NUCLEUS SPECULATIVE DECODING: PLAUSIBILITY-AWARE VERIFICATION BEYOND EXACT DISTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Speculative decoding accelerates autoregressive generation by using a lightweight draft model to propose multiple tokens that are verified by a target model in parallel. However, the standard acceptance rule focuses on exact distribution correction and rejects tokens that remain highly plausible under the target model when the draft model assigns excess probability. This conservative verification limits the number of draft tokens retained after each verification forward pass. We introduce Nucleus Speculative Decoding (NSD), a relaxed verification method that incorporates target-model plausibility into speculative decoding. NSD accepts a draft token if it satisfies the standard acceptance rule or belongs to the target model's nucleus. We theoretically characterize the distributional deviation introduced by our method and show that the single-step error is exactly determined by the draft model's excess probability within the target nucleus. We further derive sequence-level fidelity bounds that quantify how local deviations accumulate over autoregressive decoding. Experiments across multiple target models and proposal mechanisms demonstrate that NSD consistently improves speculative decoding efficiency while maintaining competitive task performance. Our method achieves throughput speedups of $4.23\times$ over autoregressive decoding and $1.85\times$ over standard speculative decoding. These improvements coincide with longer accepted lengths, allowing more output tokens to share the cost of each target verification pass. Analysis shows that plausibility-aware verification provides an effective approach for relaxed verification and speculative decoding efficiency.

1 INTRODUCTION

Large language models (LLMs) have demonstrated strong capabilities across language understanding, reasoning, and code generation (Yang et al., 2025; Chen et al., 2021), but their autoregressive decoding (AR) remains inherently sequential and computationally expensive. Speculative decoding (SD) mitigates this bottleneck by allowing a lightweight draft model to propose multiple tokens that are subsequently verified by the target model in parallel (Leviathan et al., 2023; Chen et al., 2023). The acceleration of SD depends on how many draft tokens are accepted during verification.

Despite substantial progress in improving draft models (Cheng et al., 2026), target model verification remains a major component of speculative decoding cost. As shown in Figure 1(a), target verification accounts for 57.1%–78.1% of wall-clock time. For draft proposals, verification terminates at the first rejected position and all subsequent proposals are discarded. Consequently, an early rejection can substantially reduce the number of tokens accepted from a single target-model verification.

For a token x sampled from the draft model probability distribution q, speculative decoding accepts x with probability $\min\{1, p(x)/q(x)\}$, where p denotes the target model probability distribution. This acceptance rule is determined by the relative probability assigned to x by the target and draft models, rather than by an explicit measure of the token's plausibility under the target distribution. So rejection under speculative decoding does not necessarily indicate that a proposal is implausible under target model. Consequently, when $q(x) > p(x)$, a proposal may be rejected even if x is highly ranked or lies within the target model's high-probability region.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

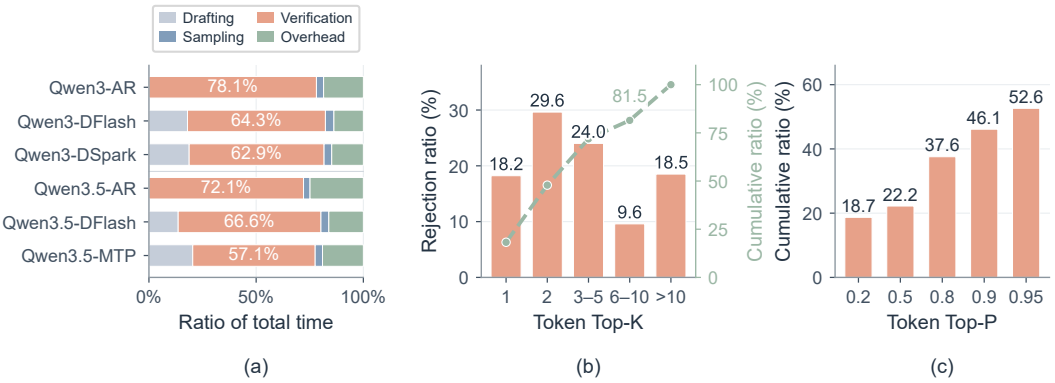


Figure 1: Execution time breakdown and rejection analysis of speculative decoding. (a) Execution time breakdown. Ratio of total time spent on drafting, verification, sampling, and overhead. (b) Target Top-K rank from rejected tokens. (c) Target Top-P distribution from rejected tokens.

Experiments reveal that this distinction appears frequently in practice. Figure 1(b) shows that 81.5% of these rejected tokens still rank within the target's top ten tokens. At the commonly used setting Top-P=0.95, Figure 1(c) shows that 52.6% of the rejected tokens still belong to this region. Thus, a substantial fraction of proposals rejected by SD remain within the target nucleus.

Together, these observations expose an inefficiency in speculative decoding: target model verification is the main bottleneck, yet it still rejects many proposals that remain plausible for target model. To fix this issue, we introduce **Nucleus Speculative Decoding** (NSD), a simple but effective method that supplements standard speculative decoding with a target-nucleus acceptance condition. At each proposed position, the target nucleus contains the highest-probability tokens whose cumulative probability mass reaches a threshold (Holtzman et al., 2020). NSD accepts a proposal if it either passes the speculative verification or belongs to target nucleus.

Although NSD relaxes the exact distribution guarantee of speculative decoding, we prove that its distributional deviation is exactly determined by the draft model's excess probability within the target nucleus. We further establish sequence-level fidelity guarantees that characterize the deviation. Across evaluations on five benchmarks with backbones from the Nemotron and Qwen families, NSD achieves a mean throughput improvement of 323.0% over AR and 85.0% over SD.

2 Related Work

Speculative Decoding. Early blockwise parallel decoding predicts several positions together and retains the longest prefix validated by a scoring model (Stern et al., 2018). Speculative decoding accelerates autoregressive generation by using a lightweight draft model to propose future tokens that will be verified in parallel by the target model (Leviathan et al., 2023; Chen et al., 2023). For a draft proposal sampled from distribution q, speculative decoding accepts tokens with probability $\min\{1, p(x)/q(x)\}$ and applies residual correction upon rejection to preserve the target distribution p. Its efficiency depends on the number of draft tokens retained from verification. SpecInfer organizes proposals into token trees for parallel target model verification (Miao et al., 2024), while Draft & Verify generates proposals by skipping target model layers (Zhang et al., 2024).

Recent advances have improved speculative decoding through stronger draft models, including feature-based drafters (Li et al., 2025a; 2024; 2025b), multi-token prediction modules (Gloeckle et al., 2024; DeepSeek-AI et al., 2025), parallel diffusion drafting (Chen et al., 2026), semi-autoregressive proposal strategies (Cheng et al., 2026), and unified autoregressive diffusion models (Fu et al., 2026). These approaches primarily improve the quality or efficiency of draft generation. In contrast, NSD modifies the verification procedure and is complementary to existing drafting mechanisms. Therefore, NSD can be applied across different speculative decoding paradigms.

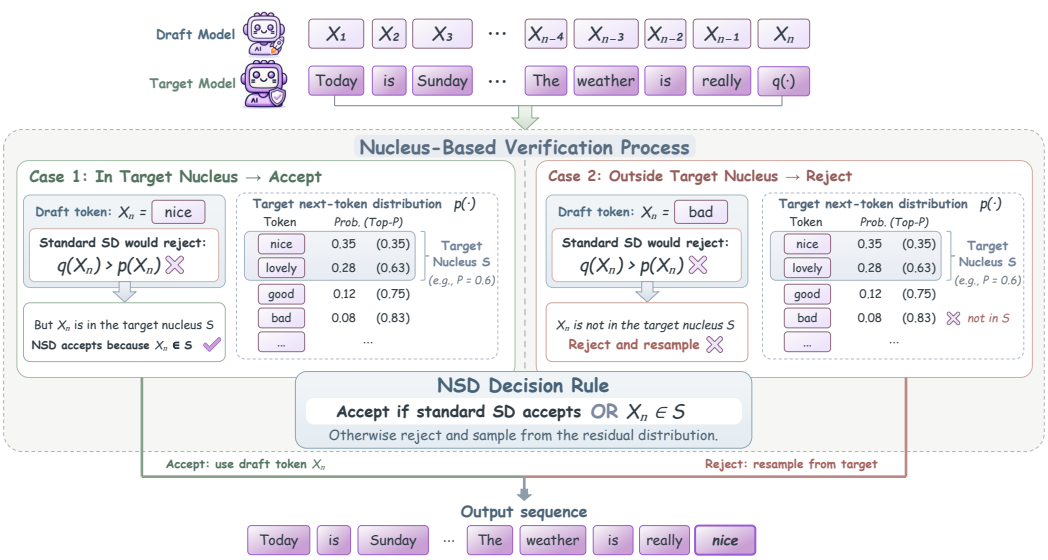


Figure 2: Overview of Nucleus Speculative Decoding (NSD).

