EAGLE-2: Faster Inference of Language Models with Dynamic Draft Trees

Yuhui Li[1,3]*, **Fangyun Wei[2], Chao Zhang[1], Hongyang Zhang[3,4]†**
[1]School of Intelligence Science and Technology, Peking University,
[2]Microsoft Research, [3]University of Waterloo, [4]Vector Institute.
yuhui.li@stu.pku.edu.cn, fawe@microsoft.com
c.zhang@pku.edu.cn, hongyang.zhang@uwaterloo.ca

Abstract

Inference with modern Large Language Models (LLMs) is expensive and time-consuming, and speculative sampling has proven to be an effective solution. Most speculative sampling methods such as EAGLE use a static draft tree, implicitly assuming that the acceptance rate of draft tokens depends only on their position. Interestingly, we found that the acceptance rate of draft tokens is also context-dependent. In this paper, building upon EAGLE, we propose EAGLE-2, which introduces a new technique of context-aware dynamic draft tree into drafting modeling. This improvement leverages the fact that the draft model of EAGLE is well-calibrated: the confidence scores from the draft model approximate acceptance rates with small errors. We conducted extensive evaluations on three series of LLMs and six tasks, with EAGLE-2 achieving speedup ratios 3.05x-4.26x, which is 20%-40% faster than EAGLE-1. EAGLE-2 also ensures that the distribution of the generated text remains unchanged, making it a **lossless** acceleration algorithm. The code is open sourced at https://github.com/SafeAILab/EAGLE.

1 Introduction

Modern Large Language Models (LLMs) (OpenAI, 2023; Touvron et al., 2023) exhibit impressive capabilities and are widely applied across various domains. However, their parameter sizes have grown substantially, even exceeding hundreds of billions. During autoregressive generation, each token generation requires accessing all model parameters. In a single dialogue, hundreds to thousands of tokens might be generated, making LLM inference slow and expensive. Speculative sampling (Leviathan et al., 2023; Chen et al., 2023a) methods aim to address this issue by rapidly generating draft tokens

---

*The work is done when Yuhui Li is an intern student at University of Waterloo.
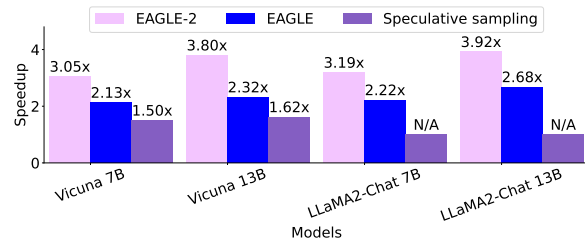
†Corresponding author.

Figure 1: Speedup ratios of different methods at temperature=1. For speculative sampling, the Vicuna series uses Vicuna-68M as the draft model. LLaMA2-Chat lacks a suitable draft model, and is marked as N/A. Methods like Medusa relax acceptance conditions under non-greedy settings, which do not guarantee lossless acceleration. *In this paper, we only compare with speculative sampling based methods ensuring the output text distribution remains constant.* In Table 1, we present comparisons with additional methods, but this figure only showcases a subset, including the fastest among these methods, EAGLE.

and then verifying them in *parallel*. These methods generate multiple tokens in a single forward pass, significantly reducing inference latency.

Standard speculative sampling (Leviathan et al., 2023; Chen et al., 2023a) uses a chain-structured draft. To improve acceptance length, recent work in speculative sampling has employed tree-structured drafts. Sequoia (Chen et al., 2024) explicitly assumes that the acceptance rate of a draft token depends only on its position in the tree. EAGLE (Li et al., 2024b) and Medusa (Cai et al., 2024) use the same static draft tree structure in all contexts: at the i-th step of the draft phase, k candidates are added, with k being fixed. This implicitly assumes the aforementioned hypothesis. However, this assumption appears to contradict the insight of speculative sampling that *some tokens are simpler and can be predicted by smaller models.* Our experiments (see Section 3.1) reveal that the acceptance rate of draft tokens is not only position-dependent but also highly context-dependent. Therefore, the static structure of draft trees has inherent limitations. Dynamically adjusting the draft tree structure based
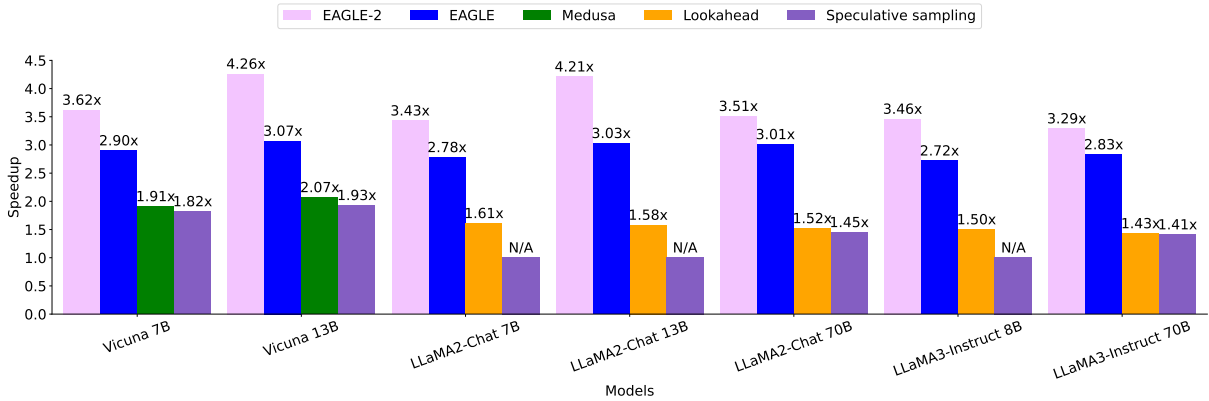
Figure 2: Speedup ratios of different methods at temperature=0. For speculative sampling, the Vicuna series uses Vicuna-68M as the draft model. LLaMA2-Chat 7B, 13B, and LLaMA3-Instruct 8B lack suitable draft models and are marked as N/A. LLaMA2-Chat 70B and LLaMA3-Instruct 70B use LLaMA2-Chat 7B and LLaMA3-Instruct 8B as draft models, respectively. In Table 1, we present comparisons with additional methods, but this figure only showcases a subset, including the fastest among these methods, EAGLE.

on the acceptance rates of draft tokens in different contexts can yield better results.

