# Top-$n\sigma$: Eliminating Noise in Logit Space for Robust Token Sampling of LLM

**Anonymous ACL submission**


## Abstract

Large language models (LLMs) rely heavily on sampling methods to generate diverse and high-quality text. While existing sampling methods like top-$p$ and min-$p$ have identified the detrimental effects of low-probability tails in LLMs' outputs, they still fail to effectively distinguish between diversity and noise. This limitation stems from their reliance on probability-based metrics that are inherently sensitive to temperature scaling. Through empirical and theoretical analysis, we make two key discoveries: (1) the pre-softmax logits exhibit a clear statistical separation between informative tokens and noise, and (2) we prove the mathematical equivalence of min-$p$ and top-(1-$p$) under uniform distribution over logits. These findings motivate the design of top-n$\sigma$, a novel sampling method that identifies informative tokens by eliminating noise directly in logit space. Unlike existing methods that become unstable at high temperatures, top-$n\sigma$ achieves temperature-invariant token selection while preserving output diversity. Extensive experiments across reasoning and creative writing tasks demonstrate that our method consistently outperforms existing approaches, with particularly significant improvements in high-temperature settings.


## 1 Introduction

Large Language Models (LLMs) have revolutionized natural language processing (NLP), demonstrating remarkable capabilities across various domains, including code generation (Chen et al., 2021), mathematical reasoning (Lewkowycz et al., 2022), and complex problem-solving (Wei et al., 2022). These advancements are largely driven by the models' text generation mechanisms, which underpin their versatility in diverse applications.

The generation process of LLMs involves a fundamental trade-off between creativity and quality, which is controlled by the temperature parameter $T$. This parameter shapes the output's sharpness

Figure 1: Temperature sensitivity analysis of min-$p$ sampling ($p = 0.1$) by prompting LLaMA-3-8B with "Randomly pick a digit between 0~9: " (verbatim prompt). Blue tokens represent valid digits (0-9), and red tokens indicate noise. The sampling space is denoted by highlighted tokens and numbers below each token show their sampling probabilities.

and influences how the model selects its next tokens (Ackley et al., 1985; Chen and Ding, 2023; Bellemare-Pepin et al., 2024). Specifically, a lower temperature causes the model to favor the most probable outputs, which may limit exploration and creativity. Conversely, a higher temperature encourages exploration and unconventional choices, though this increased diversity may increase the risk of errors and inconsistencies.

To empirically investigate the trade-off mentioned above, we employ the popular min-$p$ sampling (Nguyen et al., 2024) which claims to be more stable under high temperature. This technique truncates tokens with probabilities below $p \cdot p_{max}$ (where $p_{max}$ denotes the maximum probability), and we set $p = 0.1$ in our analysis. We prompt LLaMA-3-8B with "Randomly pick a digit between 0~9: " and visualize the output probability distribution under different temperature settings in Figure 1. Given $T = 1.0$, though min-$p$ selects the correct tokens, the model produces a highly skewed distribution that clearly favors certain digits. For example, the probability of selecting 3 or 7 is more

than twice that of 9, which contradicts the explicit randomness requirement in the prompt. When $T$ is increased to 1.5, the distribution becomes more uniform, yet this method begins including invalid tokens like `123` and `\xa0`, whose probabilities are larger than the threshold 0.004 (*i.e.*, $0.1 \times 0.04$).

Notably, we know that a perfectly uniform distribution over the entire vocabulary is theoretically achievable through infinite temperature scaling (*i.e.*, $T \to \infty$). However, existing methods fail to correctly identify valid tokens under such conditions, as they rely on temperature-dependent probability thresholds for token selection. This limitation prompts a critical question: is it possible to develop a criterion that simultaneously effectively identifies valid tokens and remains invariant to temperature scaling? If such a criterion exists, we could achieve the ideal scenario illustrated in Figure 1. Since temperature scaling operates directly on the pre-softmax logits, it naturally motivates an investigation into their structural properties.

Intriguingly, through our empirical analysis (Section 3.1), we discover that the logits naturally are separated into two distinct components: a **Gaussian-distributed background** and **several outliers**. This clear statistical separation suggests logits as a better foundation for token selection, instead of probabilities. More importantly, we prove the mathematical equivalence of top-$p$ and min-$p$ under the uniform assumption in Section 3.2. This equivalence provides valuable insight for the underlying distribution of these outliers, which are precisely the informative tokens to be preserved.

Building upon these findings, we propose top-$n\sigma$, a novel sampling approach that operates directly on logits using standard deviation as the selection criterion. Our method achieves temperature-invariant control over token selection: as temperature increases, top-$n\sigma$ only elevates the probabilities of chosen tokens without introducing additional ones, allowing separated optimizations of token selection and distribution shaping. Our method also eliminates the computational overhead of sorting and softmax transformations, ensuring computational efficiency.

Our main contributions include:

- **Novel Logit-based Perspective:** Through empirical analysis, we discover that LLM's pre-softmax logits exhibit a natural separation between informative tokens and noise.
- **Theoretical Understanding:** We prove the equivalence between min-$p$ and top-(1-$p$) under the assumption of uniform logit distribution. This reveals the deeper connection between these approaches and provides insights into the distribution of informative tokens.
- **Temperature-Invariant Dynamic Sampling:** We introduce a sampling method that selects candidate tokens dynamically using logit standard deviation, making the selection totally independent of temperature scaling.
- **Comprehensive Validation:** Through extensive experiments across diverse datasets and tasks, we demonstrate significant improvements in both generation quality and diversity compared to existing methods, especially under high temperatures.

## 2 Related Work

### 2.1 Probability-based Methods

Probability-based sampling methods directly manipulate the raw probability distribution output from LLMs, presenting the most widespread approaches. OpenAI (OpenAI, 2025), Anthropic (Anthropic, 2025), and Google (Google AI, 2025) all incorporate them as standard API parameters in their inference services. These methods typically begin with temperature scaling (Ackley et al., 1985) to balance generation quality and diversity. Subsequently, as the most straightforward approach, top-$k$ (Fan et al., 2018) restricts the sampling space to the $k$ most probable tokens. However, a fixed value of $k$ might exclude relevant tokens or include irrelevant tokens. To address this limitation, top-$p$ (nucleus) sampling (Holtzman et al., 2019) dynamically selects the smallest set of tokens whose cumulative probability exceeds a threshold $p$. However, top-$p$ exhibits high sensitivity to temperature settings, even slight increases in temperature will lead to deteriorated output quality. More recently, min-$p$ sampling (Nguyen et al., 2024) filters out tokens whose probabilities are below a fraction $p$ of the maximum probability, partially alleviating but not fully resolving the challenges in high-temperature settings, as discussed in Section 1.

### 2.2 Entropy-based Methods

In parallel with probability-based approaches, researchers have explored sampling methods grounded in information theory, particularly entropy. These methods establish their own optimiza-

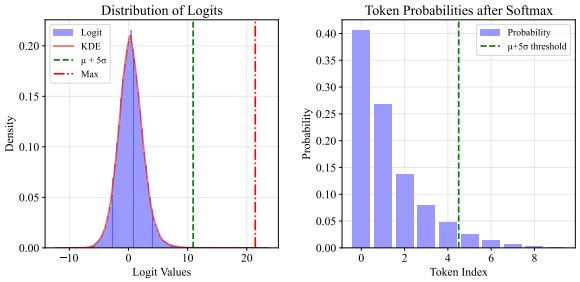



Figure 2: Distribution of logits and sorted token probabilities of LLaMA3-8B-Instruct on an AQuA sample. A vertical line at $\mu + 5\sigma$ is drawn in the logit distribution to depict the separation. Its corresponding threshold in the probability plot shows which tokens have logits exceeding this threshold.

tion criteria. For example, mirostat sampling (Basu et al., 2020) targets a constant perplexity through cross-entropy regulation, $\eta$-sampling (Hewitt et al., 2022) adjusts truncation thresholds based on token-level entropy, and REAL (Chang et al., 2024) optimizes for asymptotic entropy in the sampling process. Despite their theoretical guarantees, these entropy-based methods have not gained widespread adoption. This is mainly due to their implementation complexity, additional computational overhead, and the lack of substantial performance improvements over simpler probability-based alternatives, as reported in (Zhou et al., 2024; Nguyen et al., 2024).

