

Consultant Decoding: Yet Another Synergistic Mechanism

Chuanghao Ding¹, Jiaping Wang², Ziqing Yang³,
Xiaoliang Wang¹, Dahua Lin⁴, Cam-Tu Nguyen^{1*}, Fei Tan^{2*}

¹State Key Laboratory for Novel Software Technology, Nanjing University,
²East China Normal University, ³Independent Researcher, ⁴The Chinese University of Hong Kong
ch777.ding@smail.nju.edu.cn 51265902031@stu.ecnu.edu.cn {waxili, ncamtu}@nju.edu.cn
ziqingyang@gmail.com dhlin@ie.cuhk.edu.hk tanfei2007@gmail.com

Abstract

The synergistic mechanism based on Speculative Decoding (SD) has garnered considerable attention as a simple yet effective approach for accelerating the inference of large language models (LLMs). Nonetheless, the high rejection rates require repeated LLMs calls to validate draft tokens, undermining the overall efficiency gain of SD. In this work, we revisit existing verification mechanisms and propose a novel synergetic mechanism *Consultant Decoding* (CD). Unlike SD, which relies on a metric derived from importance sampling for verification, CD verifies candidate drafts using token-level likelihoods computed solely by the LLM. CD achieves up to a 2.5-fold increase in inference speed compared to the target model, while maintaining comparable generation quality ($\sim 100\%$ of the target model’s performance). Interestingly, this is achieved by combining models whose parameter sizes differ by two orders of magnitude. In addition, CD reduces the call frequency of the large target model to below 10%, particularly in more demanding tasks. CD’s performance was even found to surpass that of the large target model, which theoretically represents the upper bound for speculative decoding.

1 Introduction

Large Language Models (LLMs) have attracted widespread attention and adoption across diverse domains, owing to their exceptional performance and robust generalization abilities (Yang et al., 2024a; Lu et al., 2023; Wang et al., 2024b; Zhang et al., 2024; Gu et al., 2024; Li et al., 2023a; Dubey et al., 2024). As their capabilities continue to evolve, so does their scale, with some reaching unprecedented sizes. Notable examples include LLaMA3.1-405B (Dubey et al., 2024) and DeepSeek-V3/R1 models (Liu et al., 2024). The

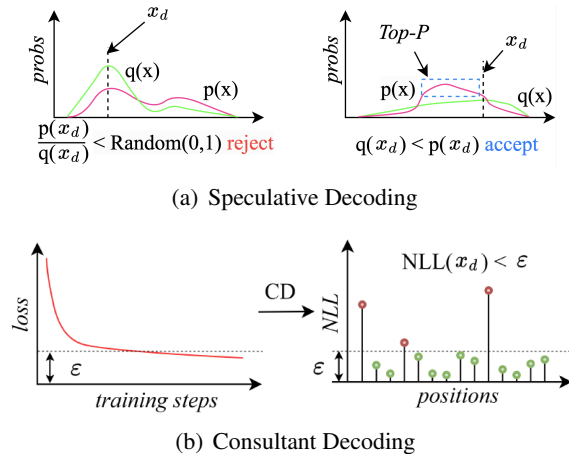


Figure 1: (a) A visual description of the sub-optimal problem of distribution-based method, where the blue box is high-quality nucleus of Top-P sampling and the x_d is the draft token. (b) The CD verification algorithm. $q(x)$ and $p(x)$ denote distribution of draft model and target model, respectively. The ϵ is the approximate convergence loss in the training phase of target model.

growing size of large models is making inference efficiency an increasingly pressing concern.

Recently, Speculative Decoding (SD) (Leviathan et al., 2023; Chen et al., 2023), based on the draft-and-verify paradigm, has been introduced to accelerate the decoding process of large language models (LLMs). It exploits the key property that parallel processing can improve GPU utilization and reduce the average latency during autoregressive decoding. SD maintains distribution consistency between the sampling results and target models. This advantage has spurred the development of numerous improved methods (Li et al., 2024b,a; Zafrir et al., 2024; Hu and Huang; Narasimhan et al., 2024; Yang et al., 2024b; Miao et al., 2024; Xiao et al., 2024a).

However, these methods rely on a rigid verification process akin to speculative sampling, leading to an excessive rejection of draft tokens and fre-

*Corresponding authors

quent target model calls. These challenges raise a crucial question: *Can we devise a better verification approach that boosts the acceptance rate and lessens reliance on target models, while maintaining a high level of performance?*

In this paper, we examine existing speculative decoding verification approaches (Leviathan et al., 2023; Tran-Thien, 2023) and observe that they predominantly evaluate the draft tokens by likelihood ratio or its variants between the target and draft models, as illustrated in Figure 1(a). This metric is derived from SD’s importance sampling to ensure that the final sampling distribution is well aligned with that of the LLM. However, the likelihood ratio does not reflect well the quality of the generated token. For example, as shown in Figure 1(a), when the target model’s prediction probability marginally exceeds that of the draft model – even if both probabilities are substantially low – the candidate tokens will still be accepted, potentially compromising output quality. In addition, the target model’s best token may still be rejected, particularly when the target language model has a relatively flat probability distribution. The enhanced SD approaches, such as Mentored Decoding (MD) (Tran-Thien, 2023), which continue to exhibit suboptimal problems, as it relies on likelihood ratio. Furthermore, it is unclear on how to choose the parameters, such as threshold and tolerance.

To address these problems, we introduce a novel approach that directly evaluates the correctness of candidate tokens using likelihood of the target model. Essentially, we leverage the negative log-likelihood (NLL) of candidate tokens predicted by target models to directly evaluate their correctness, a method we refer to as *Consultant Decoding* (CD). As illustrated in Figure 1(b), a draft token is deemed correct if its NLL approaches or falls below the convergence loss observed during the training of target models, signaling that the distribution of draft tokens closely mirrors the training data distribution of target models. Consultant Decoding is named for its approach of consulting the target model’s judgment to validate draft tokens, similar to seeking expert advice while retaining the ability to make independent decisions. This approach strikes a balance between independence (the draft model’s autonomy) and collaboration (the target model’s validation), leading to faster inference without sacrificing accuracy much even exceeding target models.

We conduct extensive empirical experiments us-

ing both publicly available datasets (covering areas such as math, coding, conversation, and instruction following) and real-world industrial scenarios, and observe significant benefits. Additionally, we compare the consistency of CD and Top-P sampling and found theoretical evidence that CD can maintain the performance of target models. Our analysis of speedup ratios and LLM-calls ratios also revealed that CD shows better scalability.

2 Related Work

Speculative decoding (Xia et al., 2023; Leviathan et al., 2023; Chen et al., 2023) has been widely adopted and extended across various works, yielding notable speedups in numerous model families and datasets. Several strategies for optimizing the draft model to accelerate the generation of draft tokens have also been explored in the literature. These approaches include training a smaller draft model specifically optimized for faster token generation (Cai et al.; Li et al., 2024b,a; Zafrir et al., 2024; Xiao et al., 2024b,a) and utilizing non-causal algorithms to generate draft tokens (Fu et al., 2024; Zhao et al., 2024; Liu and Zhu, 2024), which enable parallelization of the token generation process, reasonably boosting throughput.

Additionally, several research efforts have focused on improving the acceptance rate of draft tokens by relaxing the verification criteria (Hu and Huang; Narasimhan et al., 2024; Kim et al., 2024). Instead of strictly requiring that the distribution from speculative sampling aligns perfectly with the distribution of LLMs, these approaches allow for a certain degree of deviation within an acceptable range. These methods achieve a speedup by moderately sacrificing predictive accuracy.