Relaxed Verification. Several works relax exact verification to improve acceptance rates under an accuracy-efficiency trade-off. The lenience extension of speculative decoding modifies the likelihood ratio criterion, while typical acceptance and related methods introduce target probability or ranking thresholds (Leviathan et al., 2023; Cai et al., 2024). Other approaches explore adaptive verification rules, approximate sampling, learned judges, or regret constraints (Song et al., 2026; Urbán et al., 2026; Hao & Mou, 2026; Bachmann et al., 2025; Holsman et al., 2025; Feng et al., 2026). These methods generally rely on learned components or optimization objectives. Training-free verification based on draft-target KL divergence has also been explored as an alternative to learned judges (Sun et al., 2026). Inspired by nucleus sampling (Holtzman et al., 2020), NSD introduces a different relaxation: a draft token is accepted if it satisfies the standard acceptance rule or belongs to the target nucleus. Unlike approaches that replace the verification criterion, NSD retains the original residual correction and bonus sampling mechanisms, providing a simple nucleus-based acceptance condition using target probabilities already computed during verification.

3 Nucleus Speculative Decoding

We introduce Nucleus Speculative Decoding (NSD), a simple yet effective method for speculative decoding. NSD accepts a draft proposal if it passes the standard acceptance verification or belongs to the target model's top-p set. This method can extend the acceptance length using target probabilities already computed during verification. Figure 2 illustrates the workflow of our method.

3.1 Background on Speculative Decoding

Let $p(\cdot|h)$ and $q(\cdot|h)$ denote the target and draft distributions conditioned on a prefix h. Given a committed prefix h_0, the draft model generates a block of K candidate tokens, and the target model evaluates their conditional probabilities in parallel. To preserve the exact target distribution, standard speculative decoding accepts each draft token x_i using the stochastic rule $a_i^{\mathrm{SD}}(x_i) = \min\{1, p_i(x_i)/q_i(x_i)\}$ where p_i and q_i denote the target and draft distributions at position i. Verification continues until the first rejection, at which point a replacement token is sampled from the residual distribution. If all draft tokens are accepted, a bonus token is sampled.

Although standard speculative decoding provides exact distribution preservation, its efficiency depends on the number of draft tokens surviving verification. Moreover, a token can be rejected when $q_i(x_i) > p_i(x_i)$ even if it remains highly ranked under the target distribution. NSD addresses this limitation by introducing an additional acceptance condition based on target nucleus verification.

3.2 NUCLEUS-BASED VERIFICATION

NSD augments speculative decoding using target nucleus verification. For each verification position, we construct a nucleus from the target distribution p_i using a nucleus threshold $\theta \in (0, 1)$:

$$S_i(\theta) = \{v_{i,(1)}, \ldots, v_{i,(k_i(\theta))}\} \quad \text{where} \quad k_i(\theta) = \min \left\{ k : \sum_{j=1}^{k} p_i(v_{i,(j)}) \geq \theta \right\}. \tag{1}$$

The nucleus adapts to the target distribution at each position and is only used as an additional acceptance condition. NSD does not renormalize p_i or q_i over $S_i(\theta)$. When $q_i(x_i) > p_i(x_i)$, standard speculative decoding may reject x_i even when $x_i \in S_i(\theta)$. NSD accepts such proposals through the additional nucleus-based condition: $b_i = \mathbf{1}\{u_i \leq a_i^{\mathrm{SD}}(x_i) \vee x_i \in S_i(\theta)\}$.

Algorithm 1 gives one verification cycle. Here q_i is the actual conditional proposal distribution, and both p_i and q_i include sampling transformations. Nucleus ties use a fixed ordering. The generation wrapper stops at EOS or the output budget and discards uncommitted cache states. The implementation-specific numerical safeguards are described separately in Appendix B.

Algorithm 1 Nucleus Speculative Decoding: one verification cycle

Require: Token prefix h_0; draft length K; nucleus threshold $\theta \in (0, 1)$; target model p; draft model q
Ensure: Newly emitted token block o (at most $K + 1$ tokens)

```
Generate x_{1:K} with the draft mechanism; retain the actual conditional proposal distributions q_{1:K}
Evaluate p_i = p(· | h_0, x_{<i}) for i = 1, ..., K + 1 in one target verification pass
o ← ()
for i = 1, ..., K do
    Construct S_i(θ) from p_i using Equation (1)
    Sample u_i ~ Uniform(0, 1); set a_i^SD(x_i) ← min{1, p_i(x_i)/q_i(x_i)}
    if u_i ≤ a_i^SD(x_i) or x_i ∈ S_i(θ) then
        Append x_i to o
    else
        ρ_i(v) ← [p_i(v) − q_i(v)]_+ / Σ_{w∈V} [p_i(w) − q_i(w)]_+
        Sample y ~ ρ_i; append y to o; return o
Sample bonus token y ~ p_{K+1}; append y to o; return o
```

3.3 DISTRIBUTIONAL FIDELITY

We characterize the distributional deviation introduced by nucleus-based acceptance. Let $S = S_\theta(p)$ denote the target nucleus, and let $D_{\mathrm{TV}}(r, s) = \frac{1}{2} \sum_v |r(v) - s(v)|$. Define

$$R = D_{\mathrm{TV}}(q, p) = \sum_{v \in \mathcal{V}} [q(v) - p(v)]_+, \quad E_S = \sum_{v \in S} [q(v) - p(v)]_+. \tag{2}$$

R is the rejection probability of standard speculative decoding, while E_S is the draft excess probability mass within the target nucleus. Under a shared proposal $X \sim q$ under uniform distribution, E_S is exactly the probability that NSD accepts a proposal rejected by standard SD. Therefore, $\bar{a}_{\mathrm{SD}} = 1 - R$, $\bar{a}_{\mathrm{NSD}} = 1 - R + E_S$, and the increase in proposal acceptance is $\bar{a}_{\mathrm{NSD}} - \bar{a}_{\mathrm{SD}} = E_S$.

For $R > 0$, let $\rho(v) = [p(v) - q(v)]_+ / R$ denote the SD residual distribution. Since NSD only rejects the draft excess outside the nucleus, residual resampling decreases from R to $R - E_S$, so

$$\widetilde{p}(v) = \min\{p(v), q(v)\} + \mathbf{1}\{v \in S\}[q(v) - p(v)]_+ + (R - E_S)\rho(v). \tag{3}$$

Subtracting $p(v)$ gives $\widetilde{p}(v) - p(v) = \mathbf{1}\{v \in S\}[q(v) - p(v)]_+ - E_S \rho(v)$. The two terms have disjoint support: the first is nonzero only where $q(v) > p(v)$, while $\rho(v)$ is supported only where $p(v) > q(v)$. Hence

$$D_{\mathrm{TV}}(\widetilde{p}, p) = E_S = \bar{a}_{\mathrm{NSD}} - \bar{a}_{\mathrm{SD}}. \tag{4}$$

Equation (4) shows that the extra acceptance of NSD is exactly its local deviation from the target. Since $E_S \leq R$, this deviation is no larger than the rejection probability of standard SD.