## 3 Insights

### 3.1 Limitation from Gaussian Intuition

Modern LLMs rely on the softmax function to produce output probabilities. Due to its exponential nature, softmax aggressively pushes small logits towards zero probabilities, making it impossible to distinguish the underlying distribution of noise. To better illustrate this effect, we examine the output logits and probabilities of LLaMA-3-8B-Instruct on an AQuA sample, as illustrated in Figure 2. We observe that the majority of logits follow a Gaussian distribution in the lower-value region, which corresponds to the low-probability tails that are commonly treated as noise in the probability distribution. This pattern suggests the potential for more meaningful truncation in the logit space.

Intuitively, given that the majority of logits exhibit a Gaussian distribution, a natural first attempt would be to identify informative tokens as statistical outliers using the conventional methods, *e.g.*, the $\mu + 3\sigma$ rule (Kazmier, 2009). To formalize this intuition, let us first review how LLMs generate token probabilities and how existing sampling methods operate on them. Given an input context $x$, an LLM first generates a logit vector $l = (l_1, \cdots, l_V) \in \mathbb{R}^V$, where $V$ is the vocabulary size. These logits are firstly scaled by temperature ($l \leftarrow l/T$) and then transformed into probabilities $p = (p_1, \cdots, p_V) \in \mathbb{R}^V$ through the softmax function

$$p_i = \frac{e^{l_i}}{s}, \quad \text{where } s = \sum_{j=1}^{V} e^{l_j}, 1 \leq i \leq V \quad (1)$$

Fundamentally, all truncation sampling methods operate by determining a probability threshold $p^{(t)} \in [0, 1]$. Tokens with probabilities above this threshold form the sampling nucleus, and their cumulative probability defines the *nucleus mass*. Formally, for a threshold $p^{(t)}$, the nucleus $\mathcal{N}$ is

$$\mathcal{N} = \{i \mid p_i \geq p^{(t)}\} \quad (2)$$

Typical outlier detection approach (such as $\mu + 3\sigma$) can be generalized as selecting tokens whose logit values exceed $\mu + c\sigma$, where $\mu$ is the mean of logit values, $c$ is a constant parameter and $\sigma$ is their standard deviation. Accordingly, Equation (2) can be described as

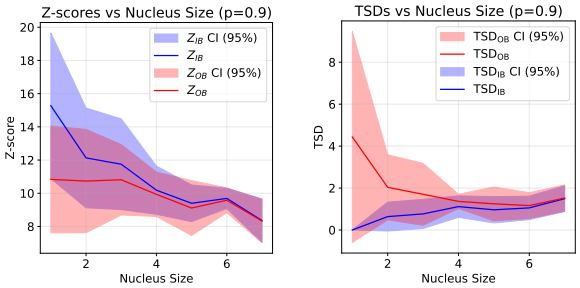
$$\mathcal{N} = \{i \mid l_i \geq \mu + c\sigma\} \quad (3)$$

To validate the feasibility of this criterion, we examine how it aligns with the nuclei produced by existing sampling methods. We introduce two critical measures: the Inner Boundary Z-score ($Z_{\text{IB}}$) and the Outer Boundary Z-score ($Z_{\text{OB}}$). Specifically, for a logit value $\beta$, $Z_\beta$ is defined as

$$Z_\beta = \frac{\beta - \mu}{\sigma} \quad (4)$$

where IB corresponds to the smallest logit value within the nucleus (*i.e.*, $\beta = \min_{l_i \in \mathcal{N}} l_i$), and OB corresponds to the largest logit value outside the nucleus (*i.e.*, $\beta = \max_{l_i \notin \mathcal{N}} l_i$). Any parameter $c$ between $Z_{\text{IB}}$ and $Z_{\text{OB}}$ will result in the same nucleus. Ideally, if this Gaussian-based criterion truly captures the underlying token distribution pattern, we should observe nearly constant Z-scores across different scenarios.

However, our empirical analysis reveals a different story. As shown in Figure 3(a), when examining LLaMA-3-8B-Instruct's behavior on an




(a) (b)

Figure 3: Comparison of Z-scores and Top-based Sigma Distances (TSDs) with their 95% confidence intervals (CI) during generation on an AQuA sample. The region between the IB curve and the OB curve represents the nucleus boundary range.

AQuA sample using top-$p$ as the reference sampling method, the Z-scores exhibit substantial variations across different nucleus sizes. This inconsistency prompts us to rethink the fundamental assumptions behind the $\mu + c\sigma$ criterion. Such a criterion implicitly assumes that the specific distribution of informative tokens is trivial. In fact, this assumption fundamentally contradicts the core purpose of sampling, which is precisely to preserve these informative tokens. Due to their inherent scarcity, statistically characterizing their distribution proves challenging. Fortunately, we have discovered an alternative approach to this problem.

### 3.2 Solution from Uniform Assumption

Our investigation starts from an empirical observation. The two popular methods min-$p$ and top-$p$ with complementary parameters (*e.g.*, min-0.1 vs. top-0.9) are frequently compared and exhibit similar behaviors in low-temperature scenarios. While this empirical connection was previously noted but dismissed as coincidental by Nguyen et al. (2024), it motivates us to investigate whether a deeper relationship exists or not.

To analyze this relationship rigorously, we need to understand how these sampling methods operate. Top-$p$ (Holtzman et al., 2019) uses the nucleus mass as the criterion. Formally, given $p$ (typically 0.9), the probability threshold $p^{(t)}$ is the solution to $\sum_{p_i \geq p^{(t)}} p_i = p$. Min-$p$ (Nguyen et al., 2024) scales the maximum probability by a fraction $p$ (typically 0.1) and uses the result as the threshold, *i.e.*, $p^{(t)} = p_{max} \cdot p$, where $p_{max} = \max_{1 \leq i \leq V} p_i$.

Since probabilities are transformed from logits by softmax, the probability threshold $p^{(t)}$ can be equivalently translated into a corresponding logit threshold $t = \ln(s \cdot p^{(t)})$, where $s$ is the sum of exponentials in Equation (1). The corresponding logit threshold of min-$p$ is thus $t = M + \ln p$, where $M = \ln(s \cdot p_{max}) = \max_{1 \leq i \leq V} l_i$. However, the solution of top-$p$ is not apparent. To make this problem tractable, we assume that the logits are independently and identically distributed according to some distribution $f$. This statistical perspective leads to a series of useful lemmas.

**Lemma 3.1.** *Assume $V$ logits $\{l_1, \cdots, l_V\}$ independently and identically distributed according to $f(x)$. For any threshold $t$, we have*

$$\sum_{l_i > t} e^{l_i} \xrightarrow{P} V \int_t^{+\infty} e^x f(x)\, \mathrm{d}x$$

The complete proof is provided in Appendix A.1. It is conceptually simple and allows us to leverage the overall distribution information instead of discrete samples.

**Lemma 3.2.** *Denote $\mathcal{I}(t) = \int_t^{+\infty} e^x f(x)\, \mathrm{d}x$, and thus $s = V \cdot \mathcal{I}(-\infty)$. The nucleus mass of a given logit threshold $t$ is*

$$p_{\mathcal{N}} = \sum_{i \in \mathcal{N}} p_i = \frac{\mathcal{I}(t)}{\mathcal{I}(-\infty)} \tag{5}$$

Lemma 3.2 is particularly useful for solving the logit threshold of top-$p$ if $\mathcal{I}$ has a closed-form solution in elementary functions. Remarkably, we discover a surprising equivalence between the two methods, presented in Theorem 3.3.

**Theorem 3.3.** *For logits following a uniform distribution $U(-\infty, M)$, min-p sampling is equivalent to top-$(1-p)$ sampling.*

*Proof.* For $l_i \sim U(-\infty, M)$, the logit threshold of min-$p$ is simple as

$$t = \ln(s \cdot p_{max} \cdot p) = \ln(e^M \cdot p) = M + \ln p \tag{6}$$

To solve the threshold $t$ of top-$p$, we can treat $U(-\infty, M)$ as a limiting case of $U(M-a, M)$ as $a \to \infty$. As derived in Appendix A.2, for any finite value of $a$, the logit threshold of top-$(1-p)$ under $U(M-a, M)$ distribution is

$$t = M - \ln\left[\frac{1}{1-(1-p)(1-e^{-a})}\right] \tag{7}$$

Taking $a \to \infty$, we obtain

$$t = M + \ln p \tag{8}$$

This is exactly the same threshold as min-$p$ sampling. □

Theorem 3.3 provides significant insights into the empirical observations of similarity between min-$p$ and top-(1-$p$) sampling, rather than being merely coincidental. More importantly, this theoretical result offers a valuable perspective on the distribution of informative tokens, suggesting uniform distribution as a reasonable approximation. Given our goal of retaining informative tokens, it can be viewed as a denoising process from a uniform distribution with a known maximum. This suggests measuring the distance from *the top* rather than the mean, motivating our investigation of the Top-based Sigma Distance (TSD) metric. Similar to Equation (4), $\text{TSD}_\beta$ is defined as

$$\text{TSD}_\beta = \frac{M - \beta}{\sigma} \tag{9}$$

where $M$ is the maximum of logits. Similarly, two variants are of particular interests: $\text{TSD}_{\text{IB}}$ and $\text{TSD}_{\text{OB}}$. As illustrated in Figure 3(b), TSD values exhibit a relatively stable pattern across different nucleus sizes, indicating superiority against Z-scores. Specifically, within the region defined by $\text{TSD}_{\text{IB}}$ and $\text{TSD}_{\text{OB}}$, there appears to be a consistent central value around 1.0, which we don't observe on Z-scores. This advantage motivates the design of our top-$n\sigma$ algorithm, which begins from the maximum value and extends downward, using the standard deviation of the distribution to dynamically adjust the boundary.