Research efforts have also been undertaken to examine the structure of the drafts with the aim of boosting the acceptance rate of draft tokens: 1) Multi-candidate draft: These methods (Yang et al., 2024b; Miao et al., 2024; Wang et al., 2024a; Brown et al., 2024; Cai et al.; Li et al., 2024b) involve using tree-based attention mechanisms to generate multiple candidate draft token sequences during each iteration for chances of accepting more tokens. 2) Block-Wise Verify: Researchers have also explored moving beyond the token-level verification of SD, opting instead for block-wise conditional probabilities (Hu and Huang; Sun et al., 2024). 3) Dynamic Draft Length: In an effort to minimize the rejection of draft tokens for im-

proving the acceptance rate (Mamou et al., 2024; Huang et al., 2024). Existing works, such as BiLD (Kim et al., 2024), have employed similar verification methods to ours by evaluating token accuracy through the distribution distance between the hard label of the draft model and the soft label of the target model.

Our work focuses on the novel non-distribution verification mechanism, which differs remarkably from the above diverse research threads.

3 Method

3.1 Preliminaries

Speculative Sampling. Speculative sampling typically requires two models: a large target model P and a small draft model Q . The process involves iterative drafting and verification.

During the drafting phase, in each iteration, given the input token sequence $x_{\leq t}$, the draft model generates γ candidate tokens in an autoregressive manner, denoted as $\{x_i, q_i(\cdot) \mid 1 \leq i \leq \gamma\}$, here x_i represents the token sampled by the draft model at position $t + i$, and $q_i(\cdot)$ indicates the corresponding probability distribution predicted by the draft model. The sequence of input tokens $x_{\leq t}$ is concatenated with the draft tokens $\{x_i \mid 1 \leq i \leq \gamma\}$ to form $\{x_{\leq t}, x_1, \dots, x_\gamma\}$, which is then passed to the target model for the parallel verification.

During the verification phase, the target model generates the probability distributions $\{p_i(\cdot) \mid 1 \leq i \leq \gamma + 1\}$ for each draft token. Then a verification method $V_{SD}(x_i)$ is invoked to decide whether to accept the draft token or resample a new token

$$V_{SD}(x_i) = r(x_i) - \text{Rand}(0, 1) \quad (1)$$

$$r(x_i) = \min(1, p_i(x_i)/q_i(x_i)) \quad (2)$$

where $r(x_i)$ is acceptance probability of token x_i , $\text{Random}(0, 1)$ is a random number between 0 and 1. If $V_{SD}(x_i) > 0$, we accept the draft token and verify the next one, otherwise we reject it and sample a new token from the distribution:

$$\pi(x_i) = \frac{\max(0, p_i(x_i) - q_i(x_i))}{\sum_{y \in \mathcal{V}} \max(0, p_i(y) - q_i(y))} \quad (3)$$

and discard the draft tokens $\{x_{i+1}, \dots, x_\gamma\}$. The summation is over the whole vocabulary \mathcal{V} .

Speculative decoding has been proven to generate the same distribution as the target model theoretically (Chen et al., 2023; Leviathan et al., 2023).

Mentored Decoding. Alternative approaches (Narasimhan et al., 2024; Tran-Thien, 2023) trade off the consistency of the distribution for a higher acceptance rate by adjusting the distance between the sampling and the target distribution flexibly. In Mentored Decoding (Tran-Thien, 2023), the draft token x_i is accepted if $V_{MD}(x_i) > 0$:

$$V_{MD}(x_i) = \min(1, \frac{p_i(x_i)}{\alpha q_i(x_i)}) - \text{Rand}(0, 1) \quad (4)$$

otherwise, we reject x_i and sample a new token from distribution $\pi(x_i)$

$$\pi(x_i) = \frac{\max(0, p_i(x_i)/\beta - q_i(x_i))}{1 - \sum_{y \in \mathcal{V}} p_i(y) \min(1, \frac{p_i(y)}{\alpha q_i(y)})} \quad (5)$$

where the coefficients of α and β are calculated by iteratively optimizing to make the KL divergence between $q_i(x)$ and $p_i(x)$ fall within the interval $[D - \gamma, D + \gamma]$ (Tran-Thien, 2023), and D and γ are predefined thresholds of the distribution distance and tolerance.

3.2 Consultant Decoding

The perspective of CD is to validate the accuracy of the draft tokens using NLL of the target model, rather than to assess them based on the likelihood ratio of the target and the draft modls. We can formalize the verification method as a token-wise prediction NLL, to align with training loss as illustrated in the Figure 1(a). As shown in Equation 6, we treat the draft tokens as pseudo ground truth and compute the negative log-likelihood with the prediction probability of the target model, where ε is the threshold of the accuracy of the draft tokens. The underlying rationale of this method is that LLMs (target models) learn the vocabulary distribution for a given context from the training data, thereby utilizing their outstanding performance to assess the probability of predicted tokens in drafts.

$$V_{CD}(x_i) = \varepsilon - (-\log(p_i(x_i))) \quad (6)$$

If $V_{CD}(x_i) > 0$, the draft token x_i is accepted, otherwise we sample a new token from the distribution given by the target model. This approach eliminates the need for additional adjustments or modifications. By relying on the LLMs distribution, we preserve the integrity of its predictions and maintain coherence in the generated output.

Furthermore, Perplexity (PPL) is a common measure used to estimate the likelihood of a given

Algorithm 1 Consultant Decoding

Require: Small draft model Q , large target model P , threshold ε , EMA decay weight β , prefix tokens $x_{\leq t}$.

- 1: **for** $i = 1$ **to** γ **do**
- 2: $q_i(\cdot) \leftarrow Q(x_{\leq t}, x_1, \dots, x_{i-1})$
- 3: $x_i, q_i(x_i) \sim q_i(\cdot)$
- 4: **end for**
- 5: $p_1(\cdot), \dots, p_{\gamma+1}(\cdot) \leftarrow P(x_{\leq t}, x_1, \dots, x_\gamma)$
- 6: Set $n = 0$.
- 7: **for** $i = 1$ **to** γ **do**
- 8: $p_i(x_i) \leftarrow p_i(\cdot), x_i$
- 9: $r_i = \text{EMA}_\beta(r_{i-1}, -\log(p_i(x_i)))$
- 10: **if** $r_i \leq \varepsilon$ **then**
- 11: $n = i$
- 12: **else**
- 13: **break**
- 14: **end if**
- 15: **end for**
- 16: $x'_{n+1} \leftarrow p_{n+1}(\cdot)$
- 17: $r_{n+1} = \text{EMA}_\beta(r_n, -\log(p_{n+1}(x'_{n+1})))$
- 18: **Return** $[x_{\leq t}, x_1, \dots, x_n, x'_{n+1}], r_{n+1}$

sentence based on the predicted tokens as studied in (Lu et al., 2023), which can be formalized as $PPL = e^{\frac{1}{N} \sum (-\log(p_i))}$, where N is sentence length. It is obvious that the acceptance criteria in Consultant Decoding can be understood as a specific case where the sentence length $N = 1$.

Given this interpretation, it is natural to introduce the smooth version of the Consultant Decoding. As shown in Equation 7, we utilize the exponential moving average (EMA) to smooth the token’s NLL and take the context NLL into account. The parameter β allows for a flexible trade-off between the ratios of context and the current NLL.

$$V(x_i) = \varepsilon - \text{EMA}_\beta(r_i = -\log(p_i(x_i))) \quad (7)$$

$$\text{EMA}_\beta(r_i) = \beta \text{EMA}_\beta(r_{i-1}) + (1 - \beta)r_i \quad (8)$$

Algorithm 1 provides a sketch of the Consultant Decoding implementation.