However, obtaining the acceptance rate of draft tokens requires the forward results from the original LLM, which conflicts with the goal of speculative sampling to reduce the number of forwards for the original LLM. Fortunately, we find that EAGLE is well-calibrated: the confidence score (probability) of the draft model is a good approximation of the acceptance rate of draft tokens (see Section 3.2). This makes it feasible to use a context-dependent dynamic draft tree structure.

We propose EAGLE-2, which leverages the confidence scores from the draft model to approximate acceptance rates. Based on this, it dynamically adjusts the draft tree structure, increasing the number of accepted tokens. We conducted comprehensive and extensive tests on six tasks: multi-turn conversation, code generation, mathematical reasoning, instruction following, summarization, and question answering. The datasets used were MT-bench (Zheng et al., 2023), HumanEval (Chen et al., 2021), GSM8K (Cobbe et al., 2021), Alpaca (Taori et al., 2023), CNN/Daily Mail (Nallapati et al., 2016), and Natural Questions (Kwiatkowski et al., 2019). Our comparisons included six advanced speculative sampling methods: standard speculative sampling (Leviathan et al., 2023; Chen et al., 2023a; Joao Gante, 2023), PLD (Saxena, 2023), Medusa (Cai et al., 2024), Lookahead (Fu et al., 2023), Hydra (Ankner et al., 2024), and EAGLE (Li et al., 2024b). We conducted experiments on three series of LLMs: Vicuna, LLaMA2-Chat, and LLaMA3-Instruct. In all experiments, EAGLE-2 demonstrated the best performance, achieving

a speedup of **2.5x-5x**. Figures 1 and 2 show the speedup ratios of EAGLE-2 and other speculative sampling methods on MT-bench. MT-bench is a multi-turn conversation dataset that closely resembles real-world scenarios for models like ChatGPT and is frequently used to evaluate state-of-the-art open-source and closed-source models. On the MT-bench dataset, EAGLE-2 is approximately **2x** faster than Medusa and about **2.3x** faster than Lookahead, while ensuring the output distribution remains unchanged.

Besides performance, EAGLE-2 offers the following advantages:

- **Out-of-the-box usability.** Comparing to EAGLE, EAGLE-2 does not require training any extra models. It does not train a separate model to predict the draft tree structure. Instead, it adjusts the draft tree structure based on the confidence scores from the draft model, which is essential for speculative sampling. Therefore, EAGLE-2 requires no additional training.

- **Reliability.** EAGLE-2 does not fine-tune or update the parameters of the original LLM, nor does it relax acceptance conditions. This ensures that the distribution of the generated text remains **exactly the same** with that of the original LLM, provably.

2 Preliminaries

2.1 Speculative Sampling

The core idea of speculative sampling (Leviathan et al., 2023; Chen et al., 2023a; Sun et al., 2024c,b)
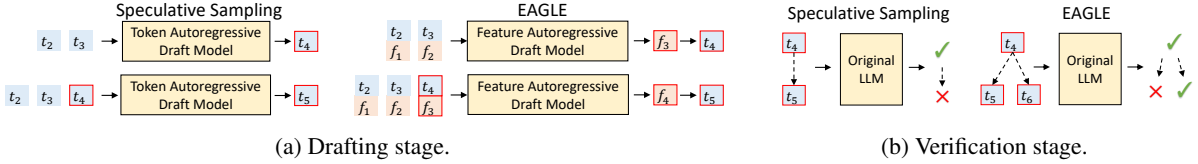
(a) Drafting stage.

(b) Verification stage.

Figure 3: Comparison of standard speculative sampling and EAGLE. For simplicity, EAGLE's tree-structured draft is shown only in the verification stage, while the illustration of the drafting stage uses a chain-structured draft. Here, t_i denotes the i-th token embedding, and f_i denotes the i-th feature vector in the second-to-top-layer of LLM before LM head.

is to first draft and then verify: quickly generate a potentially correct draft and then check which tokens in the draft can be accepted. We use t_i to denote the i-th token and $T_{a:b}$ to represent the token sequence $t_a, t_{a+1}, \cdots, t_b$. Speculative sampling alternates between drafting and verification stages.

Consider a prefix $T_{1:j}$, in the drafting stage, speculative sampling invokes a draft model (a smaller LLM than original LLM) to autoregressively generate a draft $\hat{T}_{j+1:j+k}$ with $T_{1:j}$ as the prefix, while also recording the probability $\hat{p}$ for each token. In the verification stage, speculative sampling calls the original LLM to check the draft $\hat{T}_{j+1:j+k}$ and record its probability p. Then, speculative sampling determines the acceptance of draft tokens sequentially from front to back. For token $\hat{t}_{j+i}$, the probability of it being accepted is $\min(1, p_{j+i}(\hat{t}_{j+i})/\hat{p}_{j+i}(\hat{t}_{j+i}))$. If the token is accepted, it proceeds to check the next one. Otherwise, it samples a token from the distribution $\text{norm}(\max(0, p_{j+i} - \hat{p}_{j+i}))$ to replace $\hat{t}_{j+i}$ and discards the remaining tokens in the draft. Appendix A.1 of (Leviathan et al., 2023) proves that speculative sampling is consistent with the distribution of vanilla autoregressive decoding. Both EAGLE and EAGLE-2 apply this framework.

