HeteroSpec: Leveraging Contextual Heterogeneity for Efficient Speculative Decoding

Siran Liu[1,2], **Yang Ye**[1], **Qianchao Zhu**[1], **Zane Cao**[2], **Yongchao He**[2†]
[1]Peking University [2]ScitiX AI
{liusr25, yang.ye}@stu.pku.edu.cn, dysania@pku.edu.cn, zcao@scitix.ai, yongchao-he@outlook.com

Abstract

Autoregressive decoding inherently limits the inference throughput of Large Language Model (LLM) due to its sequential dependency. *Speculative decoding* mitigates this by verifying multiple predicted tokens in parallel, but its efficiency remains constrained by what we identify as *verification heterogeneity*—the uneven difficulty of verifying different speculative candidates. In practice, a small subset of high-confidence predictions accounts for most successful verifications, yet existing methods treat all candidates uniformly, leading to redundant computation. We present **HeteroSpec**, a **hetero**geneity-adaptive **spec**ulative decoding framework that allocates verification effort in proportion to candidate uncertainty. HeteroSpec estimates verification complexity using a lightweight entropy-based quantifier, partitions candidates via a data-driven stratification policy, and dynamically tunes speculative depth and pruning thresholds through coordinated optimization. Across five benchmarks and four LLMs, HeteroSpec delivers an average **4.24**× decoding speedup over state-of-the-art methods such as EAGLE-3, while preserving exact output distributions. Crucially, HeteroSpec requires no model retraining and remains compatible with other inference optimizations, making it a practical direction for improving speculative decoding efficiency.

1 Introduction

Autoregressive decoding serves as the foundation for modern large language models (LLMs), enabling high-quality text generation across diverse applications such as dialogue systems, summarization, and question answering (Brown et al., 2020; Touvron et al., 2023; Vaswani et al., 2023; OpenAI et al., 2024). However, this sequential generation paradigm introduces a significant computational bottleneck, as each token requires a complete forward pass through the target model (Kasai et al., 2021; Shazeer, 2019). Developing inference acceleration techniques that preserve output quality and distributional correctness is therefore essential for scalable LLM deployment.

Speculative decoding has emerged as an effective approach to mitigate this sequential bottleneck (Chen et al., 2023; Leviathan et al., 2023). This paradigm employs a smaller *draft model* to propose candidate token sequences, which are subsequently validated in parallel by the target model, reducing sequential forward passes while maintaining exact distributional guarantees through rejection sampling. Recent advances have explored diverse strategies: Medusa (Cai et al., 2024) uses parallel prediction heads, EAGLE-2 (Li et al., 2024) introduces confidence-guided dynamic draft trees, and EAGLE-3 (Li et al., 2025a) leverages multi-layer feature aggregation with relaxed training constraints to improve draft quality.

Despite this progress, speculative decoding still faces a fundamental challenge stemming from the **dynamic and heterogeneous nature** of token prediction. As characterized by Zipf's Law (Zipf, 1949), natural language exhibits a highly skewed distribution: a small set of high-frequency patterns constitutes the bulk of text, while a long tail of complex, low-frequency structures proves computationally challenging. Consequently, text generation is far from uniformly predictable; it constantly transitions between highly frequent, easily anticipatable patterns and complex, low-frequency structures, leading to dynamically fluctuating "decoding difficulty" during generation.

Current dynamic methods (Brown et al., 2024; Zhang et al., 2024c; Huang et al., 2024; Zhang et al., 2024b) recognize this need for adaptation, employing confidence metrics or trained predictors to control drafting length and stopping crite-

†Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 12930–12947

ria. However, these approaches primarily focus on optimizing draft generation while the verification phase—which constitutes 67-90% of total computation—receives limited attention for fine-grained adaptation. Moreover, static thresholds and pre-trained predictors struggle with the context complexity of language, making adaptive controls fragile across diverse generation scenarios.

To understand this challenge, we empirically profile the draft acceptance process in EAGLE-3 (§3). Our analysis reveals a pronounced *verification heterogeneity* [1] in draft acceptance outcomes: a small fraction of high-confidence, top-ranked draft candidates are disproportionately responsible for accepted tokens and contribute significantly to acceleration, while the majority yield minimal or no accepted prefixes. This highlights the inefficiency of uniformly processing all candidates and strongly suggests that dynamically allocating computational resources, particularly the computationally expensive verification effort, based on predicted confidence and linguistic complexity, can substantially improve efficiency by prioritizing the most promising candidates.

Motivated by these empirical insights, we introduce **HeteroSpec** (**Hetero**geneity-Adaptive **Spec**ulative Decoding), a framework that addresses the verification bottleneck through complexity-aware adaptive optimization. HeteroSpec comprises three synergistic components that collectively transform uniform speculation into heterogeneity-adaptive resource allocation. First, it introduces a *contextual complexity quantification* module that assesses real-time generation predictability using a novel cumulative meta-path Top-K entropy metric. Second, based on this complexity score, an *adaptive decision framework* employs data-driven entropy stratification to partition the generation process into distinct, actionable regimes. Finally, *coordinated adaptive optimization* mechanisms leverage these regimes to dynamically adjust speculative depth, prune unpromising candidates, and select efficient computational graphs, thereby allocating verification resources where they are most effective.

Across five representative benchmarks spanning diverse task categories and four open-source LLMs, HeteroSpec achieves a **4.24**$\times$ average speedup, consistently outperforming the state-of-the-art EAGLE-3 across all scenarios while maintaining exact distributional guarantees. This improvement costs less than 1% additional overhead, yielding a net gain in overall inference speed. HeteroSpec is orthogonal to existing acceleration techniques and requires no model retraining for deployment. Our main contributions are:

- **Empirical Characterization of Heterogeneity**: We demonstrate that a small fraction of high-confidence candidates disproportionately drive the majority of successful speculation. We establish this heterogeneity as a critical, previously overlooked performance bottleneck, thereby creating a new, data-driven foundation for optimization. (§3)

- **The HeteroSpec Framework**: We design and implement HeteroSpec, a novel adaptive decoding framework that integrates real-time complexity quantification, data-driven decision-making, and coordinated multi-level optimizations to dynamically allocate computational resources. (§4)

- **Comprehensive Empirical Validation**: Experiments show that HeteroSpec significantly outperforms state-of-the-art methods across diverse models and benchmarks, establishing a new, more efficient standard for speculative decoding while providing in-depth analyses of its performance and robustness. (§5)

2 Preliminaries

2.1 Speculative Decoding

Speculative decoding (Chen et al., 2023; Leviathan et al., 2023) accelerates autoregressive LLM inference while preserving the exact target model probability distribution. A lightweight draft model proposes k candidate tokens $\hat{T}_{j+1:j+k}$ following prefix $T_{1:j}$, which the target model processes in parallel. In the *verification stage*, tokens are validated sequentially: $\hat{t}_{j+i}$ is accepted with probability $A_{j+i} = \min\left(1, \frac{p_{j+i}(\hat{t}_{j+i})}{\hat{p}_{j+i}(\hat{t}_{j+i})}\right)$, where p and $\hat{p}$ denote target and draft distributions. Upon rejection, a token is sampled from residual distribution $p_{j+i} - \hat{p}_{j+i}$ to maintain fidelity, subsequent drafts are discarded, and decoding resumes from $j + i$.

[1] We adopt the systems perspective of "heterogeneity," describing variance in computational requirements and resource efficiency across execution contexts. This differs from the mathematical interpretation where concentrated outcomes represent homogeneity, and aligns with systems literature conventions (e.g., "heterogeneous workloads").

2.2 EAGLEs and its Draft Tree Construction

Building on standard speculative decoding, the EAGLE family significantly advances LLM inference acceleration. Its key innovation is **dynamic draft tree construction**, introduced in EAGLE-2 (Li et al., 2024). This approach moves beyond fixed-length drafts by adaptively proposing candidate sequences in tree structures through two stages: *Expansion* and *Reranking*. EAGLE-3 (Li et al., 2025a) refines this framework, enhancing predictive power by removing feature loss constraints and incorporating multi-layer information. The dynamic tree construction process in EAGLE-2 consists of the following two phases:

- **Expansion**: EAGLE-2 constructs preliminary draft tree T_1 by expanding nodes with Top-k tokens from draft model distribution $\hat{p}$. Expansion prioritizes branches with high estimated global acceptance values V_i, defined as the product of acceptance probabilities along the path from root to node i. True acceptance probability A_j is approximated by draft model confidence score c_j, yielding $V_i \approx \prod_{t_j \in \text{Path(root},i)} c_j$. Expansion is limited by maximum depth d.

- **Reranking**: All T_1 nodes are re-evaluated by global acceptance values V_i. Top-N nodes with highest V_i form pruned subtree T_2, maintaining validity since V_i is bounded by ancestors. T_2 undergoes parallel target model verification using tree-mask attention to compute all token probabilities p simultaneously.

3 Observations

3.1 The Verification Bottleneck

The target model verification stage constitutes the primary computational bottleneck in speculative decoding. As illustrated in Figure 1(b), this stage consumes a staggering 67-90% of the total runtime across different model sizes. This substantial cost stems from the large size and complexity of target models. Consequently, improving overall inference efficiency critically depends on optimizing this stage, which can be achieved through two principles: (1) reducing the frequency of required verifications, and (2) decreasing the computational cost of each verification pass.

3.2 Characterizing Heterogeneity

To analyze successful draft acceptance dynamics, we introduce the metric of **Terminal Confidence Rank (TCR)**. As depicted in Figure 1(a), TCR is defined as the rank (among the Top-N candidate sequences generated by the reranking phase) of the longest prefix ultimately accepted by the target model in a given decoding iteration.