# 4 Algorithm

## 4.1 Algorithm Description

Our method introduces a statistical threshold to filter candidate tokens before sampling. Algorithm 1 outlines the main steps of our method. The algorithm operates directly on logits, capturing a region that extends $n\sigma$ below the maximum value and masking out all other logits (Lines 5~7), where a threshold multiplier $n$ controls the size of the sampled region. Finally, the logits are transformed via softmax into probabilities for the next token sampling. We further analyze its theoretical range and connections to existing methods in Section 4.2. Furthermore, we demonstrate that our method maintains a consistent nucleus size across different temperature settings in Section 4.3, ensuring robust sampling behavior.

## 4.2 Range of $n$

While the precise distribution of overall logits remains unknown, we propose a tractable analytical framework based on the discussions in Section 3.2. We model the logit as a random variable $L = \alpha X + (1-\alpha)Y$, where $X \sim U(M-a, M)$ represents the informative component and $Y \sim N(\mu_n, \sigma_n^2)$ captures the noise component with mixing factor $\alpha \in (0,1)$. Since the actual mixing mechanism between components is inherently unidentifiable, we opt for a clean separation assumption where all logits in $[M-a, M]$ are samples from $X$.

This separation assumption leads to an important property: $\sigma_u = \sqrt{\text{Var}(X)} \leq \sigma$, where $\sigma$ is the standard deviation of the overall logit distribution. By the law of total variance (Weiss et al., 2006), $\text{Var}(L) = E[\text{Var}(L|I)] + \text{Var}(E[L|I])$, where $I$ indicates whether a logit belongs to the informative component $X$ or not. Under our separation assumption, $\text{Var}(L|I=1) = \text{Var}(X)$, and therefore $\text{Var}(X)$ must be less than or equal to the total variance $\text{Var}(L) = \sigma^2$.

The goal of the truncation algorithm is to preserve the uniform component $X$ while eliminating the normal component $Y$. The optimal truncation is clearly $M - a$, but $a$ is unknown. Since the variance of a uniform distribution is $a^2/12$, we have

$$a = 2\sqrt{3}\sigma_u = 2\sqrt{3}\frac{\sigma_u}{\sigma}\sigma \tag{10}$$

This indicates the optimal truncation parameter should be $n = 2\sqrt{3}\frac{\sigma_u}{\sigma} \approx 3.46k$, where $k = \frac{\sigma_u}{\sigma}$. Unfortunately, determining $\sigma_u$ is impossible, as we cannot definitively attribute each sample to either distribution. Given that $\sigma_u \leq \sigma$ under our assumption, we can derive an upper bound $n \leq 3.46$. Since informative tokens and noisy tokens are typically far apart, $k$ tends to be small, suggesting that 3.46 is a rather loose upper bound.

**Algorithm 1** Top-$n\sigma$ Sampling

1: **Input:** Input context $x$, temperature $T$, threshold multiplier $n$
2: **Output:** Next token
3: Compute logits $l = \text{LLM}(x)$
4: Scale logits: $l' = l/T$
5: Calculate $M = \max(l')$ and $\sigma = \text{std}(l')$
6: Create mask: $m_i = \begin{cases} 1 & \text{if } l'_i \geq M - n\sigma \\ 0 & \text{otherwise} \end{cases}$
7: Apply mask: $l'_i = \begin{cases} l'_i & \text{if } m_i = 1 \\ -\infty & \text{otherwise} \end{cases}$
8: $p = \text{softmax}(l')$
9: Sample token from distribution $p$

To complement our theoretical analysis, we can gain practical intuition for choosing $n$ by examining the widely-adopted min-$p$ sampling method. Since min-$p$ truncation is equivalent to removing tokens with logits below $M - \ln(1/p)$, we can establish a direct correspondence by $n = \frac{\ln(1/p)}{\sigma}$. For example, with a reference logit standard deviation $\sigma = 2.2$ (see Appendix F for details), a min-$p$ threshold of 0.1 corresponds to $n \approx 1.05$. In practice, we use $n = 1.0$ as the default setting, which provides an effective quality-diversity trade-off.

### 4.3 Temperature Invariance

A key advantage of our method is its temperature invariance, as stated in the following theorem.

**Theorem 4.1.** *For any temperature $T > 0$, the nucleus of top-$n\sigma$ remains invariant.*

This temperature invariance follows from the fact that temperature scaling affects both the maximum value and standard deviation of the logits proportionally by $1/T$, thereby preserving the relative selection criterion for each token (see Appendix A.3 for details).

This invariance distinguishes our method from other common sampling approaches. While top-$k$ sampling also maintains temperature invariance, it relies on a fixed $k$ value that cannot adapt to varying token distributions across different contexts. In contrast, methods like top-$p$ and min-$p$ sampling have temperature-dependent selection sets: as temperature increases, their sampling nuclei tend to include more noise tokens due to the flattening effect on probability distributions. Our method combines the benefits of temperature invariance with adaptive token selection, better aligning with human language patterns where grammatical consistency is maintained even as vocabulary choices vary.

## 5 Experiments

### 5.1 Setup

**Models.** We evaluate our proposed top-$n\sigma$ using LLaMA-3 (Dubey et al., 2024), specifically with LLaMA-3-8B-Instruct and LLaMA-3-70B-Instruct. We also report the results of Qwen2.5 (Yang et al., 2024) in Appendix D. Besides, we use vLLM (Kwon et al., 2023) for inference.

**Benchmarks.** We conduct experiments on two distinct task categories:

- **Reasoning:** We evaluate four question-answering datasets spanning elementary to doctoral-level mathematics: AQuA (Ling et al., 2017), MATH500 (Lightman et al., 2023; Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), and GPQA-main (Rein et al., 2024). Each problem is transformed into an open-ended generation task.
- **Creative Writing:** Following (Nguyen et al., 2024), we adopt a diverse collection of 500 samples. Detailed experimental settings can be found in Appendix B.

**Baselines.** We evaluate top-$n\sigma$ against top-$k$ (Fan et al., 2018) ($k = 20$), top-$p$ (Holtzman et al., 2019) ($p = 0.9$), min-$p$ (Nguyen et al., 2024) ($p = 0.1$), $\eta$-sampling (Hewitt et al., 2022) ($\eta = 9 \times 10^{-4}$), and mirostat (Basu et al., 2020) ($\tau = 5.0$). The hyperparameter values are adopted from previous work (Hewitt et al., 2022; Nguyen et al., 2024), practical guidelines (Siml, 2024) and our empirical tests (discussed in Appendix C). These values keep fixed across different temperatures to demonstrate stability. For top-$n\sigma$, we prove a theoretical bound $n \in (0, 2\sqrt{3})$, with $n = 1.0$ as an effective default value. We omit the reasoning results of $\eta$-sampling and mirostat as they are designed for a diverse generation.

**Metrics.** We use **Exact Match** (EM) for reasoning tasks and **win rate** against greedy decoding using DeepseekV3 (Liu et al., 2024) as judge through Alpaca2.0 framework (Li et al., 2023) (see Appendix B for details).

