Kangaroo: Lossless Self-Speculative Decoding for Accelerating LLMs via Double Early Exiting

Fangcheng Liu[†] **Yehui Tang**[†] **Zhenhua Liu**[†] **Yunsheng Ni**[†] **Duyu Tang**[‡]
Kai Han[†,*] **Yunhe Wang**[†,*]
[†]Huawei Noah's Ark Lab [‡]Consumer Business Group, Huawei
{liufangcheng3,yehui.tang,kai.han,yunhe.wang}@huawei.com

Abstract

Speculative decoding has demonstrated its effectiveness in accelerating the inference of large language models (LLMs) while maintaining an identical sampling distribution. However, the conventional approach of training separate draft model to achieve a satisfactory token acceptance rate can be costly and impractical. In this paper, we propose a novel self-speculative decoding framework *Kangaroo* with *double* early exiting strategy, which leverages the shallow sub-network and the `LM Head` of the well-trained target LLM to construct a self-drafting model. Then, the self-verification stage only requires computing the remaining layers over the *early-exited* hidden states in parallel. To bridge the representation gap between the sub-network and the full model, we train a lightweight and efficient adapter module on top of the sub-network. One significant challenge that comes with the proposed method is that the inference latency of the self-draft model may no longer be negligible compared to the big model. To boost the token acceptance rate while minimizing the latency of the self-drafting model, we introduce an additional *early exiting* mechanism for both single-sequence and the tree decoding scenarios. Specifically, we dynamically halt the small model's subsequent prediction during the drafting phase once the confidence level for the current step falls below a certain threshold. This approach reduces unnecessary computations and improves overall efficiency. Extensive experiments on multiple benchmarks demonstrate our effectiveness, where Kangaroo achieves walltime speedups up to 2.04×, outperforming Medusa-1 with 88.7% fewer additional parameters. The code for Kangaroo is available at https://github.com/Equationliu/Kangaroo.

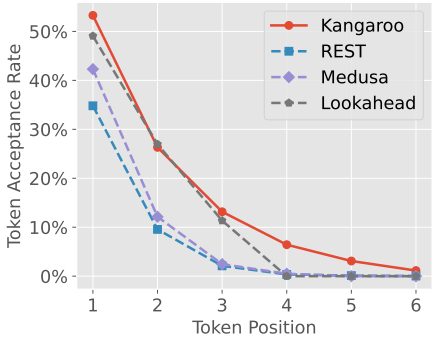
1 Introduction

Large Language Models (LLMs) [1, 2, 3, 4, 5, 6] have demonstrated remarkable performance across various natural language processing tasks, such as chain-of-thought reasoning [7] and agents [8]. Beyond model performance, recent research has also focused on inference efficiency [9, 10], particularly in scenarios involving the deployment of LLMs for service applications. However, constrained by the bottleneck of memory bandwidth [11], the primary latency for autoregressive decoding of LLMs arises mainly from memory read/write operations rather than arithmetic computations, leading to inadequate parallelism. For instance, decoding Vicuna-33B [12] on four NVIDIA V100 GPUs yields a throughput of only seven new tokens per second.

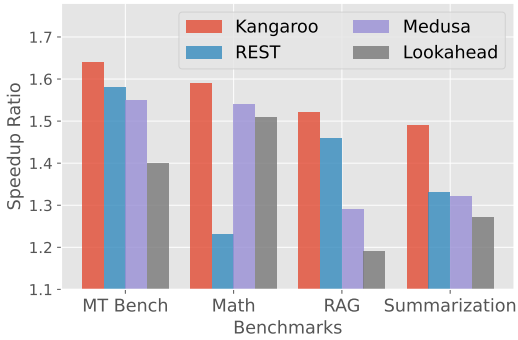
To address this challenge, Speculative Decoding (SD) techniques [13, 14] have been developed, aiming to accelerate autoregressive decoding by verifying multiple tokens generated by a draft model

[*]Corresponding Author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).



(a) Token acceptance rate on the *mathematical reasoning* subtask. Token position "2" denotes the task to predict the next-next-token.



(b) End-to-end speedup ratio on four subtasks in Spec-Bench. "Math" and "RAG" denote mathematical reasoning and retrieval-augmented generation, respectively.

Figure 1: Comparison of various self-drafting speculative decoding methods *without tree mask* on Spec-Bench [22] for Vicuna-7B [12]. Kangaroo outperforms all other methods *w.r.t.* end-to-end speedup ratio across all the four subtasks. Specifically, Kangaroo (without tree) achieves speedups of 1.68× on MT-bench [23], outperforming Medusa with 88.7% fewer additional parameters.

in parallel. Given γ draft tokens, SD can generate 1 to $\gamma + 1$ new tokens within each forward pass of the big LLM. Most of existing SD methods typically train a *tiny* draft model from scratch on a large corpus to accelerate LLMs from the same series, *e.g.*, LLaMA-68M [15] for LLaMA-7B [2]. Distillspec [16] uses knowledge distillation to better align the draft model with the target model. However, the training of such task-specific models can be costly [16, 17], limiting its application in real-world scenarios.

To mitigate these costs, several studies have proposed *self-drafting* methods that do not rely on external drafter models. LLMA [18] and REST [19] generate draft tokens by selecting text spans from references or retrieving relevant tokens from a database. Notably, Medusa [20] trains multiple time-independent Feed-Forward neural Network (FFN) heads on top of the last decoder layer. Although Medusa and REST could generate draft tokens efficiently, however, the token acceptance rate is not always satisfactory (see Figure 1(a)). On the other hand, focusing exclusively on the token acceptance rate without considering the latency of generating draft tokens can lead to suboptimal speedup ratio. For instance, as shown in Figure 1, Lookahead [21] achieves a significantly higher token acceptance rate than Medusa in the mathematical reasoning subtask. However, due to its lower efficiency in the drafting phase when compared to Medusa, its end-to-end speedup ratio is slightly lower than that of Medusa (see Figure 1(b)).