2.2 EAGLE

EAGLE (Li et al., 2024b) is an improvement over speculative sampling. At the submission of this work, EAGLE ranks first in the Spec-Bench (Xia et al., 2024), a comprehensive benchmark designed for assessing speculative decoding methods across diverse scenarios.

Drafting Stage. Unlike standard speculative sampling, which autoregressively predicts token sequences, EAGLE performs autoregression at the more structured feature (before LM head) level and then uses the LM Head of original LLM to obtain the draft tokens. The sampling process introduces uncertainty in the feature sequence. To address this, EAGLE also inputs a token sequence advanced by one time step into the draft model, as shown in
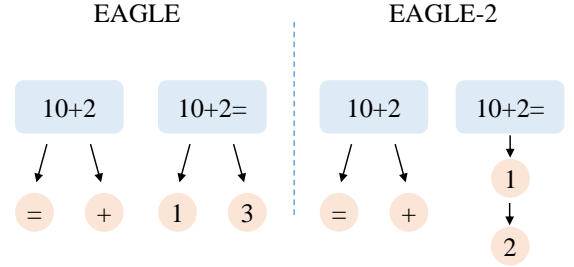

Figure 4: Differences between EAGLE and EAGLE-2. EAGLE always uses a fixed draft shape. When the query is "10+2=", the next token is very likely to be correctly predicted as "1". However, with a static draft tree, EAGLE would still add two candidates, even though the probability of the other candidate "3" being correct is very low. EAGLE-2, on the other hand, adjusts the shape of draft tree based on the context. When the query is "10+2", the next token is difficult to predict, so EAGLE-2 adds two candidates. For the simpler query "10+2=", EAGLE-2 adds only one candidate "1".

Figure 3a.

Verification Stage. In standard speculative sampling, the draft is chain-structured, requiring the discarding of all subsequent tokens if a draft token is rejected. EAGLE uses a tree-structured draft, allowing alternative branches to be attempted if a draft token is rejected. Figure 3b illustrates the differences between the two.

Differences between EAGLE and EAGLE-2. The shape of EAGLE's draft tree is fixed, with the drafting phase filling in the corresponding positions. EAGLE-2 aims to improve this by introducing a dynamically adjustable draft tree. Figure 4 illustrates the difference between EAGLE and EAGLE-2 with a simple example.

3 Observations

3.1 Context-Dependent Acceptance Rates

First, we evaluate the necessity of using a dynamic draft tree. This depends on whether the acceptance rates of draft tokens are solely related to their positions. We tested the acceptance rates of tokens at different positions in the draft tree on the Alpaca dataset and Vicuna 7B. The results are shown in

(a) Draft tree structure.

(b) Acceptance rates of tokens at different positions, with each point representing a query.
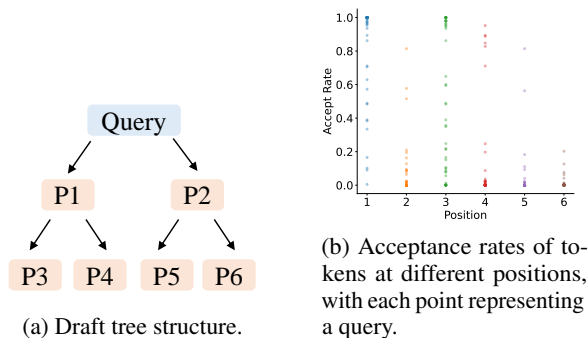
Figure 5: Acceptance rates of draft tokens at different positions. In the left figure, P1-P6 indicate positions in the token tree, corresponding to positions 1-6 on the horizontal axis in the right figure. The right figure shows the acceptance rates of draft tokens at positions P1-P6.

Figure 5. Overall, the acceptance rate of draft tokens is position-dependent, with the highest acceptance rate at position P1 and the lowest at position P6. Draft tokens in the upper left side of the draft tree (such as position P1) have higher acceptance rates, while those in the lower right side (such as position P6) have lower acceptance rates. This supports the rationale for having more nodes in the upper left and fewer in the lower right in static draft trees used by methods like EAGLE and Medusa. However, we also observed significant variance in acceptance rates at the same position, indicating that the probability of a draft token being accepted depends not only on its position but also on the context. This suggests that a context-aware dynamic draft tree has greater potential than a static draft tree.

3.2 Well-Calibrated Draft Model

To apply a dynamic draft tree, we need a low-cost method to estimate the acceptance rates of draft tokens without invoking the original LLM. We conducted experiments on the Alpaca dataset to explore the relationship between the draft model's confidence score (the output probability of LLM w.r.t. each token) and the acceptance rate. As shown in Figure 6, there is a strong positive correlation between the draft model's confidence score and the acceptance rate of the token. Draft tokens with confidence score below 0.05 have an acceptance rate of approximately 0.04, while those with confidence score above 0.95 have an acceptance rate of about 0.98. Therefore, we can use the draft model's confidence score to estimate acceptance rates without additional overhead, enabling dynamic adjustments to the draft tree. Similar phenomena are observed with draft models in other
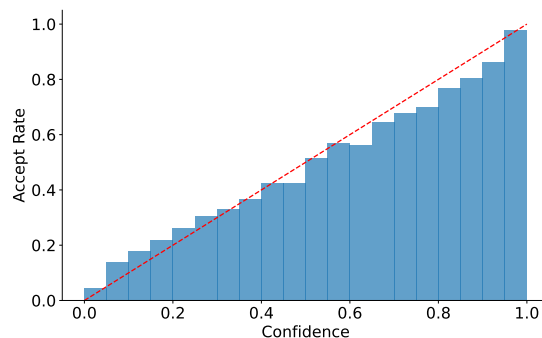

Figure 6: Average acceptance rates for different confidence score intervals of the draft model. The red dashed line connects (0,0) and (1,1) to aid in visual assessment. The original LLM is Vicuna 7B.

methods, such as GLIDE and CAPE (Du et al., 2024).

