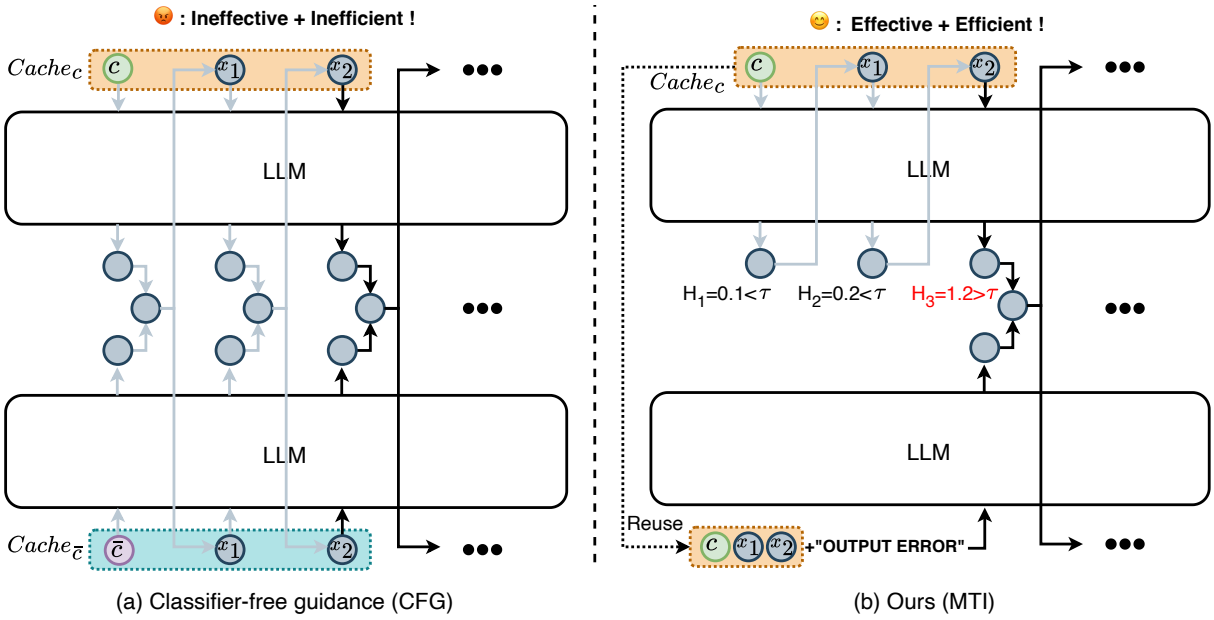


Figure 2: Comparison between CFG and MTI. MTI applies CFG exclusively to high-entropy tokens and reuses the conditional KV cache via negative prompt injection.

Classifier-free guidance in LLMs. Originally developed for diffusion models, Classifier-Free Guidance (CFG) (Ho and Salimans, 2022) balances fidelity and diversity by interpolating between conditional and unconditional predictions. This mechanism can also incorporate negative prompts to steer generation away from undesirable attributes, such as content errors (Koulischer et al., 2024) or artifacts (Miyake et al., 2025; Yang et al., 2023). However, adapting CFG to LLMs (Sanchez et al., 2023) faces two primary hurdles: the lack of a well-defined unconditional space—rendering empty negative prompts poorly calibrated—and the huge computational and memory overhead. We address these by applying CFG selectively to high-entropy tokens and introducing negative-prompt injection via KV-cache reuse, effectively eliminating redundant cache allocation and computational inefficiency while improving unconditional space approximation to suppress erroneous content generation.

Entropy-based token selection in LLMs. Recent studies emphasize that LLM tokens exhibit varying uncertainty, with token-level entropy serving as a reliable metric for identifying critical decision boundaries. While prior work leverages low-entropy tokens for redundancy pruning (Li et al., 2025c; Xia et al., 2025) or focuses on high-entropy tokens for optimization (Wang et al., 2025; Lee et al., 2025a; Qian et al., 2025), these methods typically require retraining or heuristic-based pruning. In contrast, our approach is entirely training-free and operates at inference-time. Specifically, we leverage token entropy to determine where to apply CFG—only to tokens with high entropy.

3 Method

3.1 Preliminary

We first introduce the notation used throughout this paper. Let $P(\cdot)$ denote the token distribution predicted by the LLM. Given a context c, the conditional probability of generating a token x is denoted by $P(x \mid c)$. In the setting of classifier-free guidance (CFG), $\bar{c}$ represents a negative prompt in the unconditional branch. The CFG-adjusted distribution is denoted by $\hat{P}(\cdot)$, where ω controls the strength of guidance. We use l_t to denote the logits predicted at generation step t, x_t denotes the token generated at step t, and τ denotes the entropy threshold.

Classifier-Free Guidance in LLMs. Originally proposed for diffusion models, classifier-free guidance (Ho and Salimans, 2022) improves conditional generation by interpolating between conditional and unconditional predictions. In diffusion models, an additional null condition is introduced during training to learn the global data distribution. In contrast, for LLMs, CFG is applied purely at inference time without requiring additional training objectives (Sanchez et al., 2023). The guided prediction is formulated in the log-probability space

as:

$$\begin{aligned}&\log \hat{P}\left(x_t \mid c, \bar{c}, x_{<t}\right)\\&= (1-\omega)\cdot \log P\left(x_t \mid \bar{c}, x_{<t}\right)\\&\quad + \omega \cdot \log P\left(x_t \mid c, x_{<t}\right),\end{aligned} \tag{1}$$

where $x_{<t}$ denotes the sequence of previously generated tokens up to step $t-1$. The negative prompt $\bar{c}$ encourages the model to generate outputs that deviate from the unconditional distribution.