In this paper, we propose a novel self-speculative decoding framework based on a *double* early-exiting mechanism. Rather than training the draft model from scratch, we enhance efficiency by inheriting part of the target model's knowledge through parameter sharing. Specifically, Kangaroo utilizes both the shallow sub-network and the `LM Head` of the target LLM to construct a self-drafting model. However, there is a natural discrepancy in the representation capabilities between the shallow network and the final layer of the model. To bridge this gap, we incorporate a lightweight and efficient adapter module trained on top of the sub-network. Kangaroo's adapter network is streamlined yet effective, comprising only one multi-head attention layer [24] and two normalization layers [25], utilizing just 11.3% of the parameters used by Medusa's heads. This *layer-level* early-exiting strategy significantly reduces both the training and deployment costs typically associated with traditional self-speculative decoding methods that necessitate separate draft model.

Moreover, to achieve an optimal balance between token acceptance rate and drafting efficiency, Kangaroo introduces an additional *token-level* dynamic drafting strategy via early exiting. Unlike existing methods that rely on a fixed drafting step and a predefined static token tree, our approach dynamically adjusts the depth and width of the token tree based on the conditional probability distribution of the self-drafting model. In the case of single-sequence decoding, which is a specific scenario within the token tree, the drafting phase is immediately halted if the top-1 probability of the current sample token (in the self-drafting model) falls below a predefined threshold. Extensive experiments conducted on the Spec-Bench demonstrate the effectiveness of our approach. Kangaroo

achieves speedups of up to 2.04×, significantly outperforming Medusa-1 while using 88.7% fewer additional parameters (67M compared to 591M.). These results highlight the efficiency and potential of Kangaroo in improving decoding processes without compromising performance.

2 Related Work

Inference Acceleration of Large Language Models With the rapid development of large language models, significant research effort has been dedicated to accelerating their inference speed [26]. Techniques such as knowledge distillation [27], model compression [28], and quantization [29] have also been widely applied in this area. However, these approaches often require additional training of the backbone or substantial modifications to the model architecture. FlashAttention [9] and vLLM [10] frameworks accelerate the inference speed of large language models by optimizing memory management. Recent efforts have explored early exiting on models like the T5 series [30, 31, 32] and decoder-only architectures [33]. However, since early exiting accelerates inference by saving subsequent computations, it inevitably incurs the issue of performance degradation [30].

Speculative Decoding Speculative decoding has gained significant attention due to its ability to accelerate the inference of LLMs while maintaining the same sampling distribution. Generally, speculative decoding [13, 14] involves finding or training [16, 34] a small draft model closely aligned with the target LLM. Consequently, recent research has focused on more convenient self-drafting methods. For instance, approaches like blockwise parallel decoding [35] and Medusa [20] expedite the generation of draft tokens by training multiple time-independent Feedforward Neural Networks at the second top layer. Several self-drafting acceleration techniques are inspired by early exiting. Draft & Verify [36], for example, generates draft tokens by skipping intermediate redundant layers of the target LLM. While this approach could achieve a high token acceptance rate, the inference latency of the "small model" is exceptionally high, which can hinder end-to-end acceleration efficiency. SPEED [37] adapts early exiting to pipelined speculative execution for transformer decoders that employ parameter sharing. There are also several works [38, 39, 40] that make improvement on Medusa by introducing time dependency among the draft tokens. Unlike our approach, which utilizes early exiting from shallow layers, these methods typically extrapolate directly from the features of the penultimate layer. For more related works on speculative decoding, we refer readers to a recent survey [22] for a detailed summarization.

3 Preliminaries

In this section, we introduce background and the formulations of standard speculative decoding under greedy decoding. We use x^t to denote the discrete token sequence $(x_1, \cdots, x_t)$ and $x^{i:j}$ to represent sequence $(x_i, \cdots, x_j)$. Let $\mathcal{X}$ be a discrete space over all possible tokens x in the LLM's vocabulary, we model the autoregressive process of a language model $\mathcal{M}$ by the conditional distributions $\mathcal{M}(\cdot \mid x^t) \in \mathbb{R}^{|\mathcal{X}|}$ where $|\mathcal{X}|$ is the vocabulary size. We denote the big target language model and the speculative small model as $\mathcal{M}_b$ and $\mathcal{M}_s$, respectively.

Single-Sequence Decoding Due to the memory bandwidth [11] limitations of autoregressive decoding in large language models, the latency to generate one new token is approximately the same as the time required to parallelly infer γ tokens. Based on this characteristic, speculative decoding [13, 14] leverages a low-cost small model to quickly generate γ candidates

$$\hat{x}_{t+i} = \arg\max_{x \in \mathcal{X}} \mathcal{M}_s(x \mid \hat{x}^{t+i-1}), \quad i = 1, 2, \cdots, \gamma, \tag{1}$$

where $\hat{x}^t = x^t$. Subsequently, the big model $\mathcal{M}_b$ only needs one forward pass over $(\hat{x}_t, \hat{x}_{t+1}, \cdots, \hat{x}_{t+\gamma})$ to generate $\gamma + 1$ tokens $(x_{t+1}, \cdots, x_{t+\gamma+1})$. The accept length over this single-sequence verification is

$$\max_{i=0,\cdots,\gamma} \{i \mid \hat{x}_{t+i} = x_{t+i}\} + 1, \tag{2}$$

ranging from 1 to $\gamma + 1$.

