

InfoSpec: Information-Guided Anchor Selection for Retrieval-Based Speculative Decoding

Jie Wang, Yi Ren*, Wei Li, Zhenyang Gao, Jiawei Liao

College of Computer Science and Technology

National University of Defense Technology

{wangj, renyi, liwei1, gaozhenyang, liaojiawei_24}@nudt.edu.cn

Abstract

Retrieval-based speculative decoding accelerates large language model inference without training an additional draft model, but its effectiveness is often limited by retrieval ambiguity. Existing future-guided methods typically rely on a fixed future-token position, such as a proxy for the next-next token, as the retrieval anchor. However, this fixed-anchor strategy may be suboptimal across different contexts. We propose InfoSpec, a training-free framework that formulates retrieval-anchor selection as an optimization problem performed at inference time. At each decoding step, InfoSpec constructs a lightweight candidate anchor pool from the target model, estimates the retrieval utility of each candidate via hash-index probing, and selects anchors based on model confidence, ambiguity reduction, locality, and continuation consistency under a fixed draft-tree budget. Experiments on generation, reasoning, code, and long-context benchmarks show that InfoSpec improves mean accepted tokens and end-to-end speedup over strong retrieval-based baselines. Further analyses show that InfoSpec reduces high-ambiguity matches and uses the verification budget more effectively while preserving plug-and-play deployment.

1 Introduction

Large language models (LLMs) have achieved strong performance across dialogue, code generation, mathematical reasoning, and long-context understanding. However, their inference remains fundamentally limited by autoregressive decoding, where each token must be generated after all previous tokens (Li et al., 2025a; Chen et al., 2025). This sequential dependency creates substantial latency, especially as model size and output length increase. Speculative decoding addresses this bottleneck by drafting multiple candidate tokens and verifying

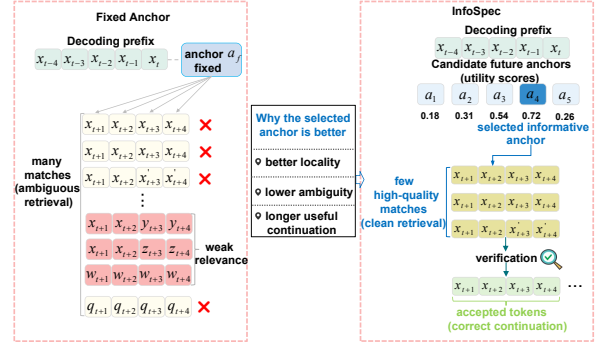


Figure 1: Fixed-anchor retrieval vs. InfoSpec. A fixed future anchor may retrieve many ambiguous or weakly relevant continuations. InfoSpec selects a more informative anchor from multiple candidates, producing a cleaner match set and enabling verification to accept a longer correct continuation.

them in parallel with the target model, while preserving the target model distribution (Leviathan et al., 2023; Chen et al., 2023; Xia et al., 2024b).

A major design choice in speculative decoding is how to obtain draft tokens. Many methods use an auxiliary draft model or additional decoding heads to propose candidates (Cai et al., 2024; Ankner et al., 2024; Xiao et al., 2024; Li et al., 2024). Although effective, these approaches often require extra parameters, task-specific training, or additional serving complexity. Retrieval-based speculative decoding offers a different trade-off: it constructs drafts by retrieving continuations from the prompt, previously generated context, or an external datastore (He et al., 2024; Fu et al., 2024; Hu et al., 2025; Luo et al., 2025). This makes it attractive for deployment because it is training-free, lightweight, and compatible with existing inference stacks.

Despite these advantages, retrieval-based speculative decoding is limited by a concrete and measurable bottleneck: retrieval ambiguity. At each decoding step, the method must form a query that retrieves useful continuations. If the query is too broad, it matches many unrelated or weakly related

*Corresponding author

continuations; if it is too narrow, it may return no match. In both cases, the resulting draft tokens are less likely to be accepted by the target model, reducing mean accepted tokens and end-to-end speedup. Existing retrieval-based methods mainly improve where to retrieve from or how to match efficiently, but they often leave the retrieval cue itself fixed once the current prefix is given (He et al., 2024; Fu et al., 2024; Hu et al., 2025; Luo et al., 2025).

Recent future-guided retrieval methods show that the retrieval cue can be improved by using information from the target model’s next-token distribution. For example, LogitSpec uses the target model’s last-step logits to infer a next-next-token signal and uses this signal as an auxiliary retrieval anchor (Liu et al., 2026). This improves retrieval specificity and demonstrates that weak future information can make retrieval-based drafting more effective. However, existing future-guided methods rely on a fixed-anchor assumption: they use a pre-determined future position, typically the next-next token, as the retrieval anchor for all contexts.

This assumption is restrictive. The most probable future token is not necessarily the most useful retrieval anchor. A high-probability token may be common and retrieve many ambiguous continuations, while a slightly lower-probability token may sharply reduce the match set and expose a more coherent continuation. Moreover, the best anchor can vary across tasks and decoding states. In some contexts, the immediate next token already disambiguates retrieval; in others, a farther future token provides a stronger cue. Therefore, the key gap is not whether future-token guidance helps, but which future token should be used as the retrieval anchor under a fixed verification budget. This gap is directly testable by comparing fixed-anchor retrieval with adaptive anchor selection in terms of match-set ambiguity, retrieval success rate, mean accepted tokens, and wall-clock speedup.

In this paper, we propose InfoSpec, an information-guided framework for retrieval-based speculative decoding. Instead of committing to a fixed future position, InfoSpec treats retrieval-anchor selection as an inference-time optimization problem. At each decoding step, it builds a small candidate anchor pool from the target model’s current decoding state, including the realized next token, top-ranked future-token candidates, and lightweight two-step approximations. For each candidate anchor, InfoSpec performs a fast hash-based retrieval probe and estimates its utility using

training-free signals: model confidence, retrieval ambiguity reduction, locality of matched contexts, and continuation consistency. It then selects anchors according to their estimated utility per verification cost and expands only the most valuable branches under a fixed draft-tree budget. Figure 1 illustrates the key difference between a fixed-anchor retrieval strategy and InfoSpec.

This design is motivated by the practical constraint of speculative decoding: the objective is not to generate as many draft tokens as possible, but to generate draft tokens that are worth verifying. Increasing the draft tree indiscriminately can improve accepted length but also increases verification overhead, which may reduce end-to-end speedup. InfoSpec therefore allocates verification capacity to anchors that are both plausible under the target model and discriminative in the retrieval space. This differs from training a stronger drafter, which introduces additional model maintenance cost, and from only improving the retrieval index, which does not decide which future cue should be used for retrieval. By optimizing the retrieval anchor itself, InfoSpec preserves the plug-and-play nature of retrieval-based decoding while making better use of the available verification budget.

We evaluate InfoSpec on diverse generation, reasoning, code, and long-context benchmarks, including Spec-Bench, CNN/DM, GSM8K, HumanEval, MATH, AIME, and LongBench. Across multiple model families, InfoSpec improves mean accepted tokens and end-to-end speedup over fixed-anchor future-guided retrieval and other retrieval-based speculative decoding baselines. Further analyses show that InfoSpec shifts retrieval toward low-ambiguity non-empty match sets, improves retrieval success rate, and uses moderate draft-tree budgets more effectively.

Our contributions are as follows:

- We identify retrieval-anchor selection as a concrete inference-time optimization problem in retrieval-based speculative decoding, showing that fixed future-token guidance is a special case of a broader anchor-selection space.
- We propose InfoSpec, a training-free framework that estimates anchor utility through lightweight retrieval probing and selects anchors under a fixed verification budget.
- We demonstrate across multiple benchmarks and model families that InfoSpec improves

accepted-token efficiency, retrieval quality, and end-to-end speedup while retaining the deployment simplicity of retrieval-based decoding.

2 Related Work

Speculative decoding with auxiliary drafters. Speculative decoding accelerates autoregressive inference by first proposing multiple draft tokens and then verifying them in parallel with the target model while preserving the target distribution (Leviathan et al., 2023; Chen et al., 2023; Xia et al., 2024b). A large body of work improves this paradigm by introducing auxiliary draft generators or additional prediction heads. For example, Medusa adds lightweight decoding heads to the target model (Cai et al., 2024), while Hydra and Clover model stronger dependencies among draft heads (Ankner et al., 2024; Xiao et al., 2024). The EAGLE series further improves draft quality through autoregressive draft heads and dedicated training objectives (Li et al., 2024, 2025b). Other methods optimize the verification process or draft-tree structure, such as OPT-Tree and PEARL, which adaptively allocate draft depth, width, or length according to acceptance behavior (Wang et al., 2025; Liu et al., 2025).

