ConfSpec: Efficient Step-Level Speculative Reasoning via Confidence-Gated Verification

Siran Liu[1,2], Zane Cao[2], Yongchao He[2†]
[1]Peking University [2]ScitiX AI
liusr25@stu.pku.edu.cn, zcao@scitix.ai, yongchao-he@outlook.com

Abstract

Chain-of-Thought reasoning significantly improves the performance of large language models on complex tasks, but incurs high inference latency due to long generation traces. Step-level speculative reasoning aims to mitigate this cost, yet existing approaches face a long-standing trade-off among accuracy, inference speed, and resource efficiency. We propose ConfSpec, a confidence-gated cascaded verification framework that resolves this trade-off. Our key insight is an asymmetry between generation and verification: while generating a correct reasoning step requires substantial model capacity, step-level verification is a constrained discriminative task for which small draft models are well-calibrated within their competence range, enabling high-confidence draft decisions to be accepted directly while selectively escalating uncertain cases to the large target model. Evaluation across diverse workloads shows that ConfSpec achieves up to 2.24× end-to-end speedups while matching target-model accuracy. Our method requires no external judge models and is orthogonal to token-level speculative decoding, enabling further multiplicative acceleration.

1 Introduction

Large language models (LLMs) (DeepSeek-AI et al., 2025b; OpenAI et al., 2024) have demonstrated remarkable reasoning capabilities by explicitly generating chains of thought (CoT) (Wei et al., 2022), enabling strong performance on complex tasks such as mathematical problem solving, scientific question answering, and program synthesis. By decomposing a problem into a sequence of intermediate reasoning steps, CoT allows models to perform multi-step inference that would otherwise be infeasible with direct answer generation. However, this expressive reasoning ability comes at a substantial computational cost. Modern reasoning-oriented models often produce thousands of tokens per example, making inference latency and serving cost a critical bottleneck for large-scale deployment (Pan et al., 2025; Fu et al., 2025).

Speculative decoding (Leviathan et al., 2023) mitigates this overhead by allowing a lightweight draft model to propose candidate tokens that are verified by a larger target model. While effective at the token level, this paradigm overlooks a key structural property of CoT reasoning: reasoning progresses through semantically coherent *steps* rather than isolated *tokens*. Token-level verification enforces strict surface-form equivalence, causing semantically valid but lexically different continuations to be frequently rejected, and leaving substantial efficiency gains untapped.

Motivated by this observation, recent work explores *step-level speculative reasoning*, where entire reasoning steps are proposed and verified at once. However, existing approaches face a fundamental limitation that goes beyond a simple speed–accuracy trade-off. Scoring-based methods (Pan et al., 2025; Liao et al., 2025; Shi et al., 2025) rely on lightweight heuristics that implicitly assume optimism in draft generation, often accepting fluent but logically incorrect steps. Lookahead-based approaches (Fu et al., 2025) rely on brute-force verification by invoking additional judge models, effectively stacking model capacity rather than exploiting structural asymmetries. Embedding-based verification (Fu et al., 2025) reduces overhead but collapses rich logical structure into low-dimensional similarity, leading to brittle semantic judgments. As a result, current methods exhibit a persistent tension among accuracy, inference speed, and resource efficiency, as summarized in Table 1. Underlying this tension is a deeper paradigm gap: scoring-based methods evaluate draft plausibility in isola-

†Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 26525–26538
July 2-7, 2026

Table 1: Comparison of speculative reasoning approaches under the accuracy–speed–resource trade-off.

Method	Accu.	Speed	Res. Effi.
Scoring-Based	✗	✓	✓
Lookahead-Based	✓	✓	✗
Embedding-Based	✗	✓	✓
ConfSpec (Ours)	✓	✓	✓

tion ($f_{\text{eval}}(\hat{s})$), whereas lookahead and embedding methods attempt equivalence verification against the target model's output ($f_{\text{eval}}(\hat{s}, s)$)—yet none exploit the structural properties that make equivalence verification fundamentally easier than generation.

Across these methods, *step-level verification is implicitly treated as uniformly hard*, an assumption that leads to either unnecessary computation or irreversible accuracy loss. In this work, we argue that the true bottleneck of step-level speculative reasoning lies not in step generation, but in overly conservative verification, which forces expensive model verification even for routine reasoning steps. This bottleneck stems from a fundamental asymmetry: while generating a correct reasoning step often requires substantial model capacity, verifying whether two candidate steps are semantically equivalent is typically easier (Zhou et al., 2025) and highly uneven in difficulty (Ding et al., 2024). Many verification instances involve routine arithmetic or local logical transformations, while only a small fraction require deep semantic scrutiny.

Crucially, and contrary to the common belief that small language models are poorly calibrated or prone to overconfidence, we make a counter-intuitive observation: *for the specific task of step-level verification, small models remain surprisingly well-calibrated within their competence range* (validated in §4.1). When a draft model assigns high confidence to a verification decision, that decision is usually correct. Low confidence, in contrast, reliably signals genuinely ambiguous or out-of-distribution cases. This property suggests that heavyweight target-model verification is unnecessary for the majority of reasoning steps and should be reserved for difficult instances.

Based on these insights, we propose ConfSpec, a confidence-gated cascaded verification framework that fully exploits the structural advantages of equivalence-based verification for step-level speculative reasoning. Instead of uniformly invoking the target model for verification, ConfSpec introduces a two-stage cascade. The draft model first performs semantic equivalence verification and produces both a decision and a confidence score. High-confidence decisions are accepted directly, while low-confidence cases are escalated to the target model for definitive verification. This confidence-aware routing enables efficient use of model capacity without sacrificing correctness.

We evaluate ConfSpec on mathematical reasoning, scientific QA, and code generation benchmarks, which require long chains of intermediate reasoning and represent diverse reasoning paradigms, including AIME, AMC, MATH, GPQA, and HumanEval (§5). Across tasks and model families, ConfSpec achieves up to 2.24$\times$ inference speedup while matching target-model accuracy. Our approach requires no external judge models and is fully compatible with token-level speculative decoding, enabling multiplicative speedups. These results demonstrate that verification-level cascading provides a principled solution to the long-standing speed–accuracy–resource trade-off in step-level speculative reasoning.

2 Background

2.1 Reasoning with LLMs

LLMs perform complex reasoning through the generation of CoT. Given an input prompt x, a model M produces a sequence of intermediate reasoning steps $S = \{s_1, s_2, \ldots, s_n\}$ before deriving the final answer y. Each reasoning step s_i consists of multiple tokens and represents a semantically coherent logical transformation.