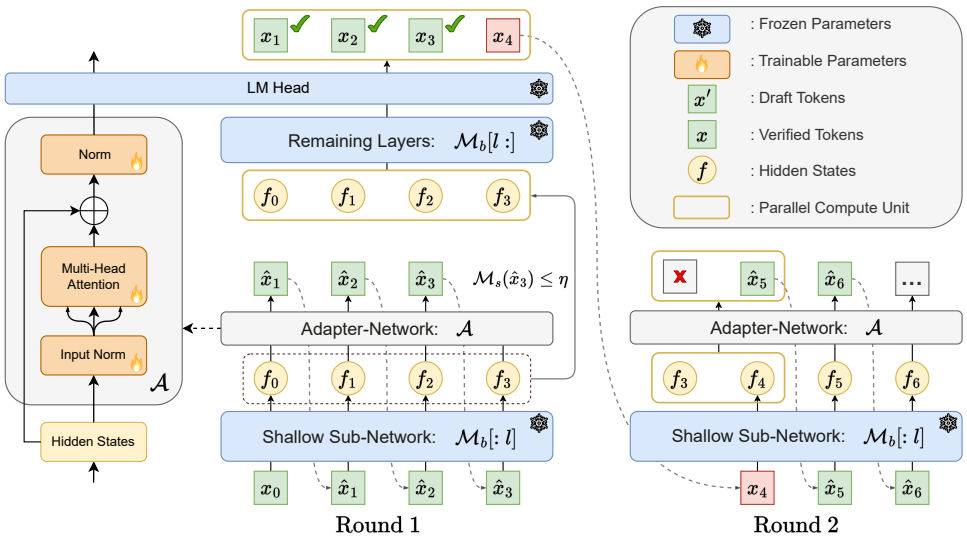


Figure 2: The framework of Kangaroo under single-sequence verification. The adapter network $\mathcal{A}$ consists of only one multi-head attention and two normalization layers. The self-drafting model $\mathcal{M}_s = \mathcal{A} \circ \mathcal{M}_b[:l]$ will reuse the `LM Head` of the target LLM $\mathcal{M}_b$ for better alignment, where l denotes the early exit layer. To avoid unnecessary costs on more difficult tokens, $\mathcal{M}_s$ stops drafting once the top-1 probability of the current sampled token falls below a certain threshold, *e.g.*, $\mathcal{M}_s(\hat{x}_3 \mid \hat{x}^{0:2}) \leq \eta$. Note that we will concatenate the stopped token's *next early feature* f_3 with all previous exited features into a parallel compute unit $[f_0, f_1, \cdots, f_3]$, which will be verified by the remaining layers $\mathcal{M}_b[l:]$ in parallel. Once all drafted tokens are accepted ($\hat{x}_i = x_i$ for $i = 1, 2, 3$), we could start the next round with x_4 rather than x_3 if we have not calculated f_3 in advance. The decoding on parallel unit $[f_3, f_4]$ could save the latency for a single forward pass of $\mathcal{A}$.

4 Kangaroo

In this section, we present Kangaroo, a novel self-speculative decoding framework based on a *double* early-exiting mechanism. We leverage the shallow sub-network and the `LM Head` of the target LLM to construct a self-drafting model. Subsequently, the self-verification stage only requires computing the remaining layers over the *early-exited* hidden states in parallel.

4.1 Early Exiting as Self-Drafting Model

Training an additional small model from scratch is often costly and impractical, thus it is worth considering sharing a portion of the parameters from the target LLM. Drawing inspiration from early exiting, we directly extract hidden states from a fixed shallow sub-network of the target LLM

$$f_t = \mathcal{M}_b[:l](x_t), \quad l \in \{1, 2, \cdots, L\},$$

where $\mathcal{M}_b[:l]$ denotes the first l layers from the target model $\mathcal{M}_b$. Note that there is a natural representation gap between the shallow sub-network and the full model. Therefore, we train a lightweight and efficient adapter $\mathcal{A}$ to bridge this gap. As shown in Figure 2, the architecture of the adapter $\mathcal{A}$ consists of only one multi-head attention and two normalization layers. Given an exited hidden states f_t, the forward pass of the adapter model $\mathcal{A}$ can be represented as

$$f'_t = f_t + \texttt{MultiHead}\left(\texttt{LayerNorm}(f_t)\right),$$

where we keep the residual connection and the multi-head attention module but remove the FFN in a standard transformer block. Besides the shallow sub-network, Kangaroo also reuses the `LM Head` of the target model to get the final conditional distribution, *i.e.*,

$$\mathcal{M}_s(x \mid x^t) = \texttt{Softmax}\left(\mathbf{W}^\top \texttt{LayerNorm}(f'_t)\right),$$

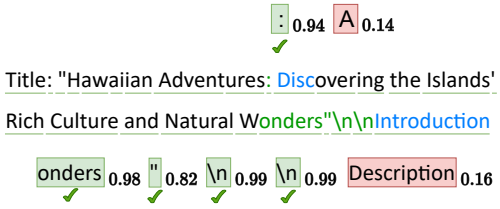


Figure 3: The difficulty of token acceptance varies across different contexts. The values on the right side of the rectangular blocks represent the top-1 probability of the tokens on the self-drafting model $\mathcal{M}_s$. Green boxes indicate accepted tokens, while red boxes represent rejected tokens. Blue tokens signify corrections made by the big target model $\mathcal{M}_b$.

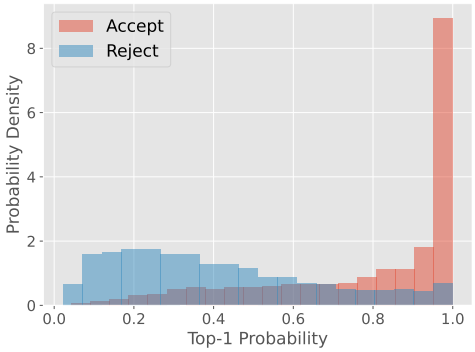


Figure 4: The density of top-1 conditional probability on the *mathematical reasoning* subtask. "Accept" denotes the top-1 confidence of accepted draft tokens while "Reject" denotes the corresponding confidence of rejected tokens.

where $\mathbf{W} \in \mathbb{R}^{N\times|\mathcal{X}|}$ is the frozen `LM Head` and N denotes the dimension of hidden states. The total parameters of $\mathcal{A}$ comes from the four projection matrix in the `MultiHead` layer and two vectors in the `LayerNorm` layers, *i.e.*, $4N^2 + 2N = 67.1$M when $N = 4096$.

Single-Sequence Self-Speculative Decoding In the drafting phase, the self-drafting model $\mathcal{M}_s$ generates γ new tokens autoregressively. Similar to the standard speculative decoding in Equation (1), we have $\hat{x}_{t+i} = \arg\max_{x\in\mathcal{X}} \mathcal{M}_s(x \mid \hat{x}^{t+i-1})$ and the exited hidden states

$$f_{t+i} = \mathcal{M}_b[:l](\hat{x}_{t+i}), \quad i = 1, 2, \cdots, \gamma,$$

Subsequently, the remaining layers $\mathcal{M}_b[l:]$ will in charge of verifying the concatenated early-exited hidden features in parallel, *i.e.*,

$$f^{t:t+\gamma} \overset{\text{def}}{=} \texttt{Concat}\left([f_t, f_{t+1}, \cdots, f_{t+\gamma}]\right) \in \mathbb{R}^{N\times(\gamma+1)},$$
$$\mathcal{M}_b(x^{t+1:t+\gamma+1} \mid x^t) = \texttt{Softmax}\left(\mathbf{W}^\top \texttt{LayerNorm}\left(\mathcal{M}_b[l:](f^{t:t+\gamma})\right)\right),$$

where $\mathbf{W}$ is the frozen `LM Head`. In the verification procedure, we keep the same as the conventional speculative decoding by comparing the top-1 candidates of $\mathcal{M}_b(x^{t+1:t+\gamma+1} \mid x^t)$ and $\hat{x}_{t+i}$ for $i = 1, 2, \cdots, \gamma$, following Equation (2).