We next give a conditional sequence-level guarantee. Let $K_t(\cdot \mid h)$ denote the actual next-emitted-token distribution of the decoder, with internal state marginalized conditional on the emitted history h. If $D_{\mathrm{TV}}(K_t(\cdot \mid h), p_t(\cdot \mid h)) \le \epsilon_t$ for every relevant history, then a sequential coupling gives

$$D_{\mathrm{TV}}\Big(P_{\mathrm{NSD}}^{(T)}, P_p^{(T)}\Big) \le 1 - \prod_{t=1}^{T}(1-\epsilon_t).$$

Applying this bound to a particular speculative backend requires establishing the conditional-kernel assumption, including proposal selection, recovery, bonus sampling, and decoder state. Equivalently, NSD and target decoding can generate identical sequences under a coupling with probability at least $\prod_{t=1}^{T}(1-\epsilon_t)$. For a uniform per-token error bound $\epsilon_t \le \epsilon$, this becomes

$$\Pr\big(X_{1:T}^{\mathrm{NSD}} = X_{1:T}^{\mathrm{target}}\big) \ge (1-\epsilon)^T \ge 1 - T\epsilon. \tag{5}$$

Therefore, when the nucleus-based relaxation introduces only small single-token deviations, NSD remains close to target-model distribution. For illustration, $\epsilon \le 10^{-3}$ and $T = 32$ imply coupling agreement at least $(0.999)^{32} \approx 0.9685$. Detailed proofs are provided in Appendix A.

4 Experiments

4.1 Experimental Setup

Models. We evaluate NSD across five different proposal mechanisms and four model sizes, covering Nemotron-Labs-Diffusion-3B, Nemotron-Labs-Diffusion-8B, Nemotron-Labs-Diffusion-14B (Fu et al., 2026), and Qwen3 models at 4B, 8B, and 14B sizes (Yang et al., 2025). In our main experiments, Nemotron employs linear self-speculation with 32-token proposal blocks, while Qwen3 is paired with the DSpark draft model (Cheng et al., 2026) using 7-token blocks. To evaluate whether NSD generalizes across different speculative decoding paradigms, we additionally conduct experiments with DFlash (Chen et al., 2026) and EAGLE-3 (Li et al., 2025b) on Qwen3, as well as MTP on Qwen3.5 (Qwen Team, 2026) . These additional results are reported in Appendix D.

Evaluation. We evaluate mathematical reasoning on GSM8K (Cobbe et al., 2021), MATH500, a subset of MATH Hendrycks et al., 2021), AIME24, AIME25, and AIME26, as well as functional code generation on MBPP (Austin et al., 2021) and HumanEval (Chen et al., 2021).

Metrics. We evaluate both task performance and efficiency. Task performance reports accuracy for mathematical reasoning tasks and functional pass@1 for MBPP and HumanEval. Speedup Ratio is the ratio of each method's token throughput to target model autoregressive (AR) throughput, where AR is normalized to $1.00\times$. The mean accepted length, denoted by τ, is the number of tokens emitted per verification cycle, which includes accepted draft tokens and any resampling or bonus token. So τ can exceed the draft block size by up to one token.

Baselines. We use target model autoregressive decoding as the primary reference for generation quality and latency. We compare Nucleus Sampling (NS), which modifies the sampling distribution through temperature and nucleus truncation. Standard speculative decoding (SD) (Leviathan et al., 2023; Chen et al., 2023) serves as the exact distribution-preserving acceleration baseline. We further compare with relaxed speculative verification methods, including ASD (Feng et al., 2026), Cactus (Hao & Mou, 2026), and MARS (Song et al., 2026), which trade exact verification guarantees for improved acceptance or efficiency. As for decoding parameters, all methods use temperature $T = 1.0$. For Nucleus Sampling, we use $top - p = 0.95$ following the common decoding configuration. For NSD, we set the nucleus threshold $\theta = 0.95$ in the main experiments.

4.2 Main Results

Consistent throughput speedup. Table 1 shows that NSD achieves the highest reported speedup among the evaluated methods in all comparisons. Across models, the mean speedups range from $3.70\times$ to $4.63\times$ relative to AR decoding. For Nemotron models, NSD achieves overall speedups of $3.70\times$, $4.35\times$, and $4.36\times$. The gains also extend to Qwen3 with DSpark drafting: NSD increases the

Table 1: Speedup, mean accepted length and accuracy performance overview. AR denotes autoregressive sampling; NS denotes nucleus sampling on AR mode; Diff denotes diffusion-based parallel decoding. Spd.: throughput speedup relative to target-model AR; τ: mean tokens accepted per draft-block verification, including any resampling token; Acc.: math accuracy or code pass@1 (%). Blue rows denote NSD; bold numbers mark the highest speedup and largest τ within each comparison.