3.3 How to Determine Generic Threshold ε

Except for SD, other verification methods have not been widely used, as it is challenging to determine thresholds that preserve inference accuracy across various LLMs and tasks. As discussed in Section 1, our CD provides a new perspective to address this issue. Specifically, the parameter ε in CD is related

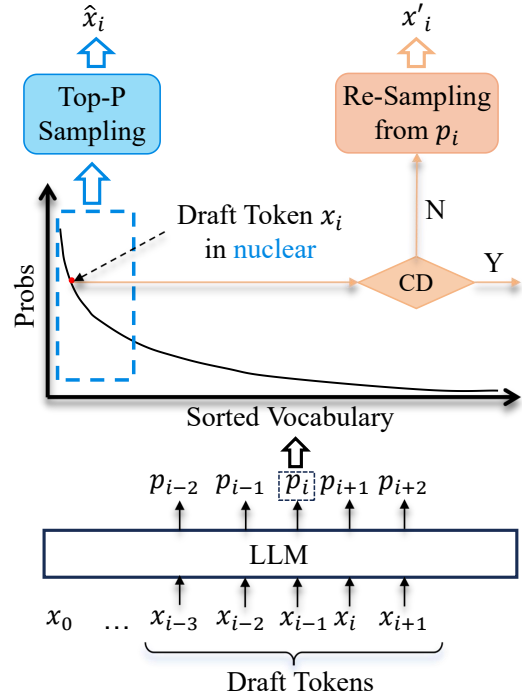


Figure 2: The comparison between Consultant Decoding and Top-P sampling. x_i , x'_i , and \hat{x}_i denote draft token, resample token by CD, and sample token by Top-P, respectively.

to the convergence loss of target model during training. Therefore, we can set ε to the convergence loss. For open-source models where convergence loss is not readily available, we estimate it using the Chinchilla scaling law proposed by Hoffmann et al. (2022), as depicted in Equation 9,

$$L(N, D) = E + \frac{A}{N^\alpha} + \frac{B}{D^\beta} \quad (9)$$

where $E = 1.69$, $A = 406.4$, $B = 410.7$, $\alpha = 0.34$, and $\beta = 0.28$. N and D are the model scale and the training data size of LLMs, respectively. Based on these calculations, we approximate the convergence loss as 2.0 for several models, and set $\varepsilon = 2.0$ without additional parameter tuning. Although Chinchilla scaling law was originally fit on different LLMs, and thus $\varepsilon = 2.0$ is a rough estimate, we find it performs reasonably well in most scenarios.

3.4 Why Generic Threshold ε can Preserve Accuracy of LLMs?

We can explain why the generic threshold of CD preserves target model accuracy through two key aspects: **1) Accuracy:** The CD threshold is aligned with the convergence loss of target model training,

ensuring that predicted tokens reflect the distribution of the training dataset. This alignment allows CD to filter the correct tokens with high precision, thereby maintaining the overall performance of the target model. **2) Consistency:** The tokens accepted by CD are drawn from the set of results produced by top-p sampling of the target model. This indicates that when CD accepts a token, it corresponds to a likely output the target model would generate in the same context. Figure 2 visualizes the relationship between Top-P sampling and CD, showing how Top-P sampling achieves high-precision text prediction by sampling from a high-quality dynamic nucleus (Holtzman et al., 2019). As β in Equation 7 approaches 0, the verification formula for CD can be reformulated as $p(x_i) - e^{-\epsilon} > 0$. So, when high-quality dynamic nucleus is set by $1 - e^{-\epsilon}$, CD shares similar property of Top-P sampling, and its verification method is equivalent to determining whether a draft token lies within the high-quality dynamic nucleus of the target model, thus also enabling high-precision text prediction. Intuitively, the small draft model can be viewed as a sampler that sample from the high-quality dynamic nucleus of the large target model, thus combining the strengths of both target model and draft model. As shown in Figure 4, appropriately expanding the size of the dynamic nucleus does not significantly degrade model performance on GSM8K.

4 Experiments

Models and Architectures. The architectures of speculative decoding can be broadly classified into two types: independent drafting and self-drafting (Xia et al., 2024). In independent drafting, a smaller, separate model is used as the draft model. In contrast, self-drafting utilizes the LLMs itself to efficiently generate draft tokens, making it a practical alternative when a dedicated draft model is not readily available. In the independent drafting setting, we use Qwen2.5 (Yang et al., 2024a) (a 72B target model, with 0.5B, 1.5B, 3B, and 7B draft models), and Llama3.1 (Dubey et al., 2024) (a 70B target model, with an 8B draft model, shown in Appendix A.4) models. In the self-drafting setting, experiments are conducted on EAGLE-2, which is one of the most effective methods for inference speedup currently available.

Tasks and Metrics. We evaluate the performance of verification methods across several tasks: mathematical reasoning GSM8K (Cobbe et al., 2021),

code generation HumanEval (Chen et al., 2021), multi-turn dialogue MT-Bench (Zheng et al., 2023), and instruction following AlpacaEval (Li et al., 2023b). For each task, we adopt the corresponding evaluation metric used in the literature (Yang et al., 2024a) for performance evaluation. The details of tasks and metrics are described in the following:

- **GSM8K.** GSM8K is designed to evaluate the mathematical reasoning abilities of LLMs, consisting of 1319 questions. We following (Wei et al., 2022) using 8-shot, as shown in Table 7, and step-by-step solution leading to the correct final answer. We use accuracy to evaluate model performance, which measures the correct rate of mathematical problem-solving.
- **HumanEval.** HumanEval is used to evaluate the ability of models to generate python code based on natural language. Each sample includes a function signature (with parameter names and type hints) and a docstring (natural language specification of the function’s intended behavior). Then we adopt pass@1 as the metric to assess the correctness of the model’s single-attempt code generation.
- **MT-Bench.** MT-Bench is a multi-turn dialogue benchmark, containing 80 samples. Responses are scored (1-10) by GPT-4o for helpfulness, relevance, accuracy, depth, creativity, and level of detail of the response, which is used by Medusa (Cai et al.).
- **AlpacaEval.** AlpacaEval is a benchmark for evaluating the instruction-following capabilities of language models. It focuses on assessing how well models can generate helpful, high-quality responses based on user instructions. We follow (Li et al., 2023b) and use the Win Rate as the evaluation metric, comparing model outputs against a baseline (i.e., text-davinci-003) via GPT-4o judgments.

In our experiments, unless otherwise specified, we normalize the accuracy, pass@1, score, and win-rate metrics across all tasks relative to the target model’s greedy decoding performance. This normalization facilitates a comparative analysis of

Model	Verify	GSM8K		HumanEval		MT-Bench		AlpacaEval	
		Accuracy	Speedup	Pass@1	Speedup	Score	Speedup	Win-Rate	Speedup
Qwen2.5-0.5B		47.7%	4.83×	39.4%	4.56×	43.6%	4.52×	29.7%	4.6×
Qwen2.5-72B		100.0%	1.0×	100.0%	1.0×	100.0%	1.0×	100.0%	1.0×
Qwen2.5 0.5B/72B	SD	99.9%	2.22×	100.7%	2.18×	100.0%	1.57×	99.8%	1.40×
	MD	96.8%	2.54×	97.9%	2.41×	95.2%	1.77×	97.5%	1.68×
	CD	96.1%	3.09 ×	97.2%	3.04 ×	95.43%	2.06 ×	97.2%	2.02 ×
Qwen2.5-7B		95.6%	3.75×	96.5%	3.59×	90.0%	4.44×	95.9%	3.48×
Qwen2.5 7B/72B	SD	99.9%	2.09×	100.0%	2.12×	100.0%	1.68×	99.4%	1.57×
	MD	98.7%	2.82×	97.2%	2.91×	95.8%	2.63×	97.8%	2.39×
	CD	99.1%	3.00 ×	100.7%	3.13 ×	96.3%	2.83 ×	98.1%	2.60 ×

Table 1: Comparing verification methods for speedup ratio, where 0.5B, 7B, and 72B all represent model sizes (the smaller is the draft model and the larger is the target model). The Qwen2.5-72B achieves accuracy of 95.53 on GSM8K dataset, pass@1 metric of 86.59 on HumanEval dataset, and scores of 8.65 on MT-Bench datasets, and win-rate of 96.71 on AlpacaEval datasets, respectively. The absolute values of other performance are shown in Table 12.

metric variations. To assess the inference speed, we employ the following metrics:

- **Walltime speedup ratio:** The speedup relative to standard autoregressive decoding of the target model.
- **LLM-calls ratio:** The ratio of calls to the LLMs compared to autoregressive decoding. Reducing target model calls enables greater parallel tasks throughput.
- **Average acceptance length:** The average ratio of tokens accepted per draft.

