CSD: CONFIDENCE- AND STABILITY-GUIDED SPECULATIVE DECODING IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Speculative decoding has emerged as a promising technique to accelerate large language model inference by leveraging a lightweight draft model to propose candidate tokens for parallel verification by a target model. Although recent methods have explored adaptive draft-length control, efficiently estimating when the draft model can safely speculate more aggressively remains challenging. In this paper, we propose **C**onfidence- and **S**tability-guided speculative **D**ecoding (CSD), a training-free method that relies only on draft-model runtime signals, including token-level prediction confidence and recent local stability. CSD adopts a lightweight two-level framework to dynamically adjust the drafting boundary during decoding. Experiments across multiple model pairs and datasets show that CSD can be integrated into both standard and parallel speculative decoding without modifying the target verification rule and achieves a $3.09\times$–$6.41\times$ speedup over autoregressive decoding. CSD achieves competitive or better throughput than strong fixed-length and parallel speculative decoding baselines across the evaluated settings.

1 INTRODUCTION

Large language models (LLMs) have achieved remarkable progress in recent years. Foundation models Touvron et al. (2023); Bai et al. (2023); DeepSeek-AI et al. (2024) have demonstrated strong capabilities across a wide range of tasks, including code generation, mathematical reasoning, summarization, and machine translation Cobbe et al. (2021); Chen et al. (2021). However, such capabilities come with substantial computational costs. LLMs typically rely on autoregressive decoding, where generating each token requires a forward pass through the model. This sequential decoding process significantly limits inference speed and increases serving costs, especially in latency-sensitive deployment scenarios.

In this work, we focus on speculative decoding Chen et al. (2023), a promising paradigm for accelerating autoregressive generation. Speculative decoding improves inference efficiency by allowing the target model to verify multiple candidate tokens in a single forward pass. Specifically, a lightweight draft model first generates a sequence of candidate tokens conditioned on the current prefix. These candidates are then verified in parallel by the target model, rather than being generated one by one through repeated target-model invocations. When multiple draft tokens are accepted, the target model can skip several sequential decoding steps, thereby reducing the number of expensive forward passes and improving inference efficiency. We define a speculation round as one draft-and-verify iteration: the draft model proposes candidate tokens, the target model verifies them, and the accepted prefix is committed. A key factor in speculative decoding is the number of tokens the draft model should generate in each speculation round. The draft length directly affects both acceptance rate and decoding efficiency: overly long drafts increase the risk of rejection, while overly short drafts fail to fully exploit parallel verification. However, standard speculative decoding usually adopts a fixed draft length γ, implicitly assuming that speculation reliability remains stable throughout generation. This assumption often breaks down in practice, because the agreement between the draft and target models can vary significantly across decoding positions. As shown in Figure 1(a), a fixed $\gamma = 3$ can be too conservative in stable regions, where the draft model could safely propose more tokens. In contrast, the same $\gamma = 3$ can become too aggressive in unstable regions, leading to wasted draft computation.

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

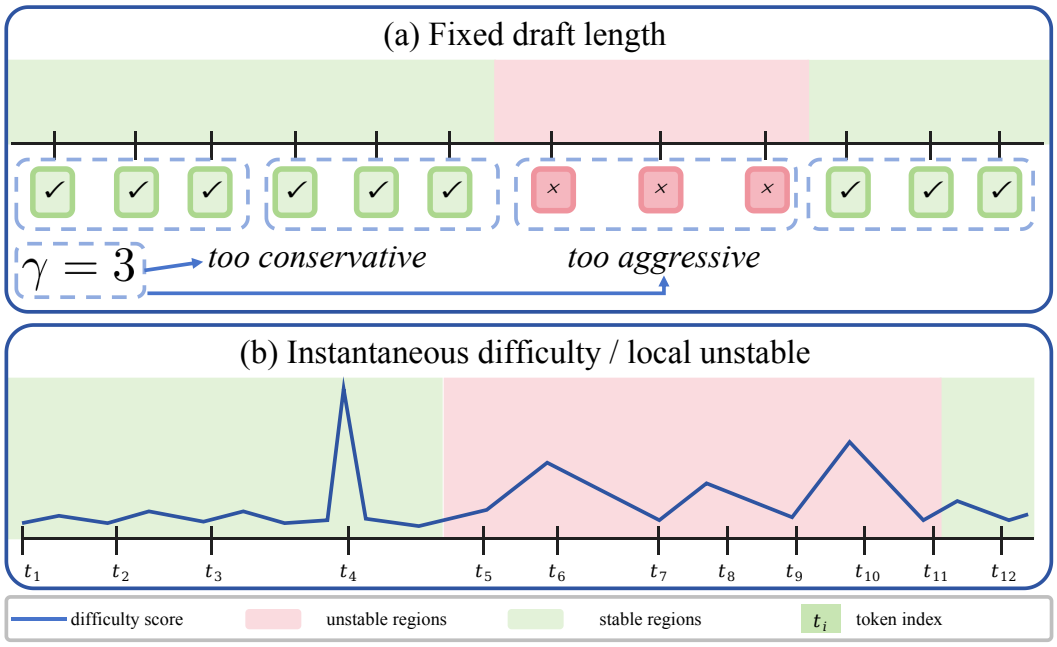

Figure 1: Fixed-length decoding leads to overly conservative drafts in stable regions and overly aggressive ones in unstable regions. Draft reliability can fail under instantaneous uncertainty or local instability.