Draft-model-free speculative decoding. To avoid the overhead of a separate drafter, another line of work reuses the target model itself to construct approximate drafts. Early-exiting and self-speculative methods exploit layer redundancy by using partial model computation to propose candidates before full-model verification (Zhang et al., 2024; Elhoushi et al., 2024; Xia et al., 2024a; Liu et al., 2024; Bae et al., 2023; Hooper et al., 2025). These approaches reduce the cost of drafting, but their effectiveness depends on how much computation can be skipped without substantially degrading draft quality. Retrieval-based speculative decoding offers a complementary draft-model-free direction: instead of approximating the target model internally, it retrieves candidate continuations from the prompt, generated history, or external datastore (Saxena, 2023; He et al., 2024; Fu et al., 2024; Hu et al., 2025; Luo et al., 2025).

Retrieval-based speculative decoding. Existing retrieval-based methods mainly differ in the source of retrieved continuations and the efficiency of matching. Prompt Lookup Decoding retrieves

continuations by exact n -gram matching over the prompt and generated context (Saxena, 2023). REST expands the retrieval source with an external database to improve coverage when useful continuations are absent from the local context (He et al., 2024). Lookahead Decoding reuses previously generated or discarded draft tokens as additional retrieval candidates (Fu et al., 2024). SAM Decoding improves matching efficiency with suffix automata and dynamic text sources (Hu et al., 2025), while Token Recycling reuses token transitions from earlier decoding trajectories through an adjacency-style structure (Luo et al., 2025).

Future-token guidance for retrieval. Recent work shows that retrieval quality can be improved by injecting limited information about future generation. LogitSpec observes that the target model’s last-step logits contain useful signals about the next-next token and uses this signal as a future retrieval anchor (Liu et al., 2026). This demonstrates that retrieval-based speculative decoding can benefit not only from better retrieval sources or data structures, but also from more informative retrieval cues.

Our work is most closely related to retrieval-based speculative decoding with future-token guidance. We formulate anchor selection as an inference-time optimization problem under a fixed verification budget. InfoSpec estimates the utility of candidate anchors through lightweight retrieval probes and selects anchors according to their expected value for ambiguity reduction and verification efficiency. This perspective generalizes fixed next-next-token guidance as one point in a broader anchor-selection space, and connects retrieval specificity, draft-tree budget allocation, and downstream verification within a unified training-free framework.

3 Preliminaries and Problem Setup

Let $x_{1:i} = (x_1, \dots, x_i)$ denote the current decoding prefix and M denote the target language model. In standard speculative decoding, a drafting procedure proposes a set of draft tokens, which are then verified in parallel by the target model using speculative sampling, preserving the target distribution exactly. Retrieval-based speculative decoding replaces the draft model with a retrieval module that matches the recent context against an internal or external corpus and uses matched continuations as draft candidates. Formally, given a retrieval index $\mathcal{R}$ storing n -grams, a query m -gram q re-

turns matched continuations by a matching operator $\text{MATCH}(\mathcal{R}, q)$. This paradigm is attractive because it is training-free and deployment-friendly, but it is fundamentally limited by retrieval ambiguity: short or common queries often produce either no match or many weakly relevant matches, leading to poor draft quality.

LogitSpec addresses this issue by fixing the next-next token as an auxiliary retrieval cue and shows that the last logit contains a surprisingly strong signal about the next-next token. These observations motivate a more general question: Which future token is the most informative retrieval anchor under a fixed verification budget? Existing methods instantiate one specific answer by always using the next-next token as the auxiliary anchor. In contrast, we elevate retrieval anchor selection to a first-class optimization problem.

3.1 The Retrieval-Anchor Selection Problem

At decoding step i , let $p_i = M(x_{1:i})$ be the next-token distribution. A retrieval anchor is any candidate future token z used to construct a more specific retrieval query. For example, after sampling or greedily selecting the next token x_{i+1} , one may query either

$$g_i^{(1)} = (x_{i-m+2}, \dots, x_i, x_{i+1}), \quad (1)$$

or augment it with a candidate future anchor z ,

$$g_i^{(2)}(z) = (x_{i-m+3}, \dots, x_i, x_{i+1}, z). \quad (2)$$

The key difference between prior work and ours is that prior work fixes z to be a next-next candidate from the last-logit top- k set, whereas we select z adaptively from a candidate pool. This distinction matters because the most probable future token is not always the most useful retrieval anchor: a token with slightly lower model probability may induce dramatically lower retrieval ambiguity and yield longer, more verifiable continuations.

We therefore define the candidate anchor pool

$$\mathcal{C}_i = \mathcal{C}_i^{(1)} \cup \mathcal{C}_i^{(2)} \cup \mathcal{C}_i^{(2')}, \quad (3)$$

where:

- $\mathcal{C}_i^{(1)}$: the realized next token x_{i+1} ;
- $\mathcal{C}_i^{(2)}$: top- k tokens from the last-logit distribution used as proxy next-next anchors, following the empirical observation of LogitSpec.

- $\mathcal{C}_i^{(2')}$: a lightweight two-step approximation of next-next candidates obtained without training.

We intentionally restrict the main method to this low-order candidate pool. This keeps the approach aligned with the empirical budget regime in retrieval-based SD.

4 Method

4.1 Overview of InfoSpec

To motivate the proposed design, we first present an overview of InfoSpec. Figure 2 illustrates the overall pipeline of InfoSpec. At each decoding step, InfoSpec consists of four stages: (1) candidate anchor pool construction, (2) utility estimation via fast retrieval probing, (3) budgeted anchor selection, and (4) tree verification. Starting from the current decoding prefix, InfoSpec first constructs a lightweight pool of candidate future tokens from the target model. It then probes the retrieval utility of each candidate anchor through fast matching, where the utility reflects not only model confidence but also ambiguity reduction, locality, and continuation quality. Under a fixed draft-tree budget, only the anchors with the highest utility per cost are selected for expansion, while low-value candidates are pruned. The retrieved continuations from the selected anchors are finally merged into a compact draft tree and verified in parallel by the target model.

Unlike fixed-anchor retrieval-based speculative decoding, which commits to a pre-specified future token as the retrieval cue, InfoSpec treats *retrieval-anchor selection itself as an optimization problem*. The method is fully training-free and does not introduce any additional model parameters.

4.2 InfoSpec: Information-Guided Retrieval Anchors

A useful anchor should satisfy two properties simultaneously:

- **Model plausibility**: it lies near the model’s likely continuation manifold;
- **Retrieval specificity**: it sharply reduces ambiguity in the retrieval space.

This suggests that anchor value is not determined by $\log p(z)$ alone, but by the joint interaction between model confidence and the reduction in retrieval uncertainty induced by z .

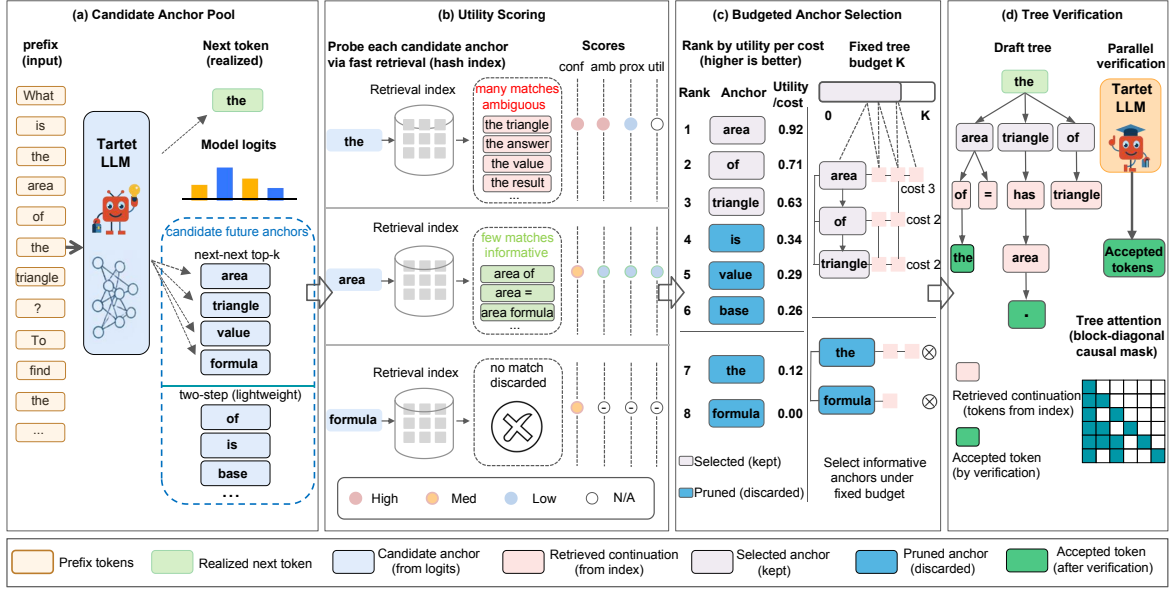


Figure 2: **Overview of InfoSpec.** InfoSpec first constructs a pool of candidate future tokens, then estimates their retrieval utility through fast probing, selects the most informative anchors under a fixed draft-tree budget, and finally verifies the resulting draft tree in parallel with the target model. The example illustrates that the best retrieval anchor is not necessarily the most probable future token: a common token such as the yields many ambiguous matches, whereas a more specific token such as area produces fewer but more useful continuations.