Experiment Setting. We present two parameter configurations for verification methods:

- **Speed-Optimal.** An extensive search through numerous experiments is conducted to find the optimal speedup for each algorithm, with parameter settings detail in Appendix C.
- **Generic.** The Generic setting is used for a fair comparison with SD, where no parameter adjustments are required. From our novel perspective, we set threshold ε to 2.0 and the EMA decay β to 0.2 for CD, and threshold D to 0.3 with a temperature of 0.3 for MD¹.

4.1 Main Results

4.1.1 Walltime Speedup Ratio of CD

To fully demonstrate the maximum acceleration ratio of different verification methods, we adjust parameters and draft lengths (see Appendix C for details), allowing a moderate performance decrease,

¹These are the default parameters provided by MD.

within 5% of the large target model, in prediction accuracy. As indicated in Table 1, the combination of Qwen2.5-0.5B/72B models achieves at least 2.02× speedup ratio in all datasets. Thanks to the fixed inference paths and code templates on mathematical reasoning and code generation datasets, CD achieves the highest inference acceleration of over 3×, which is average faster by 0.8× and 0.59× compared to SD and MD, respectively. On the more challenging MT-Bench and AlpacaEval datasets, CD achieves speedup ratios of 2.06× and 2.02× for the Qwen2.5-0.5B/72B model combination, and 2.83× and 2.60× for the Qwen2.5-7B/72B model combination, respectively.

Note that due to the minimal performance difference between the Qwen2.5-7B and Qwen2.5-72B models on GSM8K and HumanEval datasets (<5% of Qwen2.5-72B), the speedup ratios presented in Table 1 approach the limit of these model combinations, i.e., accepting all draft tokens. Although the experiment does not reveal the upper limit of CD, it suggests that with the advancement of LLMs and the enhancement of draft model prediction accuracy and speed, verification methods that accept more drafts are increasingly practical.

Furthermore, we provide the results of speedup ratio with Generic setting, in Appendix A.2, to fairly compare with SD and support practical use.

4.1.2 LLM-calls

Based on the Generic setting, we set the draft length to 20 to reduce the LLM-calls ratio (i.e., target models). Table 2 illustrates that, across most tasks, the CD method achieves a lower LLM-calls ratio compared to SD and MD, while preserving prediction accuracy of the target model. For example,

Model	Verify	GSM8K			HumanEval			MT-Bench		
		Accuracy	Speedup	LLM-calls	Pass@1	Speedup	LLM-calls	Score	Speedup	LLM-calls
Qwen2.5-0.5B		47.7%	4.83×	0.0%	39.4%	4.56×	0.0%	43.6%	4.52×	0.0%
Qwen2.5-72B		100.0%	1.0×	100%	100.0%	1.0×	100%	100.0%	1.0×	100%
Qwen2.5 0.5B/72B	SD	100.0%	1.69×	11.7%	100.0%	1.59×	11.8%	100.0%	0.80×	22.0%
	MD	99.1%	2.03×	10.2%	99.3%	1.91×	10.2%	97.9%	0.99×	19.0%
	CD	100.3%	2.39×	9.1%	99.3%	2.23×	9.1%	96.5%	1.25×	15.8%
Qwen2.5-7B		95.6%	3.75×	0.0%	96.5%	3.59×	0.0%	90.0%	4.44×	0.0%
Qwen2.5 7B/72B	SD	100.0%	1.78×	9.0%	100.0%	1.74×	8.5%	100.0%	1.01×	14.1%
	MD	100.5%	2.24×	7.5%	99.3%	2.20×	7.1%	98.6%	1.40×	9.2%
	CD	100.2%	2.53×	6.5%	103.5%	2.49×	6.2%	99.8%	1.75×	8.6%

Table 2: Comparing verification methods with draft length setting to 20, 0.5B, 7B, and 72B all represent model sizes, where the smaller is the draft model and the larger is the target model. The absolute values of performance are shown in Table 13.

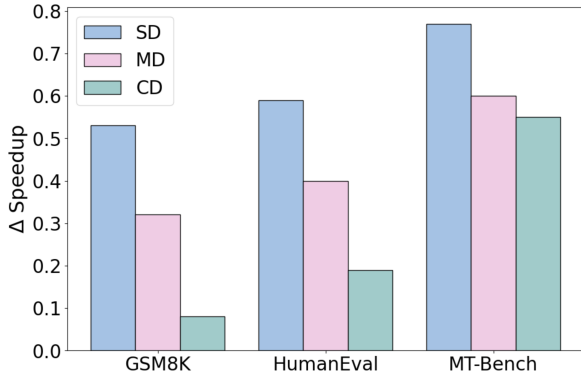


Figure 3: A comparison of the decrease in speedup ratios among SD, MD, and CD verification methods as the draft length increases from 6 to 20, using greedy decoding. Here target and draft models are 72B and 0.5B models, respectively.

on GSM8K and HumanEval datasets, using the Qwen2.5-0.5B as the draft model and the Qwen2.5-72B as the target model, CD achieves the lowest LLM-calls ratio of 9.1%. On HumanEval dataset, with the combination of Qwen2.5-7B/72B model, using only LLM-calls ratio of 6.2%, CD surpasses the prediction accuracy of target model by an additional 3.5%. This means that compared to autoregressive decoding, the computation time of the LLMs is reduced to 6%. On MT-Bench dataset, which is a more challenging dataset, CD significantly outperforms SD and MD in the LLM-calls ratio, showing a reduction of 6.2% compared to SD and 3.2% compared to MD. This indicates that in real dialogue scenarios, CD is better at identifying the correctness of draft tokens. We further validate this advantage of CD in a real-world business benchmark designed for practical industry applications. It can be regarded as a combination of multiple aspects not limited to reasoning, coding, multi-turn conversion and instruction following, as detailed in Appendix A.1.

Method	HumanEval		MT-Bench	
	Pass@1	Speedup	Score	Speedup
Llama-3-70B	100.0%	1.0×	100.0%	1.0×
Eagle-2-70B	100.0%	3.43×	100.0%	3.18×
Eagle-2-70B-CD	100.8%	3.66×	98.2%	3.51×

Table 3: Generalization of CD on EAGLE-2 structure. The Llama-3-70B achieve pass@1 accuracy of 76.83 on HumanEval, a score of 7.4 on MT-Bench.

Method	HumanEval		MT-Bench	
	Pass@1	Speedup	Score	Speedup
Baseline	100.0%	1.0×	100.0%	1.0×
+ CD	103.52%	2.51×	99.8%	1.75×
+ CD (w/o Smooth)	103.52%	2.40×	99.3%	1.70×

Table 4: Ablation study of Consultant Decoding with Qwen2.5-7B/72B.