4 Context-Aware Dynamic Draft Tree

Building on the aforementioned observations, we introduce EAGLE-2, an acceleration algorithm for LLM inference that dynamically adjusts the draft tree. EAGLE-2 does not alter the training and inference of the draft model, nor does it affect the verification stage. Its improvements focus on two aspects: how to expand the draft tree (Section 4.1) and how to rerank draft tokens (Section 4.2). During the expansion phase, we input the most promising nodes from the latest layer of the draft tree into the draft model to form the next layer. During the reranking phase, we select the tokens with higher acceptance probabilities to form the input for the original LLM during the verification phase.

In the draft tree, a node represents a token. In the following text, we use "node" and "token" interchangeably.

4.1 Expansion Phase

Thanks to tree attention, the draft model can simultaneously input all tokens from the current layer and compute the probabilities for the next tokens in a single forward pass, thereby expanding all tokens in the current layer. However, inputting too many tokens at once can slow down the draft model's forward pass, and the number of tokens in each layer of the draft tree grows exponentially. Therefore, we need to selectively expand the draft tree.

We choose the top-k tokens with the highest global acceptance probabilities from the current layer for expansion. In speculative sampling, rejecting a draft token leads to discarding all subsequent tokens; a token is ultimately accepted only if all its

prefixes are accepted. The global acceptance rate of a token t_i is the product of the acceptance rates of all tokens on the path from the root node to t_i. We define it as the value V_i:

$$V_i = \prod_{t_j \in \text{Path}(\text{root}, t_i)} p_j \approx \prod_{t_j \in \text{Path}(\text{root}, t_i)} c_j,$$

where $\text{Path}(\text{root}, t_i)$ represents the path from the root node to the node t_i in the draft tree, p_j represents the acceptance rate of the node t_j, and c_j represents the confidence score of t_j from the draft model. Experiments in Section 3.2 show that confidence score is strongly positively correlated with acceptance rate. We leverage this relationship to approximate the value.

Branches starting from tokens with higher values are more likely to be accepted. Therefore, we select the top-k nodes with the highest values in the last layer as the input to the draft model and expand the draft tree based on the output. The top of Figure 7 illustrates the expansion phase.

4.2 Reranking Phase

The purpose of the expansion phase is to deepen the draft tree. Since acceptance rates range between 0 and 1, the value of a deeper token is lower. Some shallow nodes that were not expanded may have higher values than the deeper expanded nodes. Therefore, we do not use the tokens selected during the expansion phase as the draft directly. Instead, we rerank all draft tokens and select the top m tokens with the highest values. The value of a node is always less than or equal to that of its parent node. For nodes with the same value, we prioritize selecting shallower nodes. This ensures that the top m tokens selected after reranking still form a connected tree.

Afterwards, we flatten the selected tokens into a one-dimensional sequence to serve as the input for the verification phase. To ensure consistency with vanilla autoregressive decoding, we also need to adjust the attention mask. In vanilla autoregressive decoding, each token can see all preceding tokens, resulting in a lower triangular attention matrix. When using a draft tree, tokens from different branches should not be visible to each other. Therefore, the attention mask must be adjusted according to the tree structure to ensure that each token can only see its ancestor nodes. The bottom of Figure 7 illustrates the reranking Phase.

5 Experiments
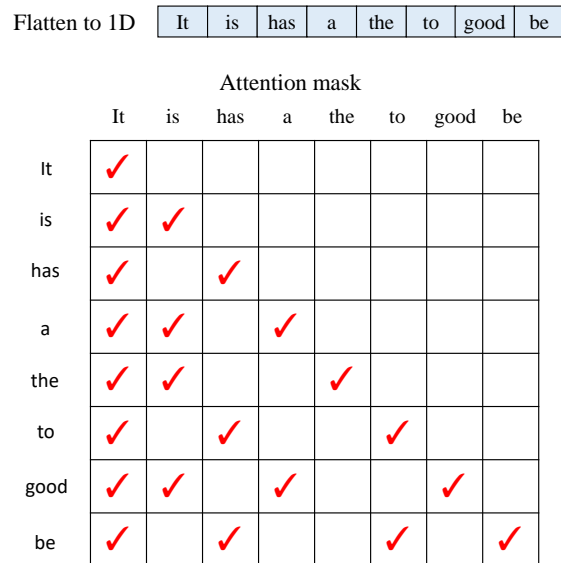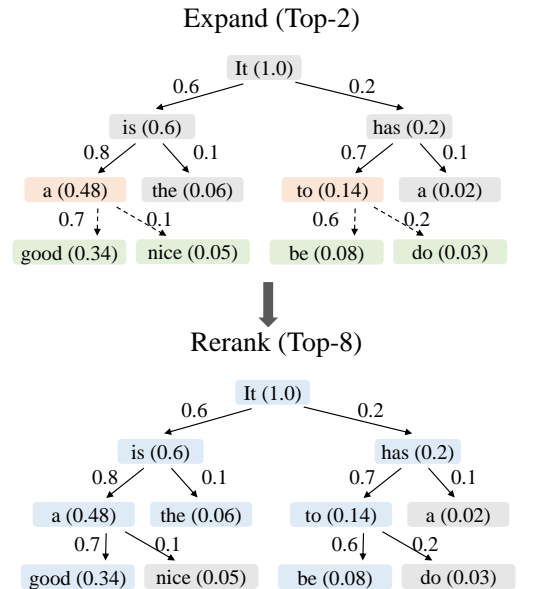
Figure 7: Illustration of EAGLE-2. The numbers beside the edges represent the confidence scores of the draft model, and the numbers in brackets within the blocks represent the value of the nodes. During the expansion phase, we select the top 2 nodes with the highest value from the current layer (orange blocks) as inputs to the draft model and connect the generated tokens (green blocks) to the draft tree. In the rerank phase, we select the top 8 nodes with the highest value from all nodes (blue blocks), flatten them into a 1-dimensional sequence to form the final draft. We then construct the attention mask according to the tree structure, ensuring each token can only see its ancestor nodes.

