

Min P Sampling: Balancing Creativity and Coherence at High Temperature

Nguyen Nhat Minh^{*1}, Andrew Baker^{*2}, Andreas Kirsch², Clement Neo^{*1},

¹Apart Research ²Independent

Abstract

Large Language Models (LLMs) generate long-form text by successively sampling the next token based on the probability distribution of the token vocabulary at each decoding step. Current popular truncation sampling methods such as **top- p sampling**, also known as **nucleus sampling**, often **struggle to balance coherence and creativity in generating text**, particularly when **using higher temperatures**. To address this issue, we propose **min- p** , a dynamic truncation sampling method, that establishes a minimum base percentage threshold for tokens, which the scales according to the probability of the top candidate token. Through experiments on several benchmarks, such as **GPQA**, **GSM8K** and **AlpacaEval Creative Writing**, we demonstrate that **min- p improves the coherence and quality of generated text even at high temperatures**, while also facilitating more creative and diverse outputs compared to top- p and other sampling methods. As of writing, **min- p has been adopted by multiple open-source LLM implementations**, and have been independently assessed by members of the open-source LLM community, further validating its practical utility and potential¹.

1 Introduction

Large Language Models (LLMs) have shown a remarkable ability to generate coherent and creative text across various domains. However, the quality and diversity of the generated text depends heavily on the choice of sampling methods (*samplers*) and hyperparameters, such as thresholds and sampling temperature, during the decoding process, where the model autoregressively samples the next token based on the probability distribution over its vocabulary (Shi et al., 2024).

^{*}Equal Contribution.

Correspondence to minh1228@gmail.com

¹Code, evaluation code and results available at https://github.com/menhguin/minp_paper/

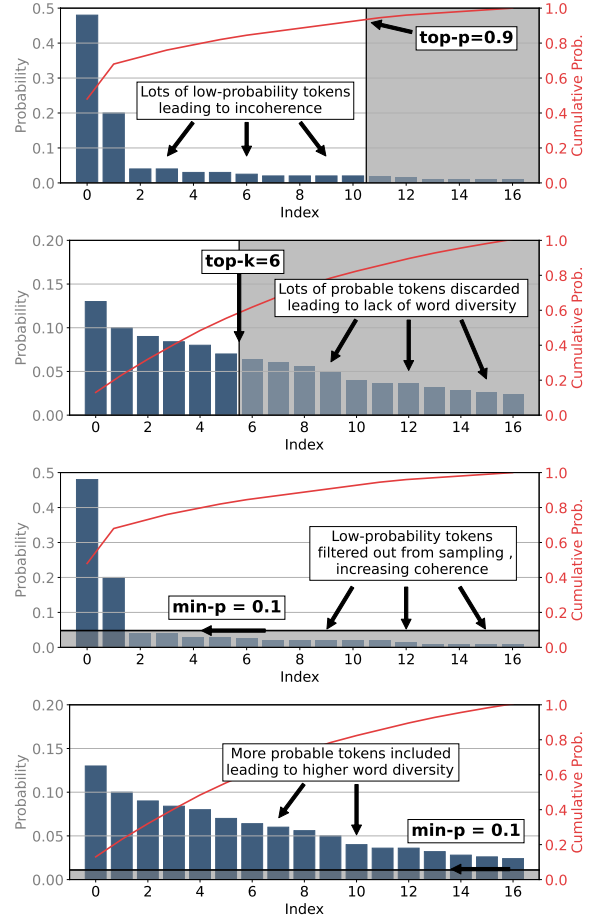


Figure 1: Effects of top- p , top- k , and min- p sampling on token probability distributions. Min- p sampling dynamically adjusts its filtering threshold based on the model’s confidence, focusing on high-probability tokens when confident and including diverse but plausible options when uncertain. This helps min- p balances coherence and diversity more effectively than top- p and top- k sampling.

In particular, **increasing the sampling temperature can enhance creativity by promoting more diverse token selection—but often at the cost of coherence**, as the quality of the generated text tends to degrade quickly at higher temperature with nucleus sampling. **This creativity-coherence trade-off**

is the main reason low conservative temperatures between 0 and 1 are often chosen for sampling. In this work, we will examine whether the creativity-coherence trade-off can be relaxed, that is whether we can achieve higher creativity while preserving coherence.

As of writing, top- p sampling (Holtzman et al., 2020) is the state-of-the-art sampling method that improves coherence by focusing on high-probability tokens and truncating the "unreliable tail" of low-probability tokens that can lead to degenerate text in top- k sampling. However, top- p sampling does not work well for higher temperatures: here, tokens from the "unreliable tail" can still enter the sampling pool.

Inspired by this observation, our proposed truncation sampling method, min- p , dynamically sets a minimum probability threshold for token selection. Min- p maintains coherence at higher temperatures without sacrificing creativity by eliminating the long tail.

In summary, our objectives for min- p are to (1) match or outperform the state-of-the-art top- p sampling across various reasoning and performance benchmarks at standard temperature settings between 0 and 1, (2) better handle the creativity-coherence trade-off at higher temperatures, and (3) provide a simple, effective sampling method without relying on additional techniques like repetition penalties to address the externalities of high-temperature sampling.

To evaluate the performance of min- p sampling, we conduct experiments comparing it with top- p and other popular sampling methods on several LLM benchmarks, spanning open-ended question answering, grade school math, and creative writing. The results demonstrate min- p 's ability to maintain coherence at higher temperatures while outperforming top- p in terms of output diversity. The practical utility of min- p is further validated by its adoption in multiple open-source LLM frameworks.