		GSM8K			MATH500			AIME			HumanEval			MBPP		
Model	Method	Spd.	τ	Acc.	Spd.	τ	Acc.	Spd.	τ	Acc.	Spd.	τ	Acc.	Spd.	τ	Acc.
Nemo-3B	AR	1.00×	1.00	87.95	1.00×	1.00	74.60	1.00×	1.00	8.00	1.00×	1.00	70.73	1.00×	1.00	56.03
	NS	0.99×	1.00	88.32	0.99×	1.00	76.16	0.99×	1.00	10.00	0.99×	1.00	74.63	0.99×	1.00	57.04
	Diff	2.67×	-	90.07	2.59×	-	73.60	2.44×	-	3.33	1.63×	-	68.90	1.52×	-	56.81
	SD	2.73×	7.30	88.63	2.17×	7.49	74.52	2.05×	6.01	10.00	1.66×	5.61	73.41	1.52×	5.04	54.86
	MARS	3.00×	9.75	85.97	3.60×	9.75	73.20	3.45×	9.12	11.11	2.63×	6.97	74.39	2.54×	6.58	54.09
	ASD	3.69×	9.58	85.22	3.69×	10.09	75.80	3.53×	9.21	13.33	2.50×	6.61	75.61	2.36×	6.12	56.81
	CACT	3.87×	10.17	85.37	3.82×	10.79	69.80	3.68×	9.54	5.56	2.90×	7.77	72.56	2.78×	7.26	52.14
	NSD	**3.96×**	**10.44**	88.26	**4.42×**	**11.51**	75.96	**3.95×**	**10.24**	13.33	**3.08×**	**7.98**	75.24	**3.08×**	**7.52**	55.25
Nemo-8B	AR	1.00×	1.00	89.84	1.00×	1.00	82.20	1.00×	1.00	24.44	1.00×	1.00	79.76	1.00×	1.00	61.09
	NS	1.00×	1.00	90.90	1.00×	1.00	82.40	0.98×	1.00	28.89	1.01×	1.00	80.12	0.97×	1.00	60.31
	Diff	2.02×	-	93.93	3.06×	-	83.80	2.83×	-	26.67	1.68×	-	82.07	1.36×	-	61.48
	SD	2.20×	5.64	93.56	1.50×	5.59	87.72	1.49×	5.08	27.78	1.50×	4.51	81.46	1.38×	4.08	59.53
	MARS	4.18×	10.52	93.03	3.38×	9.61	87.80	3.39×	10.12	28.89	2.94×	7.69	79.27	2.65×	6.68	57.59
	ASD	4.08×	10.37	92.70	4.19×	11.90	88.60	3.59×	10.48	33.33	3.43×	8.82	81.10	2.92×	7.39	61.87
	CACT	4.57×	12.01	88.64	4.14×	11.69	82.80	3.57×	10.69	30.00	3.99×	10.44	80.24	3.53×	9.70	58.37
	NSD	**4.87×**	**12.21**	92.57	**4.40×**	**13.18**	88.56	**4.50×**	**12.87**	31.11	**4.13×**	**10.98**	81.95	**3.83×**	**9.87**	62.26
Nemo-14B	AR	1.00×	1.00	93.48	1.00×	1.00	84.80	1.00×	1.00	33.33	1.00×	1.00	80.24	1.00×	1.00	60.70
	NS	1.01×	1.00	94.23	1.00×	1.00	85.00	1.00×	1.00	32.22	0.99×	1.00	80.85	1.00×	1.00	61.48
	Diff	2.29×	-	93.25	2.37×	-	84.80	2.93×	-	20.00	1.49×	-	79.14	1.54×	-	63.04
	SD	2.31×	5.80	94.31	1.72×	5.11	88.36	1.31×	4.23	36.67	1.58×	4.39	77.20	1.40×	3.84	61.48
	MARS	4.73×	12.00	92.80	3.82×	10.24	87.40	3.45×	9.99	31.11	3.17×	8.00	82.32	2.72×	6.73	58.37
	ASD	4.62×	11.87	91.58	4.21×	11.09	86.80	3.30×	9.59	34.44	3.50×	8.78	81.71	2.84×	6.92	59.53
	CACT	4.87×	12.20	90.14	4.19×	11.26	84.40	3.32×	9.37	32.22	3.92×	10.19	77.44	3.64×	9.02	58.75
	NSD	**5.16×**	**12.64**	92.87	**4.38×**	**12.71**	87.82	**4.12×**	**11.36**	36.67	**4.24×**	**11.27**	81.34	**3.92×**	**9.74**	59.92
Qwen-4B	AR	1.00×	1.00	89.84	1.00×	1.00	83.00	1.00×	1.00	20.22	1.00×	1.00	85.12	1.00×	1.00	61.63
	NS	0.99×	1.00	89.31	0.99×	1.00	84.20	1.06×	1.00	14.89	0.99×	1.00	85.24	0.99×	1.00	62.65
	SD	3.73×	6.18	90.90	2.87×	5.63	84.36	3.16×	4.39	21.11	3.01×	5.46	83.90	2.90×	5.14	62.96
	MARS	4.11×	6.87	91.58	3.29×	6.03	84.00	3.41×	5.49	18.89	3.49×	5.92	81.71	3.35×	5.54	62.65
	ASD	3.84×	6.44	89.39	3.09×	5.65	83.80	3.09×	5.02	20.00	3.37×	5.72	81.71	3.16×	5.27	59.92
	CACT	4.12×	6.99	88.62	3.63×	6.68	79.00	4.09×	6.35	13.33	3.74×	6.30	81.10	3.57×	5.90	61.48
	NSD	**4.28×**	**7.17**	89.77	**3.86×**	**6.84**	84.10	**4.13×**	**6.53**	20.00	**3.93×**	**6.43**	82.32	**3.81×**	**6.02**	62.10
Qwen-8B	AR	1.00×	1.00	92.04	1.00×	1.00	83.60	1.00×	1.00	20.00	1.00×	1.00	87.80	1.00×	1.00	63.81
	NS	0.99×	1.00	93.32	1.00×	1.00	84.00	1.02×	1.00	18.89	0.99×	1.00	84.76	0.99×	1.00	64.59
	SD	4.16×	6.32	91.66	3.25×	5.70	84.64	3.04×	4.36	17.78	3.35×	5.46	87.44	3.25×	5.20	66.15
	MARS	4.35×	6.68	90.91	3.12×	5.98	84.00	3.32×	5.51	20.00	3.79×	5.86	84.76	3.56×	5.56	63.04
	ASD	4.26×	6.50	93.56	3.36×	5.63	85.60	2.94×	4.92	21.11	3.59×	5.54	85.36	3.52×	5.35	64.12
	CACT	4.44×	7.09	91.91	4.02×	6.78	82.40	3.88×	6.38	12.22	4.21×	6.49	85.98	4.07×	6.15	64.98
	NSD	**4.76×**	**7.24**	92.30	**4.31×**	**6.87**	86.40	**4.12×**	**6.50**	21.11	**4.35×**	**6.58**	86.46	**4.14×**	**6.25**	64.59
Qwen-14B	AR	1.00×	1.00	93.93	1.00×	1.00	87.40	1.00×	1.00	22.22	1.00×	1.00	85.98	1.00×	1.00	66.93
	NS	0.99×	1.00	92.80	0.99×	1.00	87.60	1.00×	1.00	18.89	0.99×	1.00	90.24	1.00×	1.00	66.54
	SD	4.53×	6.31	93.78	3.66×	5.68	86.88	3.10×	4.25	25.56	3.66×	5.40	89.51	3.63×	5.25	67.78
	MARS	4.81×	6.74	93.56	3.98×	5.95	85.60	3.40×	5.37	21.11	4.05×	5.74	89.63	4.02×	5.59	68.09
	ASD	4.61×	6.44	93.78	3.76×	5.56	86.80	2.98×	4.71	26.67	3.86×	5.46	89.02	3.82×	5.31	68.09
	CACT	5.02×	7.12	92.19	4.56×	6.79	84.80	4.00×	6.35	17.78	4.55×	6.44	88.41	4.42×	6.15	68.09
	NSD	**5.15×**	**7.22**	93.18	**4.75×**	**6.87**	87.40	**4.16×**	**6.53**	24.44	**4.64×**	**6.53**	90.24	**4.47×**	**6.25**	68.48

overall speedups to $4.00\times$, $4.34\times$, and $4.63\times$ for Qwen3-4B, -8B, and -14B respectively. Averaging the paired throughput ratios across all 30 comparisons gives mean improvements of 323.0% over AR and 84.98% over SD. These results demonstrate efficiency gains under both evaluated proposal mechanisms, different model sizes and diverse model backbones.

Increased mean accepted length. The throughput gains coincide with increases in the reported mean accepted length τ across every model and benchmark group. For Nemotron-3B, -8B, and

Table 2: Effect of draft block size K and nucleus threshold θ on speedup, mean accepted length, and accuracy. Spd. denotes speedup relative to target-model AR; τ is the mean accepted length per draft block verification; Acc. is math accuracy or code pass@1 (%).

		Nucleus threshold θ														
		0.95			0.925			0.90			0.875			0.85		
Dataset	K	Spd.	τ	Acc.	Spd.	τ	Acc.	Spd.	τ	Acc.	Spd.	τ	Acc.	Spd.	τ	Acc.
Nemo-8B																
GSM8K	8	2.92×	6.61	91.66	2.89×	6.61	91.66	2.83×	6.67	91.66	2.69×	6.51	92.42	2.65×	6.54	92.87
	16	4.35×	10.70	91.58	3.63×	9.42	93.48	4.21×	9.66	93.94	3.93×	9.56	93.10	4.06×	9.46	92.72
	24	4.55×	11.90	92.87	4.60×	11.74	90.90	4.65×	12.09	90.14	4.70×	12.03	90.98	4.89×	12.47	91.58
	32	4.87×	12.21	92.57	5.04×	13.03	91.43	5.36×	13.57	93.63	4.88×	12.26	91.74	5.30×	13.53	92.42
MBPP	8	2.72×	6.17	61.48	2.53×	6.23	59.53	2.70×	6.27	60.70	2.48×	6.04	59.14	2.57×	6.20	62.26
	16	3.52×	8.15	59.92	3.47×	7.93	59.92	3.43×	7.82	58.75	3.30×	7.56	60.31	3.26×	7.77	62.65
	24	3.79×	9.79	58.75	3.92×	9.67	58.37	3.77×	9.48	59.53	3.94×	9.66	58.75	3.65×	8.88	62.26
	32	3.83×	9.87	62.26	3.88×	9.46	60.63	3.95×	9.66	59.92	3.77×	9.06	61.48	3.61×	8.88	61.88
Nemo-14B																
GSM8K	8	2.91×	6.65	90.90	2.97×	6.81	91.74	2.93×	6.74	92.42	2.70×	6.54	93.18	2.98×	6.61	94.01
	16	3.82×	9.33	90.14	4.04×	9.83	91.58	3.99×	9.79	92.95	4.16×	10.08	93.03	4.26×	10.29	93.78
	24	4.66×	11.40	90.45	4.67×	11.12	92.57	4.25×	10.62	92.04	4.42×	10.91	92.95	4.56×	11.19	93.48
	32	5.16×	12.64	92.87	5.00×	12.53	92.80	5.11×	12.31	92.70	4.76×	11.69	93.10	4.86×	11.81	93.86
MBPP	8	2.59×	6.11	58.37	2.60×	6.29	59.53	2.56×	5.95	63.42	2.56×	5.92	62.26	2.56×	5.95	63.81
	16	3.40×	8.16	60.70	3.32×	7.89	61.48	3.39×	8.10	62.57	3.31×	7.91	61.48	3.25×	8.03	64.20
	24	3.68×	9.08	59.14	3.81×	9.53	61.87	3.57×	8.73	61.87	3.37×	8.15	60.70	3.53×	8.63	63.04
	32	3.92×	9.74	59.92	3.59×	8.87	60.31	3.59×	8.86	62.65	3.70×	9.19	61.87	3.58×	8.89	62.26

-14B, the overall τ increases from 6.29, 4.98, and 4.67 under SD to 9.54, 11.82, and 11.54 under NSD. For Qwen3-4B, -8B and -14B, the corresponding values increase from 5.36, 5.41 and 5.38 to 6.60, 6.69 and 6.68. NSD also achieves the largest reported τ among the compared verification methods. These observations are consistent with NSD retaining more draft tokens.