Empirical analysis using EAGLE3-LLAMA3.1-8B on MT Bench (Zheng et al., 2023) reveals significant heterogeneity in the speculative decoding process (we provide additional experimental evidence of this heterogeneity across more models and datasets in Appendix A.1). Figure 1(c) and (e) show that Terminal Confidence Rank is heavily concentrated among the top 25% of Top-N candidates. Furthermore, Figure 1(d) shows a strong correlation between lower TCRs and longer average accepted lengths, often approaching the maximum acceptance length. These findings indicate that sequences originating from high-confidence, top-ranked draft candidates are substantially more likely to be accepted and yield greater length gains.

This observed heterogeneity aligns with the nature of language and phenomena like Zipf's Law, whereby a small number of high-frequency patterns (such as common words and punctuation) make up the majority of natural language text. Simple, high-frequency linguistic patterns are more accurately predicted by the draft model, resulting in higher confidence (lower TCR) and higher acceptance probabilities leading to longer accepted prefixes. Conversely, complex or low-frequency structures are harder to predict, resulting in lower confidence (higher TCR) and shorter accepted lengths. As draft models improve, they expand the scope of patterns that can be reliably predicted, effectively reclassifying previously complex structures as simple, predictable patterns, thereby potentially amplifying this heterogeneity effect.

3.3 Implications for Further Optimization

Our empirical analysis reveals a pronounced heterogeneity in the success of draft candidates. We observe that **a small fraction of high-confidence paths generates the vast majority of successfully accepted tokens**, as shown in Figure 1(c-e). These high-potential paths are often characterized by a low TCR. Conversely, a large volume of draft candidates contributes little to no accepted sequence length, representing a significant source of redun-

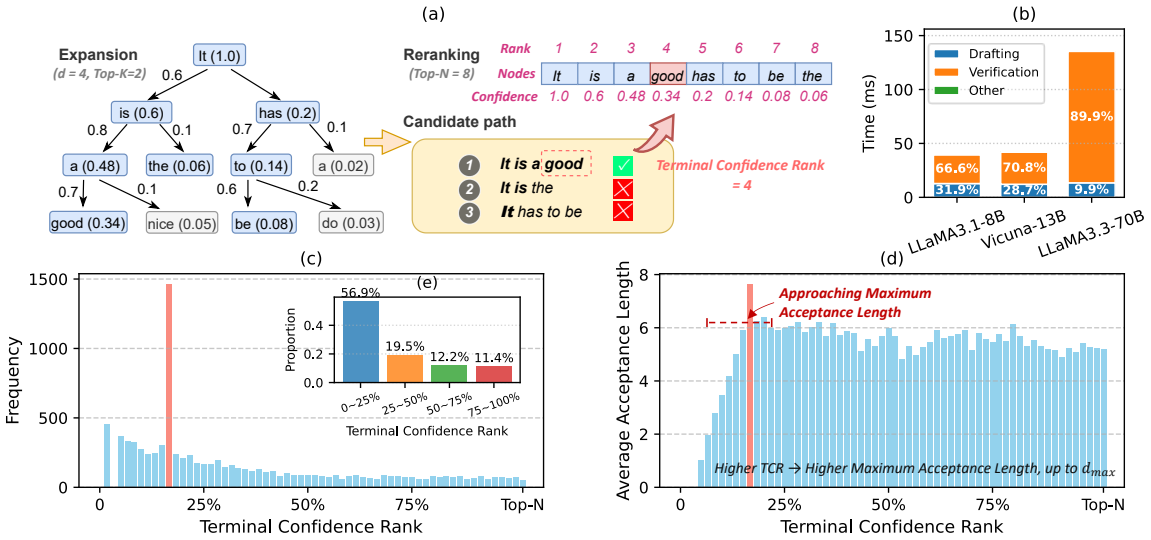

Figure 1: Key empirical observations with EAGLE-3. (a) Illustration of the Terminal Confidence Rank (TCR). (b) Breakdown of runtime overhead during single-turn speculative decoding for models of different sizes. (c) Distribution of Terminal Confidence Rank within the Top-N draft candidates (with prominent values highlighted in orange). (d) Correlation between Average Acceptance Length and Terminal Confidence Rank. The initial rise reflects increasing maximum acceptance length as TCR grows, until reaching the maximum draft depth. (e) Quantile analysis of the Terminal Confidence Rank distribution, showing concentration within the top percentiles of Top-N.

dant computation during the expensive verification stage. This clear asymmetry between a draft's potential and its final outcome is the central insight for further optimization.

Leveraging this insight, a clear optimization strategy emerges: **dynamically** allocate verification resources by prioritizing **high-potential draft paths**. Rather than treating all candidates uniformly, this approach concentrates the target model's computational budget on the branches most likely to yield long accepted sequences. This alignment of verification effort with the empirical likelihood of success directly maximizes the return on investment for each verification call.

4 Methodology

HeteroSpec comprises three synergistic components: Section 4.1 introduces contextual complexity quantification for real-time predictability assessment; Section 4.2 presents an adaptive decision framework that stratifies complexity patterns through data-driven partitioning; Section 4.3 develops coordinated adaptive optimization mechanisms that dynamically allocate computational resources based on complexity assessment. Figure 2 illustrates the unified framework.

4.1 Contextual Complexity Quantification

The fundamental challenge in adaptive speculative decoding lies in distinguishing between high-predictability contexts that enable aggressive speculation and complex contexts requiring conservative approaches. To address this challenge, we introduce the **Cumulative Meta-Path Top-K Entropy** as our complexity oracle. For a candidate speculation path $\mathcal{P} = (x_1, x_2, \ldots, x_T)$, we define:

$$\Phi(\mathcal{P}) = \sum_{t=1}^{T} \mathcal{I}_t(\mathcal{P}) = -\sum_{t=1}^{T} \sum_{i=1}^{\text{Top-}K} \tilde{p}_{t,i} \log \tilde{p}_{t,i} \quad (1)$$

where $\mathcal{I}_t(\mathcal{P})$ captures the instantaneous uncertainty at position t, and $\tilde{p}_{t,i}$ represents the normalized probabilities of the Top-K tokens.

This design is motivated by two key principles: (1) Empirical foundation—simple, well-structured contexts exhibit highly skewed probability distributions with low per-step entropy, making cumulative entropy an effective discriminator for predictable generation patterns; (2) Computational efficiency—the Top-K approximation ensures $\mathcal{O}(K \cdot T)$ complexity suitable for real-time assessment.

For the operational metric, we focus on the candidate path $\mathcal{P}^*$ with the highest final-token confidence, i.e., $\mathcal{P}^* = \arg\max_{\mathcal{P}} p_{T,\text{Top-1}}^{(|\mathcal{P}|)}$. We compute $\Phi(\mathcal{P}^*)$ as the confidence indicator for the speculation tree at that step. We provide additional experi-

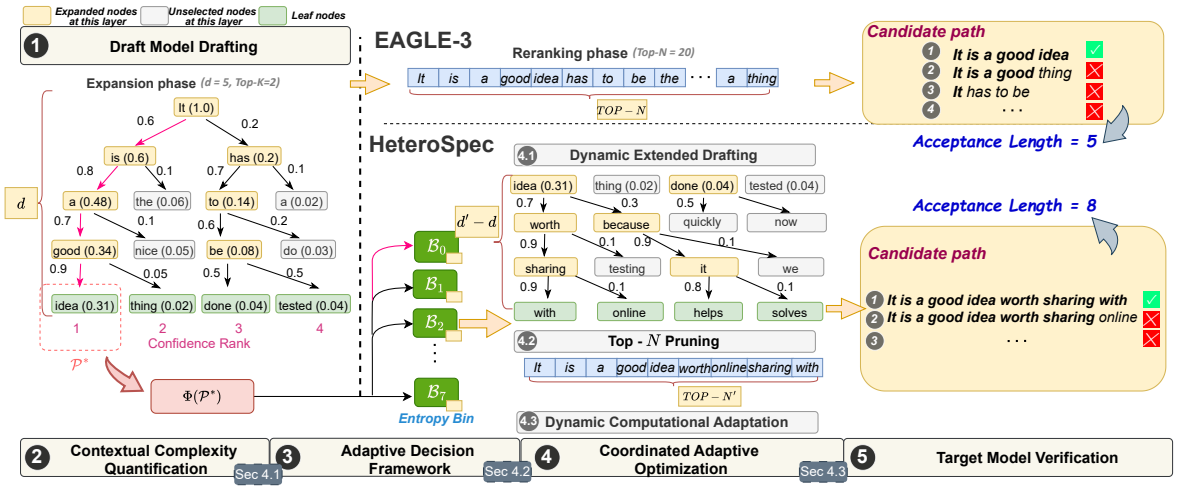

Figure 2: Illustration of the HeteroSpec framework, where ②, ③, and ④ represent our three unique modules. We demonstrate the main differences between HeteroSpec and EAGLE-3 in the inference pipeline using an example of an EAGLE drafting tree with Top-K=2, Top-N=20, and Depth=5.

mental validation of this metric's effectiveness in Appendix A.2.

4.2 Adaptive Decision Framework

The continuous nature of contextual complexity requires a principled method to discretize the complexity space into actionable decision regions that enable systematic resource allocation policies. Manual threshold specification lacks adaptability and fails to capture the nuanced relationships between complexity patterns and speculation outcomes.