7425

Models. We conduct experiments on Vicuna 7B, 13B (Chiang et al., 2023), LLaMA2-Chat 7B, 13B, 70B (Touvron et al., 2023), and LLaMA3-Instruct 8B, 70B models (Meta, 2024).

Tasks. We conduct comprehensive evaluations on six generation tasks. For multi-turn conversation, code generation, mathematical reasoning, instruction following, summarization, and question answering tasks, we chose the MT-bench (Zheng et al., 2023), HumanEval (Chen et al., 2021), GSM8K (Cobbe et al., 2021), Alpaca (Taori et al., 2023), CNN/Daily Mail (Nallapati et al., 2016), and Natural Questions (Kwiatkowski et al., 2019) datasets, respectively.

Metrics. EAGLE-2 neither fine-tunes the original LLM nor relaxes acceptance conditions, making it a **lossless** acceleration method. Therefore, we do not evaluate the generation quality and instead use the following metrics to assess acceleration performance:

- **Speedup Ratio:** The actual test speedup ratio relative to vanilla autoregressive decoding.

- **Average Acceptance Length τ:** The average number of tokens generated per drafting-verification cycle, which corresponds to the number of tokens accepted from the draft. The advantage of average acceptance length is that it is independent of hardware and runtime environment, while its disadvantage is that it does not reflect the overhead of the draft model.

Why is acceptance rate not included? The acceptance rate only reflects the performance of the draft model. Since EAGLE-2 does not modify the structure of the draft model, the acceptance rate remains the same as that of EAGLE.

Comparison. We use vanilla autoregressive decoding as the baseline, which serves as the benchmark for speedup ratios (1.00x). We compare EAGLE-2 with recent lossless speculative sampling methods, including standard speculative sampling (Leviathan et al., 2023; Chen et al., 2023a; Joao Gante, 2023), PLD (Saxena, 2023), Medusa (Cai et al., 2024), Lookahead (Fu et al., 2023), Hydra (Ankner et al., 2024), and EAGLE (Li et al., 2024b). The speedup ratio is hardware-dependent, so we tested different methods on the same devices to ensure fairness. Our comparative experiments utilized Spec-Bench (Xia et al., 2024). The implementation details of these methods and EAGLE

can be found in Appendix A.

5.1 Effectiveness

Figures 1 and 2, along with Tables 1 and 2, present the speedup ratios of different methods. Across all datasets and LLMs we tested, EAGLE-2 achieved the highest speedup ratios. Most speculative sampling methods exhibit the highest speedup on the code generation task (HumanEval), benefiting from the extensive use of fixed templates in code. EAGLE achieved a speedup of up to 5x on code generation tasks. PLD achieved the highest speedup ratio on summarization tasks (CNN/DM) when using Vicuna as the original LLM, due to PLD's retrieval-based draft generation and the high overlap in context when Vicuna performs summarization. Standard speculative sampling, using Vicuna-68M as the draft model, also achieved significant speedups but had much higher training overhead compared to other methods. PLD and Lookahead do not require training, while Medusa, Hydra, EAGLE, and EAGLE-2 use SFT datasets for training their draft models. Vicuna-68M used both pre-training and SFT datasets, with the pre-training dataset being much larger than the SFT dataset.

Tables 1 and 2 show the average acceptance lengths for different methods, which is a hardware-independent metric. Across all datasets and LLMs we tested, EAGLE-2 achieved the longest average acceptance length. Each drafting-verification cycle of EAGLE-2 generates approximately 4-5.5 tokens, significantly higher than other methods, roughly twice that of standard speculative sampling and Medusa. PLD and Lookahead have shorter average acceptance lengths, but since they either lack a draft model or their draft model is not a neural network, the overhead during the drafting phase is very low, resulting in a speedup ratio very close to their average acceptance length.

Medusa, Hydra, EAGLE, and EAGLE-2 have lower average acceptance lengths on QA (Natural Questions) and summarization (CNN/DM) tasks compared to other tasks, whereas standard speculative sampling does not show this reduction. The same pattern is observed for the speedup ratios. This discrepancy may be attributed to differences in the training data for the draft models. The draft model for standard speculative sampling uses both pretraining and SFT datasets, while Medusa, Hydra, EAGLE, and EAGLE-2 only use the SFT dataset. Natural Questions involves questions about world knowledge, such as "Where was the 2015

Table 1: Speedup ratios and average acceptance lengths τ of different methods. V represents Vicuna, L2 represents LLaMA2-Chat. SpS denotes standard speculative sampling, with its draft model being Vicuna-68M. Methods like Medusa relax acceptance conditions under non-greedy settings, which do not guarantee lossless acceleration. Therefore, we do not compare EAGLE-2 with these methods.