Token and answer entropy. We adopt Shannon entropy to quantify the model's uncertainty over predicted logits. Lower entropy indicates higher confidence, while higher entropy reflects greater uncertainty. The token entropy at step t is defined as:

$$H_t = -\sum_{i=1}^{V} p_i \log p_i, \tag{2}$$

where $[p_1, \ldots, p_V]_t = \mathrm{softmax}(l_t)$, V denotes the vocabulary size, and p_i is the probability assigned to the i-th token.

The answer-level entropy for the n-th question is defined as the sum of token entropies along its full reasoning trajectory:

$$A_n = \sum_{t=1}^{T_n} H_t, \tag{3}$$

where T_n is the length of the generated trajectory. The average answer entropy A_{avg} over N data instances is given by:

$$A_{avg} = \frac{1}{N}\sum_{n=1}^{N} A_n. \tag{4}$$

KV Cache in LLM-CFG. At generation step t, the key-value (KV) cache avoids recomputation of full-prefix attention by reusing previously stored keys and values. Under LLM-CFG, two separate KV caches are maintained—one for conditional prediction and one for unconditional prediction—to enable efficient guided inference. Specifically, the caches used to compute the outputs for the two branches at step t are:

$$\begin{aligned}\mathrm{Cache}_c &= [c, x_1, \ldots, x_{t-1}],\\ \mathrm{Cache}_{\bar{c}} &= [\bar{c}, x_1, \ldots, x_{t-1}].\end{aligned} \tag{5}$$

For simplicity, we do not distinguish between keys and values; for example, c denotes the KV pairs cached after processing the prompt c. Figure 2 follows the same notation.

3.2 Selective CFG intervention

LLMs often display chain instability in multi-step reasoning: uncertainty at a few steps amplifies and derails the entire answer. To locate where this instability originates, the inset panel in the upper-right corner of Fig. 1 shows that questions answered incorrectly have markedly higher average answer entropy than those answered correctly. The main panel further indicates that the gap is driven by high-entropy tokens in erroneous responses. These findings suggest that overall failure is concentrated at a small set of critical high-entropy nodes and that stabilizing these nodes can yield gains.

Based on this observation, Fig. 2b proposes a selective CFG intervention strategy. During decoding, we monitor token entropy and intervene only when it exceeds a threshold τ: tokens with entropy $\leq \tau$ proceed normally, whereas tokens with entropy $> \tau$ receive CFG to reduce local uncertainty and prevent error propagation. Compared with the vanilla CFG baseline in Fig. 2a, which applies guidance uniformly at every step, our approach activates selectively at unstable nodes, concentrating guidance where it matters most and improving both stability and efficiency.

3.3 Lightweight negative-prompt guidance

However, as illustrated in Fig. 2a, although selective CFG intervention reduces the frequency of application, it still requires maintaining two separate sets of KV caches because the contexts of the conditional and unconditional branches differ. This dual-KV cache mechanism substantially degrades inference efficiency in modern LLM accelerators and weakens the long-context capabilities of frameworks such as `vllm` . Furthermore, CFG originated in the field of diffusion model, where a specific "null condition" is typically trained on global data to optimize the unconditional branch. In the context of LLMs, however, CFG lacks this explicit training process, rendering the control capability of its unconditional branch inherently unstable.

To address these challenges, as shown in Fig. 2b, we propose to reuse the KV cache of the conditional branch and append a short negative prompt. This KV reuse mechanism eliminates the need for a separate unconditional KV cache, thereby significantly reducing the memory footprint. To tackle the instability of the unconditional branch space, we draw inspiration from negative prompting in diffusion model and construct a negative unconditional

branch to steer the model away from erroneous tokens. Specifically, we inject short negative cues (e.g., "`OUTPUT ERROR`") into the reused KV cache, forcing the unconditional branch to generate an undesirable probability distribution. This distribution is then used to perturb high-entropy nodes in the conditional branch, thereby refining the generation results. While we use "`OUTPUT ERROR`" as a general-purpose cue, this prompt can be further tailored to specific tasks for even greater precision. Nevertheless, our experiments show that even this simple two-word cue yields robust performance gains across various datasets.

4 Experiments

4.1 Experimental setup

Datasets. To assess robustness across diverse domains, we evaluate on three categories of benchmarks: general tasks (MMLU-Pro (Wang et al., 2024)); coding tasks (HumanEval (Chen et al., 2021), HumanEvalPlus, LiveCodeBench (Jain et al., 2024)); and STEM tasks (GPQA-Diamond (Rein et al., 2024), MATH500 (Lightman et al., 2023), AIME2024). We use `OpenCompass`[1] (Contributors, 2023) for consistent and fair evaluation; the corresponding test configuration files and all testing details are provided in Appendix A.1.

Model selection. To evaluate the effectiveness of our method, we experiment with the Qwen3 (Yang et al., 2025a) model family (Qwen3-8B, Qwen3-14B, and Qwen3-32B), DeepSeek-R1-7B (Guo et al., 2025b) and Ling-mini-2.0 (Li et al., 2025a).

Baselines. We compare against the following baselines: (1) Direct Inference (DI): generate outputs directly from the LLM; (2) Vanilla CFG with negative prompt (VC): classifier-free guidance with a specified negative prompt, set to the same prompt injected in our method. Following vanilla CFG (Sanchez et al., 2023), we set ω to 1.5 and use it as the default hyperparameter. Additionally, a detailed comparison between MTI and current SOTA test-time scaling methods is provided in Tab. 5.

Implementation details. All experiments set context length of 32768 to ensure that the models' full capabilities are preserved in all datasets. To evaluate the effectiveness of MTI and to further examine its ability to correct the model's probability distribution—where changes to the top-1 prediction are the most directly observable—while ensuring reproducibility, we uniformly adopt greedy search as the base decoding strategy in Tab. 1. Additionally, to validate the robustness of our method under stochastic decoding, we evaluate it on AIME2024 using random sampling with temperature 0.6, top-p 0.95, and top-k 20. We run eight independent tests and report the average performance in Tab. 2.