4.1.3 CD’s Resilience to Draft Length

Essentially, the draft length impacts the overall speedup of synergistic decoding in a complex way, thus tweaking draft length to achieve reasonable balance of performance and inference speedup is notoriously challenging. We study how CD method is influenced by draft length via fixing the collaboration performance to around 100%. As shown in Figure 3, when the draft length increases from 6 to 20, the speedup ratios of SD and MD on GSM8K and HumanEval decrease by 0.53×, 0.59×, and 0.32×, 0.4×, respectively. This is mainly because a larger proportion of draft tokens may be inappropriately rejected, which consumes a significant amount of time. In contrast, our CD, which benefits from a more accurate assessment of draft token correctness, experienced only a slight reduction in the speedup ratio by 0.08× and 0.19×, respectively.

For the more challenging MT-Bench dataset, as illustrated in Figure 3 and Table 2, CD shows the smallest decrease in speedup ratio while still achieving the highest speedup of 1.25×. In contrast, SD performs worse than the standard autoregressive decoding used in LLMs. Thus, our CD approach is more resilient to draft length variations

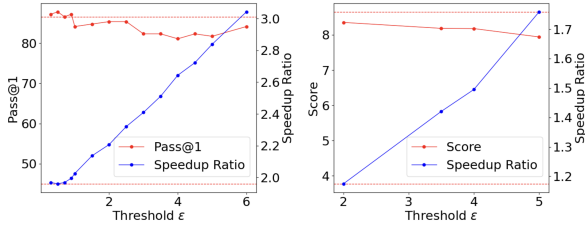


Figure 4: Ablation study of threshold ε of CD with Qwen2.5-0.5B/72B, on HumanEval and MT-Bench. The red dashed lines are the inference accuracy of the target and draft models.

in terms of maintaining the speedup ratio.

4.1.4 Results on Self-drafting Decoding

To demonstrate the generalization of CD, we replace the verification method in EAGLE-2 and compare the speedup ratios and accuracy. As presented in Table 3, CD shows strong generalization across benchmarks. Specifically, when applied to HumanEval and MT-Bench, EAGLE-2 with CD verification, the speed is improved by $0.23\times$ and $0.33\times$, respectively, with nearly no performance degradation. Additional analysis and discussion are provided in Appendix B.3.

4.2 Ablation Study

4.2.1 Threshold ε

We evaluate the performance of CD by varying the ε parameter. As depicted in Figure 4, the performance of CD does not significantly deteriorate with a more relaxed threshold. Specifically, on GSM8K dataset, performance remains largely stable as ε increases from 2 to 6. On MT-Bench, an 8% performance decrease is observed, yet it still outperforms smaller models. In practice, the optimal ε can be selected based on the trade-off between performance and efficiency required.

4.2.2 Smooth Verification

Smooth verification is used to balance the contribution of the current prediction and context tokens. If the tokens in the context have a low NLL, as calculated by Equation 6, we can somewhat relax the constraints on the current token. This helps reduce accidental errors in the model. Table 4 presents comparative results that indicate the impact of using smooth verification. Smooth verification has been shown to increase the acceleration ratio by $0.11\times$ on HumanEval and by $0.05\times$ on MT-Bench, without compromising the model performance.

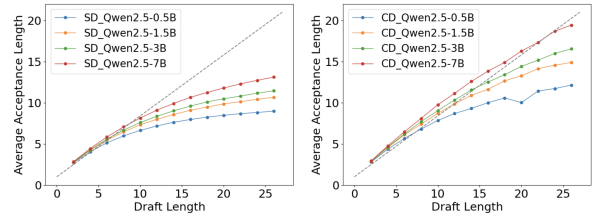


Figure 5: The figure shows the scalability of average acceptance length on HumanEval. The target model is Qwen2.5-72B, and draft models are Qwen2.5-0.5B, 1.5B, 3B, 7B. The gray dashed lines extend from the coordinates (0,1) to (27,24).

4.3 Case Study

Consultant Decoding evaluates the correctness of draft tokens using the target model, rather than strictly aligning the LLMs distribution. This approach combines the strengths of both the target and draft models for more accurate predictions. As shown in Table 5, we compare SD and CD on GSM8K dataset. In SD, the target model incorrectly connects "the number of hotdogs short" and "each guest needs 2 hotdogs", leading to an erroneous prediction. In contrast, CD allows the target model to follow the reasoning path of the draft model, resulting in a correct prediction. This illustrates why CD can surpass the performance upper bound of speculative decoding, thereby improving the overall performance of the target model.

4.4 Scalability

To investigate why CD can reduce the LLM-calls ratio without significantly compromising the speedup ratio, we fix the parameters of CD as in Generic setting to ensure high-quality predictions. We then examine the Average Acceptance Length (AAL) of draft tokens with various draft lengths.

The results, shown in Figure 5, reveal that as the draft length increases, the growth rates of AAL for SD and CD slow down. Notably, the AAL of SD saturates quickly, whereas the AAL of CD shows a sharper increase. Furthermore, as the draft model size increases, the AAL of CD grows more significantly than that of SD, indicating a more pronounced scaling behavior. For instance, with a 7B draft model, the growth of AAL for SD from draft length 2 to 26 is 10.28, while for CD it is 16.51. In other words, compared to SD, CD achieves higher AAL and a lower LLM-calls ratio more easily by increasing the draft model size.

Question:	A food caterer was told to prepare gourmet hot dogs for 36 guests. While most people would only eat one hotdog, he prepared enough for half of the guests to be able to have two hotdogs. However, 40 guests showed up, and everyone wanted a second hotdog. How many guests did not get a second hotdog?
SD	<p>First, (we) let's determine how many hotdogs (were) the caterer prepared (for) . (Since) The caterer (prepared) was told to prepare (gourmet) enough for (half) 36 guests, (and) with half of them (being) able to have two hotdogs. (So) This means () : (1) - Half of () the guests (=) (18 guests) would have (two) 2 hotdogs each (.) : () (1) $18 \times 2 = 36$) hotdogs. - The other half (of) (18 guests) would have 1 hotdog each: $18 \times 1 = 18$ hotdogs.</p> <p>So, the total number of hotdogs prepared is (()) : $36 (\wedge) + 18 = 54$</p> <p>(Next) Now, 40 guests showed up, and (everyone) each (of) wanted a second hotdog. This means () each (of) guest (who) wanted 2 hotdogs (.) . (So) The total number of hotdogs (served) needed for 40 guests is: $40 \times 2 = 80$</p> <p>(Since) The (number) caterer (prepared) only (prepared) had 54 hotdogs, so the number of hotdogs (not) short is: $80 - 54 = 26$</p> <p>(Therefore) Since () each guest (who) needs 2 hotdogs (.) to (have) be (able) satisfied, the number of guests who did not get a second hotdog is: $\frac{26}{2} = 13$</p> <p>(So) The answer is 13 (guests) . ##### 13</p>
CD	<p>First, we need to determine how many hotdogs (were) the caterer prepared. (Since) The caterer prepared enough for half of the (guests) 36 guests (.) to have two hotdogs each. So, half of 36 guests is $36 / 2 = 18$ guests. (This) These 18 guests (each) would have 2 hotdogs each, so (they) the (total) caterer prepared $18 * 2 = 36$ hotdogs (.) for (them) these guests.</p> <p>(Next) The (cater) remaining (guests) 18 guests (.) would have (2) 1 hotdog each, so the caterer prepared $18 * 1 = 18$ hotdogs for these guests.</p> <p>(Now) In total, the caterer prepared $36 + 18 = 54$ hotdogs.</p> <p>Now, 40 guests showed up, and (everyone) each (of) wanted a second hotdog. (So) This means $40 * 2 = 80$ hotdogs were (consumed) needed.</p> <p>The number of hotdogs that were not (prepared) available is (the) (5) $80 - 54 = 26$ hotdogs.</p> <p>(Therefore) Since each guest who did not get a second hotdog (would) is (counted) missing one hotdog, the number of guests who did not get a second hotdog is 26.</p> <p>The answer is 26. ##### 26</p>

Table 5: The example of comparing SD and CD with Qwen2.5-0.5B/7B, on GSM8K. The blue text represents the text modified by target model, gray text denotes candidate text generate by draft model, red text and green text are critical modifications and incorrect modifications, respectively. The underline text is different thought chain between SD and CD.