### 5.2 Main Results

#### 5.2.1 Reasoning

Table 1 compares the performance of different sampling methods across temperature settings (*e.g.*, 0.7-4.0) on four representative datasets. While conventional methods achieve competitive performance occasionally, their effectiveness is highly sensitive to temperature settings, requiring careful parameter tuning for each specific application scenario. The results demonstrate that top-$n\sigma$ sampling not only outperforms or matches the peak performance of other methods at optimal temperatures but also maintains consistent performance across all temperature settings. This robustness is particularly valuable for real-world applications, where optimal temperature parameters are typically unknown *a priori* and may vary across different tasks or user requirements. In contrast, methods that are sensitive to temperature settings provide weaker perfor-

Table 1: Performance comparison (%) of different sampling methods on LLaMA3-8B-Instruct and LLaMA3-70B-Instruct across different temperature settings.

| Dataset | Method | LLaMA3-8B-Instruct | | | | | | LLaMA3-70B-Instruct | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0.7 | 1.0 | 1.5 | 2.0 | 3.0 | 4.0 | 0.7 | 1.0 | 1.5 | 2.0 | 3.0 | 4.0 |
| AQuA | Top-$k$ | 52.76 | 51.57 | 40.94 | 20.47 | 3.15 | 0.39 | 75.59 | 70.87 | 72.05 | 64.57 | 28.35 | 3.54 |
| | Top-$p$ | 51.18 | 50.00 | 36.61 | 0.00 | 0.00 | 0.00 | 75.20 | **77.95** | 70.87 | 32.28 | 0.00 | 0.00 |
| | Min-$p$ | 50.39 | 51.18 | 47.24 | 37.80 | 11.42 | 0.00 | 74.80 | 74.41 | **73.62** | **73.23** | 63.78 | 22.83 |
| | Top-$n\sigma$ | **54.33** | **52.76** | **48.82** | **51.97** | **49.21** | **50.00** | **76.77** | 76.38 | 72.44 | **73.23** | **74.41** | **76.38** |
| GSM8K | Top-$k$ | 75.51 | 71.87 | 56.03 | 29.11 | 2.50 | 0.45 | 92.19 | 90.98 | 90.37 | 84.76 | 50.27 | 10.92 |
| | Top-$p$ | 76.65 | 75.59 | 66.34 | 0.00 | 0.00 | 0.00 | 91.51 | 90.90 | 90.60 | 53.90 | 0.00 | 0.00 |
| | Min-$p$ | **76.72** | 74.15 | 71.34 | 63.68 | 25.47 | 0.91 | 91.28 | 90.90 | 91.58 | 91.05 | 85.29 | 52.08 |
| | Top-$n\sigma$ | 76.19 | **75.97** | **75.28** | **75.28** | **74.53** | **73.24** | **92.34** | **91.28** | **91.74** | **91.74** | **91.28** | **91.89** |
| GPQA-main | Top-$k$ | 31.47 | 32.81 | 27.68 | 20.09 | 6.25 | 0.67 | **41.07** | **39.51** | 38.84 | 36.38 | 18.97 | 5.36 |
| | Top-$p$ | **32.59** | **33.26** | 16.52 | 0.00 | 0.00 | 0.00 | 37.50 | 38.39 | 40.18 | 2.01 | 0.00 | 0.00 |
| | Min-$p$ | 32.14 | 32.59 | **29.24** | 29.46 | 11.38 | 0.45 | 39.29 | 38.62 | 40.63 | **41.29** | 35.04 | 9.82 |
| | Top-$n\sigma$ | 31.70 | 31.47 | 29.02 | **30.80** | **31.47** | **29.91** | 36.38 | 38.39 | **40.85** | 39.51 | **40.63** | **41.74** |
| MATH500 | Top-$k$ | 23.20 | 21.40 | 14.00 | 5.20 | 1.20 | 0.40 | 46.40 | 40.60 | 40.00 | 33.60 | 10.00 | 1.20 |
| | Top-p | 25.40 | 22.20 | 13.20 | 0.00 | 0.00 | 0.00 | **47.60** | **45.60** | 40.40 | 6.20 | 0.00 | 0.00 |
| | Min-$p$ | 24.80 | 23.60 | 19.00 | 16.00 | 4.20 | 0.00 | 46.60 | 42.60 | 42.20 | 40.60 | 28.00 | 11.20 |
| | Top-$n\sigma$ | **26.80** | **25.00** | **24.80** | **23.40** | **22.80** | **23.20** | 45.80 | 44.80 | **45.80** | **43.00** | **47.60** | **46.80** |

mance guarantees, as their accuracy can fluctuate significantly depending on the chosen temperature parameter. Furthermore, by minimizing the impact of temperature on accuracy, top-$n\sigma$ enables flexible control over output diversity without compromising accuracy.

#### 5.2.2 Creative Writing

To examine whether the performance of top-$n\sigma$ sacrifices diversity for accuracy or not, we evaluate it on creative writing tasks following Nguyen et al. (2024). As shown in Table 2, top-$n\sigma$ achieves the highest win rates against greedy decoding (56.40% for 8B and 53.80% for 70B), demonstrating its ability to maintain both creativity and coherence. The robust performance of top-$n\sigma$ opens up a unique opportunity to explore the impact of extremely high temperatures in LLM sampling. While conventional wisdom suggests that higher temperatures lead to increased diversity (Bellemare-Pepin et al., 2024; Nguyen et al., 2024), this hypothesis has remained untested due to the instability of traditional sampling methods at high temperatures. With top-$n\sigma$'s temperature invariance, we are finally able to push the boundaries to $T = 3.0$ and even $T = 10.0$. Intriguingly, our results reveal that the benefits of temperature scaling eventually saturate, with win rates stabilizing around 55.40% and 53.40% for 8B and 70B models, respectively. This finding complements our understanding about the relationship between temperature and diversity, suggesting that further temperature increases may not yield additional benefits.

Table 2: Win rates (%) against greedy decoding on AlpacaEval Creative Writing using LLaMA-3-8B/70B-Instruct.

| | 8B | | 70B | |
|---|---|---|---|---|
| Method | $T=1.0$ | $T=1.5$ | $T=1.0$ | $T=1.5$ |
| Top-$k$ | 53.40 | 51.00 | 50.40 | **51.00** |
| Mirostat | 49.50 | 2.20 | 51.50 | 5.20 |
| $\eta$-sampling | 55.10 | 9.40 | 52.00 | 34.00 |
| Top-$p$ | 53.40 | 7.00 | 49.80 | 34.04 |
| Min-$p$ | 53.60 | **54.00** | 51.40 | 50.90 |
| Top-$n\sigma$ | **56.40** | 52.90 | **53.80** | 50.50 |
| | $T=3.0$ | $T=10.0$ | $T=3.0$ | $T=10.0$ |
| Top-$n\sigma$ | **55.40** | **55.40** | 51.30 | **53.40** |

### 5.3 Test-time Scaling Analysis

Recent work has demonstrated the effectiveness of test-time scaling techniques in enhancing model capabilities without additional training (Snell et al., 2024; Brown et al., 2024; Zhang et al., 2024). Among these techniques, majority voting with repeated sampling (Brown et al., 2024) stands out as one of the simplest yet effective approaches. However, the effectiveness of such techniques is constrained by underlying sampling methods. Traditional sampling at lower temperatures typically leads to higher single-shot accuracy but limited output diversity, reducing the potential gains from majority voting. Conversely, sampling at higher temperatures can provide more diverse outputs but often at the cost of quality.

To demonstrate how top-$n\sigma$ enhances these scaling techniques by overcoming this diversity-quality trade-off, we conduct experiments on LLaMA-3-

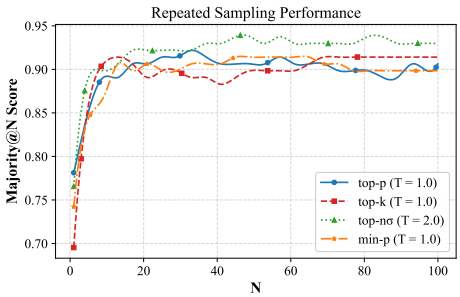



Figure 4: Repeated sampling performance of LLaMA-3-8B-Instruct on GSM8K with N (up to 100) repetitions and majority voting. Top-$n\sigma$ achieves both higher accuracy and better stability across different N values.

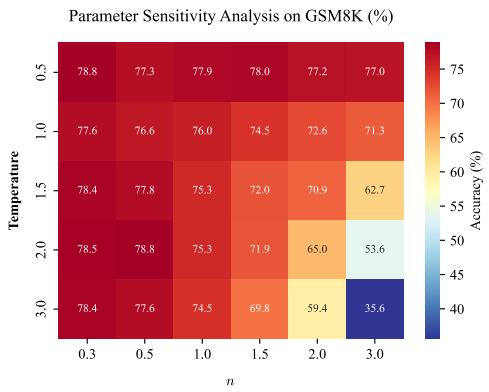



Figure 5: Parameter sensitivity analysis of top-n$\sigma$ sampling on GSM8K. The heatmap shows accuracy (%) under different combinations of $n$ (0.3-3.0) and temperature (0.5-3.0). The method maintains stable performance (~78%) with moderate $n$ values but degrades when $n \geq 2.0$, especially under high temperatures.