Task performance trade-offs. Task performance varies across models and benchmark groups. NSD achieves higher reported overall accuracy than SD on five models with increases ranging from 0.05 to 1.32 percentage points. Qwen3-4B is the exception, where overall accuracy decreases from 68.65% to 67.66% as the overall speedup increases from 3.13× to 4.00×. Across the 30 model-benchmark comparisons, NSD exceeds SD in 15 accuracy entries, ties one, and falls below SD in 14. Across all comparisons, NSD's accuracy difference relative to SD ranges from -1.58 to $+4.14$ percentage points, with NSD achieving 0.40 percentage points higher accuracy on average. These comparisons show consistent throughput gains alongside task-dependent quality changes.

4.3 Ablation Study

We vary the nucleus threshold θ over five values: 0.85, 0.875, 0.90, 0.925, and 0.95. We also vary the draft block length K over 8, 16, 24, and 32 tokens. Table 2 reports the joint effects on decoding efficiency and task performance for Nemotron-8B and Nemotron-14B on GSM8K and MBPP.

Effect of draft block length. Increasing K from 8 to 32 improves both speedup and mean acceptance length for both models and benchmarks at every tested nucleus threshold. At $\theta = 0.95$, the GSM8K speedup increases from 2.92× to 4.87× for Nemotron-8B and from 2.91× to 5.16× for Nemotron-14B. The corresponding mean accepted lengths increase from 6.61 to 12.21 and from 6.65 to 12.64. On MBPP, the same block-length change increases speedup from 2.72× to 3.83× for Nemotron-8B and from 2.59× to 3.92× for Nemotron-14B. These results show that longer draft blocks allow NSD to retain more tokens per verification. The larger speedup gains on GSM8K than on MBPP further suggest that the benefit of increasing K varies across tasks.

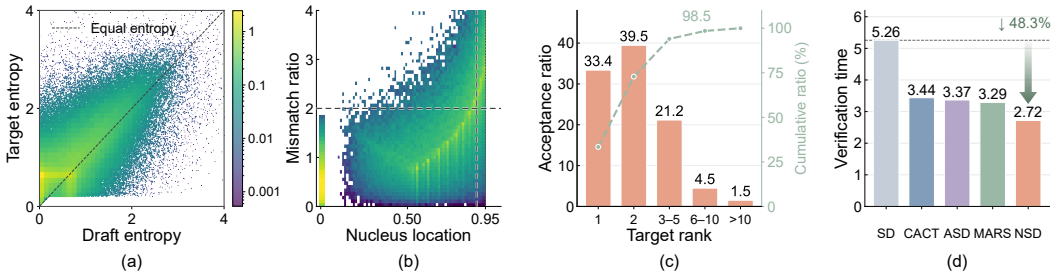


Figure 3: Nucleus-based acceptance statistics and verification efficiency. (a) Draft and target entropy. (b) Nucleus location and log probability mismatch; location zero denotes the target rank-1 token, with its strip widened for visibility. (c) Nucleus-accepted token shares and their cumulative ratio by target rank. (d) Verification time in ms/output token for Nemotron models.

Diminishing returns from longer blocks. The marginal benefit of increasing K depends on the task. For GSM8K, increasing K from 24 to 32 improves throughput for both models at all thresholds. However, the gains often diminish on MBPP. For Nemotron-8B at $\theta = 0.95$, this increase changes speedup only from $3.79\times$ to $3.83\times$ and τ from 9.79 to 9.87. For Nemotron-14B at $\theta = 0.925$, speedup instead decreases from $3.81\times$ to $3.59\times$, accompanied by a decrease in τ from 9.53 to 8.87. Thus, extending the proposed block does not always increase τ or improve throughput.

Sensitivity to the nucleus threshold. The effect of θ is model- and task-dependent rather than uniformly monotonic. For Nemotron-14B at $K = 32$, reducing θ from 0.95 to 0.85 increases GSM8K accuracy from 92.87% to 93.86% and MBPP pass@1 from 59.92% to 62.26%. These improvements are accompanied by lower speedups: $5.16\times$ to $4.86\times$ on GSM8K and $3.92\times$ to $3.58\times$ on MBPP. In contrast, for Nemotron-8B on GSM8K at $K = 32$, reducing θ from 0.95 to 0.90 improves both speedup and accuracy, from $4.87\times$ to $5.36\times$ and from 92.57% to 93.63%, while increasing τ from 12.21 to 13.57. The same threshold change on MBPP increases speedup from $3.83\times$ to $3.95\times$ but decreases pass@1 from 62.26% to 59.92%. A threshold adjustment that improves both metrics on one task can therefore introduce a quality-efficiency trade-off on another.

Joint configuration trade-offs. The configurations maximizing throughput do not necessarily maximize task performance. Although the highest observed speedup occurs at $K = 32$, the highest MBPP pass@1 for both models occurs at $K = 16$ and $\theta = 0.85$. This setting achieves 62.65% pass@1 with a $3.26\times$ speedup for Nemotron-8B and 64.20% pass@1 with a $3.25\times$ speedup for Nemotron-14B. Taken together, the ablations show that block length and nucleus threshold should be considered jointly, as larger blocks generally favor throughput by increasing the number of draft tokens that can be retained per verification step, whereas the preferred nucleus threshold depends on the model and task and can yield a more favorable balance between efficiency and performance.

4.4 Analysis

We investigate the statistics of additional acceptances under NSD and how this relaxation affects verification efficiency. An additional acceptance is a verified draft proposal that fails the standard stochastic acceptance but is retained by NSD because it belongs to the target nucleus. Figure 3(a–c) characterizes additional acceptances from Qwen3-4B with DSpark. Figure 3(d) compares verification timing on Nemotron-14B with self-speculation.

Draft distributions are often more concentrated. Figure 3(a) compares draft and target entropy at contexts where additional acceptances occur. The mean draft entropy is 0.865, compared with 1.101 for the target, and target entropy exceeds draft entropy in 70.30% of these tokens. The two entropies are positively correlated, with Pearson correlation $r = 0.668$. These observations indicate that although draft and target entropy tend to vary together, the draft distribution is often more concentrated. This pattern complements the target-rank analysis by distinguishing distributional concentration. For an additionally accepted token x, failure of the standard SD verification requires

$q(x) > p(x)$, with rejection probability $1 - p(x)/q(x)$. Thus, a proposal can be highly ranked by the target yet still face a substantial rejection probability when the draft assigns it more probability.

Most additional acceptances lie inside the nucleus core. Figure 3(b) further characterizes additional acceptances along two dimensions: nucleus location and target-draft probability mismatch. Let $C(x)$ denote the cumulative target probability mass preceding token x, and let $Y(x) = \log(q(x)/p(x))$. Within the additionally accepted tokens, we distinguish the nucleus core, $C(x) < 0.9$, and the boundary region $C(x) \geq 0.9$. Splitting the mismatch dimension at $Y = 2$ yields four categories: core-lower mismatch, core-higher mismatch, boundary-lower mismatch, and boundary-higher mismatch. Overall, 86.65% of additional acceptances lie in the nucleus core, showing that the relaxation does not operate primarily on tokens near the nucleus boundary. The available statistics also show that 79.30% satisfy both $C(x) < 0.9$ and the mismatch condition $Y(x) < 2$. The remaining tokens are distributed among the core-higher mismatch (7.36%), boundary-lower mismatch (4.84%), and boundary-higher mismatch (8.50%) regions. This concentration provides further evidence that nucleus-based acceptance predominantly retains plausible tokens under the target distribution. And the draft-target mismatch is not significant.

Additional acceptances concentrate at high target ranks. Figure 3(c) shows that NSD frequently accepts proposals ranked highly by the target model. Among the additionally accepted tokens, 33.4% are the target's rank-1 token, and 72.9% fall within its top two. The cumulative share reaches 94.1% for the top five and 98.5% for the top ten. The results illustrate that high target rank does not ensure acceptance under SD: its stochastic verification depends on $\min\{1, p(x)/q(x)\}$ rather than target rank. Thus, nucleus-based acceptance retains many highly ranked tokens that would otherwise be rejected to preserve the target distribution.