Our key contributions can be summarized as follows:

1. We introduce min- p , a dynamic truncation sampling method that balances quality and diversity, leading to more creative and coherent outputs at higher temperature settings.
2. We experimentally demonstrate min- p 's advantages over top- p and other samplers on LLM benchmarks, emphasizing its reduced

tradeoffs between creativity and coherence at temperature settings beyond 1.

By achieving these objectives, we establish min- p sampling as a viable and user-friendly alternative to top- p sampling, unlocking the benefits of high-temperature settings for enhanced creativity in LLM-generated text without sacrificing coherence.

2 Background

2.1 Language Modelling

Language models are probabilistic models that assign probabilities to sequences of words or tokens. An autoregressive language model predicts the probability distribution over the vocabulary for the next token, conditioned on the previous tokens, and iteratively samples tokens to generate a sequence. Formally, given a sequence of tokens $\mathbf{x} = (x_1, x_2, \dots, x_n)$ and a finite vocabulary \mathcal{V} , where each token $x_i \in \mathcal{V}$, an autoregressive language model models the probability distribution $P(x_t | x_1, x_2, \dots, x_{t-1})$, for each token x_t in the sequence. This probability distribution represents the likelihood of each token occurring at position t given the preceding tokens.

Greedy decoding. A deterministic way to sample tokens is *greedy decoding*, which selects the token with the highest probability at each step. Given the probability distribution $P(x_t | x_1, x_2, \dots, x_{t-1})$ over the vocabulary \mathcal{V} at position t , it chooses the token \hat{x}_t that maximizes this probability:

$$\hat{x}_t = \arg \max_{v \in \mathcal{V}} P(x_t = v | x_1, x_2, \dots, x_{t-1}). \quad (1)$$

By repeatedly selecting the most probable token at each step, we can generate a sequence of tokens $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$ of arbitrary length m . However, this greedy approach often leads to degenerate output, as it always chooses the most likely continuation without considering other plausible alternatives. This deterministic sampling has found to often result in repetitive or generic generated text that lacks diversity and creativity.

Stochastic Sampling. The stochastic alternative to greedy decoding is sampling of tokens according to the probability distribution $P(x_t | x_1, x_2, \dots, x_{t-1})$:

$$\hat{x}_t \sim P(x_t = \cdot | x_1, x_2, \dots, x_{t-1}). \quad (2)$$

In this case, there are several ways to modify and constrain the sampling distribution to improve the

quality of the generated text. We will look at the most commonly used approaches next.

2.2 Temperature Scaling

Temperature scaling modifies the sharpness of the token distribution. The term temperature is inspired by thermodynamics and the Boltzmann distribution: higher temperatures correspond to more uniform mixing that more uniform sampling while lower temperatures converge to the argmax (Ackley et al., 1985).

Formally, given a probability distribution $P(x_t|x_1, x_2, \dots, x_{t-1})$ over the vocabulary \mathcal{V} at position t , temperature sampling first divides the logits (unnormalized log probabilities) by a temperature parameter $\tau > 0$ before passing them through the softmax function to obtain a modified probability distribution $P_\tau(x_t = v|x_1, \dots, x_{t-1})$:

$$P_\tau(x_t = v|x_1, \dots, x_{t-1}) = \frac{\exp(z_v/\tau)}{\sum_{v' \in \mathcal{V}} \exp(z_{v'}/\tau)} \quad (3)$$

where $z_v = \log P(x_t = v|x_1, x_2, \dots, x_{t-1})$ is the (unnormalized) log probability of token v . The next token \hat{x}_t is then randomly sampled from the modified probability distribution:

$$\hat{x}_t \sim P_\tau(v|x_1, x_2, \dots, x_{t-1}) \quad (4)$$

The temperature parameter τ controls the randomness of the sampling process. Higher values of τ (e.g., $\tau > 1$) result in a more uniform probability distribution, increasing the chances of sampling lower-probability tokens and generating more diverse output. Lower values of τ (e.g., $\tau < 1$) make the distribution sharper, favoring higher-probability tokens and more conservative and deterministic outputs. Setting $\tau = 1$ leaves the original probability distribution unchanged.

2.3 Top-p Sampling

Top- p sampling, also known as nucleus sampling, is a stochastic truncation decoding method that aims to address the limitations of greedy decoding (Holtzman et al., 2020). Instead of always selecting the most probable token like in greedy decoding, top- p sampling introduces stochasticity while still focusing on high-probability tokens.

Given $P_\tau(x_t = v|x_1, x_2, \dots, x_{t-1})$ over the vocabulary \mathcal{V} at position t , top- p sampling first sorts the tokens in descending order of their probabilities. It then selects the smallest set of tokens whose

cumulative probability exceeds a predefined threshold p , where $p \in (0, 1]$. Formally, let $\mathcal{V}_p \subseteq \mathcal{V}$ be the smallest set such that:

$$\sum_{v \in \mathcal{V}_p} P_\tau(x_t = v|x_1, x_2, \dots, x_{t-1}) \geq p \quad (5)$$

The next token \hat{x}_t is then randomly sampled from this reduced set \mathcal{V}_p according to the renormalized probabilities:

$$\hat{x}_t \sim \frac{P_\tau(x_t = v|x_1, \dots, x_{t-1})}{\sum_{v' \in \mathcal{V}_p} P_\tau(x_t = v'|x_1, \dots, x_{t-1})} \text{ for } v \in \mathcal{V}_p$$

By restricting the sampling to the nucleus of high-probability tokens, top- p sampling allows for more diverse and coherent generated text compared to greedy decoding.