This suggests that the limitation of fixed-length speculation is not caused by a single factor. Recent methods have explored adaptive draft length control. On-the-Fly Liu et al. (2025a) adjusts the speculation window based on historical acceptance statistics, which relies on accumulated feedback and may not fully capture the instantaneous reliability variation within the current generation process. AdaEDL Agrawal et al. (2024) employs an entropy-based criterion for early draft stopping, but entropy mainly reflects overall uncertainty and does not explicitly model the local stability of consecutive draft predictions. Rather than introducing adaptive draft-length control itself, CSD focuses on exploiting lightweight draft-side signals from the current inference process to characterize drafting reliability, including instantaneous prediction confidence and recent local stability.

Autoregressive LLM generation can be viewed as a time-varying and non-stationary discrete stochastic process. In this process, the uncertainty of individual token predictions changes dynamically, while prediction reliability tends to be locally correlated across neighboring tokens. We identify two major sources of failure in speculative decoding. The first is high instantaneous uncertainty, where the current token is inherently difficult for the draft model to predict. The second is local instability, where the current token may still appear confident, but the surrounding generation region has become unstable, making further speculation more likely to be rejected by the target model. As shown in Figure 1(b), the sharp spike around t_4 reflects high instantaneous difficulty, while the unstable region from t_7 to t_{11} indicates that even seemingly confident tokens may lie in a locally unreliable generation segment.

To empirically validate these two failure modes, we further analyze token-level traces from fixed-γ speculative decoding. Using the DeepSeek-Coder-1.3B & 33B draft-target pair, we analyze 66,354 target-verified draft tokens across GSM8K Cobbe et al. (2021), HumanEval Chen et al. (2021), and CNN/DailyMail Hermann et al. (2015). We find that acceptance increases substantially with draft confidence, ranging from 50.1%–80.0% in the lowest-confidence quintile to nearly 100% in the highest-confidence quintile. Meanwhile, acceptance decreases as local instability increases, from 99.79%–100.00% in the most stable quintile to 70.84%–89.70% in the most unstable quintile. After controlling for confidence, more stable tokens still show higher acceptance on GSM8K and CNN/DailyMail, suggesting that instantaneous confidence and local stability provide complementary information for characterizing draft reliability.

Based on these observations, we propose **C**onfidence- and **S**tability-guided speculative **D**ecoding (CSD), a simple yet effective method that adapts the drafting boundary using draft-model confidence and local stability as lightweight reliability surrogates. CSD is training-free and compatible with existing speculative decoding systems. It introduces two lightweight modules: Round-level Draft Length Control determines a draft length for each speculation round, while Token-level Late Draft Extension refines this boundary. Together, these modules allow CSD to shorten drafts in uncertain or unstable regions and extend drafts in confident and stable regions, without modifying the target-model verification rule.

In summary, this work makes the following contributions:

- We introduce lightweight, training-free reliability signals based on instantaneous draft-model confidence and recent local stability, enabling online assessment of drafting reliability without an additional predictor.
- We propose a two-level drafting-boundary control mechanism that combines round-level planning with token-level late extension to adapt speculation at different granularities.
- CSD can be integrated into both standard and parallel speculative decoding without modifying the target verification rule. Experiments on multiple model pairs and datasets demonstrate that CSD achieves strong inference efficiency, with a $3.09\times$–$6.41\times$ speedup over autoregressive decoding.

2 Related Work

2.1 Training-based Speculative Decoding

Speculative decoding is highly sensitive to draft-target alignment: poor alignment or test-time distribution mismatch lowers draft-token acceptance and limits speedup. To address this, training-based methods improve the draft model itself. DistillSpec Zhou et al. (2024) uses white-box distillation for better alignment; EAGLE-3 Li et al. (2026) trains an additional draft module using multi-layer target-model features; and Online Speculative Decoding Liu et al. (2023) adapts the draft model to evolving serving distributions. However, these methods often require extra training data, distillation, architectural changes, or online updates, and may need retraining or recalibration across models, domains, or serving environments, raising concerns about cost, deployment complexity, and transferability.