We establish a **data-driven decision framework** that learns optimal complexity boundaries from empirical speculation trajectories, leveraging the interpretability and discretization strengths of Classification and Regression Tree (CART) algorithms (Breiman et al., 1984). We train a 3-layer CART regression tree on a large corpus drawn from the ShareGPT dataset, which was originally utilized for draft model training. From this corpus, we extract fully accepted draft paths to construct our training dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$, where $\mathbf{x}^{(i)} = \Phi(\mathcal{P}^{*(i)})$ represents the cumulative meta-path Top-K entropy and $y^{(i)}$ denotes the Terminal Confidence Rank for the path $\mathcal{P}^*$. For each potential split threshold s, the framework minimizes within-partition variance of success outcomes:

$$\mathcal{L}(s) = \sum_{j \in \{\text{left,right}\}} \frac{|D_j|}{|D|} \sum_{(\mathbf{x},y)\in D_j} (y - \mu_j)^2 \quad (2)$$

where D_j represents the resulting partitions and μ_j denotes the average rank in each partition.

Through recursive application of this optimization criterion, we construct a 3-layer decision tree yielding 8 non-overlapping entropy bins $\{\mathcal{B}_0, \mathcal{B}_1, \ldots, \mathcal{B}_7\}$. We concentrate optimization efforts on the low-entropy bins $\mathcal{B}_l = \bigcup_{i=0}^{2} \mathcal{B}_i$, which exhibit the highest potential for aggressive speculation strategies and align with Zipf's law observations about high-frequency pattern dominance in natural language.

The shallow decision tree achieves 2.4s training time on H20-3e GPU while maintaining computational efficiency. Once trained, the decision tree is automatically integrated into the system and **generalizes to all task types without task-specific retraining or fine-tuning**, enabling efficient and fully automated deployment. This data-driven approach eliminates manual hyperparameter tuning while ensuring theoretical consistency through empirical risk minimization. Further analysis of decision tree selection, configuration sensitivity, and cross-domain generalization (Natural Questions, SciQ, MedQA) is provided in Appendices A.3, A.4, and A.5.

4.3 Coordinated Adaptive Optimization

Having established complexity quantification and stratification, we implement three synergistic mechanisms that transform uniform speculation into adaptive resource allocation. Figure 2 presents an inference pipeline, highlighting the main differences between EAGLE-3 and HeteroSpec.

Dynamic Extended Drafting.: For contexts assigned to low-entropy bins, we extend spec-

ulation depth based on complexity assessment: $d'(\mathcal{B}_i) = d_{\text{base}} + \Delta(\mathcal{B}_i)$ where $\Delta(\mathcal{B}_i) = \alpha - i$ provides complexity-aware depth extension. Given the persistent prevalence of high-frequency and structurally simple patterns in natural language, when the current draft path falls into a low-entropy bin, subsequent speculative tokens are also likely to remain within low-entropy regions. In this scenario, increasing the speculative depth extends expected length of accepted segments, thereby reducing the total number of verifications required. Even in rare cases where the speculative extension departs from the low-entropy regime, the verification cost is only paid once for the entire segment, amortizing the computational overhead.

Top-N Pruning: Rather than uniformly processing all candidates, we dynamically adjust the verification set size: $N'(\mathcal{B}_i) = \gamma_i \cdot N_{\text{default}} + \Delta(\mathcal{B}_i)$ where γ_i implements conservative upper bounds ($\gamma_1 = 0.3, \gamma_2 = 0.6, \gamma_3 = 1.0$) for the low-entropy bins. This mapping avoids the brittleness and complexity of manual hyperparameter tuning by adopting a conservative upper-bound policy: by retaining a relatively large quantile of candidates for each bin, we reliably capture the most likely accepted branches while significantly reducing unnecessary verification on unlikely candidates.

Dynamic Computational Adaptation: The varying speculation configurations introduce per-iteration variations in computation graph structure. To maintain inference efficiency under such dynamic control flow, we employ just-in-time compilation to generate and cache specialized computation graphs $G(\mathcal{B}_i)$ for distinct complexity regions, enabling effective operator fusion and computational reuse. This ensures that high-throughput inference is preserved, even as speculative depth and candidate set size dynamically change during decoding.

Quantitative overhead analysis confirms that our approach introduces negligible computational and memory costs (see Appendix A.6).

5 Experiments

Tasks. To demonstrate the generality of HeteroSpec, we conduct a comprehensive evaluation without task-specific fine-tuning across five representative benchmarks spanning key task categories: multi-turn dialogue (MT Bench Zheng et al., 2023), code generation (HumanEval Chen et al., 2021), mathematical reasoning (GSM8K Cobbe et al., 2021), instruction following (Alpaca Taori et al., 2023), and summarization (CNN/Daily Mail Nallapati et al., 2016).

Models. We evaluate four representative LLMs spanning diverse model scales: Vicuna 13B (Chiang et al., 2023), LLaMA-Instruct 3.1 8B, LLaMA-Instruct 3.3 70B (Grattafiori et al., 2024), and DeepSeek-R1-Distill-LLaMA 8B (Guo et al., 2025). Experiments on 8B/13B models utilize $1\times$ NVIDIA H20-3e 141G GPU, while the 70B model requires $2\times$ H20-3e GPUs due to memory constraints. Additionally, experimental results on other GPU types (A800 80G, L40 40G) are provided in the Appendix A.7.

Metrics. HeteroSpec is intended to reduce the verification cost of the target model; therefore, we introduce two device-independent metrics: total validation calls and total verification tokens. HeteroSpec preserves the target model architecture and maintains strict acceptance conditions; therefore, generation quality evaluation is unnecessary.

- **Speedup Ratio**: Actual acceleration compared to vanilla autoregressive decoding.
- **Average Acceptance Length (τ)**: Mean tokens generated per drafting-verification cycle.
- **Total Validation Calls (Calls) / Total Verification Tokens(Tokens)**: The former counts target model invocations during decoding, while the latter quantifies cumulative tokens processed across all validation steps. Together, these metrics capture the computational overhead of candidate verification.

Baseline. We benchmark against state-of-the-art EAGLE-3 (Li et al., 2025a), adopting identical hyperparameters (Depth, Top-K, Top-N) from the official implementation. We also compare with alternative speculative decoding paradigms (Medusa, Hydra) and adaptive-length strategies (AdaEDL) in Appendix A.9 and A.10, demonstrating HeteroSpec's complementary advantages. Since this study does not involve training the draft model, and to avoid the confounding effect of randomness on the interpretation of our method's effectiveness, only the case of temperature=0 is considered by default throughout the following analysis. Detailed analysis of the temperature=1 case is provided in Appendix A.8.

		MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Mean	
Model	Method	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens
V 13B	EAGLE-3	3.22× 6257	6.56 327026	3.78× 3427	7.73 171598	3.31× 2421	6.40 123480	3.31× 2058	6.51 108192	2.87× 1975	6.54 101185	3.30× 3228	6.75 166296
	HeteroSpec	**3.53×** **5937**	**7.10** **278942**	**4.45×** **2948**	**9.05** **119753**	**3.42×** **2321**	**6.67** **109919**	**3.45×** **2011**	**6.70** **101778**	**3.04×** **1860**	**7.07** **86147**	**3.58×** **3015**	**7.32** **139308**
L31 8B	EAGLE-3	3.43× 8687	6.15 541384	3.95× 4118	6.83 248272	3.69× 2465	6.32 150568	3.79× 2563	6.84 161660	2.92× 4009	5.39 245204	3.56× 4368	6.31 269418
	HeteroSpec	**3.74×** **8357**	**6.44** **489985**	**4.43×** **3799**	**7.46** **196029**	**3.81×** **2379**	**6.53** **141487**	**3.99×** **2462**	**7.08** **151801**	**3.11×** **3878**	**5.47** **232705**	**3.82×** **4175**	**6.60** **242401**
L33 70B	EAGLE-3	5.33× 10099	5.64 499187	6.49× 4702	6.67 225788	5.95× 2372	6.22 115996	5.89× 2890	6.58 146264	4.36× 4345	5.01 211970	5.60× 4882	6.02 239841
	HeteroSpec	**5.38×** **9696**	**5.80** **444761**	**6.76×** **4478**	**7.02** **169249**	**6.06×** **2313**	**6.40** **103342**	**6.02×** **2829**	**6.82** **134361**	**4.43×** **4340**	**5.04** **204626**	**5.73×** **4731**	**6.22** **211268**
DSL 8B	EAGLE-3	3.50× 6965	5.88 425803	3.87× 6331	6.58 387689	4.12× 4809	7.09 295413	3.25× 7086	5.62 434004	2.82× 8139	5.02 499494	3.51× 6666	6.04 408481
	HeteroSpec	**3.78×** **6787**	**6.12** **398080**	**4.19×** **6135**	**6.82** **356517**	**4.64×** **4374**	**7.89** **226139**	**3.56×** **6962**	**5.75** **418119**	**2.99×** **8085**	**5.07** **487250**	**3.83×** **6469**	**6.33** **377221**

Table 1: Speedup ratios, average acceptance lengths (τ), total validation calls (Calls), and total verification tokens (Tokens) of different methods. V represents Vicuna, L31 represents LLaMA-Instruct 3.1, L33 represents LLaMA-Instruct 3.3, and DSL represents DeepSeek-R1-Distill-LLaMA.

5.1 Effectiveness

Table 1 demonstrates that HeteroSpec consistently outperforms EAGLE-3 across all evaluated datasets and LLMs on four key metrics: speedup ratio, average acceptance length, total validation calls, and total verification tokens. Our method achieves an average 4.24× speedup over autoregressive decoding with an average acceptance length of 6.62.