In this work, we define a *reasoning step* as a contiguous span of tokens delimited by a step boundary token. Concretely, we adopt newline-separated steps, where each step ends with newline token "\n\n". This convention aligns with the formatting of long-CoT reasoning traces (Wei et al., 2022) in modern reasoning-oriented LLMs and enables explicit step-level generation and verification. Both draft and target models generate steps autoregressively until a step boundary is reached.

Although reasoning traces are generated token by token, their semantic structure is naturally organized at the step level. This mismatch between the unit of generation and the unit of semantics motivates acceleration methods that operate beyond individual tokens.

2.2 Token-Level Speculative Decoding

Speculative decoding (Leviathan et al., 2023) accelerates autoregressive generation through a draft-then-verify paradigm. A lightweight draft model M_D first generates k candidate tokens, which are then verified by a larger target model M_T in a single batched forward pass. Token acceptance is determined via rejection sampling based on probability ratios, guaranteeing that the resulting output distribution exactly matches that of M_T.

While effective for general text generation, speculative decoding enforces strict token-wise equivalence. In reasoning tasks, however, multiple token sequences may express the same underlying logic. Consequently, semantically valid draft continuations are frequently rejected due to surface-form discrepancies, limiting the achievable speedup (Pan et al., 2025).

2.3 Step-Level Speculative Reasoning

Step-level speculative reasoning extends speculative decoding by operating on entire reasoning steps. By allowing verification based on semantic equivalence rather than exact token matching, step-level speculation better aligns with the structure of CoT reasoning.

This shift introduces a central challenge: *step-level verification*. Formally, given a draft step $\hat{s}_i$ and a target step s_i, verification determines whether the two steps are semantically equivalent. We denote this decision by a verification function $V(\hat{s}_i, s_i) \in \{\texttt{accept}, \texttt{reject}\}$.

Strict equality requires $\hat{s}_i = s_i$ at the token level, which is unnecessarily restrictive. In contrast, semantic equivalence captures the notion that two steps induce similar downstream reasoning behavior. One possible formalization is that $\hat{s}_i$ and s_i are equivalent if they induce similar conditional distributions over future outputs, i.e., $P(y \mid x, s_1, \ldots, s_{i-1}, \hat{s}_i) \approx P(y \mid x, s_1, \ldots, s_{i-1}, s_i)$. Designing an efficient and reliable approximation to this equivalence test remains the key challenge in step-level speculative reasoning.

3 Related Work

3.1 Token-Level Speculative Decoding

Speculative decoding has been extensively studied as a general mechanism for accelerating autoregressive LLM inference. Existing approaches differ in how draft tokens are generated and verified, including standalone draft models, auxiliary decoding heads (e.g., Medusa, Hydra (Cai et al., 2024; Ankner et al., 2024)), feature-aligned draft models (e.g., EAGLE (Li et al., 2025b,a)), retrieval-based methods (Saxena, 2023; Fu et al., 2024), and tree-based parallel speculation (Miao et al., 2024). Additional optimizations reuse the target model's KV cache to reduce redundant computation (Du et al., 2024; Zimmer et al., 2025).

However, strict token-wise or distribution-level matching causes semantically equivalent reasoning trajectories to be rejected due to superficial differences, motivating higher-level speculation units that tolerate semantic variation.

3.2 Step-Level Speculative Reasoning

To address token-level rigidity, recent work explores speculative reasoning at the granularity of reasoning steps. SpecReason (Pan et al., 2025) employs a lightweight speculator to propose an entire step, which is scored by the target model in a single prefill pass. While computationally efficient, this approach often sacrifices accuracy by accepting high-quality but semantically incorrect steps. Similarly, SpecCoT (Shi et al., 2025) generates multiple candidate steps via the draft model and uses the target model to select the best one, while RSD (Liao et al., 2025) employs a trained Process Reward Model to score draft steps against a reward threshold.

LookaheadReasoning (Fu et al., 2025) improves verification fidelity by generating multiple future steps and invoking an external LLM-as-a-Judge (Zheng et al., 2023) to assess semantic equivalence. However, this design incurs substantial resource overhead—requiring deployment of an additional inference engine for the judge model—and introduces redundancy: the model already possesses strong verification capability through its internal representations and output probabilities. Embedding-based alternatives (Fu et al., 2025) employ lightweight embedding models for step alignment to reduce cost, yet compress rich logical structure into low-dimensional similarity metrics, leading to unreliable alignment for complex reasoning.

At the token level, EASD (Su et al., 2025) introduces an entropy-based penalty to block error propagation; as a token-level technique, it is orthogonal and complementary to step-level methods.

These approaches highlight a persistent trade-off among accuracy, speed, and resource efficiency, suggesting that the core challenge lies in how to perform equivalence verification efficiently and ac-

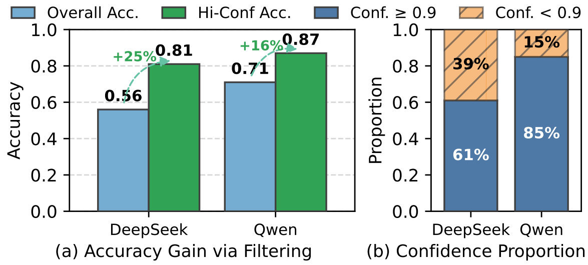

Figure 1: Confidence-based filtering improves verification accuracy. (a) Restricting to high-confidence predictions ($p_D \geq 0.9$) substantially improves draft-model accuracy, approaching target-model quality. (b) High-confidence cases cover the majority of verification instances (61–85%), enabling efficient cascading.

curately without auxiliary models.

3.3 Uncertainty and Self-Verification in LLMs

ConfSpec is closely related to work on uncertainty estimation, calibration, and self-verification in LLMs. Prior studies (Jiang et al., 2020; Kadavath et al., 2022) show that model output probabilities encode meaningful uncertainty signals and that LLMs can often identify when they are uncertain or likely to be incorrect. Such confidence estimates have been used for selective prediction, abstention, self-consistency, and iterative self-correction.

Unlike prior work that relies on external judges or additional training objectives, ConfSpec leverages intrinsic confidence estimates from the draft model to guide verification. This design aligns with findings (Zhou et al., 2025; Jiang et al., 2020; Saad-Falcon et al., 2025) that, for discriminative tasks within a model's competence range, confidence scores can serve as reliable indicators of correctness. By exploiting this property, ConfSpec achieves intrinsic verification without introducing additional models or supervision.

4 Methodology

We introduce ConfSpec, a confidence-gated cascaded verification framework for step-level speculative reasoning. The core idea is to exploit an empirical asymmetry in verification difficulty: most reasoning steps can be verified reliably by lightweight draft model, while only a small fraction require expensive target-model verification. We first present a concise preliminary study motivating confidence-based routing, then formally define the ConfSpec framework, followed by an extension to tree-structured drafting and an efficiency analysis.