However, combining truncation methods like top- p sampling with temperature scaling can lead to suboptimal results. When using top- p sampling, it is recommended to adjust either the value of p or the temperature τ , but not both simultaneously (OpenAI API, 2024) as they can lead to conflicting effects.

3 Min-p Sampling

We introduce min- p sampling, a stochastic truncation sampling method that maintains a minimum level of token diversity while still focusing on high-probability tokens. The method uses a relative probability threshold $p_{base} \in (0, 1]$ to scale the maximum token probability p_{max} to determine the absolute probability threshold p_{scaled} . Sampling is then performed on tokens with probability greater than or equal to p_{scaled} .

Formally, given the maximum probability over the token distribution $p_{max} = \max_{v \in \mathcal{V}} P(v|x_1, \dots, x_{t-1})$, the absolute probability threshold p_{scaled} is calculated as:

$$p_{scaled} = p_{base} \times p_{max} \quad (6)$$

The sampling pool \mathcal{V}_{min} is then defined as the set of tokens whose probability is greater than or equal to p_{scaled} :

$$\mathcal{V}_{min} = \{v \in \mathcal{V} : P(v|x_1, x_2, \dots, x_{t-1}) \geq p_{scaled}\} \quad (7)$$

Finally, the next token \hat{x}_t is randomly sampled from the set \mathcal{V}_{min} according to the normalized probabilities:

$$\hat{x}_t \sim \frac{P(v|x_1, \dots, x_{t-1})}{\sum_{v' \in \mathcal{V}_{min}} P(v'|x_1, \dots, x_{t-1})} \text{ for } v \in \mathcal{V}_{min} \quad (8)$$

4 Case Study

In this section, we inspect the behavior of min- p with top- p when both methods are used in conjunction with temperature. Briefly:

- **High-confidence predictions:** When the model assigns high probability to a particular next token, min- p filters out low-probability alternatives. This preserves coherence by avoiding sampling from the "unreliable tail" that could derail generation.
- **Low-confidence predictions:** When there is no clear front-runner, min- p relaxes its filter. This allows the model to sample from a more diverse set of plausible continuations, enabling creativity in open-ended settings.

We illustrate these differences in the following two examples². Table 1 shows a case where the next token probabilities are relatively low, while Table 2 shows a case with a clear high probability next token.

4.1 Case 1: Low-Certainty Next Token

Table 1 shows the token probabilities for a creative writing prompt where there are many plausible ways to continue the story. At temperature $\tau = 1$, the highest probability token "was" only has a 70.3% probability. When the temperature is increased to $\tau = 3$, this drops to 11.9% as probability mass shifts to other candidate tokens. In this low-certainty case, applying top- p and min- p to the post-temperature distribution yields similar results, as there is no clear top token for min- p to prioritize.

4.2 Case 2: High-Certainty Next Token

In contrast, Table 2 shows a case where using min- p has a clear impact. The prompt is a factual statement about rainbows, with "light" being the obvious next token with 98.25% probability when temperature $\tau = 1$.

Raising the temperature to $\tau = 3$ flattens out the probability distribution, but "light" remains dominant at 34.41%. When applying top- p , several additional tokens like "water", "sunshine", "a" still remain in the sampling pool at ~2-3% probability each. While these are vaguely plausible, they are much lower quality continuations compared to "light".

Min- p , on the other hand, truncates much more aggressively. Setting a 10% base probability threshold when "light" has 34.41% probability leads min- p to only include "light" and "sunlight" in its sampling pool, as all other tokens have less than the scaled 3.441 percent probability threshold. This showcases min- p 's ability to maintain coherence at high temperatures by calibrating its cutoff to the strength of the top candidate.

Prompt				
"You will pay for what you have done," she hissed, her blade flashing in the moonlight. The battle that ensued _____				
Sampling Method	Temperature $\tau = 1$	Temperature $\tau = 3$	+top- p $\tau = 3$ $p = 0.9$	+min- p $\tau = 3$ $p = 0.1$
was	70.3	11.9	13.1	18.5
lasted	9.5	6.1	6.7	9.5
between	6.2	5.3	5.9	8.2
left	4.5	4.8	5.3	7.4
would	3.2	4.3	4.7	6.6
could	1.2	3.1	3.4	4.8
seemed	0.5	2.3	2.5	3.5

Table 1: Comparison between temperature sampling, top- p , min- p sampling at higher temperatures, for a low-certainty case. In cases where the tokens are relatively low-certainty, min- p does not differ significantly from top- p .

Prompt				
A rainbow is an optically brilliant meteorological event, resulting from the refraction, reflection, and dispersion of _____				
Sampling Method	Temperature $\tau = 1$	Temperature $\tau = 3$	+top- p $\tau = 3$ $p = 0.9$	+min- p $\tau = 3$ $p = 0.1$
light	98.25	34.41	38.18	80.90
sunlight	1.29	8.12	9.01	19.10
water	0.10	3.44	3.82	0.00
sunshine	0.06	2.89	3.21	0.00
a	0.05	2.71	3.01	0.00
moisture	0.05	2.70	3.00	0.00

Table 2: Comparison between temperature sampling, top- p , min- p sampling at higher temperatures, for a high-certainty case. In cases where the tokens are high-certainty, min- p filters for the high probability token more aggressively than top- p . In this case, when $\tau = 3$, top- p lets initially low probability tokens (<0.10% probability) into its nucleus (~3% probability), while min- p filters them out entirely.