In the code generation task (HumanEval), HeteroSpec demonstrates the most significant performance gains. This superiority arises from the high predictability of predominantly templated code structures, with EAGLE-3 typically exhibiting high confidence and being assigned to low-entropy bins. Through dynamic extended drafting, HeteroSpec facilitates extended token sequence acceptance, achieving a 5.65% reduction in target model verifications. Additionally, Top-N pruning yields a 22.79% decrease in verification tokens while preserving generation accuracy.

For the summarization task (CNN/DM), increased output diversity and unpredictability diminish draft-target model alignment, leading to shorter average accepted lengths for EAGLE-3, with more frequent assignments to high-entropy bins. Consequently, HeteroSpec's performance gains in these tasks are comparatively modest, and its behavior partially converges towards EAGLE-3. Nevertheless, these results demonstrate HeteroSpec's robust adaptability and effectiveness across diverse models and tasks, consistently reducing target model validation overhead.

HeteroSpec maintains orthogonality to existing draft model optimization techniques, achieving correspondingly greater performance gains as draft models improve without any additional cost. EAGLE-3 further enhances mathematical reasoning capability by additionally training the DeepSeekR1-Distill-LLaMA 8B model on the OpenThoughts114k-math dataset, resulting in even more pronounced improvements for HeteroSpec on GSM8K—demonstrating this scaling effect.

5.2 Ablation Study

To systematically assess each optimization, we conducted ablation experiments progressively integrating our three adaptive mechanisms upon EAGLE-3. Table 2 demonstrates that Dynamic Extended Drafting yields the most substantial performance gains. Its underlying mechanism is that, when cumulative meta-path Top-K entropy is low, the probability that an initially drafted token is accepted at all layers increases significantly. This enables subsequent tokens' verification costs to be amortized together with the initial draft tokens, regardless of whether these subsequent tokens fall into low-entropy bins. Given the prevalence of simple patterns in natural language, subsequent tokens typically remain within low-entropy regions, further amplifying this cost-sharing benefit. This mechanism increases average accepted length and reduces total verifications by 11.09%, substantially alleviating the verification bottleneck.

Top-N Pruning achieves a 19.85% reduction in verification tokens with minimal impact on ac-

Method	LLaMA3.1-8B				Vicuna-13B				LLaMA3.3-70B			
	Speedup	τ	Calls	Tokens	Speedup	τ	Calls	Tokens	Speedup	τ	Calls	Tokens
EAGLE-3	3.95×	6.83	4118	248272	3.78×	7.73	3427	171598	6.49×	6.67	4702	225788
+DED	4.35×	7.54	3757	226796	4.38×	9.13	2900	144893	6.71×	7.18	4367	209620
+TNP	4.31×	7.46	3799	196029	4.36×	9.05	2948	119753	6.68×	7.02	4428	169249
+DCA	4.43×	7.46	3799	196029	4.45×	9.05	2948	119753	6.76×	7.02	4428	169249

Table 2: Ablation study on HumanEval dataset across three models of different sizes. DED (Dynamic Extended Drafting), TNP (Top-N Pruning), and DCA (Dynamic Computational Adaptation) are incrementally integrated; each row represents the ablation result after adding the corresponding module to the previous configuration.

cepted length (0.11 decrease). This demonstrates that the strategy can select, at key decision points, the most likely-to-be-accepted critical paths dynamically and at minimal cost. Notably, although there is a slight 0.59% decrease in the speedup metric, this reflects GPU underutilization in single-batch H20-3e scenarios rather than algorithmic limitations; In the discussion of complex multi-batch scenarios (see Section A.11), this strategy reveals greater potential for acceleration. Dynamic Computational Adaptation mitigates performance degradation from computational graph variations introduced by the preceding optimizations. These three mechanisms collectively demonstrate the synergistic effectiveness of our contextual heterogeneity exploitation framework.

5.3 Hyperparameter Study

To investigate the hyperparameter α in Dynamic Extended Drafting, we conduct experiments with varying α values (see Table 3). We find that the speedup is maximized when α is set to $\lceil depth/2 \rceil$, while both increasing or decreasing α leads to reduced acceleration. In essence, this hyperparameter balances the additional drafting overhead against the reduction in verification cost. If α is too low, the draft model underutilizes its capacity in low-entropy regions, missing opportunities to accept longer token sequences. Conversely, if α is too high, the additional drafting overhead outweighs verification savings, leading to degraded performance. These results indicate that optimal α scales with draft model capability—as models improve, a larger α can yield greater speedups.

6 Related Work

Speculative decoding (Chen et al., 2023; Leviathan et al., 2023) adopts a "drafting with lightweight model, verification with original model" paradigm for lossless acceleration. SpecInfer (Miao et al., 2024) introduced tree-based attention for efficient parallel verification. REST (He et al., 2024) and LLMA (Yang et al., 2023) employ retrieval-based drafting. Lookahead Decoding (Fu et al., 2024) accelerates inference via parallel n-gram generation. GLIDE (Du et al., 2024) and MoA (Zimmer et al., 2025) reuse target model KV cache. Medusa (Cai et al., 2024), Hydra (Ankner et al., 2024), EAGLE (Li et al., 2025b), and EAGLE-3 (Li et al., 2025a) reuse target model feature representations. Other approaches (Zhang et al., 2024a; Yi et al., 2024; Elhoushi et al., 2024; Liu et al., 2024a; Sun et al., 2024; Svirschevski et al., 2024) reuse partial target model weights. These approaches focus on training stronger draft models, which is orthogonal to our method and enables greater gains as draft performance improves.

Another research line explores dynamic draft structures. EAGLE-2 (Li et al., 2024) uses dynamic drafting trees with joint probability confidence. BiLD (Kim et al., 2023), Kangaroo (Liu et al., 2024a), DDD (Brown et al., 2024), and SVIP (Zhang et al., 2024c) introduce stopping metrics. SpecDec++ (Huang et al., 2024) and AdaEAGLE (Zhang et al., 2024b) train modules for early stopping or draft length prediction. C2T (Huo et al., 2025) proposes classifier-based tree to correct joint probability bias. However, these methods inadequately exploit linguistic heterogeneity for verification optimization, limiting performance, applicability, and timeliness. Our method adaptively optimizes computational resource allocation, achieving superior performance over existing approaches.

7 Conclusion

Based on the challenges posed by the heterogeneous computational requirements arising from natural language's varying predictability, this work presents HeteroSpec, a complexity-aware speculative decoding framework that addresses the verification bottleneck through adaptive resource allocation. Through cumulative meta-path Top-K

α	LLaMA3.1-8B				Vicuna-13B				LLaMA3.3-70B			
	Speedup	τ	Calls	Tokens	Speedup	τ	Calls	Tokens	Speedup	τ	Calls	Tokens
EAGLE-3	3.95×	6.83	4118	248272	3.78×	7.73	3427	171598	6.49×	6.67	4702	225788
$\lceil depth/2 \rceil - 1$	4.36×	7.37	3846	198486	4.35×	8.71	3049	126859	6.68×	7.00	4482	169351
$\lceil depth/2 \rceil$	4.43×	7.46	3799	196029	4.45×	9.05	2948	119753	6.76×	7.02	4478	169249
$\lceil depth/2 \rceil + 1$	4.38×	7.51	3778	195053	4.44×	9.23	2906	118657	6.71×	7.01	4473	169151

Table 3: Hyperparameter study (α) on HumanEval dataset across three models of different sizes.

entropy quantification and data-driven stratification, HeteroSpec transforms uniform verification into adaptive optimization that dynamically allocates resources to high-confidence candidates. Extensive evaluation demonstrates that HeteroSpec achieves greater speedups than the state-of-the-art EAGLE-3, with negligible computational overhead, establishing the value of leveraging linguistic heterogeneity for efficient LLM inference. Future work is discussed in Appendix A.12.

Limitations

A limitation of HeteroSpec is its current implementation within the state-of-the-art EAGLE framework, necessitating validation for generalizability to other speculative decoding methods. However, the core principle of dynamic resource allocation guided by linguistic heterogeneity is fundamentally orthogonal and broadly applicable.

References

Sudhanshu Agrawal, Wonseok Jeon, and Mingu Lee. 2024. Adaedl: Early draft stopping for speculative decoding of large language models via an entropy-based lower bound on token acceptance probability. *Preprint*, arXiv:2410.18351.

Zachary Ankner, Rishab Parthasarathy, Aniruddha Nrusimha, Christopher Rinard, Jonathan Ragan-Kelley, and William Brandon. 2024. Hydra: Sequentially-dependent draft heads for medusa decoding. *Preprint*, arXiv:2402.05109.

L. Breiman, Jerome H. Friedman, Richard A. Olshen, and C. J. Stone. 1984. Classification and regression trees.

Oscar Brown, Zhengjie Wang, Andrea Do, Nikhil Mathew, and Cheng Yu. 2024. Dynamic depth decoding: Faster speculative decoding for llms. *Preprint*, arXiv:2409.00142.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Marc Brysbaert. 2019. How many words do we read per minute? a review and meta-analysis of reading rate. *Journal of Memory and Language*, 109:104047.

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. *Preprint*, arXiv:2401.10774.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *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, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

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

Cunxiao Du, Jing Jiang, Xu Yuanchen, Jiawei Wu, Sicheng Yu, Yongqi Li, Shenggui Li, Kai Xu, Liqiang Nie, Zhaopeng Tu, and Yang You. 2024. Glide with a cape: A low-hassle method to accelerate speculative decoding. *Preprint*, arXiv:2402.02082.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, Ahmed Aly, Beidi Chen, and Carole-Jean

Wu. 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)*, page 12622–12642. Association for Computational Linguistics.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. 2024. Break the sequential dependency of llm inference using lookahead decoding. *Preprint*, arXiv:2402.02057.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Zhenyu He, Zexuan Zhong, Tianle Cai, Jason D. Lee, and Di He. 2024. Rest: Retrieval-based speculative decoding. *Preprint*, arXiv:2311.08252.