Substantial savings in verification time. In the verification timing comparison on Nemotron models, NSD reduces verification time from 5.26 to 2.72 milliseconds per output token, a 48.3% reduction relative to SD. Compared with Cactus, ASD, and MARS, whose corresponding costs are 3.44, 3.37, and 3.29 milliseconds, NSD reduces verification time per output token by 20.9%, 19.3%, and 17.3% respectively. Verification calls per output token decrease from 0.1600 under SD to 0.0835 under NSD, a 47.81% reduction, while target execution time per call changes by less than 1% (32.551 versus 32.266 milliseconds). These measurements support fewer verification calls per emitted token as the primary source of savings. NSD can continue accepting subsequent proposals using already computed target probabilities, spreading expensive verification cost over more tokens.

Taken together, these results clarify how nucleus-based verification translates additional acceptance into end-to-end efficiency. The additional proposals retained by NSD are not concentrated near the boundary of the target nucleus: 86.65% lie in the nucleus core, and 94.1% rank within the target model's top five tokens. This reduces the frequency of expensive target verification. Verification calls per output token decrease by 47.81%. Therefore, the primary efficiency gain of NSD comes from amortizing each target verification over more emitted tokens rather than making an individual target-model invocation cheaper. Competitive accuracy results suggest that NSD improves decoding efficiency by relaxing verification mainly for highly plausible proposals under the target model.

5 Conclusion

In this work, we introduce Nucleus Speculative Decoding (NSD), a simple but effective relaxed verification method that supplements stochastic acceptance with target-nucleus membership. It does not modify the proposal mechanism, residual correction, or bonus sampling procedure, allowing it to be combined with diverse speculative decoding paradigms. We provide a theoretical analysis of the distributional deviation. The deviation from the target distribution is exactly the draft model's excess probability inside the target nucleus for a single decoding step. This derivation reveals when NSD remains exact and provides conditional sequence-level fidelity bounds relative to the target distribution. Experiments across Qwen and Nvidia-Nemotron model families and proposal mechanisms demonstrate that our method improves throughput by retaining additional draft tokens while maintaining competitive task performance. Results and analysis suggest that plausibility-aware verification is a promising direction for designing efficient speculative decoding algorithms.

AI USE STATEMENT

During the preparation of this manuscript, we used AI-assisted tools to support writing-related tasks, including improving clarity, grammar, and organization of the text. The authors remain fully responsible for the research contributions, mathematical derivations, experimental design, results, interpretations, and the final content of the manuscript.

REPRODUCIBILITY STATEMENT

We have made every effort to ensure that the results presented in this paper are reproducible. Code and datasets have been made publicly available in the supplementary materials to facilitate replication and verification. The experimental setup, including models, decoding parameters and hardware details, is described in detail in the paper. All experiments are conducted on NVIDIA Pro5000 GPUs, with latency and throughput measurements collected under the same hardware configuration. We believe these measures will enable other researchers to reproduce our work.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Gregor Bachmann, Sotiris Anagnostidis, Albert Pumarola, Markos Georgopoulos, Artsiom Sanakoyeu, Yuming Du, Edgar Schönfeld, Ali Thabet, and Jonas Kohler. Judge decoding: Faster speculative sampling requires going beyond model alignment, 2025. URL https://arxiv.org/abs/2501.19309.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads, 2024. URL https://arxiv.org/abs/2401.10774.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling, 2023. URL https://arxiv.org/abs/2302.01318.

Jian Chen, Yesheng Liang, and Zhijian Liu. Dflash: Block diffusion for flash speculative decoding, 2026. URL https://arxiv.org/abs/2602.06036.

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. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Xin Cheng, Xingkai Yu, Chenze Shao, Jiashi Li, Yunfan Xiong, Yi Qian, Jiaqi Zhu, Shirong Ma, Xiaokang Zhang, Jiasheng Ye, Qinyu Chen, Chengqi Deng, Jiping Yu, Damai Dai, Zhengyan Zhang, Yixuan Wei, Yixuan Tan, Wenkai Yang, Runxin Xu, Yu Wu, Zhean Xu, Xuanyu Wang, Muyang Chen, Rui Tian, Xiao Bi, Zhewen Hao, Shaoyuan Chen, Huanqi Cao, Wentao Zhang, Anyi Xu, Huishuai Zhang, Dongyan Zhao, and Wenfeng Liang. Dspark: Confidence-scheduled speculative decoding with semi-autoregressive generation, 2026. URL https://arxiv.org/abs/2607.05147.

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. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

Yuannuo Feng, Zegang Peng, Yuxin Xie, Yubing Ye, Yizhe Chen, Wenshuai Yao, Wenyong Zhou, and Wang Kang. Approximate speculative decoding, 2026. URL https://arxiv.org/abs/2608.03447.

Yonggan Fu, Lexington Whalen, Abhinav Garg, Chengyue Wu, Maksim Khadkevich, Nicolai Oswald, Enze Xie, Daniel Egert, Sharath Turuvekere Sreenivas, Shizhe Diao, Chenhan Yu, Ye Yu, Weijia Chen, Sajad Norouzi, Jingyu Liu, Shiyi Lan, Ligeng Zhu, Jin Wang, Jindong Jiang, Morteza Mardani, Mehran Maghoumi, Song Han, Ante Jukić, Nima Tajbakhsh, Jan Kautz, and Pavlo Molchanov. Nemotron-labs-diffusion: A tri-mode language model unifying autoregressive, diffusion, and self-speculation decoding, 2026. URL https://arxiv.org/abs/2607.05722.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction, 2024. URL https://arxiv.org/abs/2404.19737.

Yongchang Hao and Lili Mou. Cactus: Accelerating auto-regressive decoding with constrained acceptance speculative sampling, 2026. URL https://arxiv.org/abs/2604.04987.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874.

Maximilian Holsman, Yukun Huang, and Bhuwan Dhingra. Fuzzy speculative decoding for a tunable accuracy-runtime tradeoff, 2025. URL https://arxiv.org/abs/2502.20704.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration, 2020. URL https://arxiv.org/abs/1904.09751.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding, 2023. URL https://arxiv.org/abs/2211.17192.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle-2: Faster inference of language models with dynamic draft trees, 2024. URL https://arxiv.org/abs/2406.16858.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle: Speculative sampling requires rethinking feature uncertainty, 2025a. URL https://arxiv.org/abs/2401.15077.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle-3: Scaling up inference acceleration of large language models via training-time test, 2025b. URL https://arxiv.org/abs/2503.01840.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. Specinfer: Accelerating generative large language model serving with tree-based speculative inference and verification, 2024. URL https://arxiv.org/abs/2305.09781.

Qwen Team. Qwen3.5-omni technical report, 2026. URL https://arxiv.org/abs/2604.15804.

Jingwei Song, Xinyu Wang, Hanbin Wang, Xiaoxuan Lei, Bill Shi, Shixin Han, Eric Yang, Xiao-Wen Chang, and Lynn Ai. Mars: Unleashing the power of speculative decoding via margin-aware verification, 2026. URL https://arxiv.org/abs/2601.15498.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. Blockwise parallel decoding for deep autoregressive models, 2018. URL https://arxiv.org/abs/1811.03115.

Shengyin Sun, Yiming Li, Renxi Liu, Weizhe Lin, Hui-Ling Zhen, Xianzhi Yu, Mingxuan Yuan, and Chen Ma. Revisiting judge decoding from first principles via training-free distributional divergence, 2026. URL https://arxiv.org/abs/2601.04766.

Oszkár Urbán, Young D. Kwon, Stylianos I. Venieris, and Cecilia Mascolo. Margins, not windows: Training-free per-step lossy speculative decoding, 2026. URL https://arxiv.org/abs/2609.02897.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. Draft & verify: Lossless large language model acceleration via self-speculative decoding, 2024. URL https://arxiv.org/abs/2309.08168.

A Theoretical Analysis and Proofs

This appendix establishes the one-step distribution induced by nucleus-based acceptance and then gives a conditional bound for sequences of emitted tokens. The local identity is exact under the sampling assumptions below.