[1] https://github.com/open-compass/opencompass

4.2 Main Results

Experimental results in Tab. 1 and Tab. 2 show that MTI consistently outperforms Direct Inference (DI) and Vanilla CFG (VC). Notably, MTI exhibits a broad "green zone" where it surpasses DI across nearly all threshold configurations, proving its robustness without meticulous tuning.

As shown in Tab. 1, on DeepSeek-R1-7B, MTI reaches an average score of 70.73% (+9.28% over DI) with 21.8% CFG usage, and on Qwen3-14B, MTI reaches an average score of 85.33% (+3.00% over DI) with 32.8% CFG usage. Crucially, MTI rectifies erroneous distributions where traditional strategies fail. For instance, unlike rank-preserving methods such as Temperature sampling, MTI reshapes the distribution by re-ranking logits to elevate suppressed candidates. For example, on GPQA-Diamond with DeepSeek-R1-7B, DI with greedy decoding leads to highly repetitive generations, resulting in a poor score of 29.29%. By contrast, MTI corrects the erroneous top-1 logits, improving performance to 51.52% (+22.23%).

On AIME2024 (Table 2), MTI demonstrates its effectiveness under random sampling, boosting DeepSeek-R1-7B from 54.17% to 62.92% (+8.75%) and Ling-mini-2.0 from 60.00% to 71.25% (+11.25%). In contrast, VC often underperforms DI in AIME2024 (e.g., 73.34% vs. 73.75% on Qwen3-8B). As analyzed in Table 6, applying CFG in low-entropy scopes (≤ 1.5) actually degrades performance to 71.67%, whereas targeting high-entropy states (> 1.5) yields 78.34%. This suggests that in long-form reasoning, excessive intervention during high-confidence steps disturbs the model's logical flow, leading to cumulative errors. By bypassing these states, MTI mitigates error propagation and achieves competitive performance against SOTA methods in Table 5. These results validate MTI as a scalable and reliable approach for enhancing LLM reasoning without the need for exhaustive hyperparameter search.

Model	Method	τ	MMLU-P	GPQA-D	MATH500	HEval	HEval+	LiveCodeBench	Avg.	CFG Usage
DeepSeek-R1-7B	DI	-	44.45	29.29	84.60	79.27	71.34	42.75/80.17	61.70	-
	VC	-	47.76	32.32	92.40	81.10	73.78	47.75/81.21	65.19	100%
	Ours	0.1	53.89	45.96	92.80	**89.02**	78.05	51.75/79.96	70.20	31.4%
		0.5	**54.29**	**51.52**	**93.60**	85.98	77.44	**54.25**/79.75	**70.98**	21.8%
		1.0	53.47	48.48	91.80	81.71	**81.71**	52.00/79.75	69.85	12.7%
		1.5	52.31	40.91	93.20	81.71	73.17	50.50/80.79	67.51	9.3%
		2.0	50.41	38.89	90.60	80.49	75.61	48.00/**81.84**	66.55	3.6%
Ling-mini-2.0	DI	-	63.93	52.02	94.00	87.20	75.00	57.00/38.62	66.82	-
	VC	-	63.48	53.54	**95.80**	85.37	78.66	56.50/40.29	67.66	100%
	Ours	0.1	63.77	**56.57**	94.80	**87.80**	78.66	56.50/42.80	**68.70**	28.2%
		0.5	**64.54**	52.02	94.80	86.59	**79.27**	57.25/**45.09**	68.51	14.8%
		1.0	63.86	52.02	95.00	86.59	76.22	55.00/41.75	67.21	4.5%
		1.5	63.18	54.04	94.60	87.20	76.22	57.50/39.46	67.46	0.9%
		2.0	63.81	51.52	95.20	87.20	75.00	**58.00**/39.67	67.20	0.2%
Qwen3-8B	DI	-	70.52	57.07	96.80	87.20	67.07	79.50/97.70	79.41	-
	VC	-	70.19	57.58	92.60	90.85	65.85	84.75/93.74	79.37	100%
	Ours	0.1	70.69	54.55	96.80	92.07	**74.39**	**88.00**/96.45	81.85	50.5%
		0.5	71.91	60.61	**97.00**	**95.12**	69.51	87.25/**98.12**	**82.79**	30.5%
		1.0	72.28	60.10	**97.00**	90.85	67.07	**88.00**/97.91	81.89	9.8%
		1.5	**72.38**	**61.11**	95.80	89.02	65.24	84.75/97.49	80.83	3.3%
		2.0	70.96	56.06	96.80	88.41	65.85	81.50/97.70	79.61	0.4%
Qwen3-14B	DI	-	75.61	57.58	96.40	92.07	70.12	86.00/98.54	82.33	-
	VC	-	74.12	60.10	94.80	90.85	69.51	89.00/**98.96**	82.48	100%
	Ours	0.1	74.86	**64.14**	97.00	95.73	72.56	90.25/97.91	84.64	48.6%
		0.5	74.63	**64.14**	97.20	**96.95**	75.00	**91.50**/97.91	**85.33**	32.8%
		1.0	**76.12**	62.12	97.00	95.12	**75.61**	90.50/97.49	84.85	10.3%
		1.5	75.91	62.12	**97.60**	95.73	69.51	88.50/97.70	83.87	3.5%
		2.0	75.64	61.62	97.20	93.90	70.73	88.00/98.33	83.63	0.3%
Qwen3-32B	DI	-	76.76	61.62	97.00	97.56	97.56	91.00/**99.16**	88.67	-
	VC	-	74.96	56.57	94.00	95.12	95.73	87.25/98.54	86.02	100%
	Ours	0.1	76.56	64.65	**98.40**	96.34	97.56	**92.00**/98.75	89.18	49.1%
		0.5	77.15	**65.66**	**98.40**	**98.17**	**98.17**	90.75/98.54	89.55	31.2%
		1.0	**78.47**	**65.66**	97.40	**98.17**	97.56	91.25/**99.16**	**89.67**	13.1%
		1.5	77.80	60.61	96.20	97.56	97.56	89.75/98.75	88.32	3.4%
		2.0	76.86	61.62	96.00	96.34	**98.17**	90.00/98.75	88.25	0.7%

Table 1: Performance comparison on general, coding and STEM benchmarks. We compare Direct Inference (DI), Vanilla CFG (VC), and Ours (MTI) under varying entropy thresholds (τ). Avg. (%) and CFG Usage denote the average performance and CFG usage ratio, respectively. Green cells indicate cases where Ours outperforms DI, and Ours consistently surpasses both DI and VC.