2.2 Training-free Speculative Decoding

Training-free speculative decoding improves inference efficiency without training additional models or modifying the target model. The original speculative decoding framework is a representative example: it uses an off-the-shelf smaller draft model and preserves the target-model distribution through exact verification. Following this paradigm, recent studies have explored more efficient candidate organization and decoding policies. SpecBranch Shen et al. (2026) introduces rollback-aware branch resampling and hybrid draft-length modeling to overlap drafting and verification while reducing wasted rollback tokens. On-the-Fly Liu et al. (2025a) dynamically selects the speculation window size and draft model based on runtime statistics. AdaEDL Agrawal et al. (2024) performs early draft stopping by estimating a lower bound on token acceptance probability from draft-logit entropy. PEARL Liu et al. (2025b) mitigates the mutual waiting between draft and target models through pre- and post-verification with dynamic draft lengths. These methods reduce the deployment cost of speculative decoding and improve its flexibility in practical inference systems.

Despite these advances, existing methods still have several limitations. Entropy-based approaches such as AdaEDL rely on uncertainty estimation but may not capture the local stability of consecutive predictions during generation. Meanwhile, acceptance-based methods such as On-the-Fly depend on historical feedback and may be less responsive to instantaneous changes in drafting reliability. Therefore, how to better exploit lightweight draft-side signals from the current inference process remains an important yet underexplored problem.

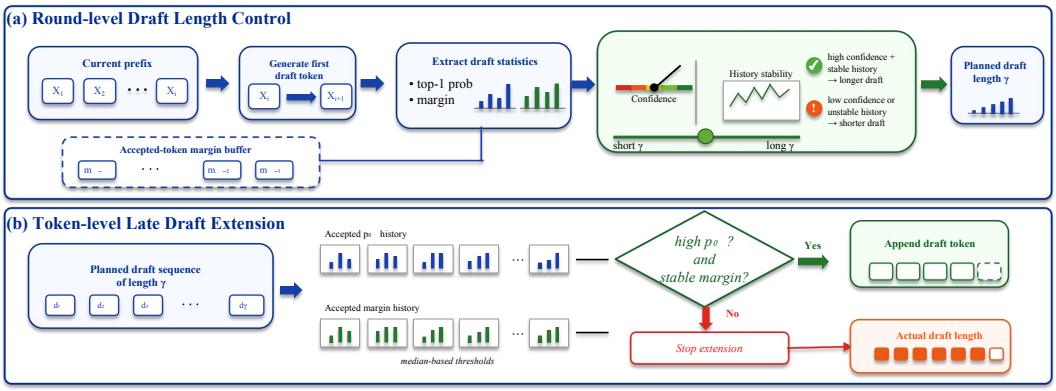

Figure 2: Overview of the Proposed Drafting Framework

3 METHODOLOGY

Standard speculative decoding (SD) typically uses a fixed draft length γ. Given the current prefix, the draft model generates γ candidate tokens, which are verified in parallel by the target model in a single forward pass. Although this design is simple and effective, it implicitly assumes that the fixed draft length is suitable for all decoding positions. However, both the generation difficulty and the agreement between the draft and target models can vary substantially across different segments of the output. As a result, a fixed draft length makes the method overly conservative in easy regions and overly aggressive in difficult regions, which results in more frequent rejections and inefficient verification.

3.1 ROUND-LEVEL DRAFT LENGTH CONTROL

To address the limitations of fixed γ, we introduce a round-level draft length control mechanism, which is training-free and does not change the verification procedure of standard SD, as shown in Figure 2(a). Instead of relying on a predetermined draft length, we exploit distributional statistics that are already available during draft generation to dynamically determine the draft length for each round.

Specifically, at each token generation position, we extract two statistics from the draft-model probability distribution: the top-1 probability p_0, which reflects prediction confidence, and the top-1/top-2 probability margin m, which measures the separation between the most likely candidate and its runner-up. We further characterize recent local stability using the first-order variation of margins over a buffer of length h containing only target-accepted tokens. Given the recent margin sequence $\mathcal{B}_t$, we define the local sensitivity S_t as the average absolute change between adjacent margins; when fewer than two accepted-token margins are available, we set $S_t = 0$ and rely only on instantaneous confidence.

$$S_t = \frac{1}{|\mathcal{B}_t| - 1} \sum_{i=2}^{|\mathcal{B}_t|} |m_i - m_{i-1}|, \tag{1}$$

where m_i denotes the margin of the i-th accepted token in the buffer.

Our reliability-based control is conceptually motivated by prior work on adaptive speculation. In particular, SpecDec++ formulates draft-length selection as a sequential decision problem and shows, under its assumptions, that the optimal policy stops drafting when the risk of continuing becomes sufficiently high. CSD follows this general principle of reliability-aware drafting control, but does not attempt to estimate the exact acceptance probability. Instead, it constructs a lightweight, training-free reliability surrogate from draft-side runtime signals.

We combine the current top-1 probability p_0 and the local sensitivity S_t into a reliability score:

$$score_t = p_{0,t}^{\alpha} \cdot \exp(-\beta S_t), \tag{2}$$

where α controls the influence of the instantaneous confidence, and β controls the strength of the penalty imposed by local instability.