²Interactive demo available at <https://artefact2.github.io/llm-sampling/index.xhtml>. Demo not affiliated with authors.

4.3 Discussion

Our case studies demonstrate that min- p sampling’s token choices differ most notably from top- p sampling when selecting high-probability tokens at elevated temperatures. This has important implications for improving the performance of language models in high-temperature settings:

Maintaining accuracy in reasoning tasks. For tasks that require factual and logical reasoning, such as mathematics, science, programming, and recalling specific facts, min- p significantly reduces the accuracy tradeoff at high temperatures. In the rainbow example (Table 2), 100% of min- p ’s probability distribution ("light" and "sunlight") can be considered accurate, with the most relevant answer "light" comprising 80.90%. In contrast, only 47.19% of top- p ’s distribution is relevant to light. This suggests that min- p can enable language models to tackle reasoning tasks at temperature settings previously considered unsuitable. We further evaluate min- p ’s reasoning capabilities in Section 5.

Mitigating cascading errors. Language models construct longform outputs by sequentially selecting tokens, with previous choices influencing future decisions. Consequently, erroneous token selections can compound, increasing the likelihood of further inaccurate or incoherent choices. The rainbow prompt illustrates this, as it contains multiple technical terms ("meteorological", "refraction", "dispersion") that require high-accuracy token selections to maintain semantic coherence and factual correctness. Min- p ’s ability to robustly select high-confidence tokens, even at high temperatures, helps prevent such cascading failures.

5 Experiments

To establish min- p sampling as a viable alternative to the state-of-the-art top- p (nucleus) sampling, we conduct experiments to demonstrate that:

1. Min- p achieves equal or better benchmark performance compared to top- p at standard temperature settings ($\tau \in [0, 1]$).
2. Min- p handles the creativity-coherence trade-off well at higher temperatures ($\tau > 1$) by achieving more diverse outputs while maintaining coherence, as demonstrated by improved performance on reasoning benchmarks at high temperatures.

5.1 Experimental Setup

We evaluate the performance of min- p sampling against top- p sampling on three generative language model tasks using a Mistral 7B (Jiang et al., 2023) model:

- **Graduate-level reasoning:** Google-Proof Q&A Benchmark - Main, Generative (GPQA Main) - 5-shot (Rein et al., 2023)
- **Grade school math:** GSM8K Chain-of-Thought (GSM8K COT) - 8-shot (Cobbe et al., 2021)
- **Creative writing:** AlpacaEval Creative Writing (Li et al., 2023b)

While 7B parameter models typically achieve less than 50% accuracy on these challenging benchmarks, this is sufficient for comparing the relative performance between truncation sampling methods.

We primarily focus on comparing min- p values in the range of 0.05 to 0.1 with top- p values between 0.9 and 0.95. These are common settings preferred by users, although they are not directly comparable. In general, a higher min- p value or a lower top- p value results in more selective token choices and a bias towards more likely outputs.

For GPQA and GSM8K, we used the EleutherAI Evaluation Harness (Gao et al., 2023).

5.2 Graduate-Level Reasoning (GPQA Main)

To evaluate min- p ’s impact on graduate-level reasoning, we employ the Google-Proof Q&A Benchmark (GPQA) (Rein et al., 2023), a challenging dataset of 448 multiple-choice questions written by domain experts in biology, physics, and chemistry. GPQA is designed to test difficult, graduate-level domain expertise with minimal dataset leakage, making it particularly suitable for our experiments, as Mistral 7B was released in September 2023, two months before GPQA’s public release.

5-shot prompting. Initial experiments revealed that the baseline Mistral 7B model at temperature 1 with zero-shot prompting was unable to answer any question correctly (0% accuracy). Few-shot prompting proved necessary to obtain statistically significant and meaningful comparisons. After discussions with GPQA’s lead author regarding standard few-shot settings for a 7B parameter model, we opted for 5-shot prompting in this evaluation.

5.3 Grade School Math (GSM8K COT)

To evaluate min- p 's impact on mathematical reasoning, we employ the GS8MK Chain-of-Thought (GSM8K COT) dataset, which consists of 8.5K high-quality, linguistically diverse grade school math word problems. We consider this a rigorous, falsifiable, quantitative test of reasoning that 7B parameter models can somewhat manage, albeit with significant performance degradations at higher temperatures. Our prompting set-up is as follows:

8-shot prompting. Initial experiments revealed that the baseline Mistral 7B model at temperature 1 with zero-shot prompting was unable to answer any question correctly (0% accuracy). **Few-shot prompting proved necessary to obtain statistically significant and meaningful comparisons.** We chose 8-shot prompting, as it is the recommended baseline on the EleutherAI Evaluations Harness and widely used in reporting GSM8K COT results.

Chain-of-Thought (COT). We utilize Chain-of-Thought prompting for two reasons. First, baseline Mistral 7B scored too low on GSM8K 8-shot for meaningful comparisons (25%), while GSM8K COT 8-shot yielded a more helpful baseline of 38% at temperature 1. Second, COT more accurately simulates real-world environments that require multi-step reasoning.