Training of the Adapter $\mathcal{A}$ To bridge the representation gap between the self-drafting model $\mathcal{M}_s$ and the full model $\mathcal{M}_b$, we train the adapter $\mathcal{A}$ with the conventional cross-entropy loss, *i.e.*,

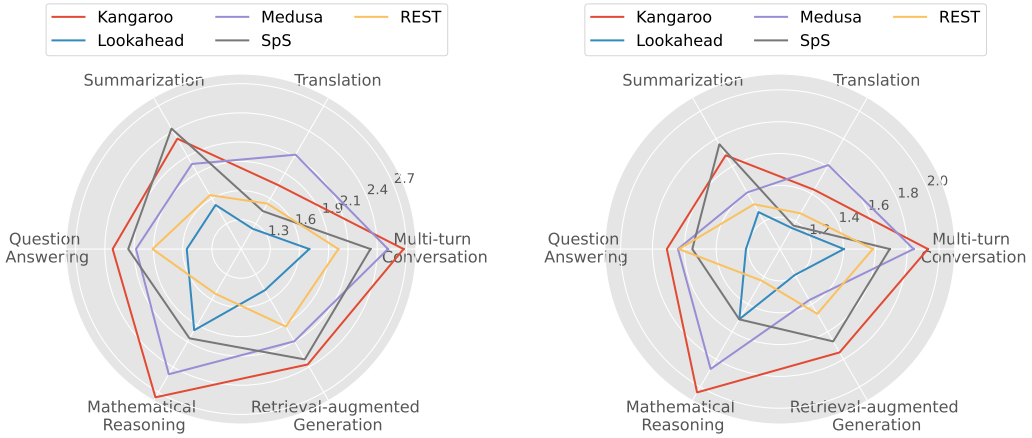
$$\mathcal{A}^* = \arg\min_{\mathcal{A}} \sum_t \sum_{x\in\mathcal{X}} -\mathcal{M}_b(x \mid x^t) \log \mathcal{M}_s(x \mid x^t),$$

where the token positions t is averaged over the whole training set and the condition probability distribution of the target LLM $\mathcal{M}_b$ serves as a distillation for the self-drafting model $\mathcal{M}_s$'s.

4.2 Dynamic Drafting Steps via Token-Level Early-Exiting

Speculative decoding typically employs a fixed drafting step γ during the drafting phase, but this often leads to local optima. Fixed-step drafting can result in unnecessary time spent on more challenging samples or missed opportunities to speculate on simpler tokens. As shown in Figure 3, the difficulty of predicting the next token varies across different contextual scenarios, necessitating the design of a token-wise dynamic drafting strategy.

Single-Sequence Decoding During drafting phase, we do not have access to the full target model $\mathcal{M}_b$ and must rely solely on the information from the self-drafting model $\mathcal{M}_s$, to determine whether to continue or terminate the drafting process. Fortunately, we observe a strong correlation between the small model's confidence level for the current sampled token and the likelihood that the token



(a) Compression rate.

(b) End-to-end speedup ratio.

Figure 5: Comparison of various speculative decoding methods on Spec-Bench [22] for Vicuna-7B, where Kangaroo outperforms other approaches in most subtasks, especially in mathematical reasoning and retrieval-augmented generation.

will be accepted by the big target model. In Figure 4, all conditional probability values of the tokens are divided into two groups. The peak values for tokens accepted by the large model skew to the right, while the conditional probability values for tokens rejected by the large model cluster at lower confidence levels. This indicates that the top-1 probability on the Kangaroo's small model is a reliable metric for determining when to stop drafting. Therefore, we stop drafting once the top-1 probability on the self-drafting model falls below a predefined threshold η, *i.e.*,

$$\max_{x \in \mathcal{X}} \mathcal{M}_s(x \mid x^t) \leq \eta. \tag{3}$$

Extension to Tree Decoding Based on the single-sequence verification procedure, the big model $\mathcal{M}_b$ can only verify one feasible path, composed of the top-1 candidates generated by the small model $\mathcal{M}_s$. Tree verification technology [15, 20] enhances parallelism and improves GPU utilization by designing a sparse token tree to simultaneously verify multiple feasible paths. Most current works that adopt token tree verification rely on handcrafted or pre-searched [41] static tree. We generalize the early stopping mechanism of Kangaroo in single-sequence decoding to tree decoding, where both the tree depth and node selection at each level of the tree are token-wise dynamic.

We define the first draft token from $\mathcal{M}_s$ as the root[2] node. To increase the token acceptance rate, we feed the Top-K[3] most probable tokens into the draft model $\mathcal{M}_s$ in parallel at each drafting step. The confidence score of a child node is computed as the product of its conditional probability in $\mathcal{M}_s$ and its parent's conditional probability, Formally,

$$c(x_i) = \prod_{x_j \in \mathrm{Path}(\mathrm{root}, x_i)} c(x_j) \ ,$$

where $\mathrm{Path}(\mathrm{root}, x_i)$ denotes the path from root to node x_i, $c(x_i)$ is the probability confidence on $\mathcal{M}_s$ and $c(\mathrm{root}) = 1$. At the first level, we begin with Top-K$_{x \in \mathcal{X}} \mathcal{M}_s(x \mid \mathrm{root})$. For level ≥ 2, we consider the Top-K child nodes for each token in the previous level's Top-K tokens. In this way, the tokens with the Top-K largest confidence score will be selected from Top-K $\times$ Top-K candidates. For the selected Top-K tokens, their parent nodes will be added into the token tree $\mathcal{T}$ directly. For tokens in the previous level's Top-K set without any child node selected, half of those with the lowest confidence scores are pruned from $\mathcal{T}$. This process continues until the tree size $|\mathcal{T}|$ surpasses a predefined maximum or the highest confidence score at the current level falls below a threshold η.