We emphasize that $score_t$ is a lightweight reliability surrogate rather than a calibrated estimate of the target-model acceptance probability. It is designed only to guide the drafting boundary using signals available from the draft model before target verification.

With high p_0 and low S_t, $score_t$ becomes larger, indicating that the draft model is more reliable in the current context. Conversely, when p_0 is low or S_t is high, $score_t$ decreases, suggesting that the current decoding position is less suitable for aggressive drafting. Finally, we linearly map this score to the allowable range of draft lengths:

$$\gamma_t = \text{round}\left(\gamma_{\min} + (\gamma_{\max} - \gamma_{\min}) \cdot score_t\right), \tag{3}$$

and clip the resulting value to the interval $[\gamma_{\min}, \gamma_{\max}]$. With this design, the draft length can be dynamically adjusted according to the evolving generation state. In stable regions, the method tends to increase γ, allowing each target verification step to cover more candidate tokens. In unstable regions, it instead tends to decrease γ, thereby reducing rejected tokens and avoiding ineffective draft generation.

3.2 Token-level Late Draft Extension

Round-level draft length control determines the planned number of draft tokens at the beginning of each round. However, once the planned length γ is set, the draft model stops after generating exactly γ tokens. While this is useful in unstable regions to avoid excessive rejection, it can be overly conservative in high-confidence and stable regions, where further draft tokens may still be reliable.

To address this limitation, we introduce a token-level draft extension mechanism, as shown in Figure 2(b). When draft generation reaches the planned length γ, we allow the draft model to generate a few additional tokens if the current state remains reliable. Specifically, we build an adaptive reliability baseline from recently accepted tokens. Let $\mathcal{H}_t$ denote the cache of recent accepted-token statistics. We compute the medians of the top-1 probabilities and margins in this cache, respectively:

$$\tau_p^{(t)} = \underset{i \in \mathcal{H}_t}{\text{Median}}\, p_{0_i}, \tag{4}$$

$$\tau_m^{(t)} = \underset{i \in \mathcal{H}_t}{\text{Median}}\, m_i. \tag{5}$$

where $\tau_p^{(t)}$ and $\tau_m^{(t)}$ denote the recent baselines of the top-1 probability and margin for accepted tokens, respectively. If the last generated draft token satisfies the following condition:

$$p_{0,t} \geq \tau_p^{(t)} \quad \text{and} \quad m_t \geq \tau_m^{(t)}, \tag{6}$$

we consider the draft model to remain locally reliable at the current position and allow it to append one additional draft token. Otherwise, the current draft stage is terminated and the target verification step is performed. This criterion is repeatedly applied during extension until the reliability condition is no longer satisfied or the number of extra extension steps reaches the predefined maximum of 5.

3.3 Integration with Speculative Decoding

CSD is implemented as two lightweight modules in the draft generation stage. Since it only relies on probability-related statistics from the draft model, it can be integrated into existing speculative decoding frameworks in a plug-and-play manner, without modifying the target model or changing the target verification rule. In this work, we instantiate CSD in both standard speculative decoding Leviathan et al. (2023) and parallel speculative decoding Liu et al. (2025b).

Method	Llama3.1 8B & 70B	CodeLlama 7B & 70B	Llama2 7B & 70B	DeepSeek-Coder-1.3B & 33B	Qwen3 0.6B & 32B
Autoregressive	1.00×	1.00×	1.00×	1.00×	1.00×
Speculative Decoding	3.18×	3.22×	3.01×	3.09×	2.01×
On-The-Fly	-	-	3.26×	-	-
PEARL	4.25×	3.81×	3.98×	3.71×	2.79×
SpecBranch	4.21×	3.83×	5.25×	3.84×	2.96×
CSD (Ours)	6.41×	4.95×	5.58×	5.15×	3.09×

Table 1: Speedup comparison across different draft-target model pairs on HumanEval dataset. Each model pair is denoted as "draft model & target model", where the smaller model is used as the draft model and the larger model is used as the target model.

With standard speculative decoding, the draft model first generates one token and obtains the probability and margin from the distribution at this position. Then, we determine the draft length γ_t for the current round according to the p_0 and S_t of this token, as mentioned above. After obtaining γ_t, the draft model continues to generate the remaining $\gamma_t - 1$ tokens, forming the planned draft sequence for the current round. We further apply Token-level Late Draft Extension based on historical accepted-token statistics before target verification.

With parallel speculative decoding, the draft process computes a planned draft length γ_t for the current round from its locally maintained buffer of accepted-token statistics, and synchronizes this planned length to the target process. This length controls how many new candidate tokens the draft process attempts to generate. During generation, the Token-level Late Draft Extension module may produce an actual draft length γ_t', which is then synchronized and used for token packing and for determining the verification length in the next pipeline round.