8B-Instruct using a 128-sample subset of GSM8K benchmark (following the same experimental setup as Brown et al. (2024)), comparing different sampling methods across varying numbers of sampling repetitions (up to 100). We use the Majority@N score, where we generate N independent responses, extract their answers in a standardized format, and select the most frequent one as the final prediction. For each method, we explore temperature settings ranging from 0.5 to 3.0 and report the performance curve with the optimal temperature.

As shown in Figure 4, while conventional methods show unstable performance with fluctuations and drops in the intermediate stage when repetitions increase, top-n$\sigma$ maintains robust performance improvements and reaches higher final accuracy (~93%). Notably, top-$n\sigma$ achieves its best performance at relatively higher temperatures, successfully maintaining output quality while providing the diversity necessary for effective majority voting

### 5.4 Sensitivity Analysis of $n$

To provide practical guidance for implementing top-$n\sigma$ sampling and validate our theoretical analysis, we conduct a sensitivity study of the key hyperparameter $n$. Specifically, we investigate how different combinations of $n$ and temperature affect model performance on GSM8K using Llama-3-8B-Instruct, with $n$ ranging from 0.3 to 3.0 and temperature from 0.5 to 3.0. We are particularly interested in the critical threshold of $n$ where performance significantly degrades under high temperatures, as the model's learned probability distribution has minimal impact in this regime, and therefore such degradation indicates the quality of the nucleus. This critical threshold can be used to verify our theoretical bounds.

As shown in Figure 5, we observe that the method exhibits robust performance when $n$ is within a moderate range (0.3-1.0), maintaining an accuracy of approximately 77-78% across different temperature settings. However, as $n$ increases beyond 1.5, we witness a noticeable performance degradation, with accuracy dropping to around 70% and falling below 60% when $n$ reaches 2.0. This empirical observation aligns well with our theoretical bound of $n = 2\sqrt{3}\sigma_u/\sigma \leq 3.46$. The observed critical threshold around $n = 1.5$ suggests that $\sigma_u \approx 0.43\sigma$, which is reasonably consistent with our theoretical expectation given that informative tokens are typically far from the noisy tokens. While this alignment is not exact due to the unknown true token distribution, it provides a practical validation of our theoretical analysis.

## 6 Conclusion

Based on empirical and theoretical analysis of LLM's output logits, we propose top-$n\sigma$, which achieves temperature-invariant sampling and preserves output diversity. The significance of our analysis extends beyond sampling strategies, as it reveals fundamental characteristics and limitations of softmax-based approaches in practice. The observed separation between informative and noise components in logits can be valuable for any scenario involving softmax operations, opening up promising directions for future research, such as leveraging these properties in model training to improve their robustness and effectiveness.

## Limitations

Several limitations deserve attention in our work. First, a static $n$ might not capture the precise boundary especially when the model is not confident. We have a more detailed discussion about $\sigma$ and the sampling process in Appendix F. However, we still have not been able to draw any definitive conclusion. We acknowledge that we know little about the relationship between informative tokens and Gaussian noise, and the assumption presented in the main text is too ideal. Second, our evaluation primarily focused on reasoning tasks and creative writing, leaving its effectiveness in other domains to be verified, like code generation. Third, while we analyzed and improved token-level sampling behavior, the impact of our method on sequence-level generation remains poorly understood, suggesting the need for further investigation into how local sampling decisions affect global generation quality. Finally, top-$n\sigma$ aims to eliminate Gaussian noise in logits. In other words, it attempts to preserve the model's own output as much as possible without modification. This may potentially retain the model's inherent hallucinations and biases. For domain-specific fine-tuned models, top-$n\sigma$ will amplify their bias on OOD (out-of-distribution) data (due to enhanced background noise)[1]. This might be beneficial for researchers as it helps identify OOD problems, but it will amplify the issues that fine-tuned models face with OOD data.

## References


David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. 1985. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169.

Anthropic. 2025. Release notes - api.

Sourya Basu, Govardana Sachitanandam Ramachandran, Nitish Shirish Keskar, and Lav R Varshney. 2020. Mirostat: A neural text decoding algorithm that directly controls perplexity. *arXiv preprint arXiv:2007.14966*.

Antoine Bellemare-Pepin, François Lespinasse, Philipp Thölke, Yann Harel, Kory Mathewson, Jay A Olson, Yoshua Bengio, and Karim Jerbi. 2024. Divergent creativity in humans and large language models. *arXiv preprint arXiv:2405.13012*.

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*.

Haw-Shiuan Chang, Nanyun Peng, Mohit Bansal, Anil Ramakrishna, and Tagyoung Chung. 2024. Real sampling: Boosting factuality and diversity of open-ended generation via asymptotic entropy. *arXiv preprint arXiv:2406.07735*.

Honghua Chen and Nai Ding. 2023. Probing the "creativity" of large language models: Can models produce divergent semantic association? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12881–12888.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

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

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

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Google AI. 2025. Gemini api reference.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.


[1] Due to double-blind review requirements, we cannot provide the link at this time

John Hewitt, Christopher D Manning, and Percy Liang. 2022. Truncation sampling as language model desmoothing. *arXiv preprint arXiv:2210.15191*.

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

Leonard Kazmier. 2009. *Schaum's Outline of Business Statistics*, 4 edition. McGraw-Hill, New York.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Minh 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*.

OpenAI. 2025. Openai api documentation.

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*.

Jan Siml. 2024. Is there an optimal temperature and top-p for code generation with paid LLM APIs? Accessed: 2024-03-17.

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*.

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

Neil A Weiss, Paul T Holmes, and Michael Hardy. 2006. *A course in probability*. Pearson Addison Wesley Boston, MA, USA:.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Di Zhang, Jianbo Wu, Jingdi Lei, Tong Che, Jiatong Li, Tong Xie, Xiaoshui Huang, Shufei Zhang, Marco Pavone, Yuqiang Li, et al. 2024. Llama-berry: Pairwise optimization for o1-like olympiad-level mathematical reasoning. *arXiv preprint arXiv:2410.02884*.

Yuxuan Zhou, Margret Keuper, and Mario Fritz. 2024. Balancing diversity and risk in llm sampling: How to select your method and parameter for open-ended text generation. *arXiv preprint arXiv:2408.13586*.

# A Theoretical Analysis

## A.1 Proof of Theorem 3.1

*Proof.* We can rewrite $\sum_{l_i > t} e^{l_i}$ to $\sum_{1 \le i \le V} e^{l_i} \mathbb{I}(l_i > t)$, where $\mathbb{I}(\cdot)$ is the indicator function, its value is 1 if the condition holds, 0 otherwise.

And,

$$\sum_{1 \le i \le V} e^{l_i} \mathbb{I}(X_i > t) = V \cdot \sum_{1 \le i \le V} \frac{e^{l_i} \mathbb{I}(l_i > t)}{V} \tag{11}$$

By Weak Law of Large Numbers, we can assert the second terms converges to $\mathbb{E}[e^L \mathbb{I}(L > t)]$ *in probability*, where $L$ denotes the random variable whose realizations are $\{l_i\}_{i=1}^V$.

Since

$$\begin{aligned} \mathbb{E}[e^L \mathbb{I}(L > t)] &= \int_{-\infty}^{+\infty} e^x \mathbb{I}(x > t) f(x) \, \mathrm{d}x \\ &= \int_t^{+\infty} e^x f(x) \, \mathrm{d}x \end{aligned} \tag{12}$$

We finally conclude:

$$\sum_{l_i > t} e^{l_i} \xrightarrow{P} V \int_t^{+\infty} e^x f(x) \, \mathrm{d}x \tag{13}$$

□

## A.2 Threshold Calculation

For data following uniform distribution $U(M - a, M)$, where $M$ is the maximum value and $a$ is the range of distribution support, we first utilize the shift invariance of softmax ($c$ is any constant):

$$\begin{aligned} \mathrm{Softmax}(X) &= \frac{e^{x_i}}{\sum_j e^{x_j}} = \frac{e^{x_i - c}}{\sum_j e^{x_j - c}} \\ &= \mathrm{Softmax}(X - c) \end{aligned} \tag{14}$$

By setting $c = M$, we can reduce our analysis to $U(-a, 0)$. The probability density function is

$$f(x) = \begin{cases} \frac{1}{a} & -a \le x \le 0 \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

Computing the integral as

$$\int_t^{+\infty} e^x f(x) \, \mathrm{d}x = \int_t^0 \frac{e^x}{a} \, \mathrm{d}x = \frac{1}{a}(1 - e^t) \tag{16}$$

After solving the equation and transforming back to the original scale, we obtain

$$t = M - \ln \left[ \frac{1}{1 - p(1 - e^{-a})} \right] \tag{17}$$

## A.3 Proof of Temperature Invariance

Here we provide the proof for Theorem 4.1.