4.1 Confidence-Guided Verification

Our design is motivated by a critical asymmetry in step-level verification, which gives rise to a key empirical finding: model confidence reliably signals verification difficulty.

Verification Asymmetry and Heterogeneity. In practice, step-level verification exhibits a strong asymmetry relative to generation. Judging whether two reasoning steps are semantically equivalent reduces to a constrained discriminative decision, whereas generating a valid step requires exploring a substantially larger output space. Moreover, verification difficulty is highly heterogeneous: many instances involve routine arithmetic or local logical transformations, while only a small fraction require deeper semantic reasoning. Together, these properties suggest that a lightweight model can handle a large fraction of verification cases reliably.

Confidence as a Reliability Signal. Crucially, we observe that for step-level verification, the draft model's confidence strongly correlates with correctness. Let p_D denote the probability assigned to the predicted verification decision and let $\gamma \in [0, 1]$ represent the confidence threshold. Our empirical analysis shows that high-confidence predictions ($p_D \geq \gamma$) are overwhelmingly correct, while low-confidence cases correspond to genuinely ambiguous or out-of-distribution instances.

We evaluate two model families on the MATH500 benchmark (Hendrycks et al., 2021): DeepSeek-R1-Distill (1.5B draft, 32B target) and Qwen3 (1.7B draft, 32B target), using target model decisions as ground truth to compare draft verification accuracy (Overall) versus high-confidence instances only (Hi-Conf, $p_D \geq 0.9$). As shown in Figure 1(a), filtering by confidence substantially improves accuracy: DeepSeek-1.5B improves from 56% to 81% (+25%), while Qwen3-1.7B improves from 71% to 87% (+16%). Importantly, Figure 1(b) shows that high-confidence predictions cover 61% and 85% of all verification instances, respectively. This indicates that the majority of steps can be verified reliably by the draft model alone, while only a small fraction requires escalation. These findings motivate confidence-gated cascaded verification: accept draft decisions when confident, and defer to the target model otherwise.

4.2 The ConfSpec Framework

We now formally describe ConfSpec. Algorithm 1 provides the full procedure.

Algorithm 1 Cascaded Speculative Verification

Require: Input x, draft model M_D, target model M_T, draft steps k, confidence threshold γ
Ensure: Reasoning trace S

1: $S \leftarrow \emptyset$, $\texttt{context} \leftarrow x$
2: **while not** $\texttt{terminated}$ **do**
3:     *# Stage 1: Step-Level Drafting*
4:     $\{\hat{s}_1, \dots, \hat{s}_k\} \leftarrow M_D.\texttt{generate}(\texttt{context}, k)$
5:     *# Stage 2: Dual-Tier Cascaded Verification*
6:     **for** $j = 1$ **to** k **in parallel do**
7:         $s_j \leftarrow M_T.\texttt{generate}(\texttt{context} \oplus \hat{s}_j)$
8:     **end for**
9:     $m \leftarrow 0$
10:     **for** $j = 1$ **to** k **do**
11:         $r_D, p_D \leftarrow M_D.\texttt{verify}(\hat{s}_j, s_j)$
12:         $V \leftarrow r_D$ **if** $p_D \geq \gamma$ **else** $M_T.\texttt{verify}(\hat{s}_j, s_j)$
13:         $m \leftarrow m + 1$ **if** $V = \texttt{accept}$ **else break**
14:     **end for**
15:     *# Stage 3: Context Management and Fallback*
16:     **if** $m > 0$ **then**
17:         $S \leftarrow S \cup \{\hat{s}_1, \dots, \hat{s}_m\}$
18:         $\texttt{context} \leftarrow \texttt{context} \oplus \hat{s}_1 \oplus \dots \oplus \hat{s}_m$
19:     **else**
20:         $s \leftarrow M_T.\texttt{generate}(\texttt{context})$
21:         $S \leftarrow S \cup \{s\}$, $\texttt{context} \leftarrow \texttt{context} \oplus s$
22:     **end if**
23: **end while**
24: **return** S

Notation. Let M_D denote a lightweight draft model and M_T a high-capacity target model. Given an input x, reasoning proceeds as a sequence of steps $S = \{s_1, \dots, s_n\}$, where each step is a contiguous token span terminated by "\n\n". At step i, the draft model proposes candidate steps $\hat{s}_i$, which are verified against the target model's continuation.

Stage 1: Step-Level Drafting. At each iteration, the draft model autoregressively generates k candidate reasoning steps conditioned on the current context (Algorithm 1, line 4):

$$\hat{s}_i, \dots, \hat{s}_{i+k-1} \sim P_{M_D}(\cdot \mid x, s_1, \dots, s_{i-1}). \quad (1)$$

Each draft step is generated until a delimiter is produced, which explicitly defines the step boundary.

Stage 2: Dual-Tier Cascaded Verification. For each draft step $\hat{s}_j$, the target model generates a corresponding step s_j in parallel (Algorithm 1, lines 6–8). Verification is then performed using a two-tier cascade.

(1) Draft Verification. The draft model M_D acts as a binary verifier, taking as input the context, the draft step $\hat{s}_j$, and the target step s_j. It outputs a decision (line 11)

$$r_D \in \{\texttt{accept}, \texttt{reject}\}, \quad (2)$$

along with a confidence score

$$p_D = p_{M_D}(r_D \mid x, s_{<i}, \hat{s}_j, s_j). \quad (3)$$

In practice, r_D is represented by a single verification token (e.g., "Yes" or "No"), and p_D is its softmax probability computed directly from the raw logits. This single-token design ensures that p_D is intrinsically independent of decoding hyperparameters such as temperature, top-k, and top-p, which are applied only downstream during token sampling.

(2) Cascaded Decision Rule. We define the cascaded verification function as

$$V_{\text{cascade}}(\hat{s}_j, s_j) = \begin{cases} r_D, & \text{if } p_D \geq \gamma, \\ r_T, & \text{otherwise,} \end{cases} \quad (4)$$

where r_T is the target model's verification decision and $\gamma \in [0, 1]$ is a confidence threshold. This cascading mechanism preserves target-model accuracy while substantially reducing computational cost (§5). High-confidence draft decisions are accepted directly, while low-confidence cases are escalated to the target model.

As shown in Algorithm 1 (lines 12–13), verification proceeds sequentially over the drafted steps until a rejection occurs.

Stage 3: Context Management and Fallback. If the first m draft steps are accepted, they are appended to the reasoning context (lines 16–18). Upon the first rejection, the target model regenerates the next step from the current context (lines 19–21), ensuring correctness is preserved. This fallback mechanism guarantees that the final reasoning trace matches the target model's distribution up to verification errors.