5 Conclusion

This paper develops Consultant Decoding (CD), a novel decoding method based on NLL, that improves speculative decoding for model inference speedup. Unlike traditional methods that trade efficiency for performance, CD aligns with the convergence loss of target model training, achieving higher speedup without compromising or even enhancing the target model performance by theoretical analysis and case study. Comprehensive experiments show the acceleration advantage, generalization, resilience and scalability of CD. Furthermore, we provide a comparison of CD with Top-P sampling, providing a theoretical basis for how CD maintains the inference accuracy of the target model with a threshold ϵ of 2.0. This advantage provides convenience for the use of CD in practical scenarios. The generalization of CD to other architectures (e.g., EAGLE-2) further highlights its robustness and scalability.

Limitations

Our propose perspective aligns the prediction NLL of the draft token with the convergence loss of target model training, facilitating the replacement of the strict SD verification methd with a fixed-threshold CD ($\epsilon = 2.0$), which achieves significant inference acceleration. However, this fixed threshold is not optimal. Figure 4 illustrates that increasing the threshold can further enhance the efficiency of CD without significantly compromising the performance of LLMs. The optimal threshold is highly dependent on both the questions and the combination of models, and determining this threshold during inference to fully leverage CD remains challenging. Moreover, another limitation is the occurrence of unnecessary modifications. While CD has reduced the cost of unnecessary modifications, as seen in Table 5, “Therefore” is changed to “Since”. Hence, improving the capability to distinguish the correct tokens is another direction for future study.

References

- Oscar Brown, Zhengjie Wang, Andrea Do, Nikhil Mathew, and Cheng Yu. 2024. Dynamic depth decoding: Faster speculative decoding for llms. *arXiv preprint arXiv:2409.00142*.
- Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D Lee, Deming Chen, and Tri Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads, 2024. URL <https://arxiv.org/abs/2401.10774>.
- Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. 2024. Break the sequential dependency of llm inference using lookahead decoding. *arXiv preprint arXiv:2402.02057*.
- Jiawei Gu, Zacc Yang, Chuanghao Ding, Rui Zhao, and Fei Tan. 2024. [CMR scaling law: Predicting critical mixture ratios for continual pre-training of language models](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16143–16162.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.
- Zhengmian Hu and Heng Huang. Accelerated speculative sampling based on tree monte carlo. In *Forty-first International Conference on Machine Learning*.
- Kaixuan Huang, Xudong Guo, and Mengdi Wang. 2024. Specdec++: Boosting speculative decoding via adaptive candidate lengths. *arXiv preprint arXiv:2405.19715*.
- Sehoon Kim, Karttikeya Mangalam, Suhong Moon, Jitendra Malik, Michael W Mahoney, Amir Gholami, and Kurt Keutzer. 2024. Speculative decoding with big little decoder. *Advances in Neural Information Processing Systems*, 36.
- Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.
- Dedong Li, Rui Zhao, and Fei Tan. 2023a. Cwseg: An efficient and general approach to chinese word segmentation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 1–10.
- Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023b. AlpacaEval: An automatic evaluator of instruction-following models.
- Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024a. Eagle-2: Faster inference of language models with dynamic draft trees. *arXiv preprint arXiv:2406.16858*.
- Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024b. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*.
- Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.
- Chengbo Liu and Yong Zhu. 2024. Sdsat: Accelerating llm inference through speculative decoding with semantic adaptive tokens. *arXiv preprint arXiv:2403.18647*.
- Jinghui Lu, Dongsheng Zhu, Weidong Han, Rui Zhao, Brian Mac Namee, and Fei Tan. 2023. What makes pre-trained language models better zero-shot learners? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2288–2303.
- Jonathan Mamou, Oren Pereg, Daniel Korat, Moshe Berchansky, Nadav Timor, Moshe Wasserblat, and Roy Schwartz. 2024. Dynamic speculation lookahead accelerates speculative decoding of large language models. *arXiv preprint arXiv:2405.04304*.
- Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, et al. 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*, pages 932–949.
- Harikrishna Narasimhan, Wittawat Jitkrittum, Ankit Singh Rawat, Seungyeon Kim, Neha Gupta, Aditya Krishna Menon, and Sanjiv Kumar. 2024. Faster cascades via speculative decoding. *arXiv preprint arXiv:2405.19261*.
- Ziteng Sun, Uri Mendlovic, Yaniv Leviathan, Asaf Aharoni, Ahmad Beirami, Jae Hun Ro, and Ananda Theertha Suresh. 2024. Block verification accelerates speculative decoding. In *Workshop on Efficient Systems for Foundation Models II@ ICML2024*.
- Vivien Tran-Thien. 2023. [An optimal lossy variant of speculative decoding](#).
- Jikai Wang, Yi Su, Juntao Li, Qingrong Xia, Zi Ye, Xinyu Duan, Zhefeng Wang, and Min Zhang. 2024a. Opt-tree: Speculative decoding with adaptive draft tree structure. *arXiv preprint arXiv:2406.17276*.
- Shiqi Wang, Zhengze Zhang, Rui Zhao, Fei Tan, and Nguyen Cam-Tu. 2024b. [Reward difference optimization for sample reweighting in offline RLHF](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2109–2123.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.
- Heming Xia, Tao Ge, Peiyi Wang, Si-Qing Chen, Furu Wei, and Zhifang Sui. 2023. Speculative decoding: Exploiting speculative execution for accelerating seq2seq generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3909–3925.
- Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. 2024. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*.
- Bin Xiao, Lujun Gui, Lei Su, and Weipeng Chen. 2024a. Clover-2: Accurate inference for regressive lightweight speculative decoding. *arXiv preprint arXiv:2408.00264*.
- Bin Xiao, Chunan Shi, Xiaonan Nie, Fan Yang, Xiangwei Deng, Lei Su, Weipeng Chen, and Bin Cui. 2024b. Clover: Regressive lightweight speculative decoding with sequential knowledge. *arXiv preprint arXiv:2405.00263*.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024a. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.
- Sen Yang, Shujian Huang, Xinyu Dai, and Jiajun Chen. 2024b. Multi-candidate speculative decoding. *arXiv preprint arXiv:2401.06706*.
- Ofir Zafrir, Igor Margulis, Dorin Shteyman, and Guy Boudoukh. 2024. Fastdraft: How to train your draft. *arXiv preprint arXiv:2411.11055*.
- Hengyuan Zhang, Yanru Wu, Dawei Li, Sak Yang, Rui Zhao, Yong Jiang, and Fei Tan. 2024. Balancing speciality and versatility: a coarse to fine framework for supervised fine-tuning large language model. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 7467–7509.
- Weilin Zhao, Yuxiang Huang, Xu Han, Wang Xu, Chaojun Xiao, Xinrong Zhang, Yewei Fang, Kaihuo Zhang, Zhiyuan Liu, and Maosong Sun. 2024. Ouroboros: Generating longer drafts phrase by phrase for faster speculative decoding. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13378–13393.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

A Other Results

A.1 Results on Business Dataset

As shown in Table 8, we present the performance of CD on the business dataset. To maintain high prediction accuracy, we employ the Generic setting with a draft length of 6. CD achieves the highest speedup ratio while preserving the prediction accuracy of the LLMs. For example, compared to SD, the speedup of CD is improved by $0.49\times$ with only a 1.7% decrease in score. Relative to MD, the speedup of CD is enhanced by $0.36\times$, with a score increase of 2.8%. In pursuit of a lower LLM-calls ratio, we utilize the Speed-Optimal setting with a draft length of 20. It is noteworthy that both MD and CD offer a trade-off between inference speed and prediction accuracy, however, at a similar speedup, CD exhibits a 16.1% higher inference accuracy than MD.