4.2.1 Feasible Anchor Set

For each $z \in \mathcal{C}_i$, we perform a constant-time hash probe on the retrieval index and obtain the matched continuation set

$$\mathcal{M}_i(z) = \text{MATCH}(\mathcal{R}, g_i(z)). \quad (4)$$

We define the feasible anchor set as

$$\mathcal{F}_i = \{z \in \mathcal{C}_i \mid |\mathcal{M}_i(z)| > 0\}. \quad (5)$$

This feasibility condition is important. In retrieval-based drafting, an anchor with zero matches has no direct drafting utility and should not be rewarded merely because its match set is small. This avoids the pathological case where a no-match anchor is incorrectly treated as maximally disambiguating.

4.2.2 Training-Free Anchor Utility

For each feasible anchor $z \in \mathcal{F}_i$, we first compute four raw utility components:

$$C_i(z) = \log p_i(z), \quad A_i(z) = -\log(|\mathcal{M}_i(z)| + 1),$$

$$P_i(z) = \text{PROX}_i(z), \quad L_i(z) = \text{LEN}_i(z),$$

where $C_i(z)$ measures model confidence, $A_i(z)$ measures retrieval ambiguity, $P_i(z)$ measures locality, and $L_i(z)$ measures continuation consistency.

Because these components have different numerical scales, we apply per-step min-max normalization over the feasible anchor set $\mathcal{F}_i$. For each $f_i \in \{C_i, A_i, P_i, L_i\}$, let

$$f_i^{\min} = \min_{z' \in \mathcal{F}_i} f_i(z'), \quad f_i^{\max} = \max_{z' \in \mathcal{F}_i} f_i(z').$$

The normalized component is defined as

$$\tilde{f}_i(z) = \begin{cases} \frac{f_i(z) - f_i^{\min}}{f_i^{\max} - f_i^{\min}}, & \text{if } f_i^{\max} > f_i^{\min}, \\ 0, & \text{otherwise.} \end{cases}$$

If all feasible anchors have the same value for one component, that component provides no discriminative information at the current decoding step and is therefore set to zero. The final anchor utility is

$$S_i(z) = \alpha \tilde{C}_i(z) + \beta \tilde{A}_i(z) + \gamma \tilde{P}_i(z) + \delta \tilde{L}_i(z), \quad (6)$$

where $\alpha, \beta, \gamma, \delta \geq 0$ control the contributions of model confidence, ambiguity reduction, locality, and continuation consistency, respectively. The four components capture complementary aspects of anchor quality. Model confidence keeps the selected anchor on a plausible target-model continuation. The ambiguity component favors anchors that produce small but non-empty match sets. Locality prioritizes matches that are consistent with the

recent decoding context, while continuation consistency rewards retrieved continuations that share stable prefixes. Normalization prevents any component from dominating the score solely because of its numerical scale.

4.3 Budgeted Anchor Selection and Tree Construction

The central design constraint in speculative decoding is not merely draft quality, but draft quality per unit verification cost. Prior work has shown that larger draft trees increase mean accepted tokens (MAT), yet beyond a moderate capacity they reduce wall-clock speedup because verification overhead grows faster than the acceptance gain.

We therefore formulate anchor selection under a fixed draft-tree budget K :

$$\max_{\mathcal{A}_i \subseteq \mathcal{F}_i} \sum_{z \in \mathcal{A}_i} S_i(z) \quad \text{s.t.} \quad \sum_{z \in \mathcal{A}_i} c_i(z) \leq K. \quad (7)$$

Here $c_i(z)$ is the verification cost contributed by anchor z , approximated by the number of draft nodes allocated to the branch rooted at z . Since solving Eq. 7 exactly at each decoding step is unnecessary, we use a greedy gain-per-cost policy:

$$\rho_i(z) = \frac{S_i(z)}{c_i(z) + \tau}, \quad (8)$$

where $\tau > 0$ is a small constant to prevent unstable preference for very short branches.

At each step, InfoSpec sorts feasible anchors by $\rho_i(z)$, then allocates branch width and continuation depth until the global budget K is exhausted. The resulting draft sequences are merged into a tree and verified in parallel with standard tree attention.

Practical Budget Policy In the main setting, we use:

- a root branch for the realized next token x_{i+1} ;
- several auxiliary anchor branches from top-ranked $z \in \mathcal{F}_i$;
- adaptive continuation truncation such that the total node count does not exceed K .

This design keeps the method faithful to the practical operating regime already validated by LogitSpec: retrieval can be kept cheap via hash probes, while the dominant runtime remains the target-model forward pass. Algorithm 1 summarizes the complete end-to-end pipeline of InfoSpec drafting,

including candidate construction, utility scoring, and budget allocation. The algorithm is provided in Appendix A.

4.4 Lightweight Two-Step Approximation

The candidate set $\mathcal{C}_i^{(2')}$ is designed to provide a slightly more faithful approximation of next-next candidates than raw last-logit ranking, without introducing model retraining or full lookahead expansion.

Given the selected next token x_{i+1} , obtained either by greedy decoding or sampling, we construct a lightweight approximate two-step score for each candidate z :

$$\hat{p}_i^{(2)}(z) = \eta \tilde{p}_i(z) + (1 - \eta) \tilde{\phi}(x_{i+1}, z), \quad (9)$$

where $\tilde{p}_i(z)$ and $\tilde{\phi}(x_{i+1}, z)$ are min-max normalized within the candidate set. We instantiate ϕ using retrieval-index bigram support:

$$\phi(x_{i+1}, z) = \frac{N_i(x_{i+1}, z) + \epsilon_b}{N_i(x_{i+1}) + \epsilon_b |\mathcal{Z}_i|}, \quad (10)$$

where $N_i(x_{i+1}, z)$ is the number of times z follows x_{i+1} in the retrieval buffer, $N_i(x_{i+1})$ is the total number of outgoing bigrams from x_{i+1} , ϵ_b is a smoothing constant, and $\mathcal{Z}_i$ is the restricted candidate set considered for two-step approximation. We then select the top- k' candidates under $\hat{p}_i^{(2)}$ to form $\mathcal{C}_i^{(2')}$.

This approximation is intentionally lightweight. Its role is not to replace the target model, but to enrich the anchor pool with a small number of candidates that may be retrieval-useful even when raw last-logit rank alone is imperfect.

4.5 Verification

After anchor selection, InfoSpec retrieves the continuation set associated with each selected anchor and organizes these continuations into a draft tree:

$$\mathcal{D}_i = \bigcup_{z \in \mathcal{A}_i} \mathcal{D}_i(z). \quad (11)$$

As in prior tree-based speculative verification, each branch is causally visible only to itself and to the shared prefix. If the j -th branch has length m_j , the branch-local attention mask is block-diagonal:

$$A_{\text{draft}} = \text{diag}(\Lambda(m_1), \dots, \Lambda(m_{|\mathcal{A}_i|})), \quad (12)$$

with the shared root token visible to all branches. This enables parallel verification of all selected draft sequences in a single forward pass.

Models	Method	CNN/DM		GSM8K		HumanEval		Spec-Bench		Average	
		MAT	Speedup	MAT	Speedup	MAT	Speedup	MAT	Speedup	MAT	Speedup
Vicuna 7B	Lookahead	1.53	1.26	1.92	1.63	1.77	1.56	1.66	1.26	1.72	1.43
	REST	1.63	1.16	1.55	1.12	1.95	1.50	1.81	1.46	1.74	1.31
	PLD	2.61	2.24	1.79	1.60	1.83	1.63	1.71	1.58	1.99	1.76
	SpS	2.31	1.56	2.07	1.46	2.55	1.80	2.27	1.46	2.30	1.57
	LogitSpec	3.27	2.60	2.66	2.18	2.61	2.23	2.44	2.00	2.75	2.25
	InfoSpec	3.34	2.66	2.83	2.31	2.78	2.37	2.58	2.12	2.88	2.37
Vicuna 13B	LogitSpec	2.88	2.15	2.58	2.11	2.86	2.46	2.32	1.91	2.66	2.16
	InfoSpec	2.96	2.23	2.75	2.24	3.01	2.58	2.47	2.04	2.80	2.27
Vicuna 33B	LogitSpec	2.66	2.00	2.45	1.91	2.40	1.93	2.12	1.75	2.41	1.90
	InfoSpec	2.72	2.07	2.58	2.00	2.55	2.06	2.24	1.85	2.52	2.00

Table 1: Main results on Spec-Bench, CNN/DM, GSM8K, and HumanEval.

5 Experiments

5.1 Experimental Setup

We evaluate InfoSpec on a broad suite of text generation benchmarks covering dialogue, translation, summarization, question answering, code generation, mathematical reasoning, and long-context understanding. Following prior retrieval-based speculative decoding work, our primary benchmark is Spec-Bench (Xia et al., 2024b), which contains six representative task families: multi-turn conversation (MT), translation (Trans), summarization (Sum), question answering (QA), mathematical reasoning (Math), and retrieval-augmented generation (RAG). We further report results on CNN/DM (Nalapati et al., 2016), GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021), MATH (Hendrycks et al., 2021), AIME (Mathematical Association of America, 2025), and LongBench (Bai et al., 2024) to evaluate robustness across heterogeneous generation regimes.