Model	Method	MT-bench		HumanEval		GSM8K		Alpaca		CNN/DM		Natural Ques.		Mean	
		Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ
								Temperature=0							
V 13B	SpS	1.93x	2.27	2.23x	2.57	1.77x	2.01	1.76x	2.03	1.93x	2.33	1.66x	1.88	1.88x	2.18
	PLD	1.58x	1.63	1.85x	1.93	1.68x	1.73	1.16x	1.19	2.42x	2.50	1.14x	1.17	1.64x	1.69
	Medusa	2.07x	2.59	2.50x	2.78	2.23x	2.64	2.08x	2.45	1.71x	2.09	1.81x	2.10	2.07x	2.44
	Lookahead	1.65x	1.69	1.71x	1.75	1.81x	1.90	1.46x	1.51	1.46x	1.50	1.36x	1.39	1.58x	1.62
	Hydra	2.88x	3.65	3.28x	3.87	2.93x	3.66	2.86x	3.53	2.05x	2.81	2.11x	2.88	2.69x	3.40
	EAGLE	3.07x	3.98	3.58x	4.39	3.08x	3.97	3.03x	3.95	2.49x	3.52	2.42x	3.11	2.95x	3.82
	EAGLE-2	**4.26x**	**4.83**	**4.96x**	**5.41**	**4.22x**	**4.79**	**4.25x**	**4.89**	**3.40x**	**4.21**	**3.13x**	**3.74**	**4.04x**	**4.65**
L2 13B	PLD	1.42x	1.46	1.63x	1.70	1.41x	1.44	1.16x	1.20	1.42x	1.45	1.12x	1.15	1.36x	1.40
	Lookahead	1.58x	1.64	1.80x	1.85	1.65x	1.69	1.47x	1.50	1.46x	1.53	1.42x	1.45	1.56x	1.61
	EAGLE	3.03x	3.90	3.76x	4.52	3.20x	4.03	3.01x	3.83	2.70x	3.59	2.83x	3.47	3.09x	3.89
	EAGLE-2	**4.21x**	**4.75**	**5.00x**	**5.52**	**4.31x**	**4.90**	**4.13x**	**4.61**	**3.45x**	**4.24**	**3.51x**	**4.04**	**4.10x**	**4.68**
V 7B	SpS	1.82x	2.36	1.99x	2.61	1.71x	2.26	1.65x	2.21	1.81x	2.44	1.60x	2.16	1.76x	2.34
	PLD	1.61x	1.68	1.82x	1.87	1.82x	1.99	1.21x	1.31	2.53x	2.72	1.23x	1.44	1.70x	1.84
	Medusa	1.91x	2.52	2.02x	2.67	1.89x	2.59	1.79x	2.48	1.42x	2.02	1.51x	2.09	1.76x	2.40
	Lookahead	1.63x	1.69	1.72x	1.77	1.84x	1.99	1.38x	1.57	1.44x	1.53	1.45x	1.60	1.58x	1.69
	Hydra	2.69x	3.60	2.98x	3.79	2.73x	3.66	2.66x	3.58	2.01x	2.70	2.25x	2.86	2.55x	3.37
	EAGLE	2.90x	3.94	3.33x	4.29	3.01x	4.00	2.79x	3.89	2.33x	3.42	2.31x	3.21	2.78x	3.79
	EAGLE-2	**3.62x**	**4.98**	**3.95x**	**5.33**	**3.63x**	**4.97**	**3.46x**	**4.86**	**2.94x**	**4.12**	**2.76x**	**3.82**	**3.39x**	**4.68**
L2 7B	PLD	1.38x	1.43	1.52x	1.59	1.32x	1.37	1.15x	1.19	1.48x	1.52	1.15x	1.20	1.33x	1.38
	Lookahead	1.61x	1.66	1.72x	1.77	1.58x	1.65	1.49x	1.52	1.49x	1.54	1.48x	1.53	1.56x	1.61
	EAGLE	2.78x	3.62	3.17x	4.24	2.91x	3.82	2.78x	3.71	2.43x	3.41	2.61x	3.44	2.78x	3.71
	EAGLE-2	**3.43x**	**4.70**	**4.03x**	**5.39**	**3.52x**	**4.77**	**3.45x**	**4.66**	**3.01x**	**4.12**	**3.15x**	**4.19**	**3.43x**	**4.64**
								Temperature=1							
V 13B	SpS	1.62x	1.84	1.72x	1.97	1.46x	1.73	1.52x	1.78	1.66x	1.89	1.43x	1.70	1.55x	1.82
	EAGLE	2.32x	3.20	2.65x	3.63	2.57x	3.60	2.45x	3.57	2.23x	3.26	2.14x	3.06	2.39x	3.39
	EAGLE-2	**3.80x**	**4.40**	**4.22x**	**4.89**	**3.77x**	**4.41**	**3.78x**	**4.37**	**3.25x**	**3.97**	**3.07x**	**3.54**	**3.65x**	**4.26**
L2 13B	EAGLE	2.68x	3.45	2.89x	3.78	2.82x	3.67	2.66x	3.55	2.41x	3.39	2.37x	3.31	2.64x	3.53
	EAGLE-2	**3.92x**	**4.51**	**4.58x**	**5.29**	**4.21x**	**4.80**	**3.85x**	**4.48**	**3.31x**	**4.08**	**3.43x**	**3.89**	**3.88x**	**4.51**
V 7B	SpS	1.50x	1.87	1.55x	1.95	1.53x	1.82	1.56x	1.85	1.63x	1.91	1.33x	1.72	1.52x	1.85
	EAGLE	2.13x	3.17	2.39x	3.43	2.34x	3.29	2.21x	3.30	2.08x	3.12	1.95x	2.86	2.18x	3.20
	EAGLE-2	**3.05x**	**4.28**	**3.33x**	**4.65**	**3.07x**	**4.49**	**3.08x**	**4.43**	**2.63x**	**3.76**	**2.48x**	**3.56**	**2.94x**	**4.20**
L2 7B	EAGLE	2.22x	3.30	2.61x	3.79	2.40x	3.52	2.29x	3.33	2.19x	3.15	2.22x	3.12	2.32x	3.37
	EAGLE-2	**3.19x**	**4.41**	**3.67x**	**5.06**	**3.35x**	**4.62**	**3.20x**	**4.48**	**2.73x**	**3.85**	**2.81x**	**4.01**	**3.15x**	**4.41**