A.2 What is the maximum walltime speedup ratio of CD on Generic setting?

To ensure a fair comparison with SD, where no parameter adjustment is required, and to support the practical application of CD, we teste the optimal speedup ratio for various varification methods based on the Generic setting with draft length setting to 6. As shown in Table 14, CD achieves the

Dataset	Prompt
GSM8K	{Instruction}\n Let's think step by step. At the end, you MUST write the answer as an integer after ####.
HumanEval	You are an intelligent programming assistant to produce Python algorithmic solutions.\n Can you complete the following Python function? \n```python\n {Instruction}\n```
MT-Bench	You are a helpful, respectful and honest assistant. Always answer as helpfully as possible, while being safe. Your answers should not include any harmful, unethical, racist, sexist, toxic, dangerous, or illegal content. Please ensure that your responses are socially unbiased and positive in nature. \n\n If a question does not make any sense, or is not factually coherent, explain why instead of answering something not correct. If you don't know the answer to a question, please don't share false information.
AlpacaEval	{Instruction}

Table 6: The prompt corresponding to the dataset.

best speedup ratio in all tasks while maintaining the prediction accuracy of LLMs. For example, with the combination of Qwen2.5-7B/72B, the speedup of CD exceeds SD by 0.41, 0.39, and 0.25, respectively, and exceeded MD by 0.26, 0.29, and 0.25, respectively, on GSM8K, HumanEval, and MT-Bench.

A.3 Results with Sampling Decoding

Table 9 presents a comparison of verification methods using sampling decoding for two combination of the Qwen2.5 model, Qwen2.5-0.5B/72B and Qwen2.5-7B/72B, based on a Generic setting with a draft length of 6 and temperature of 0.7. For both model sizes, the CD method exhibits a speedup of over $2\times$ on GSM8K and HumanEval datasets, while maintaining accuracy above 99%, even more than 100%. This demonstrates that CD can reliably verify the accuracy of draft tokens, and as long as the draft model produces correct tokens, the decoding method may be selected based on specific application requirements.

A.4 Results On Llama-3.1

We provide the results of Consultant Decoding with model combination of Llama-3.1-8B/70B, based on the Generic setting with greedy decoding. As shown in Table 10, the CD demonstrates excellent generalization on Llama3.1. For example, CD achieves the speedup ratio by $2.36\times$, $2.38\times$, and $2.22\times$ on GSM8K, HumanEval and MT-Bench, respectively, while preserving the inference accuracy of target model.

B Performance Analysis

B.1 Outperformance Analysis

As shown in Table 2, the CD approach demonstrates better performance compared to the target model in three specific scenarios: (1) GSM8K with Qwen2.5-0.5B/72B, (2) GSM8K with Qwen2.5-7B/72B, and (3) HumanEval with Qwen2.5-7B/72B. Notably, the HumanEval case exhibits a notable improvement of 3.5%.

We propose that when the target model encounters uncertainty (i.e., when multiple solutions with comparable probabilities exist), the SLM can effectively guide the decoding process toward higher-probability solutions based on its own modeling. As CD only accepts the draft model solution if it is high enough according to the target model, CD acts as a voting mechanism, allowing CD to combine the strength of the draft and the target model. This intriguing phenomenon presents a promising direction for future research.

B.2 Error Analysis

We examine 22 error samples from HumanEval and 56 from GSM8k, categorizing them into three distinct types:

- **Target Model Errors:** Cases where the target model produce incorrect predictions.
- **Draft Model Errors:** Cases where the target model was correct but the draft model made mistakes.
- **CD Errors:** Cases where both models were correct individually, yet errors emerge from the CD process.

As shown in Table 11, our analysis reveals that:

- The predominant source of CD prediction errors originates from target model errors.
- A minor proportion stems from either draft model errors or CD-specific errors.

We draw two key observations from these findings:

- Draft model errors that lead to CD failures indicate the existence of error tokens that challenge the LLMs identification capability.
- CD errors likely arise from the language models' sensitivity to contextual variations generated through the draft-target model cooperation.

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today? Let's think step by step. At the end, you MUST write the answer as an integer after #####.

A: We start with 15 trees. Later we have 21 trees. The difference must be the number of trees they planted. So, they must have planted $21 - 15 = 6$ trees. The answer is 6.\n

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot? Let's think step by step. At the end, you MUST write the answer as an integer after #####.

A: There are 3 cars in the parking lot already. 2 more arrive. Now there are $3 + 2 = 5$ cars. The answer is 5.\n

Q: Leah had 32 chocolates and her sister had 42. If they ate 35, how many pieces do they have left in total? Let's think step by step. At the end, you MUST write the answer as an integer after #####.

A: Leah had 32 chocolates and Leah's sister had 42. That means there were originally $32 + 42 = 74$ chocolates. 35 have been eaten. So in total they still have $74 - 35 = 39$ chocolates. The answer is 39.\n

Q: Jason had 20 lollipops. He gave Denny some lollipops. Now Jason has 12 lollipops. How many lollipops did Jason give to Denny? Let's think step by step. At the end, you MUST write the answer as an integer after #####.

A: Jason had 20 lollipops. Since he only has 12 now, he must have given the rest to Denny. The number of lollipops he has given to Denny must have been $20 - 12 = 8$ lollipops. The answer is 8.\n

Q: Shawn has five toys. For Christmas, he got two toys each from his mom and dad. How many toys does he have now? Let's think step by step. At the end, you MUST write the answer as an integer after #####.

A: He has 5 toys. He got 2 from mom, so after that he has $5 + 2 = 7$ toys. Then he got 2 more from dad, so in total he has $7 + 2 = 9$ toys. The answer is 9.\n

Q: There were nine computers in the server room. Five more computers were installed each day, from monday to thursday. How many computers are now in the server room? Let's think step by step. At the end, you MUST write the answer as an integer after #####.

A: There are 4 days from monday to thursday. 5 computers were added each day. That means in total $4 * 5 = 20$ computers were added. There were 9 computers in the beginning, so now there are $9 + 20 = 29$ computers. The answer is 29.\n

Q: Michael had 58 golf balls. On tuesday, he lost 23 golf balls. On wednesday, he lost 2 more. How many golf balls did he have at the end of wednesday? Let's think step by step. At the end, you MUST write the answer as an integer after #####.

A: Michael initially had 58 balls. He lost 23 on Tuesday, so after that he has $58 - 23 = 35$ balls. On Wednesday he lost 2 more so now he has $35 - 2 = 33$ balls. The answer is 33.\n

Q: Olivia has \$23. She bought five bagels for \$3 each. How much money does she have left? Let's think step by step. At the end, you MUST write the answer as an integer after #####.

A: She bought 5 bagels for \$3 each. This means she spent $5 * 3 = \$15$ on the bagels. She had \$23 in beginning, so now she has $\$23 - \$15 = \$8$. The answer is 8.\n

Table 7: The 8-shot prompt of GSM8K.