We compare InfoSpec against strong retrieval-based speculative decoding baselines, including Lookahead Decoding, REST, PLD, and LogitSpec, together with vanilla speculative sampling (SpS) as a draft-model-based reference. We report two primary metrics throughout the paper: mean accepted tokens per decoding step (MAT) and end-to-end speedup over vanilla autoregressive decoding. Higher MAT indicates better draft utility, while higher speedup reflects practical latency reduction after accounting for retrieval, tree construction, verification, and KV-cache update overheads.

We follow the standard inference configuration used in prior work: greedy decoding (temperature 0), batch size 1, float16 inference, and maximum generation length 1024. For the main experiments, models from 7B to 33B are evaluated on a single NVIDIA A800 80GB GPU, while 70B models use two A800 80GB GPUs. We use the same

evaluation protocol and hardware setting as prior retrieval-based speculative decoding work for fair comparison.

For InfoSpec, the retrieval database is constructed online from the user prompt and previously decoded tokens, following the same query-independent setting as prior prompt-based retrieval methods. The key difference is that, instead of using a fixed future anchor, InfoSpec scores candidate future anchors using an information-aware utility function that favors small but non-empty match sets under a fixed tree-verification budget.

5.2 Main Results

Table 1 reports the main end-to-end results on Spec-Bench, CNN/DM, GSM8K, and HumanEval across Vicuna backbones. Overall, InfoSpec achieves the best performance among all compared methods, showing that adaptive retrieval-anchor selection further improves retrieval-based speculative decoding beyond fixed-anchor future guidance. On Vicuna-7B, InfoSpec reaches the strongest overall result, achieving 2.88 MAT and $2.37\times$ speedup on average, outperforming LogitSpec (2.76 MAT / $2.26\times$) as well as earlier retrieval-based baselines such as PLD and REST. The gains are particularly clear on reasoning- and generation-intensive benchmarks, such as GSM8K and HumanEval, where retrieval quality is more sensitive to future-token choice. A similar trend holds at larger model scales. On Vicuna-13B and Vicuna-33B, InfoSpec improves over LogitSpec in both MAT and end-to-end speedup, indicating that the benefit of adaptive anchor selection remains stable as the target model becomes larger. Although the absolute speedup decreases with model size, the relative advantage of InfoSpec over fixed-anchor retrieval is preserved. As shown in Figure 3, InfoSpec achieves the highest MAT and end-to-end speedup across all evaluation tasks on Vicuna-7B, demon-

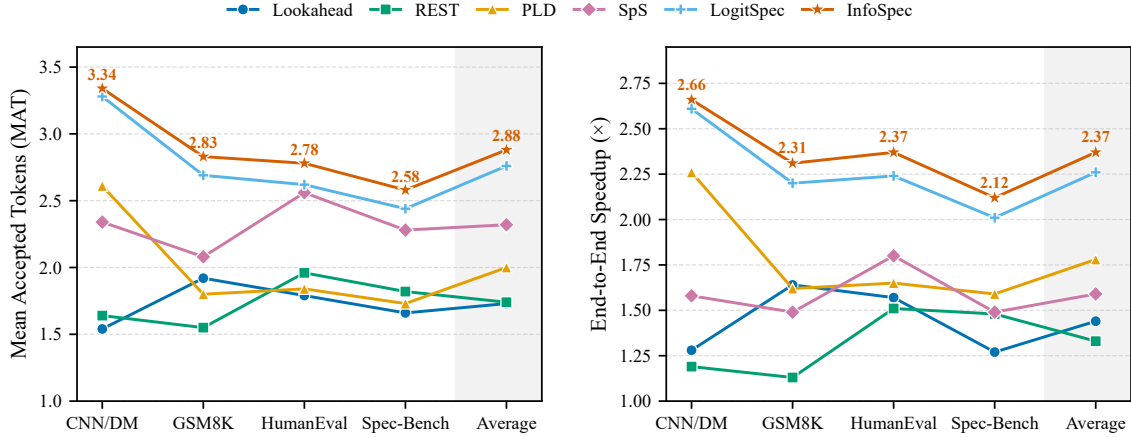


Figure 3: Vicuna-7B performance comparison across benchmarks.

Models	Method	MT	Trans	Sum	QA	Math	RAG	MAT	Speedup
Vicuna 7B	Lookahead	1.40	1.13	1.17	1.22	1.54	1.08	1.65	1.26
	REST	1.61	1.30	1.36	1.65	1.20	1.71	1.81	1.46
	PLD	1.63	1.02	2.43	1.13	1.60	1.71	1.73	1.58
	SpS	1.65	1.11	1.62	1.46	1.48	1.54	2.25	1.48
	LogitSpec	1.90	1.39	2.67	1.70	2.21	1.86	2.42	2.00
	InfoSpec	1.98	1.49	2.73	1.79	2.35	1.93	2.58	2.12
Vicuna 13B	LogitSpec	1.88	1.43	2.33	1.42	2.21	1.92	2.30	1.91
	InfoSpec	1.95	1.53	2.39	1.53	2.35	1.99	2.46	2.04
Vicuna 33B	LogitSpec	1.75	1.37	2.14	1.35	1.97	1.69	2.11	1.75
	InfoSpec	1.83	1.47	2.22	1.46	2.12	1.76	2.24	1.85

Table 2: Per-task Spec-Bench breakdown. We report task-wise speedup together with overall MAT and end-to-end speedup.

strating that information-guided adaptive anchor selection improves draft candidate quality while effectively enhancing the practical inference efficiency of retrieval-based speculative decoding.

5.3 Per-Task Breakdown on Spec-Bench

Table 2 reports the task-wise breakdown on the six Spec-Bench categories. Overall, InfoSpec improves over LogitSpec across all task groups and model scales, indicating that the benefit of adaptive retrieval-anchor selection is not limited to a specific generation setting.

The gains are relatively more visible on Translation and Math, where retrieval ambiguity is more sensitive to anchor choice. InfoSpec also yields stable gains on overlap-rich tasks such as Summarization and RAG, suggesting that adaptive anchor selection remains useful even when retrieval is abundant, since it can still reduce ambiguity and improve branch quality under a fixed verification budget. The improvements on MT and QA are smaller but consistent across all Vicuna scales. This pattern suggests that the advantage of InfoSpec is

strongest when the retrieval cue plays a larger role in disambiguating candidate continuations, while still remaining robust in settings where retrieval is already moderately effective. InfoSpec also remains effective under non-zero-temperature sampling; detailed results are provided in Appendix G.

6 Conclusion

We present InfoSpec, a training-free retrieval-based speculative decoding framework that formulates retrieval-anchor selection as an inference-time optimization problem. Instead of using a fixed future token as the retrieval cue, InfoSpec adaptively selects the anchor that is most informative under a fixed draft-tree budget. This design improves retrieval specificity and verification efficiency while preserving the plug-and-play advantages of draft-model-free decoding. More broadly, our work suggests that efficient retrieval-based speculative decoding depends not only on proposing future candidates, but on selecting the candidates that are most valuable to verify. This perspective provides a useful step toward more adaptive and deployment-friendly

LLM inference.

Limitations

InfoSpec has several limitations. First, while it improves retrieval quality and budget efficiency, its utility score is still a hand-designed surrogate rather than a learned estimator of expected acceptance gain. Although this design preserves the training-free nature of the method, it may leave additional performance untapped. Second, InfoSpec currently focuses on a low-order candidate anchor pool for efficiency reasons. Extending the method to farther future positions may provide additional gains, but requires more careful control of combinatorial growth in the draft tree. Third, the effectiveness of InfoSpec still depends on the availability of useful retrieval continuations in the prompt or retrieval source. When the retrieval space is extremely sparse or highly out-of-domain, the room for improvement becomes smaller. Finally, our current implementation emphasizes inference-time feasibility and plug-and-play deployment; deeper integration with production inference engines and hybrid speculative decoding systems remains future work.

Acknowledgments

This work was supported by the Innovation Research Group Project of the Natural Science Foundation of Hunan Province (No. 2026JJ10007). We thank the anonymous reviewers and area chairs for their thoughtful comments on an earlier version of this manuscript. We also gratefully acknowledge the fieldworkers and community members who contributed to the documentation of these languages.