[2]The zero level of the token tree.

[3]Single-sequence decoding is a special case when Top-K equals to 1.

Table 1: Speedup comparison of various speculative decoding methods on Spec-Bench [22] for Vicuna [12]. Speedup is the walltime speedup ratio and CR denotes the compression rate.

Size	Method	Translation		QA		Summarization		Math		RAG		MT Bench		Avg.
		CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	
7B	Lookahead [21]	1.24	1.15×	1.56	1.21×	1.53	1.27×	1.96	1.51×	1.49	1.19×	1.70	1.40×	1.29×
	Medusa *w/o Tree*	1.58	1.41×	1.50	1.34×	1.49	1.32×	1.73	1.54×	1.51	1.29×	1.76	1.55×	1.41×
	REST [19]	1.54	1.26×	1.91	1.63×	1.64	1.33×	1.53	1.23×	1.92	1.46×	2.00	1.58×	1.43×
	Kangaroo *w/o Tree*	1.41	1.24×	1.87	1.43×	1.87	1.50×	2.14	1.61×	2.05	1.52×	2.22	1.68×	1.50×
	SpS [13]	1.45	1.17×	2.16	1.55×	2.43	**1.76**×	2.06	1.51×	2.31	1.67×	2.33	1.69×	1.56×
	Medusa [20]	2.12	**1.60**×	2.08	1.64×	2.01	1.41×	2.48	1.87×	2.09	1.37×	2.51	1.84×	1.63×
	Kangaroo	1.76	1.43×	2.32	**1.71**×	2.31	1.68×	2.76	**2.04**×	2.37	**1.75**×	2.67	**1.93**×	**1.72**×
13B	Lookahead [21]	1.25	1.02×	1.39	0.99×	1.50	0.98×	1.94	1.24×	1.52	0.94×	1.68	1.08×	1.04×
	REST [19]	1.53	1.07×	1.92	1.41×	1.66	1.14×	1.55	1.06×	1.87	1.34×	1.98	1.36×	1.23×
	Medusa [20]	2.19	1.21×	2.11	1.17×	2.08	1.21×	2.59	1.41×	2.12	1.12×	2.58	1.38×	1.24×
	Medusa *w/o Tree*	1.61	1.33×	1.49	1.25×	1.53	1.25×	1.80	1.48×	1.53	1.23×	1.82	1.48×	1.34×
	Kangaroo *w/o Tree*	1.45	1.18×	1.79	1.34×	2.00	1.41×	2.42	1.63×	2.16	1.40×	2.44	1.66×	1.44×
	SpS [13]	1.44	1.21×	1.83	1.49×	2.32	**1.62**×	2.15	1.63×	2.43	1.61×	2.25	1.65×	1.54×
	Kangaroo	1.79	**1.39**×	2.25	**1.66**×	2.41	1.57×	2.82	**1.87**×	2.49	**1.68**×	2.71	**1.79**×	**1.65**×

5 Experiments

Models and Benchmarks We conduct experiments on Vicuna [12] models with size of 7B and 13B. We select several speculative decoding approaches for comparison including Lookahead [21], Medusa [20], REST [19] and SpS [13] using Vicuna-68M [17] as the draft model. For fair and comprehensive comparison, we evaluate the acceleration performance with the recently proposed Spec-Bench [22], which consists of six subtasks including Multi-turn Conversation, Translation, Summarization, Question Answering, Mathematical Reasoning and Retrieval-augmented Generation.

Evaluation Metrics Speculative decoding is often evaluated using two primary metrics: walltime speedup ratio and compression rate. Given a speculative decoding algorithm, we execute it to generate T new tokens and record the accepted tokens per forward pass of the big model as a list $S = [s_1, s_2, \cdots, s_{|S|}]$ where $\sum_k s_k = T$. The Compression Rate (CR) is defined as

$$\mathrm{CR} = \frac{1}{|S|} \sum_k s_k, \tag{4}$$

which does not accurately reflect the acceptance levels of the drafting algorithm for tokens at varying distances. Thus, we propose a new metric consistent token acceptance rate $\mathrm{CTAR}(w)$, given a prefix and a following window with size w, is the probability that the w guessed tokens from the draft model are *all* accepted by the target model:

$$\mathrm{CTAR}(w) = \frac{1}{|S|} \sum_k \mathbb{I}(s_k - w > 0), \tag{5}$$

which is a decreasing function *w.r.t.* the window size w. We plot the empirical CTARs of several self-drafting speculative decoding algorithms on the mathematical reasoning subtask of Spec-Bench in Figure 1, where we observe a trade-off between the token acceptance rate and drafting efficiency.

Training and Inference As shown in Figure 2, the big target model is frozen and the trainable parameters are the multi-head attention and two normalization layers. For Kangaroo, we train the adapter $\mathcal{A}$ for 10 epochs with the AdamW [42] optimizer on the ShareGPT dataset following Medusa [20]. The training of the adapter $\mathcal{A}$ for Vicuna-7B takes around 24 hours on 8 NVIDIA V100 GPUs. During the inference stage, we set $\ell = 2$ for Vicuna-7B and $\ell = 3$ for Vicuna-13B. For the single-sequence decoding in Kangaroo, we set $\gamma = 6$ and $\eta = 0.6$. For the dynamic tree decoding scenario, we set Top-K as 10, and $\eta = 0.4$.

5.1 Effectiveness: Comparison with other Speculative Decoding Methods

The Spec-Bench results for Vicuna families[4] are available at Figure 5 and Table 1, where we can conclude that:

[4] Additional result for Llama2-13B-Chat [2] is available at Table 4.

- For both 7B and 13B model sizes, Kangaroo achieves the highest average speedup across all tasks, with 1.72× and 1.65× respectively. Furthermore, at both model sizes, Kangaroo with dynamic tree verification demonstrates approximately a 12% improvement in speedup compared to its single-sequence decoding version, indicating its effectiveness.
- In both single-sequence and tree verification scenarios, Kangaroo outperforms Medusa across most datasets, despite Kangaroo's additional parameters comprising only 11.3% of Medusa's heads. Kangaroo shows consistent high performance across different tasks such as QA, Math, and MT Bench, indicating its robustness and versatility.
- Under single-sequence decoding, Kangaroo (67M) achieves comparable results to the SpS method using Vicuna-68M, which is specifically pre-trained from scratch. Furthermore, with tree verification, Kangaroo significantly outperforms SpS across most datasets.