Setting	Verify	Score	Speedup	LLM-calls
G-6	SD	100.0%	1.29×	36.53%
	MD	95.5%	1.42×	30.42%
	CD	98.3%	1.78 ×	26.89%
S-20	SD	98.9%	0.79×	28.45%
	MD	67.4%	2.14×	10.38%
	CD	83.5%	2.15 ×	10.57%

Table 8: Comparing verification methods on business dataset, where "G-6" and "S-20" denote the employment of the Generic setting with a draft length of 6 and Speed-Optimal setting with draft length of 20.

Datasets	Accuracy	Speedup	LLM-calls
<i>Qwen2.5-0.5B/72B</i>			
GSM8K	99.61%	2.25×	18.78%
HumanEval	101.41%	2.25×	18.93%
MT-Bench	97.54%	1.60×	25.78%
<i>Qwen2.5-7B/72B</i>			
GSM8K	100.16%	2.17×	16.06%
HumanEval	102.82%	2.16×	16.01%
MT-Bench	98.34%	1.82×	18.59%

Table 9: Results of Consultant Decoding with sampling decoding, based on Generic setting with draft length of 6 and temperature of 0.7.

Datasets	Accuracy	Speedup	LLM-calls
GSM8K	99.05%	2.36×	5.80%
HumanEval	100.79%	2.38×	5.45%
MT-Bench	100.0%	2.22×	7.48%

Table 10: Results of Consultant Decoding with combination of Llama-3.1-8B/70B, based on Generic setting, with draft length of 20 on GSM8K and HumanEval, draft length of 16 on MT-Bench.

Error Type	HumanEval	GSM8K
Target model errors	15	44
Draft model errors	4	10
CD errors	3	2
Total errors	22	56

Table 11: The number of error samples of each type on HumanEval and GSM8K datasets.

These insights suggest that future improvements in CD threshold optimization and LLMs robustness could effectively mitigate such error cases.

B.3 Discussion about EAGLE-Style model.

As shown in Table 3, CD demonstrates good generalization on the Eagle-style framework, indicating that CD can be well adapted to both self-drafting and dynamic tree methods. However, it should be noted that the acceleration capability of CD is related to the draft length. The Eagle-style method faces challenges in extending the draft length, which limits the potential of CD.

C Parameters for Best Speedup

As shown in Table 15, we present the parameters required for the experiments in Table 1: threshold and draft length. These parameters were obtained using grid search to demonstrate the optimal speedup ratio for each validation method. For the other parameters, we use the default settings provided in the Generic setting in Section 4.

Model	Verify	GSM8K		HumanEval		MT-Bench		AlpacaEval	
		Accuracy	Speedup	Pass@1	Speedup	Score	Speedup	Win-Rate	Speedup
Qwen2.5-0.5B		45.56	4.83×	34.15	4.56×	3.77	4.52×	28.70	4.6×
Qwen2.5-72B		95.53	1.0×	86.59	1.0×	8.65	1.0×	96.71	1.0×
Qwen2.5 0.5B/72B	SD	95.45	2.22×	87.20	2.18×	8.65	1.57×	96.51	1.40×
	MD	92.50	2.54×	84.76	2.41×	8.23	1.77×	94.28	1.68×
	CD	91.81	3.09 ×	84.15	3.04 ×	8.26	2.06 ×	93.98	2.02 ×
Qwen2.5-7B		91.36	3.75×	83.54	3.59×	7.79	4.44×	92.79	3.48×
Qwen2.5 7B/72B	SD	95.45	2.09×	86.59	2.12×	8.65	1.68×	96.13	1.57×
	MD	94.31	2.82×	84.15	2.91×	8.29	2.63×	94.58	2.39×
	CD	94.64	3.00 ×	87.20	3.13 ×	8.33	2.83 ×	94.84	2.60 ×

Table 12: Comparing verification methods for speedup ratio, where 0.5B, 7B, and 72B all represent model sizes, where the smaller is the draft model and the larger is the target model. The Qwen2.5-72B achieve accuracy of 95.53 on GSM8K dataset, pass@1 metric of 86.59 on HumanEval dataset, and scores of 8.65 and 96.71 on MT-Bench and AlpacaEval datasets, respectively.

Model	Method	GSM8K			HumanEval			MT-Bench		
		Accuracy	Speedup	τ	Pass@1	Speedup	τ	Score	Speedup	τ
Qwen2.5-0.5B		45.56	4.83×	-	34.15	4.56×	-	3.77	4.52×	-
Qwen2.5-72B		95.53	1.0×	-	86.59	1.0×	-	8.65	1.0×	-
Q2.5 -0.5B-72B	SD	95.53	1.69×	8.55	86.59	1.59×	8.50	8.65	0.80×	4.54
	MD	94.63	2.03×	9.77	85.99	1.91×	9.83	8.47	0.99×	5.28
	CD	95.83	2.39×	10.99	85.97	2.23×	10.97	8.35	1.25×	6.34
Qwen2.5-7B		91.36	3.75×	-	83.54	3.59×	-	7.79	4.44×	-
Q2.5 -7B-72B	SD	95.53	1.78×	11.18	86.59	1.74×	11.79	8.65	1.01×	7.08
	MD	95.98	2.24×	13.31	85.99	2.20×	14.16	8.53	1.40×	9.18
	CD	95.68	2.53×	15.37	89.63	2.49×	16.25	8.63	1.75×	11.66

Table 13: Comparing verification methods with draft length setting to 20, 0.5B, 7B, and 72B all represent model sizes, where the smaller is the draft model and the larger is the target model. τ is the average acceptance length.

Model	Verify	GSM8K		HumanEval		MT-Bench	
		Accuracy	Speedup	Pass@1	Speedup	Score	Speedup
Qwen2.5-0.5B		45.56 (47.7%)	4.83×	34.15 (39.4%)	4.56×	3.77 (43.6%)	4.52×
Qwen2.5-72B		95.53 (100.0%)	1.0×	86.59 (100.0%)	1.0×	8.65 (100.0%)	1.0×
Qwen2.5 0.5B/72B	SD	95.45 (99.9%)	2.22×	87.20 (100.7%)	2.18×	8.65 (100.0%)	1.57×
	MD	94.92 (99.4%)	2.35×	84.76 (97.9%)	2.31×	8.34 (96.4%)	1.59×
	CD	96.13 (100.6%)	2.47 ×	84.15 (99.3%)	2.42 ×	8.32 (96.2%)	1.80 ×
Qwen2.5-7B		91.36 (95.6%)	3.75×	83.54 (96.5%)	3.59×	7.79 (90.0%)	4.44×
Qwen2.5 7B/72B	SD	95.45 (99.9%)	2.09×	86.59 (100.0%)	2.12×	8.65 (100.0%)	1.68×
	MD	95.53 (100.0%)	2.24×	88.41 (102.1%)	2.22×	8.41 (97.2%)	1.79×
	CD	96.06 (100.6%)	2.50 ×	89.63 (103.5%)	2.51 ×	8.53 (98.6%)	1.93 ×

Table 14: Comparing verification methods for speedup ratio, based on Generic setting, where 0.5B, 7B, and 72B all represent model sizes, where the smaller is the draft model and the larger is the target model.

Model	Verify	GSM8K		HumanEval		MT-Bench		AlpacaEval	
		Threshold	Length	Threshold	Length	Threshold	Length	Threshold	Length
Qwen2.5 0.5B/72B	SD	-	6	-	6	-	6	-	6
	MD	0.7	10	0.6	10	0.8	8	0.7	8
	CD	6.0	15	6.0	20	4.0	8	4.0	8
Qwen2.5 7B/72B	SD	-	8	-	8	-	6	-	6
	MD	5.0	25	5.0	25	6.0	20	6.0	20
	CD	6.5	25	7.5	25	6.0	25	5.0	20

Table 15: The table provide the threshold and draft length for Optimal-Speedup Setting, where "Length" is draft length.