Exact Match. We primarily report Exact Match scores, in line with standard reporting practices, although Flexible Match scores generally favor min- p even more than Exact Match scores.

5.4 Creative Writing (AlpacaEval Creative Writing)

To evaluate min- p 's impact on creative writing tasks, we employ the AlpacaEval Creative Writing Benchmark (Li et al., 2023b). AlpacaEval is an LLM-based automatic evaluation framework that is fast, cost-effective, replicable, and validated against 20K human annotations. The Creative Writing Benchmark utilizes a frontier LLM evaluator (in this case, Anthropic's Claude) to select between two outputs generated from creative writing prompts.

We chose this benchmark due to the significant number of users within the open-source community who have qualitatively reported a preference for min- p sampling on high-temperature creative writing tasks. This preference has contributed to min- p 's rapid adoption within the open-source language model community. At the time of writing,

we do not have access to other creativity or creative writing evaluation datasets, whether automated or human-annotated.

We note that these AlpacaEval Creative Writing results are primarily reproduced with permission from another independent evaluator within the open-source community (Gusev, 2023). This independent evaluator is one of many open-source community members who have taken interest in min- p within the months after its release, conducting their own evaluation which we then reproduced via the same procedure.

6 Results

We find that min- p sampling outperforms top- p across various temperature settings and benchmarks, maintaining coherence while allowing for increased diversity at higher temperatures (>1).

6.1 Graduate-Level Reasoning (GPQA Main)

Figure 3 presents the results of the GPQA Main Generative evaluation. We observe that min- p sampling slightly outperforms top- p sampling at a temperature of 1.

As the temperature increases, both truncation sampling methods experience a degradation in benchmark performance. However, min- p 's performance degradation is noticeably lower compared to top- p , indicating that it more reliably answers graduate-level reasoning questions at higher temperatures.

It is notable that min- p sampling exhibits a slight improvement in reasoning performance between temperatures 1 and 3. This suggests that the increased diversity in token selection introduced by min- p at these temperature levels may be beneficial for complex reasoning tasks, as it allows the model to explore a broader range of plausible solutions without significantly sacrificing coherence.

6.2 Grade School Math (GSM8K COT)

Figure 2 shows the results of the GSM8K Chain-of-Thought evaluation. Similar to the GPQA results, min- p sampling slightly outperforms top- p sampling at a temperature of 1. As the temperature increases, both truncation sampling methods experience a decline in benchmark performance. However, min- p 's performance degradation is noticeably lower compared to top- p , suggesting that it more reliably derives the correct answer to mathematical word problems at higher temperatures.

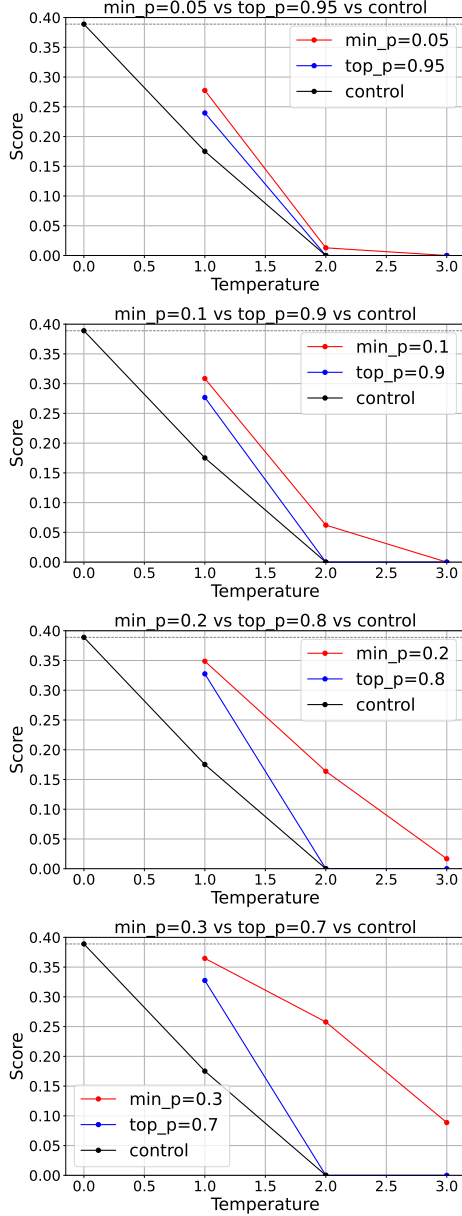


Figure 2: Results of running min- p vs top- p on GSM8K. The control method used is pure sampling. A single plot with all results is available in Figure 4.

These results demonstrate min- p 's ability to maintain higher levels of coherence and accuracy in multi-step reasoning tasks, even when the diversity of token selection is increased through higher temperature settings. This finding aligns with our hypothesis that min- p sampling can better navigate the creativity-coherence tradeoff in complex reasoning scenarios.