4.3 Extension: Tree-Structured Drafting

ConfSpec naturally extends to tree-structured speculative reasoning. Instead of generating a single linear sequence, the draft model proposes multiple candidate steps at each layer. Cascaded verification is applied independently to all candidates in parallel.

When multiple candidates pass verification, we select the step with the highest confidence score:

$$\hat{s}^* = \arg\max_{\hat{s} \in \mathcal{C}} p(\hat{s}), \quad (5)$$

where $p(\hat{s})$ denotes the verification confidence (from M_D or M_T depending on cascading). Beyond verification, this confidence score also serves as a principled criterion for path selection, favoring more reliable reasoning branches.

4.4 Efficiency Analysis

We analyze the efficiency gains of ConfSpec. Let c_{vD} and c_{vT} denote the cost of draft and target verification ($c_{vD} \ll c_{vT}$), and let $\alpha \in [0, 1]$ be the cascade rate. The expected verification cost per step is $\mathcal{O}(c_{vD} + \alpha \cdot c_{vT})$, approaching draft-only cost when α is small. The threshold γ controls this trade-off: higher γ improves reliability but increases α.

Beyond per-step savings, accurate verification sustains high acceptance rates. Erroneous acceptance of low-quality steps corrupts the reasoning trajectory; while the target model may recover through reflection, the draft model typically cannot—causing repeated rejections as subsequent drafts diverge from the corrupted context, ultimately degrading speedup. By maintaining verification accuracy, ConfSpec preserves coherent trajectories and high acceptance rates throughout.

Importantly, verification-level cascading is orthogonal to token-level speculative decoding, enabling multiplicative acceleration when combined.

5 Experiments

5.1 Experimental Setup

Models. We evaluate ConfSpec on two widely used open-source model families with long-CoT reasoning capabilities: DeepSeek-R1-Distill (DeepSeek-AI et al., 2025a) and Qwen3 (Yang et al., 2025). For both families, we fix the target model to the 32B variant and use a substantially smaller draft model (1.5B for DeepSeek-R1-Distill and 1.7B for Qwen3). This setting reflects a realistic deployment scenario where a lightweight model assists a high-capacity reasoning model, while keeping the target model identical across all methods.

Datasets. We consider five benchmarks spanning mathematical reasoning, scientific question answering, and code generation. Specifically, we evaluate on AIME24 (Zhang and Math-AI, 2024), AMC23 (AMC12, 2025), and MATH500 (Hendrycks et al., 2021) for mathematical reasoning; GPQA Diamond (Rein et al., 2023) for graduate-level scientific QA; and HumanEval (Chen et al., 2021) for code generation. These datasets require long intermediate reasoning chains and cover diverse paradigms, enabling assessment of domain generalization.

Generation Parameters. Unless otherwise specified, all methods use the official recommended configurations of their respective model families. We fix the maximum token budget to 32K for DeepSeek-R1-Distill and 37K for Qwen3 to ensure complete reasoning traces. For step-level speculative methods, we set the speculation depth to 5 steps per iteration for ConfSpec, while baselines follow their original settings. All generation parameters are kept identical across methods to ensure fair comparison.

Baselines. We compare against vanilla target-model inference as the primary accuracy and speedup baseline. In addition, we evaluate five representative step-level speculative reasoning methods: (i) **SpecReason**, which performs scoring-based verification (evaluated at score thresholds $s \in \{7, 9\}$); (ii) **RSD**, which uses a trained Process Reward Model (PRM) to score draft steps and accepts those above a reward threshold ($\delta = 0.7$); (iii) **SpecCoT**, which generates multiple candidate steps in parallel for target-model optimal selection; (iv) **LookaheadReasoning**, which relies on an external LLM-as-a-Judge for semantic equivalence verification; and (v) **Embedding Alignment**, which verifies steps via embedding-based semantic similarity (evaluated at thresholds $\alpha \in \{0.8, 0.9\}$). To demonstrate orthogonality with token-level speculative decoding, we also report results for ConfSpec + speculative decoding (SD), which combines our method with n-gram-based token-level speculation (prompt lookup decoding Saxena, 2023) in the main experiments, with additional methods such as EAGLE-3 (Li et al., 2025a) discussed in Appendix A. All baselines are evaluated under the same token budgets and hardware constraints.

Evaluation. All experiments are conducted on NVIDIA H200 GPUs. The target model is deployed using tensor parallelism across two GPUs, while the draft model runs on a single GPU. Importantly, ConfSpec does not require additional hardware resources beyond those used by the target and draft models. Our implementation builds on vLLM 0.10.1 (Kwon et al., 2023), and all methods are evaluated within the same serving framework. Following prior work (Pan et al., 2025; Fu et al., 2025), we report pass@1 accuracy (averaged over 16 samples per problem at temperature 0.6), along with relative inference speedup normalized to the target-model baseline.

5.2 Main Results

Pareto Frontier of Accuracy and Speed. The core challenge in speculative reasoning is not a smooth accuracy–latency trade-off, but the risk of

Table 2: Main Results: Performance comparison. We report **Pass@1 Accuracy (%)** and **End-to-End Speedup (×)**. The target model baseline represents the upper bound for accuracy and the baseline (1.00×) for speed.