Kaixuan Huang, Xudong Guo, and Mengdi Wang. 2024. Specdec++: Boosting speculative decoding via adaptive candidate lengths. *Preprint*, arXiv:2405.19715.

Feiye Huo, Jianchao Tan, Kefeng Zhang, Xunliang Cai, and Shengli Sun. 2025. C2t: A classifier-based tree construction method in speculative decoding. *Preprint*, arXiv:2502.13652.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2020. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Preprint*, arXiv:2009.13081.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah A. Smith. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. *Preprint*, arXiv:2006.10369.

Sehoon Kim, Karttikeya Mangalam, Suhong Moon, Jitendra Malik, Michael W. Mahoney, Amir Gholami, and Kurt Keutzer. 2023. Speculative decoding with big little decoder. *Preprint*, arXiv:2302.07863.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, and 1 others. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. *Preprint*, arXiv:2211.17192.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024. Eagle-2: Faster inference of language models with dynamic draft trees. *Preprint*, arXiv:2406.16858.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2025a. Eagle-3: Scaling up inference acceleration of large language models via training-time test. *Preprint*, arXiv:2503.01840.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2025b. Eagle: Speculative sampling requires rethinking feature uncertainty. *Preprint*, arXiv:2401.15077.

Zikun Li, Zhuofu Chen, Remi Delacourt, Gabriele Oliaro, Zeyu Wang, Qinghan Chen, Shuhuai Lin, April Yang, Zhihao Zhang, Zhuoming Chen, Sean Lai, Xupeng Miao, and Zhihao Jia. 2025c. Adaserve: Slo-customized llm serving with fine-grained speculative decoding. *Preprint*, arXiv:2501.12162.

Chaofan Lin, Zhenhua Han, Chengruidong Zhang, Yuqing Yang, Fan Yang, Chen Chen, and Lili Qiu. 2024. Parrot: Efficient serving of llm-based applications with semantic variable. *Preprint*, arXiv:2405.19888.

Fangcheng Liu, Yehui Tang, Zhenhua Liu, Yunsheng Ni, Kai Han, and Yunhe Wang. 2024a. Kangaroo: Lossless self-speculative decoding via double early exiting. *Preprint*, arXiv:2404.18911.

Xiaoxuan Liu, Cade Daniel, Langxiang Hu, Woosuk Kwon, Zhuohan Li, Xiangxi Mo, Alvin Cheung, Zhijie Deng, Ion Stoica, and Hao Zhang. 2024b. Optimizing speculative decoding for serving large language models using goodput. *Preprint*, arXiv:2406.14066.

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. 2024. Specinfer: Accelerating large language model serving with tree-based speculative inference and verification. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, ASPLOS '24, page 932–949. ACM.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *Preprint*, arXiv:1602.06023.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Ranajoy Sadhukhan, Jian Chen, Zhuoming Chen, Vashisth Tiwari, Ruihang Lai, Jinyuan Shi, Ian En-Hsu Yen, Avner May, Tianqi Chen, and Beidi Chen. 2025. Magicdec: Breaking the latency-throughput tradeoff for long context generation with speculative decoding. *Preprint*, arXiv:2408.11049.

Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *Preprint*, arXiv:1911.02150.

Hanshi Sun, Zhuoming Chen, Xinyu Yang, Yuandong Tian, and Beidi Chen. 2024. Triforce: Lossless acceleration of long sequence generation with hierarchical speculative decoding. *Preprint*, arXiv:2404.11912.

Ruslan Svirschevski, Avner May, Zhuoming Chen, Beidi Chen, Zhihao Jia, and Max Ryabinin. 2024. Specexec: Massively parallel speculative decoding for interactive llm inference on consumer devices. *Preprint*, arXiv:2406.02532.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. *Preprint*, arXiv:1707.06209.

Nan Yang, Tao Ge, Liang Wang, Binxing Jiao, Daxin Jiang, Linjun Yang, Rangan Majumder, and Furu Wei. 2023. Inference with reference: Lossless acceleration of large language models. *Preprint*, arXiv:2304.04487.

Hanling Yi, Feng Lin, Hongbin Li, Peiyang Ning, Xiaotian Yu, and Rong Xiao. 2024. Generation meets verification: Accelerating large language model inference with smart parallel auto-correct decoding. *Preprint*, arXiv:2402.11809.

Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. 2024a. 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)*, page 11263–11282. Association for Computational Linguistics.

Situo Zhang, Hankun Wang, Da Ma, Zichen Zhu, Lu Chen, Kunyao Lan, and Kai Yu. 2024b. Adaeagle: Optimizing speculative decoding via explicit modeling of adaptive draft structures. *Preprint*, arXiv:2412.18910.

Ziyin Zhang, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Rui Wang, and Zhaopeng Tu. 2024c. Draft model knows when to stop: A self-verification length policy for speculative decoding. *Preprint*, arXiv:2411.18462.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. 2024. Distserve: Disaggregating prefill and decoding for goodput-optimized large language model serving. *Preprint*, arXiv:2401.09670.

Matthieu Zimmer, Milan Gritta, Gerasimos Lampouras, Haitham Bou Ammar, and Jun Wang. 2025. Mixture of attentions for speculative decoding. *Preprint*, arXiv:2410.03804.

George Kingsley Zipf. 1949. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, Reading, MA.

A Appendix

A.1 Additional Experiments on Heterogeneity

To further validate the heterogeneity observations presented in Section 3, we conduct comprehensive experiments across multiple models and datasets. Figure 3 presents Terminal Confidence Rank (TCR) analysis results for five additional model-dataset combinations: LLaMA-Instruct 3.1-8B on HumanEval, Vicuna-13B and LLaMA-Instruct 3.3-70B on both MT Bench and HumanEval.

The left panels show that Terminal Confidence Rank is heavily concentrated among the top 25% of Top-N candidates across all model-dataset combinations. Furthermore, the right panels show a strong correlation between lower TCRs and longer average accepted lengths, often approaching the maximum draft depth. These findings demonstrate remarkable consistency in heterogeneity patterns across different model scales (from 8B to 70B parameters) and task types, confirming that a small fraction of high-confidence candidates drives the majority of successful speculation. This indicates that heterogeneity in draft acceptance is an intrinsic property of speculative decoding rather than an artifact of specific model configurations.

Interestingly, the heterogeneity effect appears more pronounced in code generation tasks (HumanEval) compared to dialogue tasks (MT Bench). This aligns with the structured and predictable nature of code syntax patterns, which enables draft models to achieve higher confidence and longer acceptance sequences for well-formed code structures. These comprehensive results establish that the heterogeneity phenomenon identified in our main analysis is not limited to specific model-dataset combinations but represents a fundamental characteristic of speculative decoding across diverse settings. This broad validation strengthens the motivation for our heterogeneity-adaptive optimization approach and confirms that the observed patterns provide a robust foundation for the HeteroSpec framework.

A.2 Contextual Complexity Oracle Validation

To empirically validate the effectiveness of the proposed Cumulative Meta-Path Top-K Entropy as a contextual complexity oracle, we conduct extensive experiments on three models: LLaMA-Instruct 3.1 8B, Vicuna 13B, and LLaMA-Instruct 3.3 70B using two representative datasets, MT Bench and CNN/Daily Mail. Results are shown in Figure 4, which plots average acceptance length and average Cumulative Meta-Path Top-K Entropy with respect to Terminal Confidence Rank (TCR).

We find that low-entropy regions, corresponding to simple and predictable contexts identified by our metric, consistently yield substantially longer average acceptance lengths. This demonstrates that the metric effectively isolates high-confidence contexts amenable to aggressive speculation. In contrast, higher entropy is associated with much greater variability in acceptance length, highlighting the unpredictability of complex contexts.

Notably, as illustrated in Figure 1(a), high-confidence, low-complexity contexts make up the majority of cases, mirroring the dominance of simple linguistic patterns in natural language. By providing a reliable and actionable measure of contextual complexity, our metric reveals significant optimization potential within these frequent low-entropy regions. Accordingly, our adaptive optimization framework is specifically designed to prioritize and exploit these regions for maximal inference acceleration.

A.3 Rationale for Adopting CART Decision Trees for Complexity Stratification

To systematically partition the contextual complexity space, we adopt a data-driven CART decision tree framework in lieu of neural networks or other continuous models. This choice is theoretically motivated by both the structural properties uncovered by our contextual complexity oracle and the operational requirements of efficient speculative decoding.