5.2 Ablation Studies

To fully explore the sensitivity of different hyperparameters in the Kangaroo framework, we conducted comprehensive ablation experiments focusing on aspects such as the depth of shallow sub-Network ℓ, the early stopping threshold η, and the architectural choices of the adapter network $\mathcal{A}$. In the settings of the ablation experiments described below, we focus on single-sequence decoding scenario.

The Depth of Shallow Sub-Network. The capacity of the self-drafting model $\mathcal{M}_s$ highly depends on the depth of the shared shallow sub-network. However, selecting deeper early exiting layers, such as half layers of $\mathcal{M}_b$, would result in excessively high inference latency. As shown in Figure 7(a), the choice of the optimal early exitlayer l significantly influences the trade-off between token acceptance rate and drafting efficiency. For Kangaroo, we set $l = 2$ for Vicuna-7B and $l = 3$ for Vicuna-13B. In general cases, the early exit layer ℓ can be set based on an empirical ratio derived from the target model's depth N. To determine the optimal depth for shallow sub-networks relative to the full model depth, we conducted multiple comparative experiments across models with different architectures and sizes. Figure 6 records the average speedup achieved on Spec-Bench with early exits at various depths. We recommend setting $\frac{\ell}{N}$ between $\frac{1}{16}$ and $\frac{1}{10}$ in practical applications.

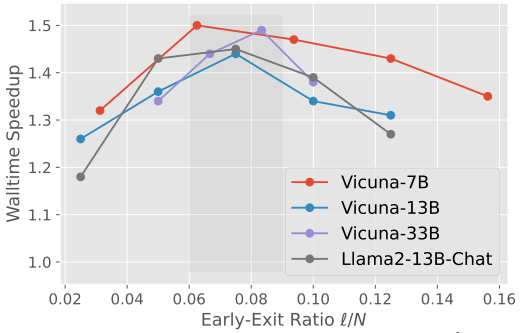


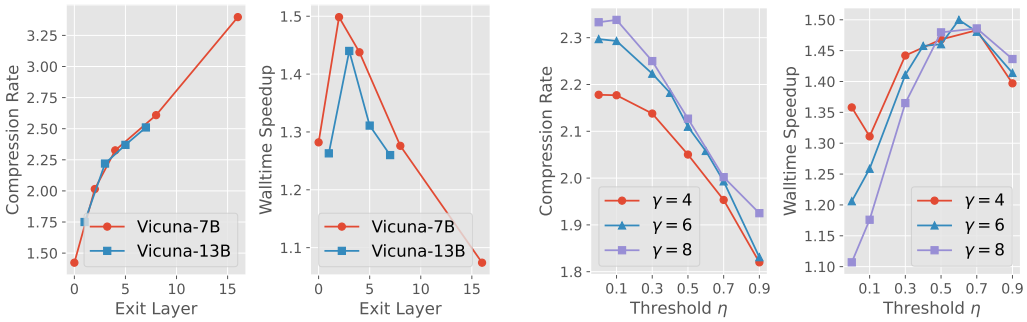
Figure 6: The optimal early-exit ratio $\frac{\ell}{N}$.

The Architecture of the Adapter Module. In a transformer block, the Feed-Forward Network (FFN) component counts for 67% of the whole parameters. As shown in Table 2, removing the FFN component and sharing the `LM Head` from the target Large Language Model (LLM) has been proven to be highly effective and efficient. Specifically, Kangaroo's adapter module $\mathcal{A}$ for Vicuna-7B achieves a higher "Speedup" of 1.50 ×, compared to 1.41 × for Medusa, while using only about one-tenth of the additional parameters as Medusa.

Table 2: Ablation studies on the architecture of the adapter module $\mathcal{A}$ for Vicuna-7B. "Speedup" denotes the average speedup ratio on Spec-Bench [22].

Architecture	Input LN	Attention	Post LN	FFN	*Linear*	Last LN	Head	# Parameters	Speedup
Medusa	✗	✗	✗	✗	× 4	✗	× 4	591M	1.41 ×
Kangaroo	✓	✓	✗	✗	✗	✓	✗	**67M**	**1.50** ×
Kangaroo + Head	✓	✓	✗	✗	✗	✓	✓	198M	1.44 ×
1-Layer Transformer	✓	✓	✓	✓	✗	✓	✗	202M	1.37 ×
MLP Only	✓	✗	✗	✗	× 2	✓	✓	165M	1.22 ×

Dynamic Exiting *v.s.* Fixed Step Drafting. Kangaroo uses a static threshold to determine the timing of the second early stopping, based on the observation that the confidence of draft tokens in the small model is strongly correlated with their acceptance by the large model. To validate the effectiveness of dynamic drafting steps with fixed threshold, we plot the comparison for various η



(a) Optimal exit-layer l. (b) Optimal threshold η.

Figure 7: Ablation studies on hyper-parameters. The compression rate and walltime speedup is averaged across all sub-benchmarks in Spec-Bench [22]. It can be seen that there is a trade-off between token acceptance rate and drafting efficiency. While deeper early-exit layer ℓ can enhance the expressive power of the equivalent draft model in Kangaroo, the increased inference latency can actually hinder the overall acceleration performance of the system.

in Figure 7(b). The fixed step strategy ($\eta = 0$) achieves the maximum compression rate, however, leading to sub-optimal end-to-end walltime speedup. Overall, the optimal threshold η is consistent across different maximum steps. For Kangaroo, we set $\gamma = 6$ and $\eta = 0.6$. Besides, the static threshold is also robust across different subtasks and architectures. We visualized the conditional distributions of the small model's Top-1 confidence across different model architectures and subtasks in Figures 8 and 9 in the appendix, where the thereshold is stable, ranging from 0.6 to 0.8.