Method	**Dataset (Acc. ↑ / Speedup ↑)**					
	AIME24	**AMC23**	**MATH500**	**GPQA**	**HumanEval**	**Mean**
Draft: Deepseek-R1-Distill 1.5B / Target: Deepseek-R1-Distill 32B						
Target Model	68.54 / 1.00×	94.22 / 1.00×	92.43 / 1.00×	61.71 / 1.00×	93.12 / 1.00×	81.99 / 1.00×
SpecReason ($s = 7$)	38.33 / 1.54×	80.78 / 2.06×	78.23 / 1.89×	51.09 / 1.41×	75.61 / 1.72×	64.81 / 1.72×
SpecReason ($s = 9$)	60.21 / 0.82×	88.91 / 1.06×	84.61 / 1.13×	58.59 / 0.76×	90.17 / 1.05×	76.50 / 0.96×
RSD ($\delta = 0.7$)	48.33 / 1.17×	78.91 / 1.29×	76.99 / 1.34×	49.27 / 1.06×	83.05 / 1.23×	67.31 / 1.22×
SpecCoT	46.46 / 1.21×	82.97 / 1.28×	81.14 / 1.31×	53.95 / 1.09×	82.44 / 1.30×	69.39 / 1.24×
Embedding ($\alpha = 0.8$)	61.67 / 1.41×	92.97 / 1.52×	90.88 / 1.50×	56.63 / 1.23×	88.99 / 1.31×	78.23 / 1.39×
Embedding ($\alpha = 0.9$)	68.54 / 1.27×	93.75 / 1.28×	92.13 / 1.27×	61.02 / 1.09×	89.63 / 1.27×	81.01 / 1.24×
LookaheadReasoning	67.29 / 1.38×	94.06 / 1.51×	92.41 / 1.48×	60.84 / 1.21×	91.63 / 1.36×	81.25 / 1.39×
ConfSpec	**68.75 / 1.73×**	**94.53 / 1.76×**	**92.35 / 1.83×**	**60.57 / 1.54×**	**91.33 / 1.53×**	**81.51 / 1.68×**
SD (Lossless)	≡ / 1.48×	≡ / 1.46×	≡ / 1.43×	≡ / 1.52×	≡ / 1.37×	≡ / 1.45×
ConfSpec + SD	≡* / **2.12×**	≡* / **2.03×**	≡* / **2.24×**	≡* / **2.01×**	≡* / **1.81×**	≡* / **2.04×**
Draft: Qwen3 1.7B / Target: Qwen3 32B						
Target Model	80.21 / 1.00×	97.34 / 1.00×	92.54 / 1.00×	67.83 / 1.00×	93.68 / 1.00×	86.32 / 1.00×
SpecReason ($s = 7$)	40.21 / 1.29×	83.13 / 1.53×	82.40 / 1.54×	54.24 / 1.06×	79.87 / 1.37×	67.97 / 1.36×
SpecReason ($s = 9$)	69.38 / 1.08×	91.41 / 1.11×	87.29 / 1.33×	63.18 / 0.77×	91.15 / 1.15×	80.48 / 1.09×
RSD ($\delta = 0.7$)	49.17 / 1.17×	84.84 / 1.10×	81.90 / 1.22×	53.19 / 1.07×	82.15 / 1.11×	70.25 / 1.13×
SpecCoT	52.29 / 1.14×	84.06 / 1.08×	84.10 / 1.29×	56.28 / 1.02×	84.68 / 1.14×	72.28 / 1.13×
Embedding ($\alpha = 0.8$)	70.42 / 1.23×	95.16 / 1.24×	91.67 / 1.30×	66.23 / 1.13×	91.06 / 1.15×	82.91 / 1.21×
Embedding ($\alpha = 0.9$)	74.17 / 1.10×	95.83 / 1.14×	91.72 / 1.17×	66.41 / 1.03×	92.07 / 1.06×	84.04 / 1.10×
LookaheadReasoning	79.58 / 1.19×	96.41 / 1.24×	92.40 / 1.28×	67.17 / 1.10×	92.73 / 1.17×	85.66 / 1.20×
ConfSpec	**79.79 / 1.27×**	**96.41 / 1.35×**	**92.82 / 1.42×**	**67.53 / 1.18×**	**94.35 / 1.27×**	**86.18 / 1.30×**
SD (Lossless)	≡ / 1.39×	≡ / 1.36×	≡ / 1.34×	≡ / 1.39×	≡ / 1.39×	≡ / 1.37×
ConfSpec + SD	≡* / **1.62×**	≡* / **1.73×**	≡* / **1.79×**	≡* / **1.54×**	≡* / **1.58×**	≡* / **1.65×**

Note: '≡' denotes lossless accuracy matching the target model. '≡*' denotes accuracy matching the base method.

sharp and irreversible accuracy collapse under aggressive verification shortcuts. Table 2 summarizes the resulting accuracy and speed characteristics.

- **SpecReason:** Aggressive thresholds yield substantial speedups (e.g., 1.72× with $s = 7$), but cause severe accuracy degradation on complex tasks (e.g., from 68.46% to 38.33% on AIME24). RSD and SpecCoT exhibit similar accuracy collapse. This failure stems from utility-based scoring that favors fluent yet semantically incorrect reasoning paths, which irreversibly misguides subsequent generation.

- **Embedding:** Despite avoiding additional inference engines, embedding-based verification struggles to capture semantic equivalence beyond lexical similarity. Consequently, logically consistent but paraphrased steps are frequently misclassified, leading to a reduced accuracy of 78.23% at $\alpha = 0.8$ despite a modest 1.39× speedup.

- **LookaheadReasoning:** This method preserves accuracy most effectively among baselines, closely matching the target model. However, its dependence on an external LLM-as-a-Judge incurs substantial computational redundancy, limiting its practical efficiency gains.

- **ConfSpec (Ours):** ConfSpec achieves a superior Pareto frontier, matching the target model's accuracy distribution while delivering significant acceleration. It attains average speedups of 1.68× and 1.30× on the DeepSeek and Qwen families, respectively. This indicates that our confidence-gated cascade effectively identifies "easy" steps that the draft model can handle reliably, while correctly routing "complex" reasoning steps to the target model for rigorous verification.

Orthogonality with Token-Level Speculation. Table 2 demonstrates that combining ConfSpec with speculative decoding (SD) yields multiplicative speedup gains. On DeepSeek-R1-Distill, ConfSpec + SD achieves 2.04× mean speedup while preserving base method accuracy, compared to 1.45× for standalone SD. On Qwen3, the combination achieves 1.65× speedup versus 1.37× for SD

Table 3: Ablation study on verification strategies. We compare static strategies (Draft/Target Only) with ConfSpec under different confidence thresholds γ. Cell format: **Accuracy / Speedup**.

Method	**Dataset (Acc. ↑ / Speedup ↑)**					
	AIME24	**AMC23**	**MATH500**	**GPQA**	**HumanEval**	**Mean**
Target Model	68.46 / 1.00×	94.22 / 1.00×	92.43 / 1.00×	61.71 / 1.00×	93.12 / 1.00×	82.03 / 1.00×
Draft Verification	35.89 / 2.34×	73.91 / 2.18×	81.92 / 2.19×	43.03 / 2.15×	69.68 / 2.00×	60.89 / 2.17×
ConfSpec ($\gamma = 0.8$)	49.38 / 1.94×	82.97 / 1.86×	90.60 / 1.96×	56.05 / 1.68×	84.14 / 1.70×	72.63 / 1.83×
ConfSpec ($\gamma = 0.9$)	**68.75 / 1.73×**	**94.53 / 1.76×**	**92.35 / 1.83×**	**60.57 / 1.54×**	**91.33 / 1.53×**	**81.51 / 1.68×**
ConfSpec ($\gamma = 0.95$)	68.46 / 1.54×	94.69 / 1.72×	92.35 / 1.72×	61.36 / 1.36×	91.42 / 1.46×	81.66 / 1.56×
Target Verification	68.75 / 1.42×	94.84 / 1.48×	92.43 / 1.59×	61.47 / 1.27×	91.39 / 1.35×	81.78 / 1.42×

Table 4: Ablation study on tree width W. Cell format: **Accuracy / Speedup**.