Comprehensive experimental analysis of the cumulative meta-path Top-K entropy metric (see Appendix A.2) reveals a distinct asymmetry in the complexity landscape: low-entropy regions correspond to stable, highly predictable contexts with consistently high draft acceptance, while high-entropy regions lack discernible structure and exhibit wide variability. Such step-like, non-smooth transitions in system behavior are well-suited to models with strong capacity for learning discrete partition boundaries. Decision trees, particularly the CART algorithm, are able to capture these sharp entropy thresholds in a principled, data-driven manner, directly optimizing for empirical risk minimization on partitioned complexity outcomes. In contrast, neural networks are fundamentally biased toward smooth function approximation, which hinders their ability to sharply distinguish abrupt

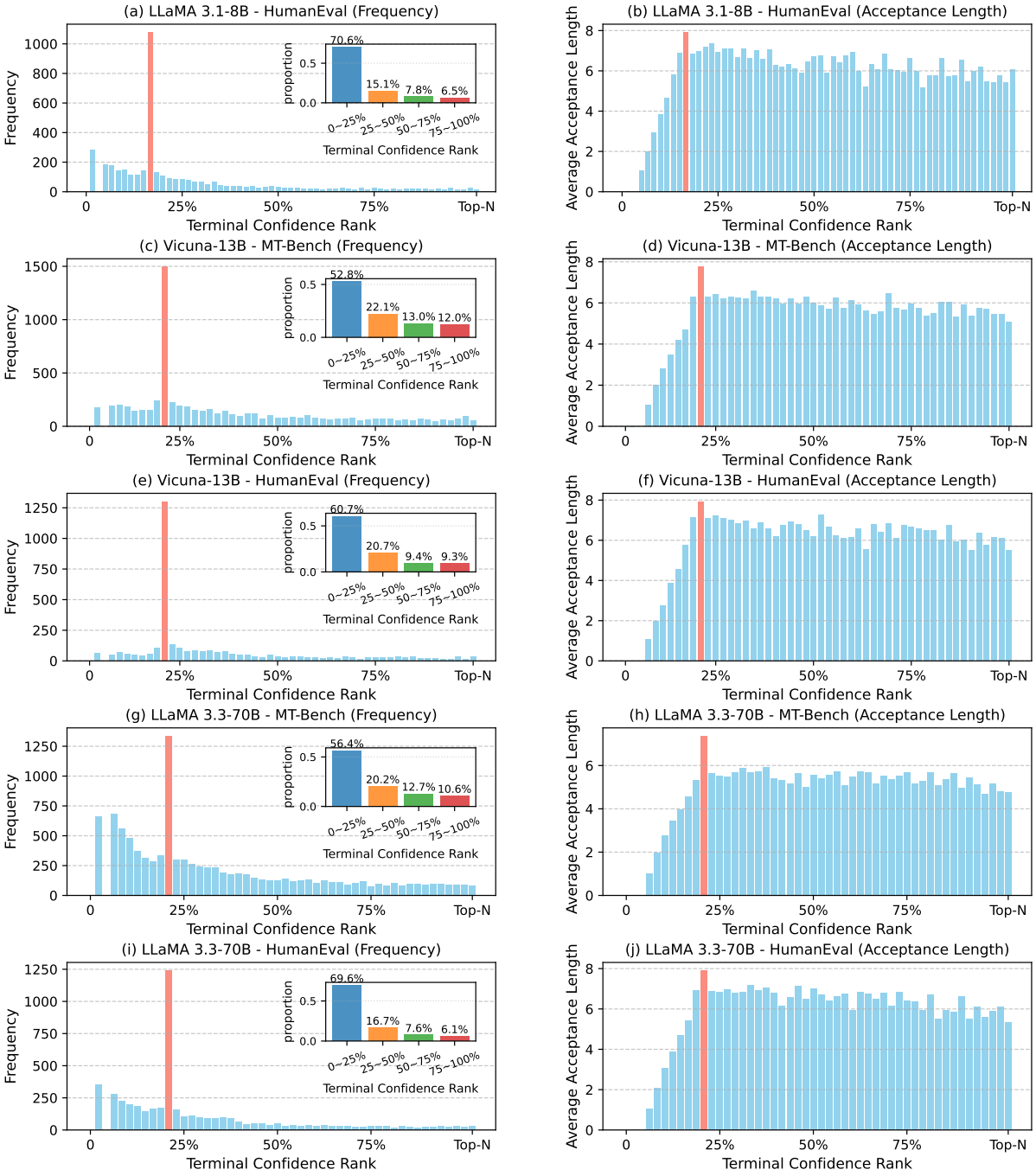

Figure 3: Terminal Confidence Rank analysis across multiple models and datasets. Left panels: Distribution of Terminal Confidence Rank within Top-N draft candidates (bar chart, with prominent values highlighted in orange), with inset quantile analysis showing concentration within top percentiles. Right panels: Correlation between Average Acceptance Length and Terminal Confidence Rank (bar chart). (a,b) LLaMA 3.1-8B on HumanEval; (c,d) Vicuna-13B on MT-Bench; (e,f) Vicuna-13B on HumanEval; (g,h) LLaMA 3.3-70B on MT-Bench; (i,j) LLaMA 3.3-70B on HumanEval.

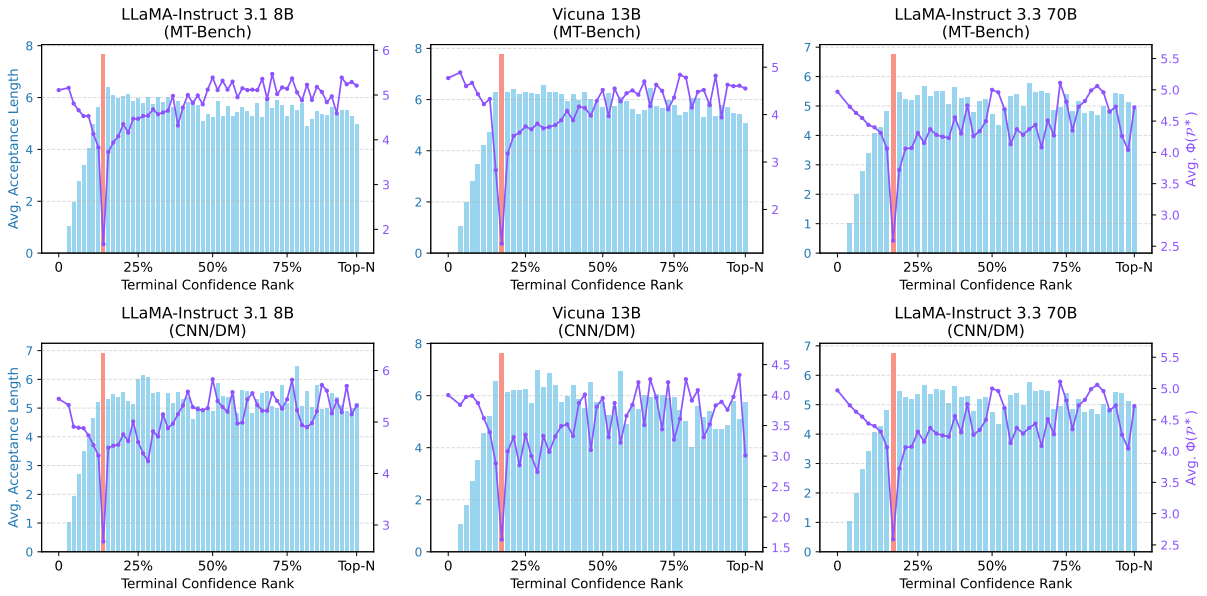

Figure 4: Validation results of the proposed Cumulative Meta-Path Top-K Entropy metric across LLaMA 8B, Vicuna 13B, and LLaMA 70B models on MT-Bench and CNN/Daily Mail datasets. For each subplot, the x-axis shows Terminal Confidence Rank (TCR), the left y-axis indicates average acceptance length (bar chart, with prominent values highlighted in orange), and the right y-axis shows average Cumulative Meta-Path Top-K Entropy (line plot).

changes or encode actionable thresholding policies.

From a systems perspective, most state-of-the-art draft models prioritize minimal inference latency and computational cost—typically employing a single-layer transformer for maximal efficiency. Introducing an additional neural network for online complexity partitioning would incur significant overhead, contradicting the primary goal of fast speculative decoding. In comparison, evaluating a shallow decision tree imposes only a few conditional checks per draft step, resulting in negligible runtime cost. As evidenced by our quantitative analysis in Appendix A.6, this approach introduces virtually no latency overhead in practice. Moreover, decision tree training itself is highly efficient; constructing a shallow CART tree typically completes within seconds.

A.4 Entropy Stratification Granularity Analysis

To validate the robustness of our data-driven entropy stratification module, we investigate the consistency of low-entropy region identification across different stratification granularities. We employ the **Jaccard similarity coefficient** as our primary evaluation metric, defined as:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3}$$

where A and B represent the sets of samples identified as low-entropy by different stratification depths. Using our 3-layer stratification as the reference baseline, we examine consistency with deeper architectures by maintaining proportional low-entropy region selection across all granularities. Specifically, we compute $J(\mathcal{S}_{3L}, \mathcal{S}_{dL})$ for $d \in \{4, 5, 6\}$, where $\mathcal{S}_{dL}$ denotes the low-entropy sample set identified by d-layer stratification with equivalent proportional selection criteria. This coefficient ranges from 0 (no overlap) to 1 (perfect agreement), providing a quantitative measure of consistency across granularity variations.

Model	3L-4L Overlap	3L-5L Overlap	3L-6L Overlap
L31 8B	1.00	0.89	0.98
V 13B	0.96	0.97	1.00
L33 70B	1.00	0.98	0.96

Table 4: Low-entropy region overlap consistency across stratification granularities

As shown in Table 4, our analysis reveals remarkably high overlap consistency across all model configurations, with Jaccard coefficients consistently exceeding 0.89. Notably, multiple configurations achieve perfect agreement (coefficient = 1.00), demonstrating that different stratification granularities often converge to identical low-entropy region identification. This consistency indicates that critical complexity boundaries are intrinsic to model be-

havioral patterns rather than stratification artifacts, with deeper architectures providing minimal additional discriminative information for low-entropy detection.

These findings establish the granularity insensitivity of our entropy stratification framework, demonstrating three key properties: (1) Structural Stability: our method captures fundamental complexity patterns that persist across granularity variations; (2) Computational Efficiency: deeper stratification provides negligible gains while increasing overhead, demonstrating that our 3-layer design achieves an effective trade-off between accuracy and computational cost; (3) Universal Applicability: consistent patterns across model scales confirm broad applicability of our approach. This empirical evidence strongly justifies our architectural choice and establishes the robustness of complexity-aware speculative decoding.