A.1 Assumptions and Notation

Fix a finite vocabulary $\mathcal{V}$ and an emitted prefix. Let p be the target sampling distribution and q the actual proposal distribution after any temperature or sampling transformations. The proposal is sampled as $X \sim q$; using a different distribution in the acceptance ratio does not satisfy this assumption. Let S be a fixed set at this prefix, such as the target nucleus in Equation (1), with ties resolved by a fixed ordering. The proof holds for any such set, including the empty and full vocabulary sets.

For $q(x) > 0$, NSD accepts x with probability

$$a_S(x) = \begin{cases} 1, & x \in S, \\ \min\{1, p(x)/q(x)\}, & x \notin S. \end{cases} \tag{6}$$

Define

$$\begin{aligned} d_+(x) = [q(x) - p(x)]_+, \quad d_-(x) = [p(x) - q(x)]_+, \\ R = D_{\mathrm{TV}}(q, p), \qquad E_S = \sum_{x \in S} d_+(x). \end{aligned} \tag{7}$$

Both d_+ and d_- sum to R, and their supports are disjoint. When $R > 0$, every remaining rejection is followed by a draw from the unchanged SD residual $\rho(x) = d_-(x)/R$, independent of the rejected token's identity. The analysis concerns the resulting emitted token, including this recovery draw. It does not condition on acceptance alone.

A.2 Single-Step Output Distribution and Exact Error

Theorem A.1 (Exact single-step NSD error and acceptance gain)**.** *Under the assumptions above, for $R > 0$ the emitted law is*

$$\widetilde{p}(x) = \min\{p(x), q(x)\} + \mathbf{1}_S(x) d_+(x) + (R - E_S)\rho(x). \tag{8}$$

It is normalized and satisfies

$$D_{\mathrm{TV}}(\widetilde{p}, p) = E_S = \bar{a}_{\mathrm{NSD}} - \bar{a}_{\mathrm{SD}} \le R, \qquad \bar{a}_{\mathrm{NSD}} = 1 - R + E_S. \tag{9}$$

Here $\bar{a}$ is the probability of accepting the proposal. If $R = 0$, then $\widetilde{p} = p = q$ and the same identity holds with $E_S = 0$.

Proof. The mass emitted by accepting a proposal is

$$q(x) a_S(x) = \min\{p(x), q(x)\} + \mathbf{1}_S(x) d_+(x).$$

This mass identity also holds for $q(x) = 0$, without evaluating $p(x)/q(x)$. Summing over the vocabulary gives $1 - R + E_S$, leaving rejection probability $R - E_S$. Multiplying this probability by ρ yields Equation (8). All terms are nonnegative because $0 \le E_S \le R$, and their sum is $(1 - R) + E_S + (R - E_S) = 1$.

Using $p(x) = \min\{p(x), q(x)\} + d_-(x)$, the signed deviation is

$$\widetilde{p}(x) - p(x) = \mathbf{1}_S(x) d_+(x) - \frac{E_S}{R} d_-(x).$$

The positive and negative terms have disjoint support, so

$$D_{\mathrm{TV}}(\widetilde{p}, p) = \frac{1}{2} \left(E_S + \frac{E_S}{R} \sum_x d_-(x) \right) = E_S.$$

Standard SD accepts with probability $1 - R$, giving the claimed acceptance gain. If $R = 0$, then $p = q$ and all proposals are accepted, so a residual distribution is unnecessary. This also covers partial-support distributions. □

The result proves the local identity in Equation (4): each unit of added acceptance mass introduces an equal amount of one-step TV deviation. This concerns proposal acceptance, not the number of tokens committed per verification or the end-to-end throughput gain.

A.3 Exactness and Threshold Monotonicity

Corollary A.2 (Exact single-step sampling)**.** *Under the same assumptions, $\widetilde{p} = p$ if and only if $E_S = 0$, equivalently $q(x) \le p(x)$ for every $x \in S$.*

Proof. Theorem A.1 gives $D_{\mathrm{TV}}(\widetilde{p}, p) = E_S$. The finite sum defining E_S vanishes exactly when each nonnegative summand vanishes. □

Equality $p = q$ is sufficient but not necessary. If $S = \varnothing$, the procedure reduces to exact SD. If $S = \mathcal{V}$, then $E_S = R$ and every proposal is accepted, giving $\widetilde{p} = q$. For nested sets $S \subseteq S'$, $E_S \le E_{S'}$; both mean proposal acceptance and one-step TV error are therefore nondecreasing as the nucleus expands. These statements hold at fixed p, q and prefix. They do not imply monotonic accuracy or throughput across independently generated trajectories.

A.4 Sequence-Level Coupling Bound

Let $K_t(\cdot \mid h)$ be the decoder's true distribution for its next *emitted* token given history h, with internal state marginalized conditional on that history. Let $p_t(\cdot \mid h)$ be the reference target kernel. For each position, assume a uniform bound

$$\sup_h D_{\mathrm{TV}}\big(K_t(\cdot \mid h), p_t(\cdot \mid h)\big) \le \epsilon_t, \qquad 0 \le \epsilon_t \le 1. \tag{10}$$

The supremum covers histories relevant to either process; conditional kernels must be specified there. Write $P_K^{(T)}$ and $P_p^{(T)}$ for the joint laws of the first T emitted tokens. Here T is an output horizon, not the draft block size K. EOS can be made absorbing so that variable-length responses are represented within a common finite horizon.

Theorem A.3 (Conditional sequence fidelity)**.** *Under Equation* (10)*,*

$$D_{\mathrm{TV}}\big(P_K^{(T)}, P_p^{(T)}\big) \le 1 - \prod_{t=1}^{T}(1-\epsilon_t) \le \sum_{t=1}^{T}\epsilon_t. \tag{11}$$

There exists a coupling of the two sequences with complete agreement probability at least $\prod_{t=1}^{T}(1-\epsilon_t)$.

Proof. Construct the sequences sequentially. Whenever their histories agree at h, maximally couple $K_t(\cdot \mid h)$ and $p_t(\cdot \mid h)$. Their next tokens then agree with probability at least $1 - \epsilon_t$. After histories diverge, use any coupling with the required conditional marginals. If M_t is the event of agreement through position t, then

$$\Pr(M_t) \ge (1-\epsilon_t)\Pr(M_{t-1}), \qquad \Pr(M_0) = 1.$$

Induction gives $\Pr(M_T) \ge \prod_t(1-\epsilon_t)$, including cases where a factor is zero. The coupling inequality bounds TV by the probability of disagreement. Finally, $1 - \prod_t(1-\epsilon_t) \le \sum_t \epsilon_t$. □

Applying the bound to speculative blocks. Theorem A.1 identifies the conditional error as $E_{S_t(h)}$ only when K_t is the complete one-proposal acceptance-and-recovery kernel specified in Section A.1. In a block decoder, conditioning on earlier acceptances can change the law of pending proposals. The relevant proposal law and verifier ratio must still match. Recovery and bonus positions, cached proposals, and rollback state must also be included in the next-output kernel; a recovery-only kernel is not generally equal to p_t.

A sufficient state-aware condition is that, for every reachable internal state z, the next-output kernel $K_t(\cdot \mid h, z)$ has TV error at most ϵ_t relative to the same $p_t(\cdot \mid h)$. Convexity then gives

$$D_{\mathrm{TV}}\left(\sum_z \mu_t(z \mid h) K_t(\cdot \mid h, z), p_t(\cdot \mid h)\right) \le \sum_z \mu_t(z \mid h) D_{\mathrm{TV}}\big(K_t(\cdot \mid h, z), p_t(\cdot \mid h)\big) \le \epsilon_t,$$

where μ_t is the conditional state distribution. Exact target bonus sampling has zero local error when it uses that same target conditional. The sequence theorem is thus conditional on an output-kernel bound; it does not establish such a bound for every implementation from token diagnostics alone.

A.5 Uniform-Error Corollaries and Interpretation

If $\epsilon_t \le \epsilon$ for all t, Theorem A.3 gives

$$D_{\mathrm{TV}}\big(P_K^{(T)}, P_p^{(T)}\big) \le 1 - (1-\epsilon)^T \le \min\{1, T\epsilon\}. \tag{12}$$

For $0 < \delta < 1$ and integer $T \ge 1$, a sufficient condition for coupling agreement at least $1 - \delta$ is

$$\epsilon \le 1 - (1-\delta)^{1/T}; \qquad \text{the simpler condition } \epsilon \le \delta/T \text{ also suffices.}$$

For $0 < \epsilon < 1$, the product bound certifies this agreement whenever $T \le \log(1-\delta)/\log(1-\epsilon)$. This is a sufficient bound on the output horizon, not a validated prescription for choosing a speculative block size.