Method	τ	DS-R1-7B	Ling-mini-2.0	Qwen3-8B	Qwen3-14B	Qwen3-32B
DI	–	54.17 / –	60.00 / –	73.75 / –	78.33 / –	79.58 / –
VC	–	48.33 / 100%	65.00 / 100%	73.34 / 100%	75.42 / 100%	72.92 / 100%
Ours	0.1	57.08 / 51.6%	65.00 / 15.8%	76.60 / 37.8%	79.58 / 40.0%	82.92 / 41.7%
	0.5	51.67 / 36.9%	68.34 / 7.6%	77.08 / 22.5%	80.83 / 24.4%	**84.58** / 25.7%
	1.0	**62.92** / 21.1%	**71.25** / 1.4%	75.42 / 8.0%	**81.67** / 8.2%	82.92 / 9.0%
	1.5	60.00 / 13.2%	63.33 / 0.2%	**78.34** / 1.9%	78.33 / 2.0%	82.50 / 2.2%
	2.0	58.75 / 6.9%	65.00 / 0.01%	75.00 / 0.3%	80.00 / 0.3%	77.92 / 0.3%

Table 2: AIME2024 accuracy (%) and CFG usage for different models and methods. Results are averaged over 8 random sampling runs. Green cells indicate cases where Ours (MTI) outperforms DI, and Ours consistently surpasses both DI and VC.

4.3 Ablation Study

Entropy threshold τ. Fig. 3a shows the ablation study on τ, with the curve representing the average accuracy from DeepSeek-R1-7B results in Tab. 1. As τ increases, performance follows a unimodal pattern, rising initially before declining. A low τ over-modifies low-entropy tokens, destabilizing the reasoning chain, while a high τ leaves high-entropy tokens uncorrected, degrading performance. The optimal τ can be determined via cross-validation on a held-out dataset. Additionally, the "green zone" in Tab. 1 reveals that MTI consistently outperforms DI across most threshold configurations, demonstrating robustness without extensive tuning. As shown in Tab. 1 and Sec. 5, at an entropy threshold of 0.5, MTI outperforms all comparison methods.

CFG hyperparameters ω. Fig. 3b investigates the impact of ω on the average accuracy of DeepSeek-R1-7B across the GPQA-Diamond, Math500, and HumanEvalPlus datasets, following

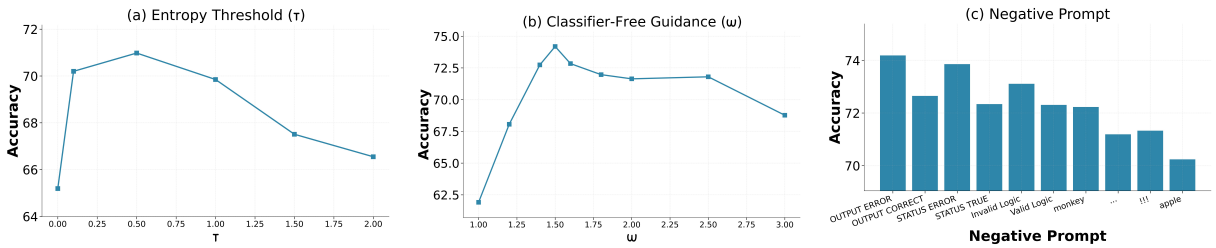


Figure 3: Ablation study results on the selection of (a) entropy threshold, (b) ω in classifier-free guidance and (c) negative prompt in MTI.

the experimental settings in Tab. 1, where the performance exhibits an initial increase followed by a subsequent decline. The model attains its peak accuracy when $\omega = 1.5$. Notably, the performance is fairly stable around this value, indicating that ω is not sensitive and typically does not require extensive tuning.

Negative prompt. Fig. 3c shows the impact of negative prompts on the average accuracy of DeepSeek-R1-7B across the GPQA-Diamond, Math500, and HumanEvalPlus datasets. Experimental results reveal two key observations. First, negative prompts (e.g., `OUTPUT ERROR`, `Invalid Logic`) consistently outperform positive prompts (e.g., `OUTPUT CORRECT`, `Valid Logic`). Second, negative prompts that are semantically grounded perform significantly better than meaningless or weakly informative tokens (e.g., `apple`, `!!!`). These findings confirm that, within the MTI framework, negative prompts effectively guide the model away from undesirable conditional distributions. By explicitly encoding errors or logical failures, these prompts suppress the generation probability of incorrect tokens more reliably. In contrast, meaningless tokens introduce only unspecific perturbations, which makes them less effective. While task-specific negative prompts can yield additional improvements (Appendix Sec. A.6), `OUTPUT ERROR` demonstrates strong universality and robustness, making it a powerful, general-purpose heuristic across tasks.