These implementations show that CSD can be attached to both serial and parallel speculative decoding pipelines while preserving the original target-model verification procedure.

4 Experiments

4.1 Implementation Details

To evaluate CSD, we conduct experiments under both standard speculative decoding (SD) and parallel SD settings. In standard SD, the draft model autoregressively generates candidate tokens for verification by the target model in one forward pass. In parallel SD, we build on an existing parallel SD framework Liu et al. (2025b), where draft generation and target verification run concurrently. CSD is applied only to the draft stage, dynamically selecting the draft length and allowing limited extension in locally reliable regions. Since the target verification procedure remains unchanged, CSD preserves the lossless property of speculative decoding and the target-model output distribution.

We evaluate CSD on DeepSeek, Qwen3, Llama2, and CodeLlama, using the smaller model in each series as the draft model and the larger one as the target model. The evaluation covers HumanEval Chen et al. (2021), GSM8K Cobbe et al. (2021), CNN/DailyMail Hermann et al. (2015), and ShareGPT Chiang et al. (2023) for open-domain dialogue. We use the full HumanEval and GSM8K evaluation sets, and randomly sample about 200 instances from CNN/DailyMail and ShareGPT to keep their evaluation sizes comparable to HumanEval while controlling experimental cost.

All experiments are conducted on 8× NVIDIA L40S GPUs. Due to memory constraints, the maximum generation length is set to 1024 for the Qwen series and 512 for the Llama and CodeLlama series. Unless otherwise specified, we use a history window size of $h = 16$ and set the reliability-score hyperparameters to $\alpha = 1$ and $\beta = 5$, without dataset-specific tuning. Decoding throughput, measured in generated tokens per second, is used as the primary efficiency metric.

For fair comparison, all methods are evaluated in our local environment under the same hardware, model, dataset, and decoding configurations. Prior methods with publicly available implementations are reproduced on our machines.

For parallel speculative decoding, we use the official PEARL implementation Liu et al. (2025b), retaining its pre-verification, post-verification, concurrent draft/target execution, and pipeline-control mechanisms. The PEARL baseline uses its default base window size $\gamma = 5$, while CSD modifies

Dataset	Method	Round-level Draft Length Control	Token-level Late Draft Extension	Window Size	Draft Length	Throughput
HumanEval	PEARL			–	5	19.92
	+ Token-level Control		✓			25.02
	+ Round-level Control	✓		16	2–10	24.46
	Full CSD		✓			27.91
GSM8K	PEARL			–	5	18.55
	+ Token-level Control		✓			23.38
	+ Round-level Control	✓		16	2–10	21.71
	Full CSD		✓			24.79
CNN/DailyMail	PEARL			–	5	17.01
	+ Token-level Control		✓			19.12
	+ Round-level Control	✓		16	2–10	18.61
	Full CSD		✓			20.43
ShareGPT	PEARL			–	5	18.89
	+ Token-level Control		✓			23.77
	+ Round-level Control	✓		16	2–10	24.41
	Full CSD		✓			26.94

Table 2: Ablation study of Round-level Draft Length Control and Token-level Late Draft Extension under parallel speculative decoding. All experiments are conducted with Llama2 7B & 70B. Throughput is measured by generated tokens per second.

only the drafting boundary and preserves PEARL's execution pipeline and target-side verification rule.

4.2 Main Results

To evaluate the overall inference efficiency of CSD, we compare it with autoregressive decoding, standard speculative decoding, and recent strong baselines Liu et al. (2025b); Shen et al. (2026). Table 1 reports the results on HumanEval across different draft-target model pairs. Except for On-the-Fly, all reported results are obtained in our local environment under identical hardware, model, dataset, and decoding configurations. Since the official implementation of On-the-Fly is not publicly available, we report its result from the original paper for reference; this number is therefore not a strictly controlled comparison.

As shown in Table 1, CSD achieves the best speedup among the compared methods with publicly available implementations across all evaluated model pairs on HumanEval. Compared with autoregressive decoding, CSD delivers a $3.09\times$–$6.41\times$ speedup, demonstrating that dynamically controlling the drafting boundary can substantially reduce the cost of autoregressive generation while preserving the target-model verification rule. Moreover, CSD consistently outperforms recent strong speculative decoding baselines, showing that confidence-stability-aware draft control provides additional efficiency gains beyond existing parallel decoding strategies. A fixed-length strategy may be overly conservative in stable regions or overly aggressive in unstable regions, leading to suboptimal draft-token utilization.

Since speculative decoding is expected to preserve the behavior of the target model, we additionally compare target-only autoregressive decoding and CSD on HumanEval using DeepSeek-Coder-1.3B/33B. Both methods use identical prompts, tokenizer, stopping criteria, and greedy decoding settings. Target-only decoding and CSD both solve 114 out of 164 problems, corresponding to an identical pass@1 of 69.51%.