References

- Zachary Ankner, Rishab Parthasarathy, Aniruddha Nrusimha, Christopher Rinard, Jonathan Ragan-Kelley, and William Brandon. 2024. Hydra: Sequentially-dependent draft heads for medusa decoding. *arXiv preprint arXiv:2402.05109*.
- Sangmin Bae, Jongwoo Ko, Hwanjun Song, and Se-Young Yun. 2023. Fast and robust early-exiting framework for autoregressive language models with synchronized parallel decoding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5910–5924.
- Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2024. Longbench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 3119–3137.
- Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D Lee, Deming Chen, and Tri Dao. 2024. Medusa: Simple llm inference acceleration framework with multiple decoding heads. In *International Conference on Machine Learning*, pages 5209–5235. PMLR.
- Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.
- Xiaoshu Chen, Sihang Zhou, Ke Liang, and Xinwang Liu. 2025. Distilling reasoning ability from large language models with adaptive thinking. *IEEE Transactions on Neural Networks and Learning Systems*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, et al. 2024. Layerskip: Enabling early exit inference and self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12622–12642.
- Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. 2024. Break the sequential dependency of llm inference using lookahead decoding. In *Forty-first International Conference on Machine Learning*.
- Zhenyu He, Zexuan Zhong, Tianle Cai, Jason Lee, and Di He. 2024. Rest: Retrieval-based speculative decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1582–1595.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.
- Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Hasan Genc, Kurt Keutzer, Amir Gholami, and Yakun Sophia Shao. 2025. Speed: Speculative

- pipelined execution for efficient decoding. In *Enhancing LLM Performance: Efficacy, Fine-Tuning, and Inference Techniques*, pages 19–32. Springer.
- Yuxuan Hu, Ke Wang, Xiaokang Zhang, Fanjin Zhang, Cuiping Li, Hong Chen, and Jing Zhang. 2025. Sam decoding: Speculative decoding via suffix automaton. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12187–12204.
- Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.
- Xiaopeng Li, Shasha Li, Shangwen Wang, Shezheng Song, Bin Ji, Huijun Liu, Jun Ma, and Jie Yu. 2025a. Identifying knowledge editing types in large language models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 1553–1564.
- Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*.
- Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2025b. Eagle-3: Scaling up inference acceleration of large language models via training-time test. *arXiv preprint arXiv:2503.01840*.
- Jiahao Liu, Qifan Wang, Jingang Wang, and Xunliang Cai. 2024. Speculative decoding via early-exiting for faster llm inference with thompson sampling control mechanism. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3027–3043.
- Tianyu Liu, Yun Li, Qitan Lv, Kai Liu, Jianchen Zhu, Winston Hu, and Xiao Sun. 2025. Pearl: Parallel speculative decoding with adaptive draft length. In *International Conference on Learning Representations*, volume 2025, pages 1085–1104.
- Tianyu Liu, Qitan Lv, Hao Li, Xing Gao, Xiao Sun, and Xiaoyan Sun. 2026. Logitspec: Accelerating retrieval-based speculative decoding via next next token speculation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 33070–33092.
- Xianzhen Luo, Yixuan Wang, Qingfu Zhu, Zhiming Zhang, Xuanyu Zhang, Qing Yang, and Dongliang Xu. 2025. Turning trash into treasure: Accelerating inference of large language models with token recycling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6816–6831.
- Mathematical Association of America. 2025. Maa invitational competitions. <https://maa.org/maa-invitational-competitions/>. Accessed: 2025-09-09.
- Ramesh Nallapati, Bowen Zhou, Cicero Dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pages 280–290.
- Apoorv Saxena. 2023. [Prompt lookup decoding](#).
- Jikai Wang, Yi Su, Juntao Li, Qingrong Xia, Zi Ye, Xinyu Duan, Zhefeng Wang, and Min Zhang. 2025. Opt-tree: Speculative decoding with adaptive draft tree structure. *Transactions of the Association for Computational Linguistics*, 13:188–199.
- Heming Xia, Yongqi Li, Jun Zhang, Cunxiao Du, and Wenjie Li. 2024a. Swift: On-the-fly self-speculative decoding for llm inference acceleration. *arXiv preprint arXiv:2410.06916*.
- Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. 2024b. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7655–7671.
- Bin Xiao, Chunan Shi, Xiaonan Nie, Fan Yang, Xiangwei Deng, Lei Su, Weipeng Chen, and Bin Cui. 2024. Clover: Regressive lightweight speculative decoding with sequential knowledge. *arXiv preprint arXiv:2405.00263*.
- Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. 2024. Draft& verify: Lossless large language model acceleration via self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11263–11282.

A Concrete definitions of utility components.

We now instantiate the locality and continuation-utility terms used in Eq. 6. All quantities are computed from the same retrieval index used for drafting and do not require additional model forward passes or training.

Let $P_i(z) = \{p_1, \dots, p_{n_z}\}$ denote the set of positions in the retrieval index where the query $g_i(z)$ is matched, where $n_z = |\mathcal{M}_i(z)|$. Each position p corresponds to an occurrence of the matched query in the prompt or previously generated history, and exposes a retrieved continuation

$$y_p = (x_{p+1}, x_{p+2}, \dots, x_{p+L_{\max}}),$$

truncated to a maximum probing length $L_{\max}$. In all experiments, we only use the first r matched positions after sorting them by recency, which bounds the cost of utility estimation.

Locality score. We define the locality score of anchor z as an exponential recency-weighted proximity score:

$$\text{PROX}_i(z) = \frac{1}{|\bar{P}_i(z)|} \sum_{p \in \bar{P}_i(z)} \exp\left(-\frac{i-p}{\sigma}\right),$$

where $\bar{P}_i(z)$ contains the r most recent matched positions in $P_i(z)$, and σ is a decay temperature. This term assigns larger values to anchors whose matched continuations come from nearby context positions. The motivation is that nearby matches are more likely to reflect the current discourse, formatting pattern, or local reasoning trajectory than distant matches. For matches from an external data-store where token position is unavailable, we set $\text{PROX}_i(z) = 0$ by default or replace $i - p$ with the retrieval rank.

Continuation-consistency score. We define the continuation utility of anchor z by the average pairwise longest common prefix length among the top- r retrieved continuations:

$$\text{LEN}_i(z) = \frac{1}{\binom{|\bar{P}_i(z)|}{2}} \sum_{\substack{p, q \in \bar{P}_i(z) \\ p < q}} \min(\text{LCP}(y_p, y_q), L_{\max}). \quad (13)$$

when $|\bar{P}_i(z)| \geq 2$. If $|\bar{P}_i(z)| = 1$, we set

$$\text{LEN}_i(z) = \min(\ell_p, L_{\max}),$$

where ℓ_p is the available continuation length after position p before reaching the end of the retrieval buffer. The longest common prefix operator $\text{LCP}(y_p, y_q)$ counts the number of consecutive identical tokens shared by two retrieved continuations. This conservative definition rewards anchors whose matched continuations agree on the next few tokens, while avoiding over-rewarding a single long but potentially spurious match.

Algorithm 1 InfoSpec Drafting with Budgeted Anchor Selection

Require: Prefix $x_{1:i}$, target model M , retrieval index $\mathcal{R}$, tree budget K , candidate sizes k, k' , score weights $\alpha, \beta, \gamma, \delta$
Ensure: Accepted tokens after one speculative decoding step

- 1: Compute next-token distribution $p_i \leftarrow M(x_{1:i})$
- 2: Select x_{i+1} by greedy decoding or sampling
- Candidate anchor construction**
- 3: $\mathcal{C}_i^{(1)} \leftarrow \{x_{i+1}\}$
- 4: $\mathcal{C}_i^{(2)} \leftarrow \text{TopK}(p_i, k) \setminus \{x_{i+1}\}$
- 5: $\mathcal{C}_i^{(2')} \leftarrow \text{LightweightTwoStepApprox}$
 $(x_{1:i}, x_{i+1}, p_i, k')$
- 6: $\mathcal{C}_i \leftarrow \mathcal{C}_i^{(1)} \cup \mathcal{C}_i^{(2)} \cup \mathcal{C}_i^{(2')}$
- Utility estimation by retrieval probing**
- 7: $\mathcal{F}_i \leftarrow \emptyset$
- 8: **for all** $z \in \mathcal{C}_i$ **do**
- 9: Construct retrieval query $g_i(z)$
- 10: $\mathcal{M}_i(z) \leftarrow \text{MATCH}(\mathcal{R}, g_i(z))$
- 11: **if** $|\mathcal{M}_i(z)| > 0$ **then**
- 12: Estimate $\text{PROX}_i(z)$, $\text{LEN}_i(z)$, and branch cost $c_i(z)$
- 13: Compute utility score
 $S_i(z) = \alpha \log p_i(z) - \beta \log(|\mathcal{M}_i(z)| + 1)$
 $\quad + \gamma \text{PROX}_i(z) + \delta \text{LEN}_i(z)$.
- 14: $\rho_i(z) \leftarrow S_i(z) / (c_i(z) + \tau)$
- 15: $\mathcal{F}_i \leftarrow \mathcal{F}_i \cup \{z\}$
- 16: **end if**
- 17: **end for**
- Budgeted anchor selection and verification**
- 18: Sort $\mathcal{F}_i$ by $\rho_i(z)$ in descending order
- 19: Greedily select anchors $\mathcal{A}_i \subseteq \mathcal{F}_i$ under budget K
- 20: Retrieve continuations for anchors in $\mathcal{A}_i$
- 21: Truncate branches so that the total draft-tree size is at most K
- 22: Merge branches into a draft tree $\mathcal{D}_i$
- 23: Construct the tree attention mask A_{draft}
- 24: Verify $\mathcal{D}_i$ with one forward pass of M
- 25: Accept or reject tokens using standard speculative sampling
- 26: Update the KV cache and retrieval index $\mathcal{R}$
- 27: **return** accepted tokens

B Computational Complexity

A naive implementation would enumerate all anchors and scan the full prompt for every query, which is prohibitively expensive. Following the same systems insight as LogitSpec, we build a hash table over prompt m -grams once per sequence, mapping each m -gram to its continuation positions. This reduces per-anchor probing to constant-time

lookup plus bounded continuation copy. Under this design, retrieval accounts for only approximately 1.42% of the total decoding time, whereas model forward computation dominates the overall run-time.