Temperature Sampling It's well known that speculative sampling suffers from decreased speedup ratio when increasing the sampling temperature [22]. We consider two sets of comparative experiments ion Kangaroo for the temperature sampling case. One strategy uses the original top-1 confidence as a stopping criterion, while the other uses an adjusted top-1 confidence which is computed by applying softmax to adjusted logits with a temperature scaling factor. As shown in Table 3, it can be seen that Kangaroo still achieves a speedup effect very similar to that of greedy decoding when $T = 0.2$. Moreover, the top-1 confidence used for the second early stopping mechanism should remain unadjusted because when $T < 1$, the adjusted top-1 confidence tends to be overestimated, making it more difficult to trigger the early stopping mechanism.

Table 3: Speedup comparison of various temperature T on Spec-Bench [22] for Vicuna [12]. Speedup is the walltime speedup ratio and CR denotes the compression rate.

T	Confidence	Translation		QA		Summarization		Math		RAG		MT Bench		Avg.
		CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	
0.0	Original	1.41	1.24×	1.87	1.43×	1.87	1.50×	2.14	1.61×	2.05	1.52×	2.22	1.68×	1.50×
0.2	Original	1.41	1.23×	1.88	1.41×	1.88	1.48×	2.19	1.58×	2.01	1.50×	2.21	1.67×	1.48×
0.2	Adjusted	1.45	1.04×	1.90	1.25×	2.00	1.31×	2.32	1.48×	2.18	1.40×	2.41	1.53×	1.34×
0.5	Original	1.41	1.18×	1.86	1.38×	1.88	1.43×	2.21	1.55×	2.01	1.46×	2.23	1.62×	1.43×

6 Conclusion

In conclusion, Kangaroo presents a novel self-speculative decoding framework that significantly enhances the efficiency of autoregressive decoding for large language models. By leveraging a double early-exit mechanism, Kangaroo achieves remarkable speedups compared to existing methods, even with significantly fewer additional parameters. Experimental results demonstrate the effectiveness of Kangaroo in balancing the token acceptance rate and drafting efficiency, making it a promising approach for accelerating inference of LLMs in real-world scenarios.

Limitations Although we introduce a lightweight adapter module to bridge the gap between the shallow network and the final layer of the model, the effectiveness of Kangaroo relies on the target model and the specific task. In certain complex tasks, this representation gap may still persist.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1

[2] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1, 2, 7, 13

[3] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. 1

[4] Yunhe Wang, Hanting Chen, Yehui Tang, Tianyu Guo, Kai Han, Ying Nie, Xutao Wang, Hailin Hu, Zheyuan Bai, Yun Wang, et al. Pangu-π: Enhancing language model architectures via nonlinearity compensation. *arXiv preprint arXiv:2312.17276*, 2023. 1

[5] Yehui Tang, Fangcheng Liu, Yunsheng Ni, Yuchuan Tian, Zheyuan Bai, Yi-Qi Hu, Sichao Liu, Shangling Jui, Kai Han, and Yunhe Wang. Rethinking optimization and architecture for tiny language models. *arXiv preprint arXiv:2402.02791*, 2024. 1

[6] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023. 1

[7] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 1

[8] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):1–26, 2024. 1

[9] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022. 1, 3

[10] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626, 2023. 1, 3

[11] Noam Shazeer. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019. 1, 3

[12] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 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. 1, 2, 7, 9, 13

[13] 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. 1, 3, 7, 13

[14] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR, 2023. 1, 3

[15] Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Rae Ying Yee Wong, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. Specinfer: Accelerating generative llm serving with speculative inference and token tree verification. *arXiv preprint arXiv:2305.09781*, 2023. 2, 6

[16] 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. *arXiv preprint arXiv:2310.08461*, 2023. 2, 3

[17] Sen Yang, Shujian Huang, Xinyu Dai, and Jiajun Chen. Multi-candidate speculative decoding. *arXiv preprint arXiv:2401.06706*, 2024. 2, 7

[18] Nan Yang, Tao Ge, Liang Wang, Binxing Jiao, Daxin Jiang, Linjun Yang, Rangan Majumder, and Furu Wei. Inference with reference: Lossless acceleration of large language models. *arXiv preprint arXiv:2304.04487*, 2023. 2

[19] Zhenyu He, Zexuan Zhong, Tianle Cai, Jason D Lee, and Di He. Rest: Retrieval-based speculative decoding. *arXiv preprint arXiv:2311.08252*, 2023. 2, 7

[20] 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. *arXiv preprint arXiv:2401.10774*, 2024. 2, 3, 6, 7

[21] Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. Break the sequential dependency of llm inference using lookahead decoding. *arXiv preprint arXiv:2402.02057*, 2024. 2, 7

[22] Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*, 2024. 2, 3, 6, 7, 8, 9, 13

[23] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[25] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[26] Zixuan Zhou, Xuefei Ning, Ke Hong, Tianyu Fu, Jiaming Xu, Shiyao Li, Yuming Lou, Luning Wang, Zhihang Yuan, Xiuhong Li, et al. A survey on efficient inference for large language models. *arXiv preprint arXiv:2404.14294*, 2024. 3

[27] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*, 2023. 3

[28] Yehui Tang, Yunhe Wang, Jianyuan Guo, Zhijun Tu, Kai Han, Hailin Hu, and Dacheng Tao. A survey on transformer compression. *arXiv preprint arXiv:2402.05964*, 2024. 3

[29] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023. 3

[30] Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022. 3

[31] Sangmin Bae, Jongwoo Ko, Hwanjun Song, and Se-Young Yun. Fast and robust early-exiting framework for autoregressive language models with synchronized parallel decoding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5910–5924, 2023. 3

[32] Shengkun Tang, Yaqing Wang, Zhenglun Kong, Tianchi Zhang, Yao Li, Caiwen Ding, Yanzhi Wang, Yi Liang, and Dongkuan Xu. You need multiple exiting: Dynamic early exiting for accelerating unified vision language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10781–10791, 2023. 3

[33] Neeraj Varshney, Agneet Chatterjee, Mihir Parmar, and Chitta Baral. Accelerating llama inference by enabling intermediate layer decoding via instruction tuning with lite. *arXiv e-prints*, pages arXiv–2310, 2023. 3

[34] Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. Spectr: Fast speculative decoding via optimal transport. *arXiv preprint arXiv:2310.15141*, 2023. 3

[35] Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. Blockwise parallel decoding for deep autoregressive models. *Advances in Neural Information Processing Systems*, 31, 2018. 3