rugby union world cup held?", and world knowledge is primarily acquired through pretraining rather than SFT. Summarization tasks are also less represented in the SFT dataset. This suggests the potential benefits of expanding the draft model's training data. Despite this, EAGLE-2 still outperforms standard speculative sampling on these two datasets.

5.2 Ablation Study

In this section, we conduct the ablation study.

5.2.1 Value and Confidence Score

EAGLE's draft model provides a good approximation of acceptance rates, but it is local and cannot reflect the actual probability of a draft token being accepted. Therefore, when selecting nodes for expansion, we use the value, which is the product of

a draft token's confidence score and its ancestor nodes' confidence scores, as the basis for ranking. In this section, we compare the performance impact of expanding based on value versus confidence score. The experimental results in Table 3 show that the speedup ratio and average acceptance length are both higher when expanding based on value, demonstrating the rationale behind the EAGLE-2 approach.

5.2.2 Reranking

The purpose of EAGLE-2's expansion phase is to deepen the draft tree, but the tokens selected may be globally less optimal than shallow nodes that were not selected. Therefore, during the reranking phase, we rerank all the draft tokens. We conducted an ablation study on this operation using the MT-bench and GSM8K dataset. As shown in Table 3,

Table 2: Speedup ratios and average acceptance lengths τ with LLaMA2-Chat 70B, LLaMA3-Instruct 70B, and LLaMA3-Instruct 8B as the original LLMs, with the temperature set to 0, on the MT-bench dataset.

Model	Method	Speedup	τ
LLaMA2-Chat 70B	PLD	1.31x	1.39
	Lookahead	1.52x	1.64
	EAGLE	3.01x	3.81
	EAGLE-2	**3.51x**	**4.48**
LLaMA3-Instruct 70B	EAGLE	2.83x	3.62
	EAGLE-2	**3.29x**	**4.16**
LLaMA3-Instruct 8B	EAGLE	2.72x	3.65
	EAGLE-2	**3.46x**	**4.53**

reranking improved both the average acceptance length and the speedup ratio.

Table 3: Ablation experiment results with temperature set to 0 on Vicuna 7B. "w/o value" indicates not using value and directly using confidence, "w/o reranking" indicates not performing reranking, and "w/o both" indicates neither value nor reranking is used.

	MT-bench		GSM8K	
Method	Speedup	τ	Speedup	τ
w/o both	2.81x	3.92	2.85x	3.93
w/o value	3.21x	4.39	2.93x	3.96
w/o reranking	3.48x	4.86	3.50x	4.85
EAGLE-2	**3.62x**	**4.98**	**3.63x**	**4.97**

6 Related Work

With widespread applications of LLMs, there has been significant work (Liu et al., 2023b) focused on accelerating LLM inference, such as low-bit quantization (Hubara et al., 2018; Shen et al., 2020; Kim et al., 2021; Zadeh et al., 2020; Zafrir et al., 2019), pruning (Gale et al., 2019; Sanh et al., 2020), and knowledge distillation (Hinton et al., 2015). These methods reduce generation latency by decreasing the computational cost of each forward pass of the LLM. However, these approaches often degrade LLM performance to some extent, resulting in a trade-off between generation quality and computational overhead.

Speculative sampling methods achieve lossless acceleration by using the original LLM for verification. Early speculative decoding methods (Stern et al., 2018; Sun et al., 2021) accelerated generation in greedy settings, while Leviathan et al. (2023); Chen et al. (2023a) proposed speculative sampling

to extend the draft-verification framework to non-greedy generation. Subsequent work has largely focused on reducing draft overhead and enhancing consistency between the draft and the original LLM. SpecInfer (Miao et al., 2023) integrates multiple small models as the draft model, aggregating their drafts into a tree and using tree attention for parallel verification. Medusa (Cai et al., 2024) trains a set of MLPs to parallelly predict multiple tokens using the original LLM's features, significantly reducing the latency during the drafting phase. EAGLE (Li et al., 2024b) autoregressively predicts feature sequences instead of token sequences and inputs the sampling results into the draft model to address uncertainty at the feature level, substantially improving the draft model's accuracy. This principle of eliminating uncertainty is also used in Hydra (Ankner et al., 2024) and Recurrent Drafter (Zhang et al., 2024). Parallel Decoding (Santilli et al., 2023), Lookahead (Fu et al., 2023), Ouroboros (Zhao et al., 2024), and CLLMs (Kou et al., 2024) generate drafts using Jacobi iterations. Methods (Hooper et al., 2023; Yang et al., 2023b; Monea et al., 2023; Li et al., 2024a; Yi et al., 2024; Liu et al., 2024; Sun et al., 2024a; Elhoushi et al., 2024; Svirschevski et al., 2024) like Draft & Verify (Zhang et al., 2023) utilize techniques such as layer skipping or early exit, using parts of the original LLM's parameters as the draft model. REST (Fu et al., 2024) and LLMA (Yang et al., 2023a) generate drafts through retrieval. Online Speculative Decoding (Liu et al., 2023a) and DistillSpec (Zhou et al., 2024) further align the draft model with the original LLM through additional training. Cascade Speculative Drafting (Chen et al., 2023b) and Staged Speculative Decoding (Spector and Re, 2023) cascade draft models of different sizes.