4.4 Analysis

Not all tokens are created equal in CFG. To investigate the intrinsic mechanism and provide a principled justification for our method, we conduct a greedy search experiment using Qwen3-8B on the HumanEvalPlus dataset, as illustrated in Fig. 4. We specifically analyze the tokens generated during the inference phase, focusing on the relationship between their logits entropy and the effectiveness

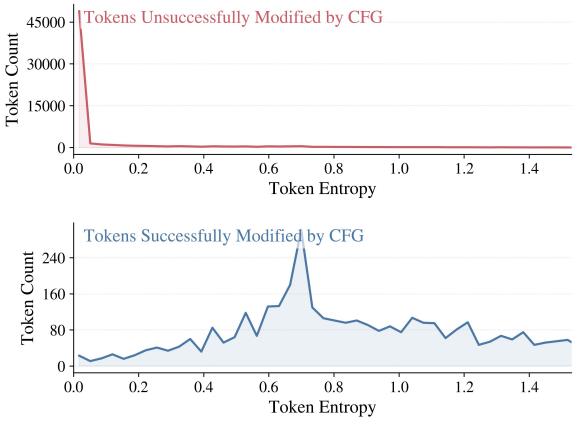
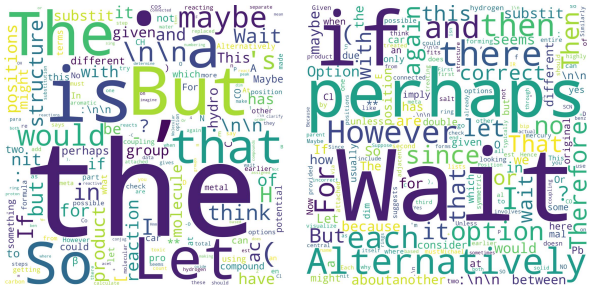


Figure 4: Entropy distributions of tokens categorized by vanilla CFG modification success. x-axis: token entropy; y-axis: token count per bin.

of CFG in altering the top-1 prediction. The empirical results reveal a clear divergence: as shown in the top panel of Fig. 4, the vast majority of tokens remain unchanged after applying CFG, with these instances predominantly concentrated in the low-entropy region. Conversely, the bottom panel demonstrates that tokens whose predictions are successfully steered by CFG are primarily those with high entropy. We attribute this phenomenon to the confidence levels inherent in the logits distribution. Low-entropy logits typically contain a single dominant mode, indicating that the model is already highly confident in its prediction; this makes the token naturally resistant to CFG-based modification. These observations resonate with our core motivation: since low-entropy tokens are often "immune" to CFG, it is both reasonable and efficient to apply CFG selectively to high-entropy tokens, thereby avoiding redundant computations on predictions that are unlikely to change.

Word-cloud analysis. As shown in Fig. 5, we conduct a word cloud visualization on Qwen3-8B using the GPQA-Diamond dataset, specifically focusing on questions where the model initially fail



(a) Before-MTI. (b) After-MTI.

Figure 5: Token word clouds before vs. after MTI.

but reach the correct answer after applying MTI. By comparing the high entropy token distributions before and after intervention, we observe that MTI significantly enhances lexical diversity: generic tokens (e.g., the, a) are frequently replaced by semantically dense reasoning terms and connectives. Notably, there is a distinct shift away from simple sequential connectives (e.g., so) toward more diverse and critical connectives, such as adversative conjunctions and hedging terms (e.g., however, if, perhaps, alternatively, wait). This transition suggests that MTI helps the model abandon erroneous prior reasoning chains and initiate new lines of thought, effectively steering the trajectory toward correct inferences. Overall, applying MTI yields more varied and balanced vocabularies, which expand the hypothesis space for reasoning trajectories and ultimately result in better outcomes.

Model	Method	τ	Accuracy (%)	Latency
Qwen-14B	DI	–	78.262	4547s
	VC	–	76.296	12411s
	Ours	0.1	79.082	6081s
		0.5	80.660	5289s
		1.0	**80.698**	**4551s**
		1.5	80.052	3954s
Ling-mini-2.0	DI	–	74.556	1288s
	VC	–	75.460	3142s
	Ours	0.1	**76.400**	**1495s**
		0.5	75.338	1444s
		1.0	74.736	1370s
		1.5	75.308	1302s

Table 3: Inference Cost. Ours (MTI) consistently outperforms DI and VC with negligible computational overhead.

Inference cost. We evaluate the latency of Qwen-14B and Ling-mini-2.0 on the GPQA-Diamond, Math500, HumanEval, Winogrande, and HumanEvalPlus datasets. Here, latency refers to the total time required to evaluate all datasets on a single GPU following the setting in Tab. 1. As shown in Tab. 3, we observe that MTI does not necessarily increase the inference time. This is because MTI can guide the reasoning trajectory more effectively and reduce unnecessary or incorrect reasoning chains, resulting in shorter output.

4.5 Applying MTI to vision language models

Method	Where2Place	RefSpatial	VaBenchPoint
DI	61.39/–	42.45/–	38.87/–
VC	61.76/100%	39.86/100%	35.88/100%
Ours	**64.36**/3.7%	**44.24**/3.7%	**40.86**/4.5%

Table 4: Evaluation of Ours (MTI) with vision-language model. Results are reported as Accuracy(%) / CFG Usage. Ours consistently outperforms DI and VC.

To examine whether MTI generalizes across pretraining objectives, we extend it to the vision-language model (VLM) setting. We evaluate MTI on three spatial understanding benchmarks—Where2Place (Yuan et al., 2024), RefSpatial (Zhou et al., 2025), and VaBenchPoint (Yuan et al., 2025)—using Qwen3-8B-VL-Instruct. We strictly follow the official evaluation prompts of Qwen3-8B-VL-Instruct and report results in the same format as Tab. 4. With an entropy threshold of $\tau = 2.0$, MTI is activated for only 3.7%–4.5% of generated tokens, yet consistently improves accuracy: 61.39%→64.36%(+2.97%) on Where2Place, 42.45%→44.24%(+1.79%) on RefSpatial, and 38.87%→40.86%(+1.99%) on VaBenchPoint.