Method	**Dataset (Acc. ↑ / Speedup ↑)**					
	AIME24	**AMC23**	**MATH500**	**GPQA**	**HumanEval**	**Mean**
Target Model	68.54 / 1.00×	94.22 / 1.00×	92.43 / 1.00×	61.71 / 1.00×	93.12 / 1.00×	82.00 / 1.00×
ConfSpec ($W = 1$)	**68.75 / 1.73×**	**94.53 / 1.76×**	**92.35 / 1.83×**	**60.57 / 1.54×**	**91.33 / 1.53×**	**81.51 / 1.68×**
ConfSpec ($W = 2$)	68.54 / 1.39×	94.69 / 1.56×	92.35 / 1.63×	60.81 / 1.28×	91.99 / 1.38×	81.68 / 1.45×
ConfSpec ($W = 4$)	68.96 / 0.70×	95.16 / 0.61×	92.84 / 0.77×	60.92 / 0.83×	92.27 / 0.65×	82.03 / 0.71×

alone. These results validate that ConfSpec is fully orthogonal to token-level speculation, enabling the stacked acceleration as discussed in Section 4.4.

Generalization and Robustness. ConfSpec exhibits robust and consistent improvements across both model families and a wide range of task domains, including mathematical reasoning, scientific question answering, and code generation. While absolute speedups vary across datasets (1.53×–1.83× on DeepSeek, 1.18×–1.42× on Qwen3), this variation is expected and reflects differences in reasoning complexity and step acceptance rates rather than method fragility. Differences across model families primarily stem from draft-model capability and alignment with the target model. Crucially, despite these variations, the relative ranking of methods remains consistent across all settings, with ConfSpec achieving the best accuracy–speed trade-off in every configuration.

5.3 Ablation Studies

Effect of Verification Strategy and Threshold. We investigate the impact of different verification strategies and confidence thresholds on the accuracy-speedup trade-off. Table 3 presents results comparing: (1) **Draft Verification**, which exclusively uses the draft model for all verification decisions; (2) **Target Verification**, which always escalates to the target model; and (3) ConfSpec with varying confidence thresholds $\gamma \in \{0.8, 0.9, 0.95\}$.

The baselines expose the limitations of relying on a single model. Draft-only verification achieves the highest raw speedup (2.17×) but suffers from catastrophic accuracy degradation, particularly on complex reasoning tasks like AIME24 (dropping from 68.46% to 35.89%). This result highlights that the small model lacks sufficient reasoning capability to reliably adjudicate complex logical steps, often misclassifying invalid paths as correct. In contrast, Target-only verification ensures rigorous correctness (Mean Acc. 81.78%) but limits the efficiency gain (1.42×), as it incurs high computational costs by activating the large model for every verification instance regardless of its difficulty.

The confidence threshold γ provides fine-grained control over the speculation aggressiveness, allowing ConfSpec to navigate the space between these two extremes. Lower threshold ($\gamma = 0.8$) adopts an aggressive policy, boosting speedup to 1.83× but compromising reliability (Mean Acc. drops to 72.63%) by allowing the draft model to accept plausible but incorrect steps. Higher threshold ($\gamma = 0.95$) forces a conservative behavior, recovering accuracy to near-perfect levels (81.66%) but reducing speedup to 1.56× due to frequent cascading. Our default setting of $\gamma = 0.9$ achieves the optimal balance, maintaining 99.7% of the target model's accuracy while delivering a robust 1.68× mean speedup, effectively filtering "easy" steps from "hard" ones.

Effect of Tree-Structured Drafting. We investigate the impact of tree width on the accuracy-speedup trade-off by evaluating branching factors $W \in \{1, 2, 4\}$. ConfSpec extends to tree-structured speculation, generating multiple candidate branches at each step. While a larger W theoretically captures superior reasoning paths via confidence-based selection, the number of candidates grows exponentially with depth, imposing substantial overhead for parallel verification.

Table 4 presents the results of this comparison. Increasing the tree width to $W = 4$ indeed yields a marginal accuracy improvement (Mean Acc. rises from 81.51% to 82.03%), matching or even marginally exceeding the target model on some tasks. This confirms that a wider search space helps capture superior reasoning trajectories. However, this gain comes at a prohibitive computational cost: the overhead of verifying multiple long steps in parallel overwhelms the speed gains from speculation, resulting in a net slowdown ($0.71\times$ speedup). Even a modest width of $W = 2$ significantly dampens the acceleration, reducing it to $1.45\times$ compared to the linear baseline's $1.68\times$.

These results suggest that for current model configurations, linear drafting ($W = 1$) provides the optimal trade-off. Tree-structured drafting may become more beneficial when draft model quality is lower or when reasoning tasks require extensive exploration, as the increased candidate diversity can compensate for individual draft quality limitations.

6 Conclusion

We revisited step-level speculative reasoning from the perspective of verification and showed that treating all steps as uniformly hard leads to unnecessary computation. Observing that verification difficulty is heterogeneous and draft-model confidence can guide selective escalation, we proposed ConfSpec, a confidence-gated framework that escalates uncertain steps to the target model. Experiments across workloads show that ConfSpec matches target-model accuracy while achieving substantial inference speedups, remaining orthogonal to token-level speculative decoding.

Limitations

ConfSpec relies on draft-model confidence to route step-level verification. While informative in our experiments, this signal may not generalize across all model families, architectures, or task distributions. Step-level verification approximates semantic equivalence via discriminative judgment rather than exact logical validation, which can be brittle for highly nuanced or implicitly structured reasoning steps. Efficiency gains also depend on alignment between the draft and target models; under substantial domain shifts or consistently low draft-model confidence, the system may frequently fall back to target-model inference, limiting speedups. Future work includes systematically characterizing confidence calibration across models, improving verification for nuanced reasoning, and extending ConfSpec to diverse domains.

References

AMC12. 2025. Amc 12 problems and solutions, 2025.

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.

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.

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.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025b. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Mucong Ding, Chenghao Deng, Jocelyn Choo, Zichu Wu, Aakriti Agrawal, Avi Schwarzschild, Tianyi Zhou, Tom Goldstein, John Langford, Anima Anandkumar, and Furong Huang. 2024. Easy2hard-bench: Standardized difficulty labels for profiling llm performance and generalization. In *Advances in Neural*

Information Processing Systems, volume 37, pages 44323–44365. Curran Associates, Inc.

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.

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

Yichao Fu, Rui Ge, Zelei Shao, Zhijie Deng, and Hao Zhang. 2025. Scaling speculative decoding with lookahead reasoning. *Preprint*, arXiv:2506.19830.