6.3 AlpacaEval

Table 3 shows that min- p displays strong performance on the AlpacaEval Creative Writing Benchmark, especially at $p = 0.1$ at $\tau = 1.5$, a temper-

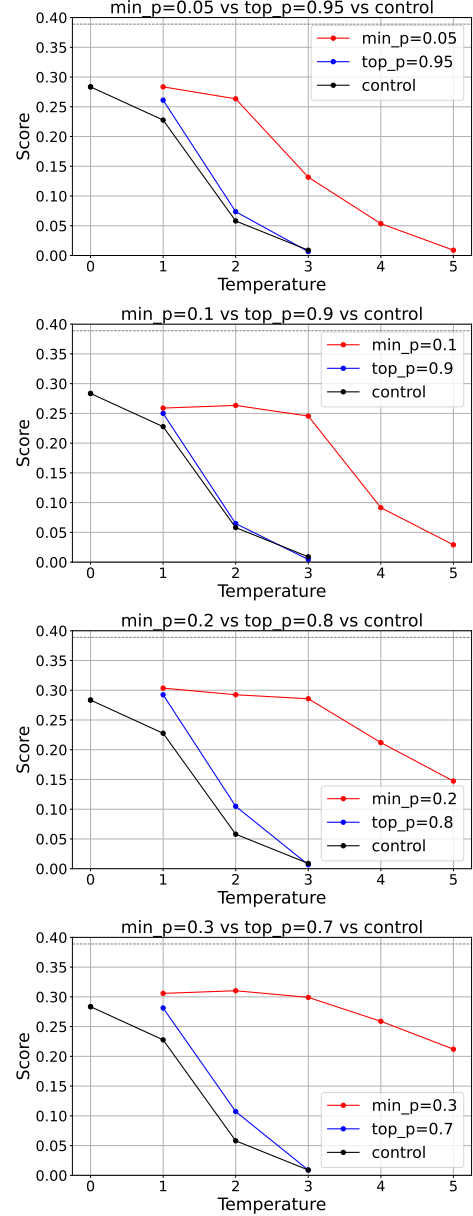


Figure 3: Results of running min- p vs top- p on GPQA. The control method used is pure sampling. A single plot with all results is available in Figure 5.

ature where top- p struggles to produce coherent creative writing.

7 Different Min- p Configurations

As of June 2024, users of open-source LLM inference engines generally recommend min- p settings of 0.05-0.1 and temperature 1 for normal usage, temperature 1-1.5 for varied text generation while balancing factual and logical tasks, and min- p 0.05-0.1 with temperature 2-5 for roleplay and creative writing. Higher values of min- p base percentage (e.g., min- $p = 0.9$) result in significantly reduced degradation for temperature scaling values >20 .

Temp	Constraints	Winrate (LC)	Winrate
1.5	min_p=0.1	58.12	56.54
1.0	min_p=0.05	55.07	52.01
0.8	top_p=0.98	54.65	51.29
1.0	min_p=0.1	53.24	50.14
1.0	top_p=0.98	53.00	50.43
1.0	top_p=0.9	52.07	50.07
0.8	top_p=0.95	51.80	50.22
1.0	min_p=0.02	51.62	50.43
1.0	min_p=0.02	51.46	48.85
0.8	min_p=0.05	50.99	47.84
0.8	top_p=0.95	50.76	48.78
1.0		50.0	50.0
0.0		49.90	46.64
1.5	min_p=0.05	48.13	48.57
1.5	min_p=0.02	42.09	44.83
1.0	mirostat_tau=4.0	16.04	16.69
1.0	mirostat_tau=5.0	16.03	16.40
1.5		FAIL	FAIL
1.5	top_p=0.98	FAIL	FAIL
1.5	top_p=0.95	FAIL	FAIL
1.5	top_p=0.9	FAIL	FAIL
1.5	top_p=0.8	FAIL	FAIL

Table 3: AlpacaEval on creative writing prompts with different sampling parameters. LC-winrate is winrate with length control, which alleviates the length biases of LLM-based judging. Min- p results are in **bold**.

Sampling Order. Min- p can be used with any other sampling method, in any order. Applying min- p after temperature scaling is recommended for steerability. Applying min- p before temperature scaling results in the same token pool as temperature 1, with temperature only shifting relative probabilities, leading to higher accuracy for high-certainty tokens but constrained diversity.

Combining Sampling Methods. Combining multiple sampling methods is possible, it has not been thoroughly tested, and no particular benefits have been observed or reported, likely because sampling methods are designed to be used in isolation.

8 Related Work

In addition to temperature sampling and top- p sampling, several alternative sampling methods have been proposed, which Shi et al. (2024) classify as deterministic and stochastic. This section focuses on stochastic sampling methods, as min- p sampling falls under this category.

Top- k sampling (Fan et al., 2018) is a truncation-based method that restricts the sampling pool to the k most probable tokens at each step.

Mirostat sampling (Basu et al., 2021) builds upon top- k sampling by dynamically adjusting the value of k at each generation step to maintain a target perplexity level, enabling the generation of coherent and diverse text without requiring manual tuning of the sampling parameters.

Locally typical sampling (Meister et al., 2023) is a sampling method that limits the sampling pool to words with negative log-probability within a certain range from the conditional entropy of the distribution at that time step, which helps to avoid both highly unlikely and overly generic tokens.

Contrastive decoding (Li et al., 2023a) uses an auxiliary language model to penalize tokens that are unlikely under the main language model, which guides the selection of higher-quality text while ensuring the generated output remains plausible.

η -sampling (Hewitt et al., 2022) is an entropy-dependent truncation method that sets a probability threshold based on the entropy of the conditional distribution at each step, truncating words below this threshold.