Under this indexing scheme, the incremental cost of InfoSpec over fixed-anchor retrieval is:

$$\mathcal{O}(|\mathcal{C}_i|) \text{ hash probes} + \mathcal{O}(|\mathcal{C}_i| \log |\mathcal{C}_i|) \text{ ranking}, \quad (14)$$

which is negligible so long as $|\mathcal{C}_i|$ remains small. In our intended regime, $|\mathcal{C}_i|$ is bounded by a small top- k plus a small two-step candidate set, keeping CPU-side anchor selection well below the cost of an extra target-model forward pass.

In contrast to prior retrieval-based speculative decoding methods that fix the future retrieval cue a priori, InfoSpec formulates retrieval-anchor selection as a budgeted optimization problem. The key idea is to allocate verification capacity not to the most probable future token, but to the future token that yields the greatest reduction in retrieval ambiguity while remaining plausible under the target model. By combining model confidence, ambiguity reduction, locality, and continuation utility into a training-free anchor score, InfoSpec selects and expands only the most informative anchors under a fixed draft-tree budget. This design preserves the plug-and-play nature of retrieval-based drafting while improving budget efficiency, retrieval precision, and end-to-end decoding speed.

C Ablation Studies

We conduct two complementary ablation studies. First, we separate the effects of fixed-anchor guidance, adaptive candidate construction, and information-aware anchor selection. This analysis distinguishes the gain from considering additional future-token candidates from the gain provided by the proposed scoring criterion. Second, we perform a leave-one-component-out analysis of the anchor utility to quantify the contributions of model confidence, retrieval ambiguity, locality, and continuation consistency.

C.1 Ablation of Adaptive Anchor Selection

We conduct ablations on the two core ingredients inherited from future-guided retrieval-based speculative decoding—future-token guidance and retrieval-based continuation drafting—and extend them to the proposed information-aware anchor selection

mechanism. Prior work shows that removing either retrieval or predictive guidance degrades both MAT and speedup. For example, on Vicuna-7B / Spec-Bench, LogitSpec reaches 2.44 MAT / $2.01\times$ speedup, while removing last-logit guidance reduces performance to 1.72 / $1.27\times$, and removing retrieval reduces it further to 1.57 / $1.24\times$. Similar degradations hold across CNN/DM, GSM8K, and HumanEval, indicating that speedup arises from the interaction between predictive guidance and retrieval rather than from either component alone.

For InfoSpec, the corresponding ablation isolates three components: (1) fixed future anchor, (2) adaptive anchor without information scoring, and (3) full information-aware anchor selection. This ablation is critical because it distinguishes the effect of adding more future candidates from the effect of selecting more useful future candidates.

Table 3 serves as the main component-level analysis of InfoSpec, separating the contribution of adaptive anchor generation from that of the proposed information-aware scoring function. It shows that simply introducing adaptivity is not sufficient to achieve the best performance; the anchor-selection criterion also plays an important role. The gap between adaptive anchor selection without information scoring and full InfoSpec estimates the contribution of the proposed scoring criterion. The consistent gap across tasks supports the hypothesis that effective retrieval anchors should balance model plausibility with retrieval-uncertainty reduction, rather than relying solely on token probability.

C.2 Component-wise Ablation of Anchor Utility

To quantify the contribution of each signal in the InfoSpec utility function, we conduct a leave-one-component-out ablation on Vicuna-7B. For each variant, we disable one utility component by setting its corresponding coefficient to zero, while keeping the other coefficients unchanged. All variants use the same candidate anchor pool, retrieval index, draft-tree budget $K=64$, branch-construction policy, and verification procedure. Therefore, the observed differences can be attributed directly to the removed utility component.

Table 4 reports MAT and end-to-end speedup on CNN/DM, GSM8K, HumanEval, and Spec-Bench. Full InfoSpec achieves the best performance on every benchmark, reaching an average of 2.88 MAT and $2.37\times$ speedup.

Removing model confidence causes the largest

Model	Method	Spec-Bench		CNN/DM		GSM8K		HumanEval	
		MAT	Speedup	MAT	Speedup	MAT	Speedup	MAT	Speedup
Vicuna 7B	w/o last-logit	1.72	1.27	2.66	2.24	1.81	1.63	1.89	1.69
	w/o retrieval	1.57	1.24	1.57	1.23	1.71	1.39	1.69	1.39
	LogitSpec	2.44	2.01	3.28	2.61	2.69	2.20	2.62	2.24
	adaptive anchor w/o info score	2.49	2.05	3.29	2.62	2.73	2.23	2.66	2.27
	InfoSpec	2.58	2.12	3.34	2.66	2.83	2.31	2.78	2.37

Table 3: Ablation study on retrieval guidance and anchor selection.

Method	CNN/DM	GSM8K	HumanEval	Spec-Bench	Average
InfoSpec (Full)	3.34 / 2.66	2.83 / 2.31	2.78 / 2.37	2.58 / 2.12	2.88 / 2.37
w/o Confidence	2.85 / 2.15	2.31 / 1.88	2.45 / 2.06	2.11 / 1.76	2.43 / 1.96
w/o Ambiguity	2.91 / 2.22	2.40 / 1.95	2.38 / 1.98	2.23 / 1.85	2.48 / 2.00
w/o Locality	3.22 / 2.54	2.76 / 2.25	2.65 / 2.24	2.49 / 2.05	2.78 / 2.27
w/o Continuation Consistency	3.28 / 2.60	2.79 / 2.28	2.70 / 2.31	2.52 / 2.07	2.82 / 2.32

Table 4: Component-wise ablation of the InfoSpec anchor utility on Vicuna-7B. Each entry reports MAT / end-to-end speedup. All variants use the same candidate pool, retrieval index, draft-tree budget $K = 64$, branch-construction policy, and verification procedure; only the indicated utility component is removed.

degradation, reducing the average result from $2.88 \text{ MAT} / 2.37\times$ to $2.43 \text{ MAT} / 1.96\times$. This result shows that retrieval usefulness alone is insufficient: an anchor must also remain plausible under the target model. Without the confidence term, the selector may prefer highly specific but unlikely future tokens whose retrieved continuations are rarely accepted.

Removing the ambiguity component produces the second-largest decrease, reducing average performance to $2.48 \text{ MAT} / 2.00\times$. This confirms that match-set contraction is central to InfoSpec. High-probability anchors can still be poor retrieval cues when they retrieve many divergent or weakly relevant continuations, consuming the fixed tree budget without producing long accepted prefixes.

Locality and continuation consistency provide smaller but consistent improvements. Removing locality reduces the average result to $2.78 \text{ MAT} / 2.27\times$, indicating that recent or context-consistent matches are generally more useful than distant matches. Removing continuation consistency reduces performance to $2.82 \text{ MAT} / 2.32\times$, showing that agreement among retrieved continuations provides an additional signal for identifying stable and mergeable draft branches.

Overall, the four components play complementary roles. Model confidence prevents the selection of implausible anchors, ambiguity reduction favors discriminative retrieval queries, locality prioritizes context-relevant matches, and continuation consistency rewards anchors whose retrieved branches agree on a useful shared prefix. The results support combining model-side plausibility with retrieval-

side utility rather than relying on any single heuristic.

D Additional Results on More Recent Backbones

To further assess whether the proposed anchor-selection strategy generalizes beyond the Vicuna and Llama-2 families, we additionally evaluate on Llama-3.1-8B-Instruct and Qwen3-8B. These two models are particularly useful test cases because they differ not only in pretraining recipe and instruction tuning, but also in vocabulary design and decoding behavior. Therefore, improvements on these backbones would indicate that the proposed retrieval-anchor selection mechanism does not depend on a specific model family. Table 5 reports mean accepted tokens per decoding step (MAT) and end-to-end speedup on Llama-3.1-8B-Instruct and Qwen3-8B across Spec-Bench, CNN/DM, GSM8K, and HumanEval. The results show that InfoSpec improves over fixed-anchor future-guided retrieval on both model families, indicating that the proposed retrieval-anchor selection strategy generalizes across newer backbones with different vocabularies and decoding characteristics. On Llama-3.1-8B-Instruct, LogitSpec already achieves strong performance, reaching $1.88\times$ speedup on Spec-Bench and $2.04\times$ overall on CNN/DM, GSM8K, and HumanEval, while InfoSpec further improves these results to $1.97\times$ and $2.13\times$, respectively. On Qwen3-8B, the corresponding gains are from $1.75\times$ to $1.84\times$ on Spec-Bench and from $1.82\times$ to $1.90\times$ overall.