Corollary A.4 (Exact sequence sampling)**.** *If $K_t(\cdot \mid h) = p_t(\cdot \mid h)$ at every relevant history and position, then $P_K^{(T)} = P_p^{(T)}$ for every finite horizon T. In particular, this holds for a sequence of complete one-step NSD kernels satisfying the stated proposal and recovery assumptions with $E_{S_t(h)} = 0$ everywhere.*

Proof. Set every $\epsilon_t = 0$ in Theorem A.3. □

The agreement statement concerns an explicitly constructed coupling, not independently sampled sequences or runs sharing an arbitrary random seed. It is not a statistical confidence level or a guarantee of unchanged task accuracy. For a fixed correctness event A, a valid sequence-TV bound does imply $|P_K^{(T)}(A) - P_p^{(T)}(A)| \le D_{\mathrm{TV}}(P_K^{(T)}, P_p^{(T)})$; however, using that inequality requires a justified sequence-level bound. Diagnostic averages over observed nucleus-accepted tokens do not supply the uniform bounds in Equation (10).

B METHOD CONVENTIONS AND IMPLEMENTATION CORRESPONDENCE

Cumulative-mass boundaries. The shared gate sums transformed target mass over tokens with strictly greater target logits. Under an unrestricted positive-temperature softmax, this ordering agrees with Equation 1. All equally scored tokens share membership, so the result need not be the smallest sorted set reaching the threshold. Ordinary top-p sampling is a separate transformation that truncates and renormalizes a distribution. NSD's nucleus threshold does not truncate replacement or bonus sampling. The MTP adapter accepts thresholds only in $(0, 1]$. The zero-threshold examples describe the mathematical/shared membership convention.

DSpark. The proposal routine returns sampled tokens and associated sampling logits, including the optional Markov-head path. The backbone uses a noncausal draft pass, while the target evaluates the proposed prefix causally. An optional confidence gate can select a prefix after sampling. When interpreting q_i as an actual proposal law, such selection must be disabled or its conditioning accounted for. The historical confidence and checkpoint settings are not recoverable from the current source alone.

Nemotron linear self-speculation. The sampled path uses a bidirectional draft pass with LoRA enabled and a causal target pass with adapters disabled. It samples proposal positions from the draft probabilities and treats the target seed separately. The cache is cropped to the seed plus accepted drafts. Output wrappers trim EOS and length-budget overshoot. The reported SD configuration can invoke an external official generation routine, so equivalence to the local sampled path with nucleus-based acceptance disabled requires that external source and historical configuration.

Qwen3.5-MTP. The adapter invokes the vLLM rejection kernel one proposal at a time and retains its replacement output when the nucleus-based condition is not satisfied. Its gate takes a softmax of the target-logit row after the adapter's processors. The standard sampler may apply further transformations. The reported temperature-one, unrestricted policy is the correspondence considered in the main text. The adapter supports one active sequence and delegates proposal generation, bonus sampling, stopping, and caches to vLLM. The historical external kernel is unavailable locally. Adapter inspection does not certify those branches.

Numerical and state conventions. The shared ratio uses $\max(q_i(x_i), 10^{-8})$ in the denominator, and residual mass at most 10^{-8} triggers sampling from p_i. The exact acceptance–deviation identity assumes neither modification and requires residual recovery independent of the rejected token's identity. Symbolic trace checks verify the example's first rejection, full acceptance, EOS, and budget behavior. They do not execute backend caches or GPU kernels.

C MECHANISM COMPARISON WITH CLOSELY RELATED WORK

The comparison below separates the rule for retaining a proposal from the rule for generating the next token after a failure. This distinction matters because an acceptance criterion alone does not determine an output distribution. Entries describe the cited formulations, not interchangeable implementations or a controlled performance comparison.

Method	Selection or intervention	Recovery and scope
SD	Stochastic ratio acceptance $\min(1, p/q)$.	Sample from normalized $[p - q]_+$ at rejection; preserves the chosen target law under the usual assumptions.
NSD	Retain ratio acceptance and accept in-nucleus failures.	Keep the same residual; local added acceptance and TV error equal E_S. No request-level budget.
ASD (Feng et al., 2026)	Greedy exceptions constrained by logit regret, a block cap, and a persistent request budget.	Target argmax recovery or bonus under the committed prefix. Zero budget restores greedy verification; regret accounting is not an NSD TV bound.
MARS (Song et al., 2026)	Accept the target top-1 token, or its top-2 token when the raw-logit ratio $z_{(2)}/z_{(1)} > \theta$.	On failure, emit the target top-1 token and discard the remaining draft. The stated rule relaxes token matching; it does not use the stochastic $[p - q]_+$ residual.
Cactus (Hao & Mou, 2026)	Select an adjusted distribution under a divergence constraint.	Uses generalized acceptance/recovery equations. NSD instead fixes the original residual and analyzes the induced deviation.

Table 3: Mechanism distinctions. Here p and q always denote target and actual proposal laws; notation used in other papers is not imported into these columns.

D ADDITIONAL PROPOSAL MECHANISMS

We compare SD with NSD at an acceptance threshold of 0.95 using DFlash-b16 and EAGLE-3-TTT7 with Qwen3-4B as the target. Both comparisons use temperature 1.0, target sampling top-$p = 1.0$, and five seeds.

Table 4: Task performance and mean accepted length for additional proposal mechanisms. Task performance is mathematical accuracy or functional pass@1 (%), reported as the mean over five seeds. τ includes the resample or bonus token.

		Acc. (%)		τ	
Proposal	Benchmark	SD	NSD	SD	NSD
DFlash-b16	MATH500	83.88	84.28	5.75	6.32
	AIME25	17.33	18.67	4.61	5.13
	HumanEval	84.51	83.17	5.48	6.20
	MBPP	62.41	63.50	4.77	5.29
EAGLE-3-TTT7	MATH500	83.64	83.32	4.54	5.32
	AIME25	20.00	20.00	3.83	4.78
	HumanEval	84.39	83.90	4.16	4.72
	MBPP	62.65	61.63	3.70	4.16

Table 5: Output throughput (tokens/s) under additional proposal mechanisms. DFlash reports per-benchmark results for seed 30. Both methods ran unusually slowly on this seed; these are observed throughputs, not measurements under verified uncontended conditions. EAGLE-3 reports per-benchmark throughput for uncontended seeds 3, 7, and 22.

Proposal	Benchmark	SD	NSD
DFlash-b16	MATH500	115.92	140.77
	AIME25	87.48	115.15
	HumanEval	118.54	141.18
	MBPP	102.36	121.65
EAGLE-3-TTT7	MATH500	157.25	188.61
	AIME25	115.63	155.62
	HumanEval	153.56	184.06
	MBPP	139.97	164.93

D.1 MTP WITH TWO PROPOSAL TOKENS

We additionally report the Qwen3.5-4B MTP comparison (Qwen Team, 2026) with two proposal tokens per block using the vLLM backend. Temperature is 1.0, target sampling top-p is 1.0, and the NSD acceptance threshold is 0.95.

Table 6: Qwen3.5-4B MTP results with block size $K = 2$. Acc. is mean accuracy or pass@1 (%); τ is weighted by verification-block counts. The maximum committed length is three, including the continuation token.

	Acc. (%)		τ	
Benchmark	SD	NSD	SD	NSD
MATH500	92.73	92.33	2.63	2.95
AIME25	35.56	34.44	2.57	2.94
HumanEval	78.25	80.89	2.55	2.95
MBPP	55.12	57.85	2.42	2.92

The throughput for NSD is 179.90 tokens/s, which is $1.70\times$ the SD throughput of 105.90 tokens/s.

E LIMITATIONS

Although our experiments demonstrate the effectiveness of Nucleus Speculative Decoding (NSD) across multiple settings, one limitation of the current study is the range of model sizes evaluated. Because of hardware constraints and the availability of Nemotron and DSpark checkpoints, our evaluation focuses primarily on relatively compact target models and corresponding draft models.