4.3 Module Ablation Study

To evaluate the contribution of the two core components in CSD, we conduct an ablation study under the parallel speculative decoding setting. All experiments in this subsection use Llama2-7B as the draft model and Llama2-70B as the target model. We report decoding throughput on HumanEval, GSM8K, CNN/DailyMail, and ShareGPT, as shown in Table 2.

In this ablation, we examine the effects of Round-level Draft Length Control and Token-level Late Draft Extension. When round-level control is disabled, the method uses a fixed draft length of $\gamma = 5$; when enabled, CSD selects γ from $[2, 10]$ according to the confidence-stability score, with the history window size set to $h = 16$. This range keeps the fixed baseline near the center while allowing shorter drafts in unstable regions and longer drafts in reliable regions. The results show that both modules are effective. Token-level Late Draft Extension consistently improves throughput

over PEARL, suggesting that fixed drafting boundaries can be overly conservative in locally reliable regions. Round-level Draft Length Control also brings clear gains, indicating that per-round draft length selection better adapts to changing generation states than a single fixed γ.

Full CSD achieves the best performance across all datasets. These results show that the two modules are complementary: round-level control sets a suitable planned boundary, while late extension further exploits locally reliable regions. Overall, CSD improves draft-token utilization by balancing drafting aggressiveness and verification reliability, rather than simply increasing the draft length.

Strategy	Throughput	Acc. Rate	Avg. Acc. Len.	Draft Fwd.	Target Fwd.
CSD w/o Late Ext.	27.15	95.54	7.73	157372	19451
CSD	28.31	94.78	9.28	161878	16526
Fixed $\gamma = 6$	24.38	91.11	5.46	157302	26217
Fixed $\gamma = 10$	26.59	85.87	8.58	176960	17696

Table 3: Performance analysis under speculative decoding.

4.4 Performance Analysis

To better understand how CSD affects speculative decoding, we analyze different drafting strategies on CNN/DailyMail using DeepSeek-Coder-6.7B & 33B, reporting acceptance rate, average accepted length, and draft- and target-model forward passes in Table 3.

Compared with fixed $\gamma = 6$, round-level control alone (CSD w/o Late Extension) increases draft-model forward passes by only 70, from 157,302 to 157,372, while reducing target-model forward passes from 26,217 to 19,451. This indicates that its gain mainly comes from selecting more effective drafting boundaries rather than generating more draft tokens. Full CSD increases draft-model forward passes to 161,878 through Late Draft Extension, but further reduces target-model forward passes to 16,526 and achieves the highest throughput of 28.31 tokens/s. Thus, round-level control improves verification efficiency with little additional draft-side computation, while Late Draft Extension trades modest extra draft computation for further speedup.

The acceptance statistics show a similar pattern. Compared with fixed $\gamma = 6$, round-level control improves both acceptance rate and average accepted length. Simply increasing the fixed draft length to $\gamma = 10$ raises the average accepted length to 8.58, but lowers the acceptance rate to 85.87% and increases draft-model forward passes to 176,960. These results show that CSD does not benefit from blindly generating longer drafts; instead, it adapts the draft boundary to the current generation state, drafting more aggressively in confident and stable regions and more conservatively in uncertain or unstable ones. This enables CSD to reduce expensive target-model invocations while maintaining high acceptance quality with limited additional draft-side overhead.

4.5 Robustness under Domain-Mismatched Draft Models

To further evaluate the robustness of CSD across different tasks, we conduct experiments on the CodeLlama model family using three non-code datasets: GSM8K, CNN/DailyMail, and ShareGPT. We compare PEARL, Token-level Late Draft Extension, Round-level Draft Length Control, and the full CSD method.

As shown in Table 4, both components consistently improve throughput across tasks. Token-level Late Draft Extension demonstrates that fixed draft boundaries can be overly conservative, while Round-level Draft Length Control further adapts speculation to changing generation states. The full CSD achieves the best performance in all settings, showing the complementarity of the two modules and its generalization across tasks.

4.6 Comparison with Fixed Draft Lengths

A natural question is whether a carefully tuned fixed draft length can close the performance gap with CSD. To examine this, we compare CSD with a range of fixed draft lengths on CNN/DailyMail using DeepSeek-Coder-6.7B & 33B. As shown in Table 5, the throughput of fixed-length speculative

Dataset	Method	Round-level	Token-level	Draft Length	Throughput
GSM8K	PEARL			5	18.20
	+ Token-level Control		✓		23.14
	+ Round-level Control	✓✓		2–10	21.49
	Full CSD		✓		24.26
CNN/DailyMail	PEARL			5	17.08
	+ Token-level Control		✓		19.76
	+ Round-level Control	✓✓		2–10	19.48
	Full CSD		✓		21.08
ShareGPT	PEARL			5	20.04
	+ Token-level Control		✓		27.32
	+ Round-level Control	✓✓		2–10	26.75
	Full CSD		✓		29.98

Table 4: Cross-task ablation results on non-code tasks with CodeLlama 7B & 70B. The smaller model is used as the draft model and the larger model is used as the target model.

decoding varies noticeably with γ and does not improve monotonically as the draft length increases. Among the tested settings, $\gamma = 6$ achieves the best fixed-length throughput of 36.12 tokens/s, while CSD achieves 36.45 tokens/s using the adaptive range $\gamma \in [2, 10]$.