Model	Method	Spec-Bench		CNN/DM		GSM8K		HumanEval		Overall	
		MAT	Speedup	MAT	Speedup	MAT	Speedup	MAT	Speedup	MAT	Speedup
Llama-3.1-8B-Instruct	Vanilla	1.00	1.00×	1.00	1.00×	1.00	1.00×	1.00	1.00×	1.00	1.00×
	LogitSpec	2.11	1.88×	2.04	1.84×	2.18	1.93×	2.62	2.30×	2.24	1.99×
	InfoSpec	2.23	1.97×	2.13	1.93×	2.31	2.06×	2.79	2.43×	2.37	2.10×
Qwen3-8B	Vanilla	1.00	1.00×	1.00	1.00×	1.00	1.00×	1.00	1.00×	1.00	1.00×
	LogitSpec	1.95	1.75×	1.77	1.58×	2.18	1.92×	2.17	1.91×	2.02	1.79×
	InfoSpec	2.07	1.84×	1.89	1.68×	2.31	2.05×	2.34	2.03×	2.15	1.90×

Table 5: **Additional results on more recent backbones.** We report MAT and end-to-end speedup on Llama-3.1-8B-Instruct and Qwen3-8B over Spec-Bench, CNN/DM, GSM8K, and HumanEval. InfoSpec outperforms fixed-anchor future-guided retrieval on both backbones, demonstrating strong cross-model generalization.

E Comparison with Stronger Retrieval Baselines

We further compare InfoSpec with stronger retrieval-based speculative decoding baselines beyond the classical PLD/REST setting. Table 6 reports the mean accepted tokens per decoding step (MAT) and end-to-end speedup on Llama-3.1-8B-Instruct across AIME, LongBench, and MATH. Compared with SAM Decoding and Token Recycling, LogitSpec already provides a stronger future-guided retrieval baseline, achieving an average of 3.23 MAT and $2.58\times$ speedup, outperforming Token Recycling by 0.27 MAT and $0.18\times$ speedup, and SAM Decoding by 0.31 MAT and $0.24\times$ speedup. The advantage is particularly pronounced on AIME, where LogitSpec improves over Token Recycling from 3.10 to 3.57 MAT and from $2.82\times$ to $3.09\times$ speedup, suggesting that future-token-guided retrieval is especially effective for challenging reasoning tasks where exact surface-level repetition is limited.

Building on this strong baseline, InfoSpec further improves performance across all evaluated datasets. On average, InfoSpec achieves 3.41 MAT and $2.72\times$ speedup, improving over LogitSpec by 0.18 MAT and $0.14\times$ speedup. The gains are consistent across AIME, LongBench, and MATH, with InfoSpec reaching 3.81, 3.21, and 3.20 MAT, respectively. These results indicate that information-aware anchor scoring can select more effective retrieval anchors than fixed-anchor future-guided retrieval, leading to higher token acceptance rates and better end-to-end acceleration.

F Mechanism Analysis

Retrieval Ambiguity Analysis. To analyze the mechanism behind InfoSpec, we report the empirical distribution of match-set sizes induced by different retrieval anchors. As shown by the empirical match-set distribution, fixed next-token anchors

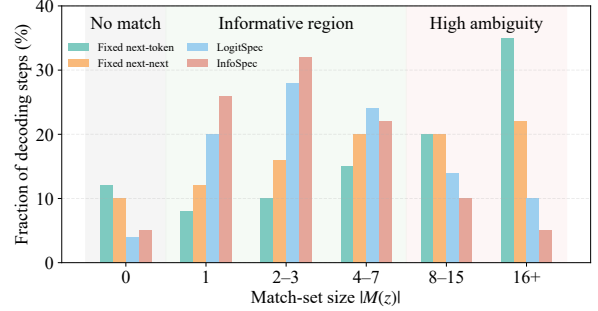


Figure 4: Distribution of match-set size under different retrieval anchors. Fixed next-token retrieval yields many highly ambiguous matches, while fixed next-next retrieval partially reduces ambiguity. LogitSpec shifts the distribution toward smaller match sets by using future-token guidance. InfoSpec further concentrates retrieval mass in the low-ambiguity but non-empty regime, indicating that adaptive anchor selection better balances retrieval specificity and match availability.

produce larger and more ambiguous match sets, while fixed future anchors reduce ambiguity only partially. InfoSpec explicitly scores candidate anchors by retrieval utility and therefore shifts the distribution toward the low-ambiguity but non-empty regime.

As shown in Figure 4, InfoSpec does not merely retrieve more candidates, it retrieves more discriminative candidates with smaller non-empty match sets. Compared with fixed-anchor strategies, InfoSpec reduces the fraction of high-ambiguity matches while preserving non-empty retrievals, providing evidence that adaptive anchor scoring improves retrieval specificity.

Budgeted Verification Efficiency. A second key question is whether better retrieval anchors translate into better use of the verification budget. Prior work shows a clear trade-off between accepted length and tree-verification cost: increasing tree capacity improves MAT, but speedup peaks at a moderate capacity and then declines once verification

Method	AIME		LongBench		MATH		Average	
	MAT	Speedup	MAT	Speedup	MAT	Speedup	MAT	Speedup
AR (Greedy)	1.00	1.00×	1.00	1.00×	1.00	1.00×	1.00	1.00×
SAM Decoding	3.10	2.72×	3.01	2.02×	2.66	2.25×	2.92	2.33×
Token Recycling	3.11	2.80×	2.96	1.96×	2.78	2.38×	2.95	2.38×
LogitSpec	3.56	3.08×	3.07	2.08×	3.02	2.55×	3.22	2.57×
InfoSpec	3.81	3.23×	3.21	2.22×	3.20	2.71×	3.41	2.72×

Table 6: Comparison with stronger retrieval-based baselines on Llama-3.1-8B-Instruct.

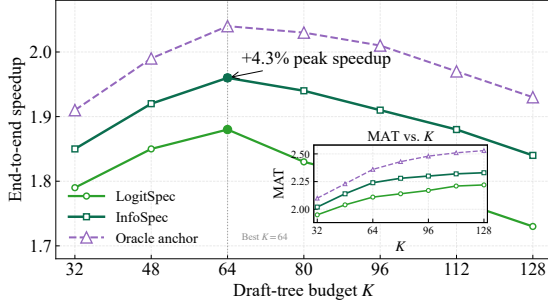


Figure 5: Effect of draft-tree budget K on end-to-end acceleration. Increasing K allows more draft candidates to be verified and improves MAT, but overly large trees introduce additional verification overhead and reduce end-to-end speedup. The Oracle anchor represents an optional upper-bound curve.

overhead dominates. On Llama-3.1-8B-Instruct, speedup peaks at $K = 64$ with $1.88\times$, while MAT continues to rise slightly for larger trees, indicating that acceptance length and practical speedup do not necessarily peak at the same point.

Table 7 reports the numerical trade-off between accepted length and end-to-end speedup under different draft-tree budgets. Figure 5 studies the effect of draft-tree budget K on end-to-end acceleration. As K increases, both LogitSpec and InfoSpec initially achieve higher speedup due to larger draft capacity. However, overly large trees introduce additional verification overhead, causing end-to-end speedup to saturate and eventually decline. InfoSpec outperforms LogitSpec across all budgets and achieves a higher peak speedup, demonstrating that information-guided anchor selection uses the limited verification budget more effectively. The oracle anchor is computed offline by selecting, among all probed candidates, the anchor that yields the longest verified continuation; it is used only as an analytical upper bound and is not available at inference time.

Retrieval Success Rate and Runtime Overhead.

We next examine whether InfoSpec’s gains are accompanied by improved retrieval success without

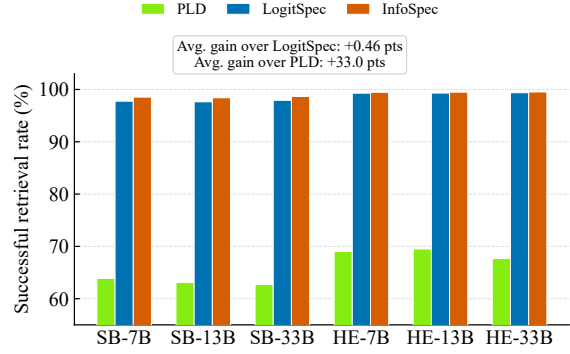


Figure 6: Successful retrieval rate on Spec-Bench and HumanEval.

introducing substantial additional engineering overhead. Table 8 reports the successful retrieval rate on Spec-Bench and HumanEval. In prior future-guided retrieval-based decoding, LogitSpec improves retrieval success over PLD, raising the rate from 63.88 / 63.08 / 62.71 to 97.76 / 97.64 / 97.93 on Spec-Bench for Vicuna-7B / 13B / 33B, and from 69.03 / 69.51 / 67.67 to 99.29 / 99.31 / 99.37 on HumanEval. These results indicate that stronger speculative decoding is closely tied to stronger retrieval quality.