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. *Preprint*, arXiv:2103.03874.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Preprint*, arXiv:1911.12543.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, and 17 others. 2022. Language models (mostly) know what they know. *Preprint*, arXiv:2207.05221.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, and 11 others. 2023. Measuring faithfulness in chain-of-thought reasoning. *Preprint*, arXiv:2307.13702.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

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.

Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. 2025. Reward-guided speculative decoding for efficient llm reasoning. *Preprint*, arXiv:2501.19324.

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.

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.

Rui Pan, Yinwei Dai, Zhihao Zhang, Gabriele Oliaro, Zhihao Jia, and Ravi Netravali. 2025. Specreason: Fast and accurate inference-time compute via speculative reasoning. *Preprint*, arXiv:2504.07891.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *Preprint*, arXiv:2311.12022.

Jon Saad-Falcon, E. Kelly Buchanan, Mayee F. Chen, Tzu-Heng Huang, Brendan McLaughlin, Tanvir Bhathal, Shang Zhu, Ben Athiwaratkun, Frederic Sala, Scott Linderman, Azalia Mirhoseini, and Christopher Ré. 2025. Shrinking the generation-verification gap with weak verifiers. *Preprint*, arXiv:2506.18203.

Apoorv Saxena. 2023. Prompt lookup decoding.

Junhan Shi, Yijia Zhu, Zhenning Shi, Dan Zhao, Qing Li, and Yong Jiang. 2025. SpecCoT: Accelerating chain-of-thought reasoning through speculative exploration. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 24405–24415, Suzhou, China. Association for Computational Linguistics.

Tiancheng Su, Meicong Zhang, and Guoxiu He. 2025. Entropy-aware speculative decoding toward improved llm reasoning. *Preprint*, arXiv:2512.23765.

Dingzirui Wang, Xuanliang Zhang, Keyan Xu, Qingfu Zhu, Wanxiang Che, and Yang Deng. 2025. Bounds of chain-of-thought robustness: Reasoning steps, embed norms, and beyond. *Preprint*, arXiv:2509.21284.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Yifan Zhang and Team Math-AI. 2024. American invitational mathematics examination (aime) 2024.

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.

Yefan Zhou, Austin Xu, Yilun Zhou, Janvijay Singh, Jiang Gui, and Shafiq Joty. 2025. Variation in verification: Understanding verification dynamics in large language models. *Preprint*, arXiv:2509.17995.

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

A Compatibility with Token-Level Speculative Decoding

We provide a detailed analysis of ConfSpec's compatibility with different token-level speculative decoding methods, comparing Prompt Lookup Decoding and EAGLE-3 to demonstrate how the choice of token-level method impacts overall acceleration.

A.1 Setup

We evaluate two representative token-level speculative decoding approaches: (1) **Prompt Lookup Decoding (PLD)**, a draft-model-free method that maintains a cache of historical tokens and performs n-gram string matching to propose speculative drafts. At each decoding step, it identifies historical n-grams starting with the current sequence's suffix as candidates, which are then verified by the target model in a single forward pass. (2) **EAGLE-3**, a draft-model-based method that trains a lightweight draft head by leveraging multi-layer features from the target model. This alignment training enables close approximation of the target model's token distribution, achieving state-of-the-art token-level speculative decoding performance on multi-turn dialogue and short-form generation benchmarks.

We conduct experiments using Qwen3 1.7B / 32B as the draft / target model pair, comparing PLD and EAGLE-3 both as standalone methods and in combination with ConfSpec. All methods are evaluated on the same five benchmarks under identical hardware and generation configurations as described in the main experiments. Results are presented in Table 5.

A.2 Results and Analysis

PLD. PLD demonstrates strong standalone performance, achieving $1.37\times$ mean speedup while preserving lossless accuracy across all benchmarks. This effectiveness stems from its cache-based design that dynamically adapts to the generation context, effectively exploiting the repetitive patterns inherent in long chain-of-thought reasoning. Mathematical derivations and structured reasoning traces naturally contain recurring phrases, notation, and expression patterns that PLD's n-gram matching mechanism can efficiently capture and reuse.

EAGLE-3. Contrary to its strong results on conversational benchmarks, EAGLE-3 produces a net slowdown ($0.94\times$) on our reasoning tasks. We attribute this to a distributional mismatch between its training data and the nature of complex reasoning. EAGLE-3 is primarily optimized for short-form, predictable conversational tasks. In contrast, long-form mathematical reasoning involves multi-step logical derivations and specialized notation, with sequences frequently exceeding 16K tokens. This distributional shift causes the draft head to be poorly calibrated, leading to high rejection rates. Ultimately, the computational overhead of rejecting these drafts outweighs the gains from valid speculations.

Orthogonality. ConfSpec + PLD achieves multiplicative speedup gains ($1.65\times$), confirming that n-gram-based methods complement step-level speculation effectively by adapting to the repetitive structure of reasoning traces. In contrast, ConfSpec + EAGLE-3 ($1.06\times$) barely overcomes EAGLE-3's standalone slowdown, as EAGLE-3's inefficiency propagates through the entire pipeline: both target model generation steps and step-level verification are subject to high rejection rates, creating cascading degradation.

These results highlight that the choice of token-level technique should account for task characteristics. Draft-model-free methods like PLD demonstrate greater robustness for complex reasoning, while draft-model-based methods require distributional alignment with the target domain to avoid negating potential speedup gains.

B Computational Overhead and Memory Analysis

B.1 Per-Step Latency Breakdown

To analyze the sources of ConfSpec's acceleration, we profile the per-step latency of each component using DeepSeek-R1-Distill (1.5B draft / 32B target) on NVIDIA H200 GPUs:

The end-to-end acceleration stems from four complementary mechanisms:

Parallel Target Generation. Single-token autoregressive decoding is severely memory-bound, operating far below GPU peak compute density. By generating k target steps in parallel across cumulative contexts, ConfSpec amortizes this overhead—achieving k-step target generation in approximately the same wall-clock time as a single step ($\sim$409.8 ms). Successfully accepting multiple fast

Table 5: Compatibility with token-level speculative decoding methods. We report **Pass@1 Accuracy (%)** and **End-to-End Speedup** (×). The target model baseline represents the upper bound for accuracy and the baseline (1.00×) for speed.