[36] Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. Draft & verify: Lossless large language model acceleration via self-speculative decoding. *arXiv preprint arXiv:2309.08168*, 2023. 3, 13

[37] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Hasan Genc, Kurt Keutzer, Amir Gholami, and Sophia Shao. Speed: Speculative pipelined execution for efficient decoding. *arXiv preprint arXiv:2310.12072*, 2023. 3

[38] Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*, 2024. 3

[39] Zachary Ankner, Rishab Parthasarathy, Aniruddha Nrusimha, Christopher Rinard, Jonathan Ragan-Kelley, and William Brandon. Hydra: Sequentially-dependent draft heads for medusa decoding. *arXiv preprint arXiv:2402.05109*, 2024. 3

[40] Aonan Zhang, Chong Wang, Yi Wang, Xuanyu Zhang, and Yunfei Cheng. Recurrent drafter for fast speculative decoding in large language models. *arXiv preprint arXiv:2403.09919*, 2024. 3

[41] Zhuoming Chen, Avner May, Ruslan Svirschevski, Yuhsun Huang, Max Ryabinin, Zhihao Jia, and Beidi Chen. Sequoia: Scalable, robust, and hardware-aware speculative decoding. *arXiv preprint arXiv:2402.12374*, 2024. 6

[42] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 7

A Appendix

Table 4: Speedup comparison of various speculative decoding methods on Spec-Bench [22] for Llama2-13B-Chat [2]. Speedup is the walltime speedup ratio and CR denotes the compression rate.

Method	Translation		QA		Summarization		Math		RAG		MT Bench		Avg.
	CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	CR	Speedup	
Draft&Verify [36]	2.61	1.22×	2.36	1.02×	2.84	1.13×	2.47	1.08×	2.44	1.15×	2.46	1.12×	1.12×
SpS [13]	1.63	1.26×	1.69	1.34×	1.47	**1.14**×	1.77	1.34×	1.81	1.32×	1.67	1.28×	1.28×
Medusa *w/o Tree*	1.85	1.53×	1.55	1.27×	1.48	1.19×	1.76	1.36×	1.54	1.25×	1.72	1.43×	1.34×
Kangaroo *w/o Tree*	1.97	1.46×	1.87	1.40×	1.97	1.35×	2.22	1.52×	2.05	1.36×	2.28	1.58×	1.45×

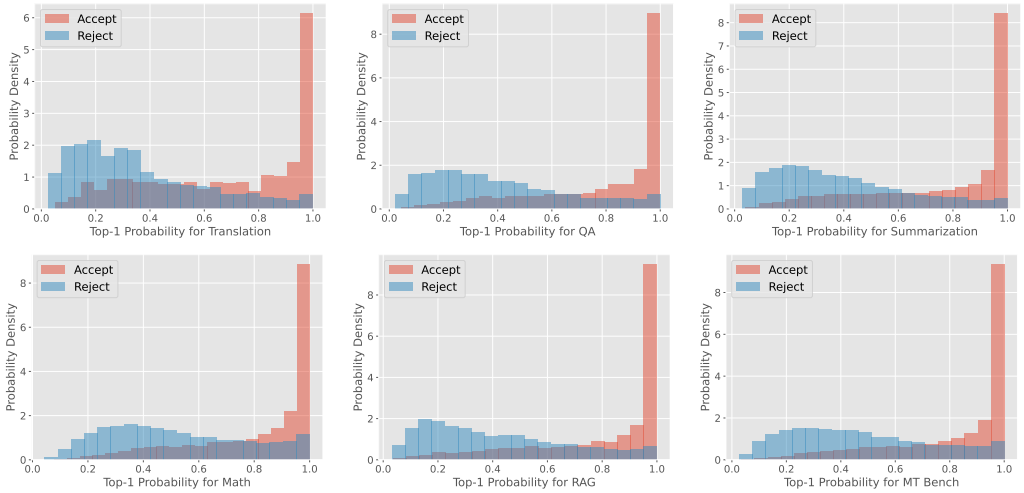


Figure 8: The density of top-1 conditional probability on various subtasks for Vicuna-7B [12].

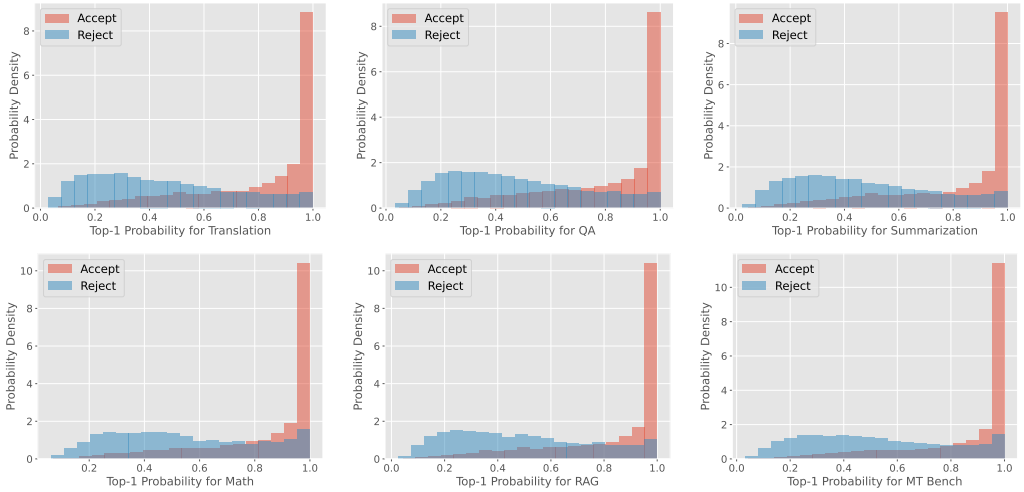


Figure 9: The density of top-1 conditional probability on various subtasks for Llama2-13B-Chat [2].

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction reflect the paper's contributions and scope accurately.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations are discussed in the Conclusion.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms

that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: The paper does not include theoretical results

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All the information needed to reproduce the main experimental results are provided in the Section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The data is open-sourced and the code implementation will be released upon accepted.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The details are provided in the Section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The error bar is not provided.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The details are provided in the Section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: It is conformed.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer:[NA]

    Justification: There is no risk in the paper.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: CC-BY 4.0 is included for each asset

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer:[NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.