5 Conclusion

We propose Minimal Test-Time Intervention (MTI), a training-free framework that improves LLM reasoning with minimal overhead. We first introduce selective CFG intervention to use token entropy to identify unstable positions and apply classifier-free guidance only at those tokens, stabilizing the reasoning trajectory without unnecessary computation. We further introduce the lightweight negative-prompt guidance to approximate the unconditional branch without additional memory cost. Experiments across general, coding and STEM benchmarks show consistent improvements. MTI is plug-and-play, compatible with modern acceleration frameworks, providing an effective mechanism for enhancing test-time reasoning reliability.

6 Limitations

In our experiments, we find that meaningless tokens, such as "apple", as well as positive prompts, improve performance over Direct Inference, though not as effectively as semantically explicit negative prompts like "OUTPUT ERROR". The underlying interpretability of this effect remains unexplored, and further investigation into this area is a key direction for future research.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Mouxiang Chen, Binyuan Hui, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Jianling Sun, Junyang Lin, and Zhongxin Liu. 2025a. Parallel scaling law for language models. *arXiv preprint arXiv:2505.10475*.

Zhuokun Chen, Zeren Chen, Jiahao He, Lu Sheng, Mingkui Tan, Jianfei Cai, and Bohan Zhuang. 2025b. R-stitch: Dynamic trajectory stitching for efficient reasoning. *arXiv preprint arXiv:2507.17307*.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James Glass, and Pengcheng He. 2023. Dola: Decoding by contrasting layers improves factuality in large language models. *arXiv preprint arXiv:2309.03883*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Tri Dao, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana Rosing, Ion Stoica, and 1 others. Efficiently scaling llm reasoning with certaindex, 2025a. *URL https://arxiv. org/abs/2412.20993*.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. 2025. Deep think with confidence. *arXiv preprint arXiv:2508.15260*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025b. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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.

Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Jiameng Huang, Baijiong Lin, Guhao Feng, Jierun Chen, Di He, and Lu Hou. 2025. Efficient reasoning for large reasoning language models via certainty-guided reflection suppression. *arXiv preprint arXiv:2508.05337*.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*.

Felix Koulischer, Johannes Deleu, Gabriel Raya, Thomas Demeester, and Luca Ambrogioni. 2024. Dynamic negative guidance of diffusion models: Towards immediate content removal. In *Neurips Safe Generative AI Workshop 2024*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Ayeong Lee, Ethan Che, and Tianyi Peng. 2025a. How well do llms compress their own chain-of-thought? a token complexity approach. *arXiv preprint arXiv:2503.01141*.

Hakyung Lee, Subeen Park, Joowang Kim, Sungjun Lim, and Kyungwoo Song. 2025b. Uncertainty-aware contrastive decoding. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26376–26391.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.

Ang Li, Ben Liu, Binbin Hu, Bing Li, Bingwei Zeng, Borui Ye, Caizhi Tang, Changxin Tian, Chao Huang, Chao Zhang, and 1 others. 2025a. Every activation boosted: Scaling general reasoner to 1 trillion open language foundation. *arXiv preprint arXiv:2510.22115*.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 12286–12312.

Xianzhi Li, Ethan Callanan, Abdellah Ghassel, and Xiaodan Zhu. 2025b. Entropy-gated branching for efficient test-time reasoning. *arXiv preprint arXiv:2503.21961*.

Zeju Li, Jianyuan Zhong, Ziyang Zheng, Xiangyu Wen, Zhijian Xu, Yingying Cheng, Fan Zhang, and Qiang Xu. 2025c. Compressing chain-of-thought in llms via step entropy. *arXiv preprint arXiv:2508.03346*.

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*.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, and 1 others. 2024. Starcoder 2 and the stack v2: The next generation. *arXiv preprint arXiv:2402.19173*.

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. 2025. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*.

Daiki Miyake, Akihiro Iohara, Yu Saito, and Toshiyuki Tanaka. 2025. Negative-prompt inversion: Fast image inversion for editing with text-guided diffusion models. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2063–2072. IEEE.

Minh Nhat Nguyen, Andrew Baker, Clement Neo, Allen Roush, Andreas Kirsch, and Ravid Shwartz-Ziv. 2024. Turning up the heat: Min-p sampling for creative and coherent llm outputs. *arXiv preprint arXiv:2407.01082*.

Chen Qian, Dongrui Liu, Haochen Wen, Zhen Bai, Yong Liu, and Jing Shao. 2025. Demystifying reasoning dynamics with mutual information: Thinking tokens are information peaks in llm reasoning. *arXiv preprint arXiv:2506.02867*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Guillaume Sanchez, Honglu Fan, Alexander Spangher, Elad Levi, Pawan Sasanka Ammanamanchi, and Stella Biderman. 2023. Stay on topic with classifier-free guidance. *arXiv preprint arXiv:2306.17806*.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik R Narasimhan, Shunyu Yao, Thomas Lieberum, Yuan Le, Deepak Gopinath, Karthik Ramesh, and 1 others. 2023. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*.

Chenxia Tang, Jianchun Liu, Hongli Xu, and Liusheng Huang. 2025. Eliminating noise in logit space for robust token sampling of llm. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10758–10774.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, and 1 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, and 1 others. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and

Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. 2025. Tokenskip: Controllable chain-of-thought compression in llms. *arXiv preprint arXiv:2502.12067*.

Minghao Yan, Saurabh Agarwal, and Shivaram Venkataraman. 2024. Decoding speculative decoding. *arXiv preprint arXiv:2402.01528*.

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

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. 2025b. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*.

Zhen Yang, Ganggui Ding, Wen Wang, Hao Chen, Bohan Zhuang, and Chunhua Shen. 2023. Object-aware inversion and reassembly for image editing. *arXiv preprint arXiv:2310.12149*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Karthik Narasimhan, Mohit Hodosh, Eric Edwards, Jieyu Xu, Yuan Zhao, and Karthik Duan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Guoyin Wang, Heng Li, Jiangcheng Zhu, Jianqun Chen, and 1 others. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. 2024. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*.