*Proof.* Consider any token $i$ and temperature $T > 0$. Let $l_i$ be the original logit. After temperature scaling, we have $l'_i = l_i / T$ for all tokens. For any given token $i$:

$$M' = \max_j (l'_j) = \frac{M}{T}$$

$$\begin{aligned} \sigma' &= \sqrt{\frac{1}{N} \sum_j (l'_j - \mu')^2} \\ &= \sqrt{\frac{1}{N} \sum_j (\frac{l_j}{T} - \frac{\mu_j}{T})^2} = \frac{\sigma}{T} \end{aligned}$$

Token $i$ is selected if and only if $l'_i \ge M' - n\sigma'$. Substituting $l_i$, $M$ and $\sigma$ for $l'_i$, $M'$ and $\sigma'$:

$$\begin{aligned} l'_i \ge M' - n\sigma' &\iff \frac{l_i}{T} \ge \frac{M}{T} - \frac{n\sigma}{T} \\ &\iff l_i \ge M - n\sigma \end{aligned}$$

This final condition is independent of $T$. Therefore, for any token $i$, its inclusion in the selected set is determined by the same condition regardless of temperature. □

# B Experimental Details

## B.1 Implementation

One of the major advantages of top-$n\sigma$ is its remarkable simplicity — the core algorithm can be integrated into any inference framework with merely *two* lines of code. To demonstrate this, we provide a reference implementation based on the Huggingface framework (Wolf et al., 2020), shown in Code 1.

Our experiments were primarily conducted using vLLM (Kwon et al., 2023). While the core implementation remains straightforward, it is worth noting that vLLM did not support custom samplers during our experimental phase, which necessitated some intricate adaptations. For the sake of reproducibility and transparency, we have made our vLLM implementation and the corresponding evaluation code available[2].

We note that since the initial development of this approach, independent implementations of top-$n\sigma$

[2] https://anonymous.4open.science/r/top_nsigma_anon-D2A3/readme.md

```python
from transformers import LogitsProcessor
import torch

class TopNSigma(LogitsProcessor):
    def __init__(self, nsigma: float, device: str):
        self.n = torch.tensor(nsigma, device=device)

    def __call__(self, input_ids: torch.Tensor, logits: torch.Tensor) -> torch.
        Tensor:
        M, std = logits.max(dim=-1, keepdim=True).values, logits.std(dim=-1, keepdim
            =True)
        logits[logits < M - self.n * std] = float('-inf')
        return logits
```

Code 1: TopNSigma Logits Processor Implementation

sampling have emerged in several popular open-source LLM inference frameworks and applications[3]. This independent adoption by the open-source community provides additional validation of our method's practical utility.

### B.2 Evaluation Framework

To ensure fair comparison among all algorithms and guarantee reproducibility, we developed a custom evaluation framework. The evaluation pipeline consists of the following key components and steps:

1. **Dataset-specific Preprocessing**: For each dataset, we implement a dedicated preprocessor that:
    - Converts raw data into a standardized format with questions and output format specifications.
    - For multiple-choice questions, structures the data to include questions, options, and output format controls.
    - Handles dataset-specific requirements and constraints.
2. **Input Processing**:
    - Applies template-based formatting to ensure consistent model inputs.
    - Incorporates necessary control tokens and format specifications.
3. **Algorithm Execution**:
    - Loads and configures models with appropriate parameters.
    - Processes the formatted inputs through the models.
    - Collects raw outputs.
4. **Output Processing and Evaluation**:
    - Extracts answers through predefined output pattern matching.
    - Normalize the answers.
    - Computes evaluation metrics based on extracted answers.

This standardized pipeline ensures fair evaluation across different models and datasets. While we utilize vllm's seed option and set random, numpy, pytorch and cuda's seeds to ensure reproducibility, it is important to note that reproducibility may still be affected by GPU's precision error[4].

#### B.2.1 Reasoning

We use a prompt template to instruct LLaMA to follow user's instructions (raw python string).

```
<|begin_of_text|><|start_header_id|>system<|
    end_header_id|>\n\nYou are a helpful expert
    problem solver. Please strictly follow the
    user's instructions, especially the output
    format.<|eot_id|><|start_header_id|>user<|
    end_header_id|>\n\nPlease answer the
    following question:\n\n{question}<|eot_id
    |><|start_header_id|>assistant<|
    end_header_id|>\n\n
```

where `{question}` is a placeholder subject to different datasets. It not only contains the question, but also an output format instruction for extraction. For example, we use following instruction (placed directly after the question) for GSM8K test:

```
Your response *must* end with "The final answer
    is (answer)". No units. For example:\n(
    Question and your reasoning)\nThe final
    answer is 33.
```

[3]Due to double-blind review requirements, we cannot provide specific references to these implementations at this time.

[4]https://github.com/vllm-project/vllm/pull/2514

And the corresponding output regular expression is:

```
The final answer is .*?(\d+\.?\d*|\.\d+)[\s\S
    ]*?(?<!\.)\.$
```

We use slightly different prompts for different datasets because desired answer formats are different and the Exact Match metric is very sensitive to the format. We tried our best to mitigate this issue through prompt engineering, complex regular expressions and output normalization.

**Metrics.** We use **Exact Match** (EM) metric for the four reasoning benchmarks. EM reports the ratio of identical extracted answers and targets.

**Datasets.** The detailed statistics is provided in Table 3.

#### B.2.2 Creative Writing

The pipeline of Creative Writing is slightly different, since there is no ground-truth answer. For each model, the text generated by greedy decoding is set as reference answer, and texts generated by each method would be compared with the reference, judged by DeepseekV3 (we set temperature as 0.0 to guarantee reproducibility). We use the Alpaca (Li et al., 2023) as the LLM-as-a-judge framework.

**Metrics.** We report **win rate** for the creative writing benchmark. For each instruction, we compare responses generated by each method against those produced by standard greedy decoding. The LLM judge (DeepseekV3) receives a pair of responses (A, B) and outputs a binary preference, indicating which response better satisfies the given instruction. The win rate is then calculated as the number of wins divided by the total number of comparisons.

**Datasets.** The dataset is a collection of 500 writing prompts, same as Nguyen et al. (2024).

## C Hyperparameters

The selection of appropriate hyperparameters is crucial for fair comparison among different sampling methods. Our selection criteria are based on three perspectives: (1) widely adopted parameters in the literature and production, (2) authors' recommendations from original papers, and (3) empirically optimal values from our experiments.

It is worth noting that for most sampling methods, extreme parameter settings can achieve robustness to temperature variations. For instance, using an extremely small $p$ value (e.g., 0.1) in top-$p$ sampling can maintain consistency across different temperatures. However, such settings essentially reduce the sampling method to greedy decoding, thereby compromising the method's ability to generate diverse outputs. Similarly, top-$n\sigma$ sampling with an extremely small $n$ (e.g., 0.1) exhibits comparable behavior, but this deviates from practical scenarios and results in a significant loss of diversity. Therefore, in our experimental setup, we avoid such extreme parameter settings that could potentially skew the comparison or lead to degenerate sampling behaviors.

For top-$n\sigma$, $n$ is typically less than 1.5. Its precise lower bound remains unclear as it is difficult to measure the diversity loss. We recommend $n = 1.0$ as the default setting. Although smaller values of $n$ can achieve better accuracy in our experiments (as shown in Figure 5), we choose this value for its simplicity and good balance in maintaining diversity. As part of our hyperparameter recommendations, we suggest different values of $n$ based on the intended use case. For scenarios requiring more rational and focused outputs, we recommend smaller values such as $n = 0.8$. Conversely, for applications emphasizing diversity, larger values like $n = 1.3$ are more appropriate. It is important to note that unlike probability-based metrics, top-$n\sigma$ operates on logits, making the impact of $n$ exponential rather than linear. For instance, with $\sigma = 2.2$, $n = 1.0$ corresponds to min-$p$ sampling with $p = 0.1$, while $n = 2.0$ corresponds to $p = 0.01$ in min-$p$ sampling, which is too loose to be practically useful.

For min-$p$ sampling, the value of $p$ typically ranges from 0.05 to 0.1. We adopt $p = 0.1$ based on our empirical results and its widespread adoption in combination with temperature $T = 1.5$, which has proven to be highly effective in practice.