Method / Draft Length (γ)	Throughput
CSD (with range 2-10)	**36.45**
4	31.70
6	36.12
8	35.61
10	33.68
12	32.39
14	31.08
16	29.97

Table 5: Comparison between CSD and fixed draft lengths on the CNN/DailyMail dataset using DeepSeek-Coder 6.7B & 33B.

These results indicate that the gain of CSD is not simply due to an unfavorable choice of fixed draft length. Increasing γ enlarges the candidate set but can also introduce more rejected draft tokens and additional draft-side computation, leading to a non-monotonic throughput trend. In contrast, CSD adapts the drafting boundary according to the current generation state and achieves performance comparable to or better than the best searched fixed-length setting without selecting a dataset-specific fixed γ. Additional sensitivity analyses of the reliability-score hyperparameters, late-extension statistics, and history window size are provided in the appendix.

5 Conclusion

In this paper, we show that the effectiveness of speculative decoding is fundamentally limited by the varying reliability of draft generation, which is mainly affected by instantaneous prediction uncertainty and local generation instability. Based on this observation, we propose CSD, a training-free speculative decoding framework that exploits lightweight draft-side signals to adapt the drafting boundary during inference. By combining round-level draft length control with token-level late draft extension, CSD enables more reliable speculation without modifying the target-model verification rule. Experiments across multiple model pairs and generation tasks demonstrate consistent efficiency improvements, achieving 3.09×–6.41× speedup over autoregressive decoding and outperforming strong fixed-length and adaptive speculative decoding baselines.

6 AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on research methodology and experimental design, assist with the interpretation of experimental results, and assist with translation during manuscript preparation. We did not use generative AI tools for the other required disclosure tasks, or those tasks were not applicable to this work. All AI-assisted content was reviewed and verified by the authors. We take responsibility for the final content of this work.

REFERENCES

Sudhanshu Agrawal, Wonseok Jeon, and Mingu Lee. AdaEDL: Early draft stopping for speculative decoding of large language models via an entropy-based lower bound on token acceptance probability. In Mehdi Rezagholizadeh, Peyman Passban, Soheila Samiee, Vahid Partovi Nia, Yu Cheng, Yue Deng, Qun Liu, and Boxing Chen (eds.), *Proceedings of The 4th NeurIPS Efficient Natural Language and Speech Processing Workshop*, volume 262 of *Proceedings of Machine Learning Research*, pp. 355–369. PMLR, 2024. URL https://proceedings.mlr.press/v262/agrawal24a.html.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, and Fei Huang. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, and Greg Brockman. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, and Joseph E Gonzalez. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna.lmsys.org (accessed 14 April 2023)*, 2(3):6, 2023.

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

DeepSeek-AI, Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, and Qiushi Du. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28, 2015.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pp. 19274–19286. PMLR, 2023.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle-3: Scaling up inference acceleration of large language models via training-time test. *Advances in Neural Information Processing Systems*, 38:136737–136756, 2026.

Jiesong Liu, Brian Park, and Xipeng Shen. A drop-in solution for on-the-fly adaptation of speculative decoding in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9778–9794, Vienna, Austria, 2025a. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.482. URL https://aclanthology.org/2025.acl-long.482/.

Tianyu Liu, Yun Li, Qitan Lv, Kai Liu, Jianchen Zhu, Winston Hu, and Xiao Sun. Pearl: Parallel speculative decoding with adaptive draft length. In *International Conference on Learning Representations*, volume 2025, pp. 1085–1104, 2025b.

Xiaoxuan Liu, Lanxiang Hu, Peter Bailis, Alvin Cheung, Zhijie Deng, Ion Stoica, and Hao Zhang. Online speculative decoding. *arXiv preprint arXiv:2310.07177*, 2023.

Yuhao Shen, Junyi Shen, Quan Kong, Tianyu Liu, Yao Lu, and Cong Wang. Specbranch: Speculative decoding via hybrid drafting and rollback-aware branch parallelism. In *The Fourteenth International Conference on Learning Representations*, 2026.

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. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. Distillspec: Improving speculative decoding via knowledge distillation. In *International Conference on Learning Representations*, volume 2024, pp. 32011–32050, 2024.