Yifu Yuan, Haiqin Cui, Yibin Chen, Zibin Dong, Fei Ni, Longxin Kou, Jinyi Liu, Pengyi Li, Yan Zheng, and Jianye Hao. 2025. From seeing to doing: Bridging reasoning and decision for robotic manipulation. *arXiv preprint arXiv:2505.08548*.

Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. 2023. Draft & verify: Lossless large language model acceleration via self-speculative decoding. *arXiv preprint arXiv:2309.08168*.

Enshen Zhou, Jingkun An, Cheng Chi, Yi Han, Shanyu Rong, Chi Zhang, Pengwei Wang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, and 1 others. 2025. Roborefer: Towards spatial referring with reasoning in vision-language models for robotics. *arXiv preprint arXiv:2506.04308*.

A Appendix

A.1 Testing setup and files

For our results, we use the following `OpenCompass` configuration files:

- `gpqa_gen.py`
- `math_500_gen.py`
- `aime2024_gen.py`
- `humaneval_gen.py`
- `humaneval_plus_gen.py`
- `livecodebench_gen.py`
- `mmlu_pro_gen.py`

During evaluation, the `MMLU-Pro` score is computed as the average over all subsets. For `AIME2024`, we conduct eight runs and report the mean score. The results of LiveCodeBench come from tests on `lcb_code_generation` and `lcb_code_execution`.

A.2 Comparison with SOTA test-time scaling methods

Model	Method	τ	AIME2024	MATH500	GPQA-D	Avg.
Qwen3-8B	TALE	–	68.9	92.3	59.1	70.4
	NoThinking	–	30.0	87.1	54.2	57.1
	Dynasor	–	62.2	91.7	57.7	70.5
	DEER	–	45.6	88.7	59.3	64.5
	CGRS	–	61.1	93.3	59.8	71.4
	Ours	0.5	77.1	**97.0**	60.6	78.2
		1.0	75.4	**97.0**	60.1	77.5
		1.5	**78.3**	95.8	**61.1**	**78.4**
DeepSeek-R1-7B	TALE	–	48.9	89.1	36.2	58.1
	NoThinking	–	32.2	80.9	37.9	50.3
	Dynasor	–	47.8	81.8	22.2	50.6
	DEER	–	47.8	89.6	33.1	56.8
	CGRS	–	52.2	87.6	32.8	57.5
	Ours	0.5	51.7	**93.6**	**51.52**	65.6
		1.0	**62.9**	91.8	48.5	**67.7**
		1.5	60.0	93.2	40.9	64.7

Table 5: Comparison with SOTA test-time scaling methods. Ours (MTI) consistently outperforms SOTA test-time scaling methods.

As shown in Tab. 5, we compare our method with the state-of-the-art approaches TALE (Han et al., 2025), NoThinking (Ma et al., 2025), Dynasor (Fu et al.), DEER (Yang et al., 2025b), and CGRS (Huang et al., 2025) under the same settings on Qwen3-8B and DeepSeek-R1-7B across AIME2024, Math500, and GPQA-Diamond. The results clearly demonstrate the superiority of our method.

Method	Entropy scope	AIME2024(%)
DI	–	73.75
VC	–	73.34
Ours	$Entropy \leq 1.5$	71.67
	$Entropy > 1.5$	78.34

Table 6: Performance under different entropy scopes. The performance gains in high-entropy scopes significantly outweigh those in low-entropy ones.

A.3 Comparison with contrastive decoding

Contrastive Decoding (CD) (Li et al., 2023) is introduced to suppress unreliable generations by contrasting a strong model with a weaker counterpart, thereby reducing hallucinations and improving faithfulness to the conditioning context. The formulation for CD is defined as:

$$\begin{aligned} &\log \hat{P}\left(x_t \mid c, \bar{c}, x_{<t}\right) \\ &= \log P_s\left(x_t \mid c, x_{<t}\right) \\ &\quad - \alpha \cdot \log P_w\left(x_t \mid \bar{c}, x_{<t}\right), \end{aligned} \tag{6}$$

where α is a hyperparameter and $\log p_s$ denotes the distribution from a large expert LM, while $\log p_w$ is derived from a small amateur LM. Notably, if $\log p_w$ is replaced by a version of the expert model itself, the formulation is closely resembles Classifier-Free Guidance (CFG). To evaluate the functional distinctions between these two methods, we integrate CD into the MTI framework—replacing the original CFG component—and perform a comparative analysis to observe their respective effects on generation quality.

We evaluate the performance on the Ling-mini-2.0 model. The results show that $\alpha = 0.5$ yields better performance than $\alpha = 1.5$. The optimal result is achieved when $\tau = 0.1$ and $\alpha = 0.5$; while this configuration outperforms both DI and VCD, it remains below the 71.25% achieved by the CFG-based version of MTI (see Tab. 2). Furthermore, a comparison of inference overhead reveals that MTI(CD) requires the execution of CD on 16.6% of tokens, whereas MTI(CFG) requires CFG on only 1.4% of tokens. This proves that the effectiveness of CFG is superior to that of CD.

A.4 Performance under different entropy scopes.

Following the experimental setup in Table 2, we evaluate Qwen3-8B on the AIME2024 benchmark across various entropy thresholds. The results reveal a clear performance divide: applying CFG to low-entropy tokens (entropy ≤ 1.5) leads to a significant degradation compared to Direct Inference (DI). This demonstrates that interventions on low-entropy regions are inherently detrimental, as they introduce instability that accumulates over the long-form reasoning chains required by AIME2024. In contrast, by masking out these low-entropy tokens and restricting CFG only to high-entropy regions (entropy > 1.5), MTI significantly outperforms Vanilla CFG (VC). This confirms that selective intervention is crucial; MTI achieves superior results precisely by exempting predictable tokens from guidance, thereby maintaining stability while leveraging the benefits of CFG where it is most effective.