For top-$k$ sampling, parameter recommendations from various sources are inconsistent, with $k$ ranging from 10 to 300. This parameter is inherently related to the vocabulary size, making some earlier recommendations potentially obsolete due to the evolution of model vocabularies. Our experiments with $k \in \{10, 20, 50, 180, 300\}$ reveal dramatic variations in performance. Small $k$ values make the sampling relatively insensitive to temperature changes but tend to approximate greedy decoding. Conversely, large $k$ values exhibit high temperature sensitivity, leading to rapid degradation at higher temperatures. As a result, we choose $k = 20$ as a compromise. This also explains why top-$k$ is

Table 3: Dataset Statistics

| Dataset | Description | Samples |
|---|---|---|
| AQuA | AQuA (Ling et al., 2017) is an algebraic word problems dataset. We use its test split for experiment. | 254 |
| GSM8K | GSM8K (Cobbe et al., 2021) (Grade School Math 8K) is a dataset of high-quality linguistically diverse grade school math word problems. We use its test split for experiment. | 1319 |
| GPQA-main | GPQA (Rein et al., 2024) is a multiple-choice, Q&A dataset of very hard questions written and validated by experts in biology, physics, and chemistry. | 448 |
| MATH500 | MATH (Hendrycks et al., 2021) is a new dataset of 12,500 challenging competition mathematics problems. Each problem in MATH has a full step-by-step solution which can be used to teach models to generate answer derivations and explanations. We use a 500 subset (Lightman et al., 2023) of it. | 500 |

rarely used alone in practice despite occasionally achieving good performance.

For $\eta$-sampling, we experimented with $\eta = 2 \times 10^{-4}$ and $\eta = 9 \times 10^{-4}$. At lower temperatures, $\eta = 9 \times 10^{-4}$ shows better performance, while at higher temperatures, both values lead to rapid quality degradation. Therefore, we report results using $\eta = 9 \times 10^{-4}$.

For mirostat, we set a relatively high target entropy of $5.0$ to evaluate its performance in diverse text generation. Similar to $\eta$-sampling, even slight temperature increases cause mirostat to deteriorate rapidly.

## D Expanded Experiments using Qwen2.5

We further extended our experiments to the Qwen2.5 series models (Yang et al., 2024) to validate the generalizability of our findings across different model architectures. The results (Table 4 and Table 5) largely align with Llama's. For reasoning tasks, in the low-temperature regime ($T \leq 1.5$), top-$n\sigma$ either outperforms other methods or achieves comparable results across all datasets. In the high-temperature regime ($T \geq 2.0$), top-$n\sigma$ consistently exhibits superior performance across all test scenarios.

For creative writing tasks, top-$n\sigma$ also exhibit similar behavior. It is worth noting that top-$n\sigma$ with $T = 1.0$ demonstrates compelling performance, consistently achieving optimal or near-optimal results across all evaluated models.

## E Combination with other samplers

In practice, multiple samplers are often combined to meet complex decoding requirements. The typical sampling process involves passing a logits vector as an intermediate value through multiple samplers, each performing its specific logic. For instance, a top-$k$ sampler retains the top $k$ logits while setting all others to $-\infty$. Since top-$n\sigma$ sampling requires standard deviation calculations, it must be positioned as the first sampler when used in combination with other sampling methods to avoid computing standard deviations of $-\infty$ values. This constraint is crucial for implementations that set discarded tokens to $-\infty$. However, for implementations that directly discard tokens (e.g., CPU implementations) rather than setting them to $-\infty$, the positioning of top-$n\sigma$ sampling becomes more flexible and is not strictly required to be first. Nevertheless, we recommend placing it early in the sampling pipeline to ensure meaningful standard deviation calculations based on a more complete token distribution.

Top-$n\sigma$ focuses on reflecting the model's inherent capabilities rather than introducing human priors. Therefore, we recommend using it as an alternative to top-$p$ and min-$p$ sampling, which serve similar purposes. Additionally, the community has developed many specialized samplers addressing specific issues, such as DRY (Don't Repeat Yourself)[5] which focuses on reducing repetition. These samplers typically modify the target distribution by introducing human priors, making them orthog-

[5] https://github.com/oobabooga/text-generation-webui/pull/5677

Table 4: Performance comparison of different sampling methods on Qwen2.5 models

| Dataset | Method | Qwen2.5-14B-Instruct 0.7 | 1.0 | 1.5 | 2.0 | 3.0 | 4.0 | Qwen2.5-32B-Instruct 0.7 | 1.0 | 1.5 | 2.0 | 3.0 | 4.0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AQuA | Top-$k$ | **83.46** | 81.10 | 75.20 | 55.91 | 14.96 | 1.57 | 85.43 | 81.10 | 75.98 | 47.64 | 12.60 | 0.79 |
| | Top-$p$ | 81.89 | 82.28 | 72.05 | 11.42 | 0.00 | 0.00 | 86.22 | 84.65 | 81.89 | 9.45 | 0.00 | 0.00 |
| | Min-$p$ | 80.32 | 80.32 | **82.68** | 77.56 | 36.22 | 1.57 | **87.01** | **86.22** | 80.71 | 82.28 | 33.07 | 2.76 |
| | Top-$n\sigma$ | 82.28 | **83.86** | 78.74 | **78.74** | **77.56** | **77.56** | 85.04 | 85.04 | **83.07** | **83.46** | **81.10** | **83.86** |
| GSM8K | Top-$k$ | 91.05 | 91.05 | 88.40 | 77.56 | 12.59 | 1.44 | **93.03** | 92.42 | 91.74 | 81.43 | 13.42 | 1.36 |
| | Top-$p$ | **92.04** | 90.37 | 90.37 | 34.04 | 0.00 | 0.00 | 92.95 | 92.87 | 91.96 | 39.65 | 0.00 | 0.00 |
| | Min-$p$ | 91.58 | 91.13 | **90.75** | 88.55 | 68.46 | 7.88 | 92.95 | 92.80 | **93.18** | 91.66 | 76.65 | 11.83 |
| | Top-$n\sigma$ | 91.05 | **91.21** | 90.45 | **90.67** | **89.01** | **89.84** | **93.03** | **93.18** | 92.42 | **93.03** | **92.49** | **93.33** |
| GPQA-main | Top-$k$ | **43.08** | 38.39 | 35.49 | 31.03 | 18.97 | 6.25 | 41.52 | **44.42** | 36.16 | 29.02 | 15.18 | 4.46 |
| | Top-$p$ | 39.96 | 39.73 | 22.77 | 0.00 | 0.00 | 0.00 | 43.75 | 42.41 | 33.04 | 0.22 | 0.00 | 0.00 |
| | Min-$p$ | 39.73 | 39.73 | 40.18 | 34.82 | 16.52 | 0.45 | **43.97** | 41.74 | **43.97** | 36.16 | 23.66 | 1.79 |
| | Top-$n\sigma$ | 39.29 | **42.86** | **40.85** | **38.17** | **38.84** | **34.60** | 43.75 | 43.97 | 41.74 | **42.63** | **43.97** | **43.75** |
| MATH500 | Top-$k$ | 74.20 | 75.00 | 62.80 | 30.60 | 3.40 | 1.20 | 77.00 | 75.20 | 66.80 | 30.40 | 4.60 | 1.00 |
| | Top-$p$ | 74.80 | 74.80 | 66.00 | 9.00 | 3.20 | 3.20 | 77.20 | **78.40** | 64.20 | 2.80 | 1.40 | 2.20 |
| | Min-$p$ | 73.80 | **76.00** | **72.80** | 65.40 | 15.20 | 1.00 | 76.60 | 74.60 | 71.00 | 68.00 | 17.00 | 1.40 |
| | Top-$n\sigma$ | **76.00** | 74.00 | 72.40 | **72.60** | **70.00** | **69.20** | **79.00** | 78.20 | **76.20** | **76.40** | **74.00** | **71.80** |

Table 5: Win rates (%) against greedy decoding on AlpacaEval Creative Writing using Qwen2.5-14B/32B-Instruct. We omitted the experimental results of top-$p$, mirostat, and $\eta$-sampling at temperature 1.5, as previous experiments on LLaMA have demonstrated their inherent instability at elevated temperatures.

| Method | 14B $T=1.0$ | 14B $T=1.5$ | 32B $T=1.0$ | 32B $T=1.5$ |
|---|---|---|---|---|
| Top-$k$ | 55.80 | 53.40 | 50.00 | 44.20 |
| Mirostat | 47.00 | - | 47.60 | - |
| $\eta$-sampling | 49.00 | - | 50.10 | - |
| Top-$p$ | 53.00 | - | 52.60 | - |
| Min-$p$ | 55.60 | 57.40 | 53.40 | **53.60** |
| Top-$n\sigma$ | **56.40** | **57.80** | **54.40** | 53.40 |
| | $T=3.0$ | $T=10.0$ | $T=3.0$ | $T=10.0$ |
| Top-$n\sigma$ | **57.20** | 51.80 | 52.10 | 52.43 |

onal to top-$n\sigma$'s objectives. As a result, they can complement each other to enhance overall performance.