Figure 6 visualizes the same trend and highlights the improvement of InfoSpec over fixed-anchor retrieval. Figure 7 reports the wall-clock time breakdown of InfoSpec on Llama-3.1-8B-Instruct with $K=64$. Importantly, prior results show that the retrieval stage itself accounts for only 1.42% of total decoding time, whereas model forward remains the dominant cost with 92.85%. This suggests that retrieval-guided acceleration benefits primarily from better draft quality rather than from heavy auxiliary computation. The measured gains of InfoSpec over LogitSpec, together with the small additional anchor-probing overhead, indicate that more informative anchors improve retrieval effectiveness without materially increasing system overhead.

Method	K=32	K=48	K=64	K=80	K=96	K=112	K=128
LogitSpec MAT	1.95	2.04	2.11	2.14	2.17	2.21	2.22
LogitSpec Speedup	1.79	1.85	1.88	1.83	1.80	1.77	1.73
InfoSpec MAT	2.02	2.14	2.24	2.28	2.30	2.32	2.33
InfoSpec Speedup	1.85	1.92	1.96	1.94	1.91	1.88	1.84

Table 7: Budget trade-off under different tree capacities K .

Method	SB-7B	SB-13B	SB-33B	HE-7B	HE-13B	HE-33B
PLD	63.88	63.08	62.71	69.03	69.51	67.67
LogitSpec	97.76	97.64	97.93	99.29	99.31	99.37
InfoSpec	98.54	98.41	98.66	99.46	99.49	99.52

Table 8: Successful retrieval rate (%). SB denotes Spec-Bench and HE denotes HumanEval.

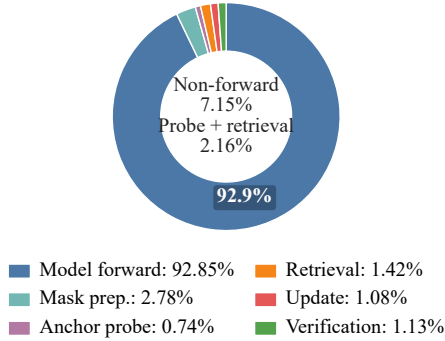


Figure 7: Wall-clock time breakdown of the decoding pipeline. Retrieval-guided speculative decoding benefits from increasing successful and useful draft retrievals, while the retrieval module contributes relatively small overhead compared with model forward.

G Robustness under Non-Zero-Temperature Sampling

The main experiments use greedy decoding with $T = 0$. To evaluate whether the proposed anchor utility remains effective under stochastic decoding, we additionally compare LogitSpec and InfoSpec on Llama-3.1-8B-Instruct under $T \in \{0.2, 0.4, 0.6, 0.8\}$. We evaluate on AIME, LongBench, and MATH. Both methods use the same sampling configuration, prompts, hardware environment, retrieval index, draft-tree budget, and verification procedure. Speedup is measured relative to target-only autoregressive decoding under the same sampling configuration.

For a fixed decoding state and $T > 0$, temperature scaling preserves the ordering of logits before probability-mass-based truncation. However, logit-rank preservation alone does not guarantee efficient speculative decoding. Temperature changes the probability mass assigned to non-top-1 tokens, and

Dataset	T	LogitSpec		InfoSpec	
		MAT	Speedup	MAT	Speedup
AIME	0.0	3.57	3.09×	3.63	3.15×
	0.2	3.45	2.84×	3.45	2.92×
	0.4	3.32	2.89×	3.62	2.96×
	0.6	3.18	2.75×	3.22	2.84×
	0.8	3.12	2.63×	3.20	2.92×
LongBench	0.0	3.08	2.10×	3.15	2.19×
	0.2	2.92	1.93×	3.10	2.07×
	0.4	2.90	1.92×	3.02	2.02×
	0.6	2.85	1.89×	2.89	1.93×
	0.8	2.82	1.85×	2.91	1.92×
MATH	0.0	3.05	2.56×	3.15	2.61×
	0.2	3.18	2.62×	3.21	2.70×
	0.4	3.02	2.53×	3.13	2.55×
	0.6	2.90	2.45×	2.98	2.51×
	0.8	2.81	2.40×	2.92	2.45×

Table 9: Non-zero-temperature decoding results on Llama-3.1-8B-Instruct. We report mean accepted tokens per decoding step (MAT) and end-to-end speedup on AIME, LongBench, and MATH. Both methods use identical sampling settings, prompts, hardware, retrieval index, draft-tree budget, and verification procedure. Speedup is measured relative to target-only autoregressive decoding under the same sampling configuration.

these tokens may be sampled frequently even when they are not the argmax. If their continuations are not represented in the draft tree, the verified prefix may become shorter. Sampling also changes the realized trajectory and therefore affects subsequent retrieval matches, ambiguity, continuation consistency, and acceptance behavior.

This setting motivates information-aware anchor selection under a finite tree budget. In addition to model probability, InfoSpec considers whether a candidate produces specific, local, and consistent retrieved continuations, thereby allocating draft capacity to branches with higher verification utility.

As shown in Table 9, InfoSpec matches or exceeds LogitSpec in MAT and achieves higher end-to-end speedup in every evaluated dataset–

Dataset	Vanilla		LogitSpec		InfoSpec	
	MAT	Speedup	MAT	Speedup	MAT	Speedup
LongBench	1.00	1.00×	3.09	2.01×	3.21	2.15×
MATH	1.00	1.00×	3.32	2.78×	3.45	2.91×
AIME24	1.00	1.00×	3.41	3.23×	3.58	3.39×
AIME25	1.00	1.00×	3.76	3.32×	3.92	3.47×

Table 10: Long-context and reasoning results.

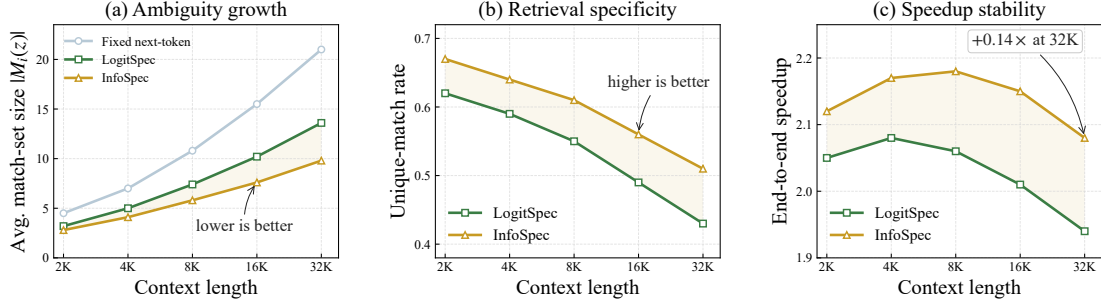


Figure 8: Long-context ambiguity scaling. As context length increases, the retrieval space becomes richer but also more repetitive, causing ambiguity to grow for fixed-anchor retrieval. InfoSpec slows down ambiguity growth by selecting more discriminative retrieval anchors, thereby preserving higher retrieval specificity and more stable end-to-end speedup in long-context settings.

temperature setting. Across the 12 non-zero-temperature settings, InfoSpec improves average MAT from 3.04 to 3.14 and average speedup from $2.39\times$ to $2.48\times$. When the three greedy settings are also included, the corresponding averages improve from 3.08 to 3.17 MAT and from $2.43\times$ to $2.52\times$ speedup.

The results demonstrate that the benefit of information-aware anchor selection is not limited to greedy decoding. In particular, InfoSpec remains effective when non-top-1 tokens receive meaningful sampling probability and alter the subsequent retrieval trajectory. The performance trend is not necessarily monotonic with temperature on every dataset because stochastic decoding introduces sampling variation, but InfoSpec consistently maintains an advantage over fixed-anchor LogitSpec.

H Long-Context and Reasoning Analysis

Finally, we evaluate the method in two challenging settings that exhibit severe retrieval ambiguity: long-context generation and hard reasoning. For InfoSpec, long-context evaluation is especially informative. As context length grows, the prompt provides a richer retrieval database, but it also introduces more repeated phrases and therefore more ambiguous matches. The results show that InfoSpec slows the growth of retrieval ambiguity with context length and preserves stronger speedup in

LongBench-style settings.

Table 10 reports results on long-context and reasoning-intensive benchmarks, and Figure 8 empirically analyzes how retrieval ambiguity evolves with context length. It shows that longer contexts amplify both retrieval opportunities and retrieval ambiguity. While larger prompts create more retrieval candidates, they also increase the probability of incorrect or weakly relevant matches. The results suggest that InfoSpec mitigates this effect by adaptively choosing more informative anchors in repetitive contexts, leading to a more stable speedup curve as sequence length grows.