Method	Dataset (Acc. ↑ / Speedup ↑)					
	AIME24	AMC23	MATH500	GPQA	HumanEval	Mean
Draft: Qwen3 1.7B / Target: Qwen3 32B						
Target Model	80.21 / 1.00×	97.34 / 1.00×	92.54 / 1.00×	67.83 / 1.00×	93.68 / 1.00×	86.32 / 1.00×
ConfSpec	**79.79 / 1.27×**	**96.41 / 1.35×**	**92.82 / 1.42×**	**67.53 / 1.18×**	**94.35 / 1.27×**	**86.18 / 1.30×**
EAGLE-3	≡ / 0.96×	≡ / 0.96×	≡ / 0.98×	≡ / 0.89×	≡ / 0.92×	≡ / 0.94×
ConfSpec + EAGLE-3	≡* / **1.07×**	≡* / **1.04×**	≡* / **1.06×**	≡* / **1.02×**	≡* / **1.09×**	≡* / **1.06×**
PLD	≡ / 1.39×	≡ / 1.36×	≡ / 1.34×	≡ / 1.39×	≡ / 1.39×	≡ / 1.37×
ConfSpec + PLD	≡* / **1.62×**	≡* / **1.73×**	≡* / **1.79×**	≡* / **1.54×**	≡* / **1.58×**	≡* / **1.65×**

Note: '≡' denotes lossless accuracy matching the target model. '≡*' denotes accuracy matching the base method.

Table 6: Per-step latency breakdown.

Component	Latency (ms)
Draft Step Generation	∼112.9
Target Step Generation	∼409.8
Draft Model Verification	∼8.7
Target Model Verification	∼45.9

draft steps (∼112.9 ms each) thus saves substantial time versus sequential target generation.

Asynchronous Execution. Step generation and semantic verification are executed in an asynchronous pipeline. As soon as a required prefix is ready, the target model immediately begins generating its corresponding step and triggers cascaded verification, effectively overlapping the workloads of both models and masking verification overhead.

Cascaded Verification Cost. Instead of unconditionally invoking the target model for verification (∼45.9 ms per step), ConfSpec routes the equivalence check first to the draft model (∼8.7 ms). By escalating to the target model only for uncertain cases, the average verification cost is substantially reduced.

Accuracy-Driven Speedup. Accurate verification sustains high acceptance rates across iterations. Erroneously accepting incorrect steps corrupts the reasoning trajectory, causing the draft model to produce misaligned subsequent drafts that are repeatedly rejected. By preventing such logical drift, ConfSpec avoids the severe latency penalties incurred by recovery loops in scoring-based methods.

B.2 Memory Footprint

Unlike frameworks that require external Process Reward Models (e.g., RSD) or third-party judge models (e.g., LookaheadReasoning), ConfSpec relies exclusively on the target and draft models. In our standard configuration, the target model (32B, FP16/BF16) requires approximately 70 GB of VRAM, while the draft model (1.5B) requires approximately 5 GB. The only additional dynamic memory overhead stems from KV cache, which scales with sequence length and parallel batch size. This lightweight footprint allows ConfSpec to be deployed on standard multi-GPU nodes without auxiliary models or specialized memory-offloading techniques.

C Theoretical Justification for Step-Level Semantic Speculation

Unlike token-level speculative decoding, which guarantees distributional equivalence via rejection sampling, step-level speculative reasoning relaxes exact token matching in favor of semantic equivalence. We ground this relaxation in three complementary theoretical perspectives.

Reasoning Path Equivalence. LLMs can converge to the same correct answer through diverse reasoning trajectories (Wang et al., 2023). A draft step that differs in surface form from the target model's output may simply represent an alternative, equivalent reasoning path. As long as cascaded verification ensures semantic alignment, the system switches to another valid trajectory without affecting final accuracy.

CoT Robustness and Information Redundancy. Recent studies on CoT faithfulness and robust-

Table 7: Effect of draft model size on ConfSpec performance (Qwen3 family, 32B target). Cell format: **Accuracy (%) / Speedup** (×).

Method	AIME24	AMC23	MATH500	GPQA	HumanEval	Mean
Target Model	80.21 / 1.00×	97.34 / 1.00×	92.54 / 1.00×	67.83 / 1.00×	93.68 / 1.00×	86.32 / 1.00×
ConfSpec (0.6B Draft)	28.33 / 1.79×	69.22 / 1.87×	68.73 / 2.04×	38.67 / 1.76×	69.65 / 1.95×	54.92 / 1.88×
ConfSpec (1.7B Draft)	**79.79 / 1.27**×	**96.41 / 1.35**×	**92.82 / 1.42**×	**67.53 / 1.18**×	**94.35 / 1.27**×	**86.18 / 1.30**×
ConfSpec (4B Draft)	80.00 / 1.17×	97.03 / 1.25×	92.84 / 1.27×	67.53 / 1.08×	93.70 / 1.19×	86.22 / 1.19×

ness (Lanham et al., 2023; Wang et al., 2025) reveal that LLMs possess significant information redundancy when deriving conclusions. Through intervention experiments—such as truncating, adding noise to, or replacing intermediate steps—these studies show that models are highly robust to local perturbations and can still arrive at the correct answer. Step-level speculation essentially performs a controlled local replacement, which falls well within the fault-tolerance threshold of CoT reasoning, especially when guarded by target-model verification.

Self-Reflection in Large Reasoning Models. Modern large reasoning models exhibit intrinsic self-reflection and error-correction capabilities (DeepSeek-AI et al., 2025a). These models can dynamically recognize and correct occasional intermediate deviations during generation. This inherent self-correction further mitigates the risk of minor semantic approximations introduced during the drafting phase.

Together, these perspectives—path equivalence, local robustness, and self-reflection—provide a strong theoretical foundation for step-level semantic speculation. Combined with our extensive empirical evidence across two model families and five diverse benchmarks, they demonstrate that relaxing strict token matching in favor of semantic equivalence is a fundamentally sound trade-off that unlocks significant latency reductions while preserving output accuracy.

D Effect of Draft Model Size

We investigate how draft model capacity affects ConfSpec's accuracy–speed trade-off. Using Qwen3-32B as the target model, we evaluate three draft model sizes from the same family (0.6B, 1.7B, and 4B) across five benchmarks. Results are presented in Table 7.

The 0.6B draft model achieves the highest speedup (1.88×) but suffers catastrophic accuracy collapse (e.g., 28.33% on AIME24 vs. the target's 80.21%), indicating that it lacks sufficient capacity to reliably judge semantic equivalence for complex reasoning steps, producing excessive false positives. The 4B draft model preserves accuracy (86.22% mean) but introduces heavier computational overhead during generation and verification, reducing speedup to 1.19×. Our default 1.7B draft model strikes the optimal balance, tightly preserving target-model accuracy (86.18%) while delivering meaningful acceleration (1.30×).

Based on these findings, we recommend that practitioners select a draft model from the same model family as the target to ensure stylistic and alignment consistency, with a parameter count of at least 1.5B to guarantee a sufficient level of reasoning and verification capability.