## F Disscussions about $\sigma$ and sampling process

Given that top-$n\sigma$ directly employs the standard deviation of logits as a measurement criterion, it is crucial to investigate the correlation between standard deviation and the sampling process. Intuitively, one might expect that when the model exhibits uncertainty, the standard deviation of its logits would increase, and conversely, decrease when the model is more confident. However, our empirical studies refute this hypothesis.

We demonstrate this by testing an AQuA example using Llama-3-8B-Instruct, shown below:

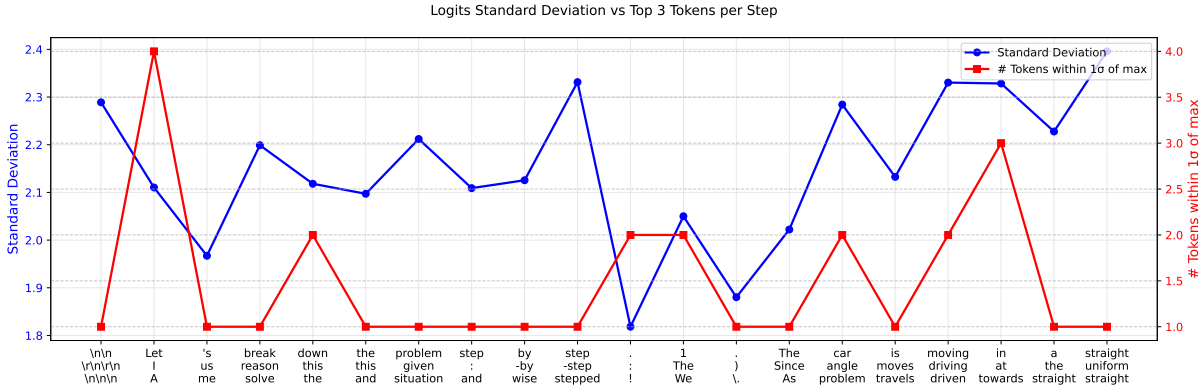
```
<|begin_of_text|><|start_header_id|>system<|
    end_header_id|>You are a helpful expert
    problem solver. Please strictly follow the
    user's instructions, especially the output
    format.<|eot_id|><|start_header_id|>user<|
    end_header_id|>Given the following problem,
    reason and give a final answer to the
    problem.

Question: A car is being driven, in a straight
    line and at a uniform speed, towards the
    base of a vertical tower. The top of the
    tower is observed from the car and, in the
    process, it takes 10 minutes for the angle
    of elevation to change from 45° to 60°.
    After how much more time will this car reach
     the base of the tower?
Choices:
(A)5(√3 + 1)
(B)6(√3 + √2)
(C)7(√3 - 1)
(D)8(√3 - 2)
(E)None of these<|eot_id|><|start_header_id|>
    assistant<|end_header_id|>
```

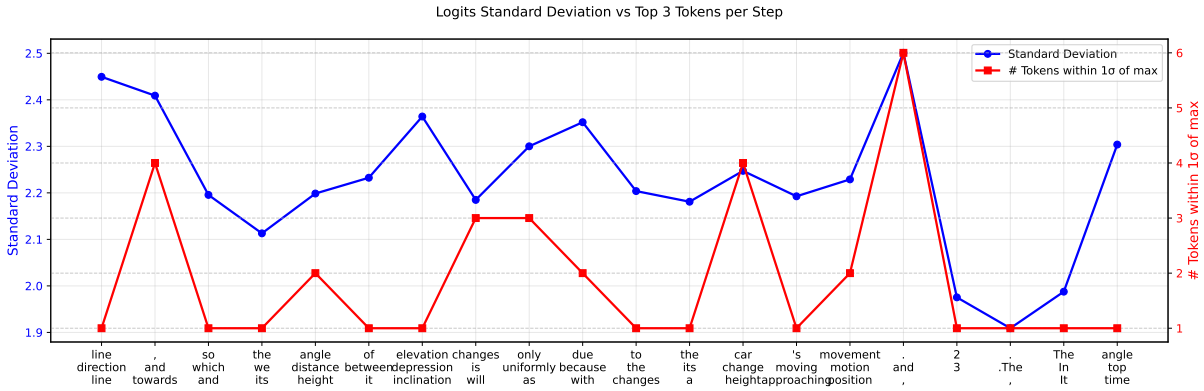
To facilitate the analysis, we visualize three metrics throughout the generation process in Figure 6: the standard deviation of logits, the number of tokens within one standard deviation ($1\sigma$) of the max, and the top-3 candidate tokens. For experimental clarity, we employed greedy decoding where the highest-probability token is selected at each step. For simplicity, we interpret the number of tokens within one standard deviation as a proxy for model confidence — fewer tokens within this range indicate a more concentrated distribution, suggesting higher confidence.

Through this lens, our analysis reveals several interesting patterns:

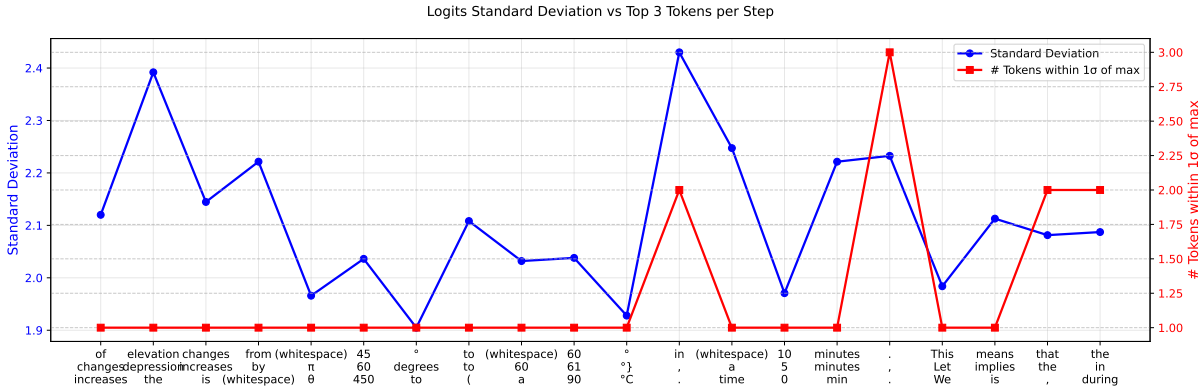
- The standard deviation of logits typically fluctuates between 1.8 and 2.5. We thus pick 2.2




(a) Logits dynamics for tokens 0-20




(b) Logits dynamics for tokens 20-40




(c) Logits dynamics for tokens 40-60

Figure 6: Visualization of logits dynamics during the generation process. Each subplot shows the standard deviation of logits, number of tokens within $1\sigma$ of max, and top-3 next tokens at different generation steps. In most steps the model is highly confident with only one candidate. Notably, multiple tokens tend to fall within one standard deviation primarily during the generation of connective words or punctuation marks, while the overall relationship between standard deviation and model confidence remains inconclusive.

as a typical value.

- There appears to be no strong correlation between the standard deviation and model confidence. We observe cases where high standard deviation coincides with high confidence, as well as cases showing the opposite pattern.

- Notably, the model exhibits lower confidence when generating connective words (e.g., “Let”, “and”) and punctuation marks. This observation is reasonable since connective words are often interchangeable with other connective words (e.g., “in” vs. “at”), and similarly, different punctuation marks can often be substituted for one another (e.g., semicolons vs. periods) while maintaining grammatical correctness.

This lack of clear pattern suggests that the relationship between logits distribution and model confidence may be more complex than initially anticipated.

Based on our preliminary observations, we can only conclude that the distribution of noisy tokens appears to be statistically independent of that of informative tokens. The underlying mechanisms driving this phenomenon and its potential implications remain unclear. Given the exploratory nature of this analysis and its inconclusive results, we present these findings in the appendix rather than the main text. We hope these initial observations, though incomplete, may stimulate future research to better understand the relationship between logits statistics and model behavior during the sampling process.