9 Conclusion

In this work, we introduced min- p sampling, a novel dynamic truncation sampling method that aims to balance creativity and coherence in text generation from large language models, particularly at high temperatures. Through experiments on multiple LLM benchmarks, including graduate-level reasoning (GPQA), grade school math (GSM8K), and creative writing (AlpacaEval), we demonstrated that min- p outperforms the popular top- p sampling method in maintaining coherence while enabling more diverse outputs at temperatures above 1. The rapid adoption of min- p by the open-source LLM community further validates its practical utility and potential. By effectively navigating the trade-off between creativity and coherence, min- p sampling opens up new possibilities for high-quality open-ended text generation.

Further Work. Future research directions for min- p sampling include investigating its performance on a wider range of NLP tasks, such as dialogue generation, story completion, and language translation. A more rigorous theoretical analysis of min- p ’s behavior may provide valuable insights for developing even more effective sampling strategies.

9.1 Limitations

Although our study of min- p has shown promising results, it is not without limitations. In particular, we focus on a single model architecture (Mistral 7B) and a limited set of benchmarks. To establish the generality of min- p 's benefits, evaluating its performance across a diverse range of model sizes, architectures, and tasks is important. Furthermore, the AlpacaEval Creative Writing benchmark, while cost-effective and validated against human judgments, may not fully capture all aspects of creativity in text generation. Human evaluation studies could provide additional insights into the quality and creativity of text generated using min- p compared to other sampling methods using the same models. At the same time, the results position min- p as a computationally cheap alternative to nucleus sampling, especially when creativity and higher diversity of the generated text are important.

References

- David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. 1985. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169.
- Sourya Basu, Govardana Sachitanandam Ramachandran, Nitish Shirish Keskar, and Lav R. Varshney. 2021. Mirostat: A neural text decoding algorithm that directly controls perplexity. *Preprint*, arXiv:2007.14966.
- 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. *Preprint*, arXiv:2110.14168.
- 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.
- Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.
- Ilya Gusev. 2023. Quantitative evaluation of modern llm sampling techniques. <https://github.com/IlyaGusev/quest>.
- John Hewitt, Christopher Manning, and Percy Liang. 2022. Truncation sampling as language model desmoothing. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3414–3427, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. *ICLR 2020*.
- Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.
- Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023a. 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, Toronto, Canada. Association for Computational Linguistics.
- Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. AlpacaEval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.
- Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2023. Locally typical sampling. *Transactions of the Association for Computational Linguistics*, 11:102–121.
- OpenAI API. 2024. Openai api reference - completions. Accessed: 2024-06-10.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *Preprint*, arXiv:2311.12022.
- Chufan Shi, Haoran Yang, Deng Cai, Zhisong Zhang, Yifan Wang, Yujiu Yang, and Wai Lam. 2024. A thorough examination of decoding methods in the era of llms. *arXiv preprint arXiv:2402.06925*.

AI Assistance. AI Assistance was used to check for typos and grammatical errors in writing, as well as propose better phrasings for existing text.

A Min- p Implementation and Documentation

Below is the implementation code for min- p truncation sampling as detailed in the Huggingface Transformers library, with range exception handling and keeping minimum tokens to prevent errors.

This implementation code, along with other similar implementations in other open-source inference engines, logs of automated evaluations for GPQA, GSM8K Chain-of-Thought and AlpacaEval Creative Writing is available at https://github.com/menhguin/minp_paper/.

```
1 class MinPLogitsWarper(LogitsWarper):
2     def __init__(self, min_p: float, filter_value: float = -float("Inf"),
3         ↪ min_tokens_to_keep: int = 1):
4         if min_p < 0 or min_p > 1.0:
5             raise ValueError(f"`min_p` has to be a float >= 0 and <= 1, but is {
6                 ↪ min_p}")
7         if not isinstance(min_tokens_to_keep, int) or (min_tokens_to_keep < 1):
8             raise ValueError(f"`min_tokens_to_keep` has to be a positive integer,
9                 ↪ but is {min_tokens_to_keep}")
10
11         self.min_p = min_p
12         self.filter_value = filter_value
13         self.min_tokens_to_keep = min_tokens_to_keep
14
15     def __call__(self, input_ids: torch.LongTensor, scores: torch.FloatTensor) ->
16         ↪ torch.FloatTensor:
17         # Convert logits to probabilities
18         probs = torch.softmax(scores, dim=-1)
19         # Get the probability of the top token for each sequence in the batch
20         top_probs, _ = probs.max(dim=-1, keepdim=True)
21         # Calculate the actual min_p threshold by scaling min_p with the top token's
22         ↪ probability
23         scaled_min_p = self.min_p * top_probs
24         # Create a mask for tokens that have a probability less than the scaled
25         ↪ min_p
26         tokens_to_remove = probs < scaled_min_p
27
28         sorted_indices = torch.argsort(scores, descending=True, dim=-1)
29         sorted_indices_to_remove = torch.gather(tokens_to_remove, dim=-1, index=
30             ↪ sorted_indices)
31         # Keep at least min_tokens_to_keep
32         sorted_indices_to_remove[..., : self.min_tokens_to_keep] = False
33
34         indices_to_remove = sorted_indices_to_remove.scatter(1, sorted_indices,
35             ↪ sorted_indices_to_remove)
36         scores_processed = scores.masked_fill(indices_to_remove, self.filter_value)
37         return scores_processed
```