A.5 Case study.

Comparing Reasoning Outputs on Math

Problem. Consider the function

$$f(x) = \begin{cases} ax^2 & \text{if } x \geq a, \\ ax + 2a & \text{if } x < a, \end{cases}$$

where a is a real number. What is the largest value of a such that the graph of $y = f(x)$ intersects every horizontal line at least once? Please reason step by step, and put your final answer within ().

Direct Inference: Upper bound miscalculated; lost the $+2a$ term.

"... $f(x) = ax + 2a,\ x < a$ gives range $(-\infty, a^2)$, hence is (1) "

Vanilla CFG: Incorrect reasoning; Repeated generation.

"... So for any $a \neq 0$, the function is surjective onto $\mathbb{R}$.
But wait... Let me think again... But wait... Let me think again... "

Ours: Correct bound and inequality; final answer is correct.

"... union is $(-\infty, a^2 + 2a) \cup [a^3, \infty)$, so the maximum a is (2) "

Figure 6: Case study: direct inference vs. vanilla CFG vs. our method on Qwen3-8B.

As shown in Fig. 6, we observe two representative failure modes. Direct Inference: symbolic information loss—when rewriting the upper bound of the piecewise range, the constant term "$+2a$" is dropped, yielding an underestimated bound and a wrong interval. Vanilla CFG: the main issue is looped generation: the model repeatedly oscillates between claims (e.g., surjectivity onto $\mathbb{R}$) and retractions without a verifiable stopping criterion, and it neglects boundary conditions of the piecewise form. This looping arises from applying the CFG to all tokens, which can induce reasoning even for low-entropy tokens, thereby disrupting an otherwise stable reasoning process. Ours: By enforcing explicit boundary/inequality checks and composing the ranges in a unified manner, it avoids these pitfalls and produces the correct final answer.

Method	DI	VCD	MTI (CD)						VCD	MTI (CD)					
α	-	0.5	0.5						1.5	1.5					
τ	-	-	0.1	0.5	1.0	1.5	2.0	-	-	0.1	0.5	1.0	1.5	2.0	-
AIME2024 (%)	60.00	65.00	**70.00**	68.34	68.34	66.67	61.67	-	3.33	1.67	1.67	10.00	51.67	66.67	-
MTI Usage	-	100%	**16.6%**	8.2%	1.4%	0.2%	0.1%	-	100%	25.9%	17.5%	11.3%	8.8%	0.1%	-

Table 7: Performance of Ours(MTI) with Contrastive Decoding (CD) on Ling-mini-2.0 using the AIME2024 dataset. VCD denotes the vanilla contrastive decoding baseline. We report results across various hyperparameter configurations for α and τ, along with the corresponding MTI usage percentage. As shown CFG outperforms CD.

A.6 Discussion on task-specific negative prompts

Following the same experimental setting as in Tab. 1, we evaluate task-specific negative prompts using DeepSeek-R1-7B on Math500 and HumanEvalPlus. As shown in Tab. 8, while `OUTPUT ERROR` is a robust general-purpose choice, aligning the negative prompt's semantics with domain characteristics can further enhance performance. Specifically, `Bad Reasoning` is more suitable for mathematical tasks (94.20% vs. 93.40% for `SYNTAX ERROR`) as it better suppresses flawed logical chains. Conversely, `SYNTAX ERROR` is more effective for coding (77.44% vs. 76.83% for `Bad Reasoning`) by directly targeting structural failure modes. These results demonstrate that performance can be further optimized by refining the semantic granularity of negative prompts to match the specific constraints of the target task.

Negative Prompt	Task	Accuracy (%)
`Bad Reasoning`	Math	**94.20**
`SYNTAX ERROR`	Math	93.40
`OUTPUT ERROR`	Math	93.60
`Bad Reasoning`	Coding	76.83
`SYNTAX ERROR`	Coding	**77.44**
`OUTPUT ERROR`	Coding	**77.44**

Table 8: Ablation study of task-specific negative prompts on math and coding benchmarks.

A.7 Discussion on decoding strategies

Decoding strategies play a critical role in inference, as they determine how effectively a model's capabilities are realized. Early methods such as greedy search select the most probable token at each step, but often suffer from repetition. Beam Search mitigates this issue by maintaining multiple high-probability candidates, yet its deterministic nature leads to outputs with limited diversity. To improve diversity, sampling-based approaches have been proposed, including temperature scaling, top-k sampling (Fan et al., 2018), and top-p sampling (Holtzman et al., 2019; Nguyen et al., 2024; Tang et al., 2025). These methods increase overall diversity, but do not correct potential misranking among candidate tokens. MTI targets high-diversity (high-entropy) generation positions during reasoning, where probability ordering is more likely to be unreliable, and applies selective distribution correction to adjust the relative likelihoods of candidate tokens. Accordingly, MTI preserves necessary diversity while mitigating errors caused by miscalibrated probability rankings, and is orthogonal to and compatible with standard sampling strategies.

Other decoding approaches rely on auxiliary models, such as Speculative Decoding (Leviathan et al., 2023; Chen et al., 2025b) for acceleration and Contrastive Decoding (Li et al., 2023; Lee et al., 2025b; Chuang et al., 2023) for reducing hallucinations. In contrast, MTI requires no additional models and focuses on improving reasoning ability rather than speed or hallucination control. LLM-CFG (Sanchez et al., 2023) guides generation via unconditional predictions but requires maintaining two KV caches and often yields unstable gains in modern reasoning models. By contrast, MTI employs Selective CFG Intervention only at high-diversity positions, achieving gains with negligible overhead, while Lightweight Negative-Prompt Guidance reuses the conditional KV cache and avoids maintaining dual KV caches, enabling easy integration with modern inference frameworks (Kwon et al., 2023).

A.8 Use of LLMs

We use LLMs to polish the paper, correct some grammatical errors, and make the language more concise and fluent.