A.5 Out-of-Domain Evaluation

To validate the cross-domain generalization of our CART decision tree trained on ShareGPT, we conducted evaluations on three out-of-domain datasets that introduce specialized terminology and knowledge contexts largely absent from the training data. All experiments were conducted on NVIDIA H20-3e GPUs under the same configuration as our main experiments.

- **RAG (Natural Questions)** (Kwiatkowski et al., 2019): Real user queries requiring factual knowledge retrieval
- **SciQ** (Welbl et al., 2017): Science exam questions with domain-specific scientific reasoning
- **MedQA** (Jin et al., 2020): USMLE medical exam questions with highly specialized clinical terminology

HeteroSpec maintains consistent improvements (5.3%-10.8%) across all out-of-domain scenarios, including the challenging MedQA with specialized medical vocabulary. These results provide strong evidence that the ShareGPT-trained CART has learned domain-agnostic complexity boundaries that generalize to unseen domains without task-specific fine-tuning.

A.6 Runtime Overhead Profiling

To quantitatively assess the computational cost of our proposed approach, we conduct comprehensive runtime profiling experiments across MT Bench and HumanEval datasets. Our analysis decomposes total inference time into three distinct components: drafting overhead, verification overhead, and additional computational cost introduced by HeteroSpec's complexity assessment and adaptive optimization mechanisms. We evaluate this decomposition across different model scales to capture performance characteristics comprehensively.

The results in Table 6 demonstrate that HeteroSpec introduces minimal additional overhead, ranging from 0.88 to 3.1 seconds across all configurations. This represents only 0.26% to 0.87% of total inference time, confirming the efficiency of our lightweight complexity quantification framework. Runtime memory usage remains essentially identical across both methods, demonstrating that our approach does not introduce significant memory overhead. While drafting overhead increases modestly due to dynamic depth extension, this is substantially offset by significant verification overhead reductions of 4.6% to 15.9%. Since verification constitutes 67.4% to 89.0% of total runtime—the primary computational bottleneck in speculative decoding—these reductions drive meaningful system-level gains.

Our profiling validates that HeteroSpec achieves performance improvements through targeted optimization with negligible additional overhead, confirming the efficiency of our data-driven complexity assessment and adaptive resource allocation, establishing that contextual heterogeneity can be effectively leveraged for speculative decoding acceleration.

A.7 More results on different GPUs

Since speedup ratio is a hardware-dependent metric, we conducted comprehensive experiments on A800 80G and L40 40G GPUs, beyond the H20-3e 141G GPU, to demonstrate HeteroSpec's excellent portability and hardware compatibility. As shown in Table 7 and Table 8, HeteroSpec maintains consistent performance advantages across all testing platforms, comprehensively outperforming the state-of-the-art EAGLE-3 across key metrics, demonstrating superior hardware generalization capabilities and cross-platform stability.

A.8 Discussion on Temperature = 1

For the temperature=1 setting, we adopt the same experimental configuration and conduct experiments on Vicuna 13B, LLaMA-Instruct 3.1

Model	Method	RAG	SciQ	MedQA
L31 8B	EAGLE-3	3.72× / 5.89	3.21× / 5.58	2.78× / 5.13
	HeteroSpec	**4.01× / 6.31**	**3.38× / 5.84**	**3.05× / 5.35**
	Improvement	+7.8% / +7.1%	+5.3% / +4.7%	+9.7% / +4.3%
V 13B	EAGLE-3	2.94× / 6.33	2.39× / 5.73	2.41× / 5.79
	HeteroSpec	**3.17× / 6.74**	**2.53× / 6.03**	**2.67× / 6.32**
	Improvement	+7.8% / +6.5%	+5.9% / +5.2%	+10.8% / +9.2%

Table 5: Out-of-domain evaluation results. Format: Speedup / Average Acceptance Length (τ)

MT Bench						
Model	Method	Draft	Verify	Add.	Total	Mem
L31 8B	EAGLE-3	98.1	207.5	-	305.6	18.9
	HeteroSpec	103.1	194.2	2.3	299.6	18.9
V 13B	EAGLE-3	79.2	207.9	-	287.1	29.8
	HeteroSpec	83.0	190.5	1.8	275.3	29.8
L33 70B	EAGLE-3	133.3	1080.7	-	1214.0	143.8
	HeteroSpec	140.3	1030.6	3.1	1174.0	143.8
HumanEval						
Model	Method	Draft	Verify	Add.	Total	Mem
L31 8B	EAGLE-3	45.3	93.7	-	139.0	18.8
	HeteroSpec	49.5	84.7	1.2	135.4	18.8
V 13B	EAGLE-3	41.7	108.7	-	150.4	29.8
	HeteroSpec	43.7	91.4	0.9	136.0	29.8
L33 70B	EAGLE-3	60.0	477.8	-	537.8	142.6
	HeteroSpec	68.9	434.2	1.3	504.4	142.6

Table 6: Runtime overhead and memory profiling across benchmarks and models. Time values are reported in seconds (s) as cumulative totals across all test cases in each dataset. Memory usage is measured in GB and represents average runtime memory. Draft, Verify, and Add. represent drafting time, verification time, and additional overhead introduced by HeteroSpec's complexity assessment and adaptive optimization, respectively. V represents Vicuna, L31 represents LLaMA-Instruct 3.1, L33 represents LLaMA-Instruct 3.3.

8B, LLaMA-Instruct 3.3 70B, and DeepSeek-R1-Distill-LLaMA. For the 8B/13B models, we employ a NVIDIA H20-3e 141G GPU. For the 70B model, we use two H20-3e GPUs due to memory limitations.

Under temperature=1, the flatter output distribution makes it significantly more challenging for the draft model to accurately predict the target model's stochastic choices, typically resulting in reduced acceleration gains from speculative decoding. Although more decoding instances fall into high-entropy bins, HeteroSpec is still capable of dynamically optimizing speculative extension and re-pruning strategies based on "decoding difficulty," thereby maximizing the accepted token length per expensive target model invocation. As demonstrated in Table 9, HeteroSpec consistently outperforms the state-of-the-art EAGLE-3 across key performance metrics, including speedup ratio, acceptance length, and verification overhead, showcasing remarkable robustness.

A.9 Comparison with Alternative Speculative Decoding Paradigms

Medusa and Hydra represent a different technical paradigm for speculative decoding, attaching multiple prediction heads to the target model for parallel token generation. In contrast, the EAGLE family employs autoregressive drafting with specialized lightweight draft models. We compare these paradigms using Vicuna-13B on NVIDIA H20-3e GPUs:

EAGLE-3's autoregressive draft model achieves 1.4-2.0× higher speedup than Medusa/Hydra by leveraging multi-layer feature aggregation for better draft-target alignment. HeteroSpec further improves upon EAGLE-3 with up to 17.7% additional gains through adaptive verification resource allocation—an optimization orthogonal to draft model architecture.

A.10 Comparison with Adaptive-Length Strategies (AdaEDL)

AdaEDL (Agrawal et al., 2024) employs entropy thresholds to trigger early stopping during drafting. While sharing the intuition of leveraging entropy, HeteroSpec differs fundamentally in optimization objective: AdaEDL minimizes *drafting cost* (the cheap component), whereas HeteroSpec minimizes *verification cost* (67-90% of total runtime). We implemented AdaEDL's early stopping logic on the EAGLE-3 backbone and conducted experiments on NVIDIA H20-3e GPUs for direct comparison:

		MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Mean	
Model	Method	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens
V 13B	EAGLE-3	4.01× 6267	6.55 327516	4.78× 3424	7.74 171451	3.95× 2433	6.40 124068	3.99× 2064	6.50 108486	3.60× 1983	6.55 101577	4.07× 3234	6.75 166620
	HeteroSpec	**4.32×** **5925**	**7.09** **281774**	**5.41×** **2971**	**9.04** **119797**	**4.15×** **2322**	**6.67** **110137**	**4.14×** **2013**	**6.70** **101910**	**3.81×** **1840**	**7.05** **84932**	**4.37×** **3014**	**7.31** **139710**
L31 8B	EAGLE-3	3.83× 8704	6.16 542387	4.47× 4124	6.83 248626	3.92× 2477	6.34 151276	4.19× 2573	6.83 162604	3.18× 3973	5.39 243080	3.92× 4370	6.31 269595
	HeteroSpec	**4.04×** **8319**	**6.47** **485731**	**4.75×** **3845**	**7.36** **198391**	**4.11×** **2380**	**6.53** **141658**	**4.39×** **2502**	**7.04** **153617**	**3.37×** **3897**	**5.45** **233771**	**4.13×** **4189**	**6.57** **242634**
L33 70B	EAGLE-3	4.24× 10017	5.66 494910	5.10× 4643	6.69 223015	4.70× 2367	6.23 115761	4.80× 2892	6.58 146358	3.61× 4392	5.01 214179	4.49× 4862	6.03 238845
	HeteroSpec	**4.35×** **9747**	**5.80** **447511**	**5.39×** **4449**	**7.04** **168115**	**4.81×** **2320**	**6.39** **103464**	**4.88×** **2838**	**6.76** **134851**	**3.66×** **4362**	**5.03** **205676**	**4.62×** **4743**	**6.20** **211923**
DSL 8B	EAGLE-3	3.67× 7013	5.85 428458	4.12× 6353	6.57 388987	4.67× 4814	7.09 295708	3.47× 7084	5.64 433001	3.09× 8118	5.02 498255	3.80× 6676	6.03 408882
	HeteroSpec	**3.88×** **6797**	**6.12** **398972**	**4.41×** **6132**	**6.82** **356129**	**5.08×** **4381**	**7.89** **228799**	**3.71×** **6924**	**5.75** **415689**	**3.25×** **8079**	**5.06** **485283**	**4.07×** **6463**	**6.33** **376974**

Table 7: Speedup ratios, average acceptance lengths (τ), total validation calls (Calls), and total verification tokens (Tokens) of different methods on A800 80G GPUs. V represents Vicuna, L31 represents LLaMA-Instruct 3.1, L33 represents LLaMA-Instruct 3.3, and DSL represents DeepSeek-R1-Distill-LLaMA.

		MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Mean	
Model	Method	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens
V 13B	EAGLE-3	2.19× 6312	6.53 329721	2.56× 3428	7.74 171647	2.15× 2432	6.39 124019	2.14× 2077	6.54 109123	1.96× 1980	6.54 101430	2.20× 3246	6.75 167188
	HeteroSpec	**2.45×** **5922**	**7.07** **278261**	**2.94×** **2988**	**9.05** **119741**	**2.31×** **2327**	**6.68** **111170**	**2.25×** **2012**	**6.70** **102442**	**2.14×** **1877**	**7.05** **87072**	**2.42×** **3025**	**7.31** **139737**
L31 8B	EAGLE-3	2.67× 8687	6.15 541384	3.06× 4129	6.83 249275	2.85× 2446	6.34 149447	2.97× 2586	6.84 163017	2.20× 3996	5.38 244437	2.75× 4369	6.31 269512
	HeteroSpec	**2.79×** **8344**	**6.48** **488135**	**3.31×** **3832**	**7.46** **198173**	**2.92×** **2386**	**6.54** **141866**	**3.16×** **2492**	**7.08** **153535**	**2.31×** **3907**	**5.49** **234268**	**2.90×** **4192**	**6.61** **243195**
DSL 8B	EAGLE-3	2.65× 6994	5.88 426983	2.97× 6327	6.59 387276	3.18× 4810	7.09 295472	2.54× 7066	5.62 432824	2.15× 8133	5.03 499140	2.70× 6666	6.04 408339
	HeteroSpec	**2.92×** **6775**	**6.14** **396712**	**3.28×** **6133**	**6.83** **353382**	**3.69×** **4341**	**7.88** **218875**	**2.79×** **6942**	**5.76** **416289**	**2.29×** **8076**	**5.08** **484295**	**2.99×** **6453**	**6.34** **373911**

Table 8: Speedup ratios, average acceptance lengths (τ), total validation calls (Calls), and total verification tokens (Tokens) of different methods on L40 40G GPUs. V represents Vicuna, L31 represents LLaMA-Instruct 3.1, and DSL represents DeepSeek-R1-Distill-LLaMA.

		MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Mean	
Model	Method	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens	Speedup Calls	τ Tokens
V 13B	EAGLE-3	2.36× 7637	5.56 393862	2.81× 4602	6.42 231721	2.36× 2131	5.66 110544	2.37× 2718	5.48 137837	2.12× 2207	5.58 112553	2.40× 3859	5.74 197303
	HeteroSpec	**2.50×** **7392**	**5.90** **363224**	**3.08×** **4154**	**7.25** **182479**	**2.48×** **1853**	**5.78** **95514**	**2.49×** **2508**	**5.78** **127051**	**2.34×** **1943**	**6.18** **92877**	**2.58×** **3570**	**6.18** **172229**
L31 8B	EAGLE-3	2.47× 14912	4.47 917450	3.42× 5414	6.04 326152	3.11× 3290	5.57 199951	3.13× 3451	5.67 213993	2.38× 4746	4.44 289631	2.90× 6363	5.24 389435
	HeteroSpec	**2.67×** **13298**	**4.84** **820799**	**3.71×** **4992**	**6.61** **265759**	**3.20×** **3607**	**5.62** **213785**	**3.26×** **3103**	**5.78** **190052**	**2.44×** **4573**	**4.54** **278893**	**3.06×** **5915**	**5.48** **353858**
L33 70B	EAGLE-3	5.12× 10375	5.50 514086	5.89× 5250	6.14 252578	5.63× 2552	5.98 123939	5.69× 2946	6.49 147345	4.21× 4355	4.90 210372	5.31× 5096	5.80 249664
	HeteroSpec	**5.21×** **9973**	**5.63** **458586**	**6.15×** **4857**	**6.54** **190939**	**5.78×** **2454**	**6.16** **108569**	**5.78×** **2925**	**6.68** **137289**	**4.35×** **4258**	**5.05** **200804**	**5.45×** **4893**	**6.01** **219237**
DSL 8B	EAGLE-3	2.77× 8408	4.96 519731	3.07× 7680	5.42 470112	3.82× 4861	6.67 298481	2.65× 8865	4.50 542741	2.31× 9280	4.33 570589	2.92× 7819	5.18 480331
	HeteroSpec	**2.98×** **8191**	**5.17** **501969**	**3.18×** **7560**	**5.54** **456374**	**4.21×** **4427**	**7.53** **238975**	**2.71×** **8480**	**4.54** **516930**	**2.39×** **9029**	**4.39** **553813**	**3.09×** **7537**	**5.43** **453612**

Table 9: Speedup ratios, average acceptance lengths (τ), total validation calls (Calls), and total verification tokens (Tokens) of different methods on H20-3e GPUs with Temperature=1. V represents Vicuna, L31 represents LLaMA-Instruct 3.1, L33 represents LLaMA-Instruct 3.3, and DSL represents DeepSeek-R1-Distill-LLaMA.

Method	Paradigm	Alpaca	GSM8K	HumanEval	MT-Bench	CNN/DM
Medusa	Multi-head	1.74×/3.57	1.78×/3.60	1.91×/3.80	1.75×/3.55	1.41×/3.09
Hydra	Multi-head	2.20×/4.64	2.22×/4.66	2.38×/4.88	2.17×/4.58	1.68×/3.83
EAGLE-3	Auto-regressive	3.31×/6.51	3.31×/6.40	3.78×/7.73	3.22×/6.56	2.87×/6.54
HeteroSpec	Auto-reg.+adaptive	**3.45×/6.70**	**3.42×/6.67**	**4.45×/9.05**	**3.53×/7.10**	**3.04×/7.07**

Table 10: Comparison across speculative decoding paradigms (Vicuna-13B). Format: Speedup / Average Acceptance Length (τ)

Model	Method	MT-Bench	HumanEval	GSM8K	Alpaca	CNN/DM
L31 8B	EAGLE-3	3.43×/6.15	3.95×/6.83	3.69×/6.32	3.79×/6.84	2.92×/5.39
	AdaEDL	3.31×/5.39	3.92×/6.47	3.59×/5.57	3.64×/5.62	2.87×/4.65
	HeteroSpec	**3.74×/6.44**	**4.43×/7.46**	**3.81×/6.53**	**3.99×/7.08**	**3.11×/5.47**
V 13B	EAGLE-3	3.22×/6.56	3.78×/7.73	3.31×/6.40	3.31×/6.51	2.87×/6.54
	AdaEDL	3.11×/5.84	3.64×/7.14	3.21×/5.87	3.07×/5.68	2.76×/5.92
	HeteroSpec	**3.53×/7.10**	**4.45×/9.05**	**3.42×/6.67**	**3.45×/6.70**	**3.04×/7.07**

Table 11: Comparison with AdaEDL on EAGLE-3 backbone. Format: Speedup / Average Acceptance Length (τ)

AdaEDL's early stopping frequently leads to performance degradation on tree-based structures because: (1) it optimizes the drafting phase which accounts for minimal latency, and (2) early exit prematurely prunes "recovery" branches where subsequent children might be highly confident. In contrast, HeteroSpec's dynamic extension exploits stable branches, maximizing accepted length per verification call.

A.11 More Discussion on HeteroSpec

HeteroSpec demonstrates strong potential for large-scale LLM service systems using speculative decoding (Liu et al., 2024b; Sadhukhan et al., 2025; Li et al., 2025c). Different applications impose diverse service-level objectives (SLOs) on inference latency: chatbots tolerate $200 \sim 500$ ms response times, while web search and autonomous driving require stricter constraints of $20 \sim 100$ ms (Brysbaert, 2019; Zhong et al., 2024; Lin et al., 2024). SLO-customized LLM service systems are thus designed to dynamically select tokens to meet these individualized latency requirements while optimizing overall throughput.

Such problems are modeled using a hardware budget—the maximum tokens processed per forward pass (Li et al., 2025c). Given a hardware budget and a batch of requests, the SLO-customized system aims to (1) satisfy diverse SLO requirements—typically measured by TPOT (Time Per Output Token)—and (2) maximize token acceptance during verification. HeteroSpec's Dynamic Extended Drafting increases acceptance for low-entropy requests, while Top-N Pruning reduces their budget requirements without sacrificing acceptance rates. This enables budget reallocation to stricter SLO requests, improving overall SLO satisfaction. As an orthogonal strategy to existing schedulers, HeteroSpec can significantly enhance throughput in SLO-customized systems, with future integration planned for inference services.

A.12 Future Work

Future work includes extending HeteroSpec to support additional orthogonal speculative decoding strategies, such as asynchronous draft-and-target, for enhanced computational efficiency through overlapping execution. We will also explore its application to ultra-long sequence generation, addressing pronounced resource and verification overheads to further demonstrate universality and scalability. Another key direction is integrating HeteroSpec into SLO-aware inference service systems. We envision it as an auxiliary module alongside existing scheduling and batching strategies. By leveraging its orthogonality, HeteroSpec can enhance system performance, provide flexible resource management, and maximize overall throughput and SLO satisfaction in real-world LLM service deployments.