A Sensitivity to Reliability-Score Hyperparameters

We further analyze the sensitivity of CSD to the reliability-score hyperparameters α and β. In the reliability score, α controls the contribution of the instantaneous confidence term, while β controls the penalty strength for local instability. The experiments are conducted with DeepSeek-Coder-6.7B & 33B as the model pair on the CNN/DailyMail dataset. The adaptive draft length range is fixed to $\gamma \in [2, 10]$. To isolate the effect of the reliability-score hyperparameters, we disable Token-level Late Draft Extension in this analysis.

Experiment	α	β	Throughput
β sensitivity	1.0	0	30.91
		1	32.44
		2	33.45
		3	33.76
		5	**34.36**
		8	32.39
α sensitivity	0	5	33.27
	0.5		33.90
	1.0		**34.36**
	1.5		33.41
	2.0		33.27

Table 6: Sensitivity analysis of the reliability-score hyperparameters α and β. Throughput is measured by generated tokens per second.

As shown in Table 6, we first fix $\alpha = 1.0$ and vary β to examine the effect of the local-stability penalty. When $\beta = 0$, the reliability score degenerates into a confidence-only criterion. This setting achieves lower throughput than the non-zero β settings. As β increases from 0 to 5, throughput improves from 30.91 to 34.36 tokens per second, and then decreases when β is further increased. This indicates that local stability provides useful information beyond instantaneous confidence, but an overly strong stability penalty may make draft length selection too conservative. These results support our design choice of incorporating recent margin variation into the draft-length control process.

We then fix $\beta = 5$ and vary α to examine the effect of the instantaneous confidence term. When $\alpha = 0$, the reliability score relies only on the local-stability penalty. This setting achieves a throughput of 33.27 tokens per second, which is lower than the 34.36 tokens per second obtained with $\alpha = 1.0$, suggesting that instantaneous confidence also contributes to draft length control. As α increases from 0 to 1.0, throughput gradually improves. However, further increasing α to 1.5 or 2.0 leads to lower throughput. This indicates that a moderate confidence weighting helps improve the effectiveness of draft length selection, while overemphasizing the current token confidence may weaken the role of local stability and make the policy less balanced.

Statistic	Throughput	Trigger Rate	Additional-token Acc.
Mean	**44.17**	54.77%	69.65%
Median	42.83	41.98%	72.53%
80th percentile	41.06	24.50%	**79.04%**

Table 7: Sensitivity of token-level late extension to different threshold statistics on GSM8K using DeepSeek-Coder-1.3B/33B. Trigger Rate denotes the proportion of positions at which late extension is activated, and Additional-token Acc. denotes the acceptance rate of tokens generated through late extension.

Overall, these results show that both instantaneous confidence and local stability contribute to the effectiveness of CSD.

B Sensitivity to Late-Extension Statistics

To examine the effect of different statistics used to determine the token-level late-extension threshold, we compare the mean, median, and 80th percentile of the recent confidence and margin statistics. Experiments are conducted using DeepSeek-Coder-1.3B/33B on GSM8K, while keeping all other settings unchanged. We report the decoding throughput, the proportion of positions at which late extension is triggered (Trigger Rate), and the acceptance rate of the additionally drafted tokens.

As shown in Table 7, different statistics lead to different levels of drafting aggressiveness. Using the mean produces the highest trigger rate (54.77%) and the highest throughput (44.17 tokens/s), but also results in the lowest acceptance rate for additionally drafted tokens (69.65%). In contrast, the 80th percentile yields a substantially more conservative extension policy, reducing the trigger rate to 24.50% while increasing the additional-token acceptance rate to 79.04%. The median lies between these two alternatives, achieving a trigger rate of 41.98% and an additional-token acceptance rate of 72.53%.

These results indicate that the statistic used for late-extension control determines the trade-off between extension frequency and the reliability of additionally drafted tokens. We therefore do not claim that the median universally maximizes throughput. Instead, we adopt it as a robust and relatively neutral default that avoids the aggressive behavior induced by the mean and the conservative behavior induced by a high percentile. Importantly, the median is fixed across all main experiments rather than selected post hoc for individual datasets or model pairs.

C Sensitivity to History Window Size

We further study the effect of the history window size in Table 8. With the same draft length range of $\gamma \in [4, 16]$, all dynamic settings outperform the fixed $\gamma = 10$ baseline, indicating that confidence-stability-aware draft length control is consistently beneficial. Among different window sizes, $h = 16$ achieves the best throughput.

Method	Window Size	Draft Length Range	Throughput
Fixed γ	–	10	27.47
CSD	32	4–16	29.29
	16		**30.07**
	8		28.50

Table 8: Effect of history window size on Qwen3-HumanEval. Throughput is measured by generated tokens per second.

A smaller window such as $h = 8$ reacts more quickly to local changes, but it may also be more sensitive to short-term fluctuations. In contrast, a larger window such as $h = 32$ provides smoother

statistics but may respond more slowly to changes in the generation state. Therefore, a moderate window size provides a better balance between responsiveness and stability.