B Full Plots for GSM8K/GPQA results

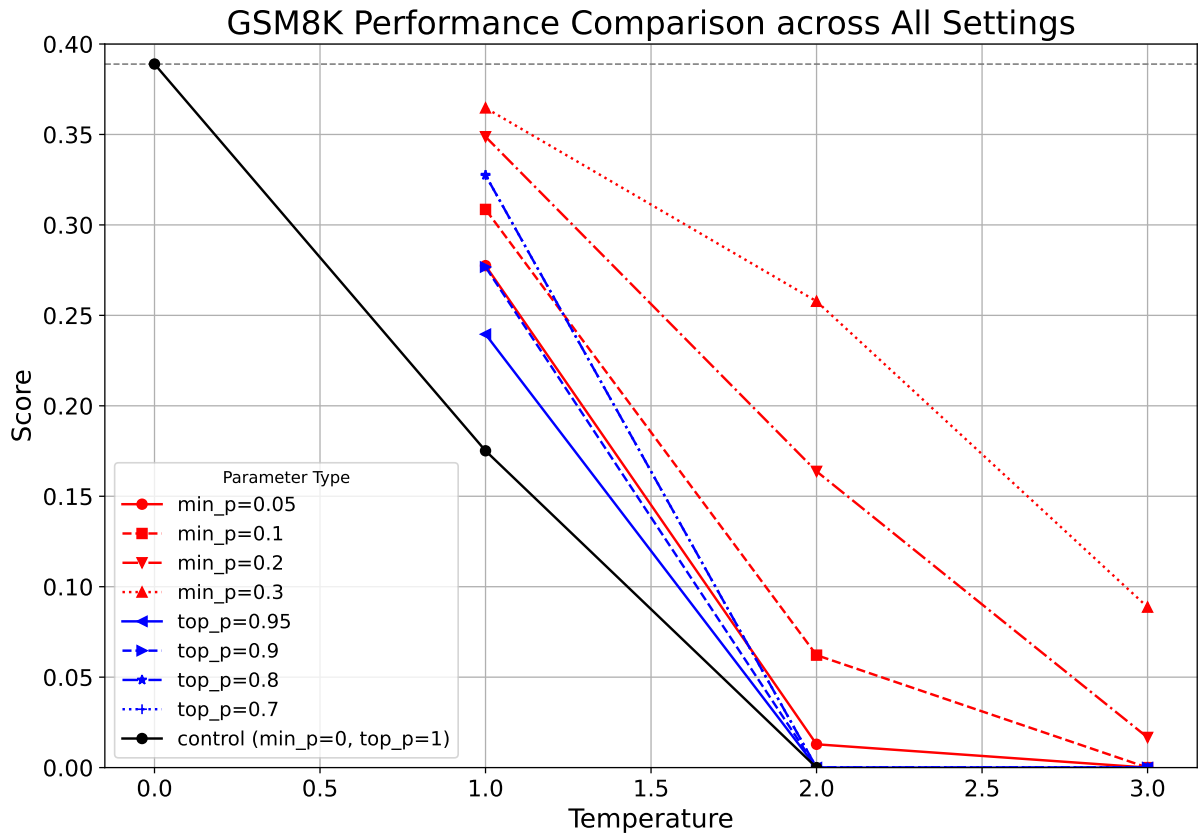


Figure 4: Results for all GSM8K experiments on a single plot.

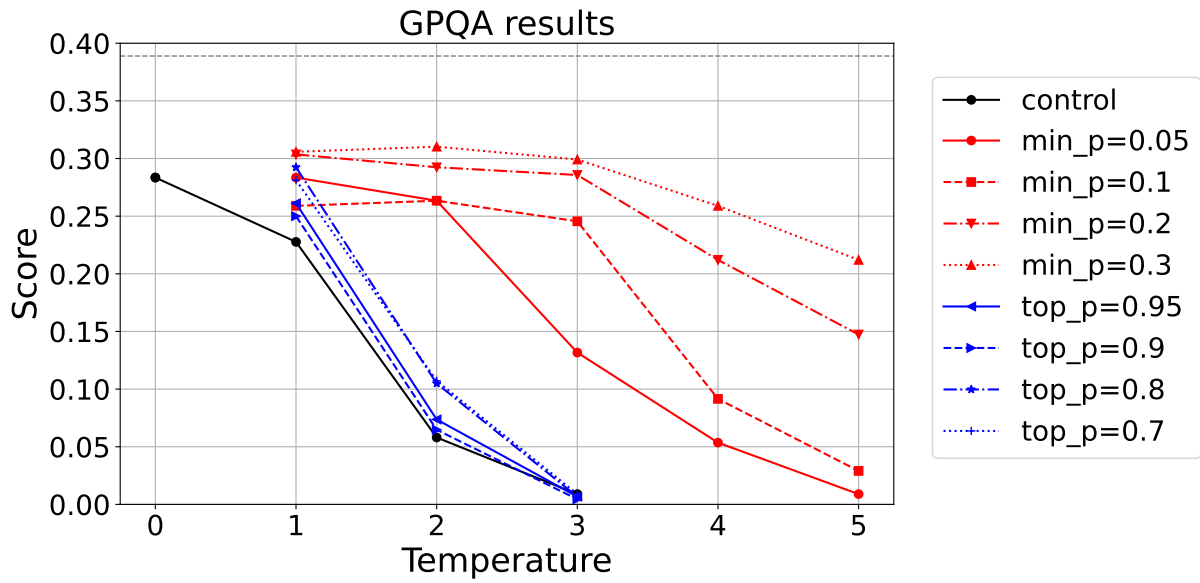


Figure 5: Results for all GPQA experiments on a single plot.