Speculative sampling methods can achieve lossless acceleration, but they can also trade off quality for higher speedup ratios. For example, BiLD (Kim et al., 2024) relaxes the acceptance conditions, while Medusa-2 (Cai et al., 2024), CLLMs (Kou et al., 2024), and SPACE (Yi et al., 2024) fine-tune the original LLMs.

Sorted Llama (Kavehzadeh et al., 2024) and LITE (Varshney et al., 2023) use confidence as an indicator of token quality. Some works have already employed partially dynamic draft trees by leveraging confidence. BiLD (Kim et al., 2024) and Kangaroo (Liu et al., 2024) use early stopping based on the draft model's confidence to control the tree's depth. GLIDE and CAPE (Du et al., 2024)

adds additional candidates when the top-1 token confidence is low, controlling the tree's depth, but the additional candidates are not further expanded, resulting in a structurally limited tree. In contrast, EAGLE-2 has no such limitations and can dynamically adjust the draft tree structure flexibly, leading to better performance.

7 Conclusion

In this paper, we introduce EAGLE-2, an efficient and **lossless** speculative sampling method. We found that EAGLE's draft model confidence is a good approximation of the acceptance rate for draft tokens. Based on this, EAGLE-2 employs a context-dependent draft tree structure, significantly increasing the number of accepted draft tokens and resulting in better speedup ratios. EAGLE-2 ensures that the generated results are consistent with the original LLMs and does not require additional training. We conducted extensive evaluations using various LLMs across multiple datasets and compared EAGLE-2 with several state-of-the-art speculative sampling methods. In all our experiments, EAGLE-2 achieved the highest speedup ratios.

Limitations

The limitations of EAGLE-2 are similar to those of EAGLE, as it requires training draft models. For training the draft model, we used the SFT dataset ShareGPT. Training the draft model also requires certain computational resources; training a draft model for a 70B original LLM requires 4 A100 (40G) GPUs for one to two days. However, in large-scale deployment and application, such training costs are negligible. Additionally, EAGLE-2 uses the confidence scores from the draft model to approximate acceptance rates, which requires the draft model to be well-calibrated.

Acknowledgements

References

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

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. 2024. Medusa: Simple llm inference acceleration framework with multiple decoding heads. *arXiv preprint arXiv: 2401.10774*.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023a. 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*.

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

Ziyi Chen, Xiaocong Yang, Jiacheng Lin, Chenkai Sun, Jie Huang, and Kevin Chen-Chuan Chang. 2023b. Cascade speculative drafting for even faster llm inference. *arXiv preprint arXiv:2312.11462*.

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

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

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

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, et al. 2024. Layer skip: Enabling early exit inference and self-speculative decoding. *arXiv preprint arXiv:2404.16710*.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. 2023. Breaking the sequential dependency of llm inference using lookahead decoding.

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*.

Trevor Gale, Erich Elsen, and Sara Hooker. 2019. The state of sparsity in deep neural networks.(2019). *arXiv preprint cs.LG/1902.09574*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

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

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2018. Quantized neural networks: Training neural networks with low precision weights and activations. *journal of machine learning research*, 18(187):1–30.

Joao Gante. 2023. Assisted generation: a new direction toward low-latency text generation.

Parsa Kavehzadeh, Mojtaba Valipour, Marzieh Tahaei, Ali Ghodsi, Boxing Chen, and Mehdi Rezagholizadeh. 2024. Sorted llama: Unlocking the potential of intermediate layers of large language models for dynamic inference. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 2129–2145.

Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. 2021. I-bert: Integer-only bert quantization. In *International conference on machine learning*, pages 5506–5518. PMLR.

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.

Siqi Kou, Lanxiang Hu, Zhezhi He, Zhijie Deng, and Hao Zhang. 2024. Cllms: Consistency large language models. *arXiv preprint arXiv:2403.00835*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

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.

Minghan Li, Xilun Chen, Ari Holtzman, Beidi Chen, Jimmy Lin, Wen-tau Yih, and Xi Victoria Lin. 2024a. Nearest neighbor speculative decoding for llm generation and attribution. *arXiv preprint arXiv:2405.19325*.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024b. Eagle: Speculative sampling requires rethinking feature uncertainty. In *International Conference on Machine Learning*.

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

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

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, et al. 2023b. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR.

Meta. 2024. LLaMA3. https://github.com/pytorch-labs/gpt-fast/.

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

Giovanni Monea, Armand Joulin, and Edouard Grave. 2023. Pass: Parallel speculative sampling. *arXiv preprint arXiv:2311.13581*.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

R OpenAI. 2023. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2(5).

Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33:20378–20389.

Andrea Santilli, Silvio Severino, Emilian Postolache, Valentino Maiorca, Michele Mancusi, Riccardo Marin, and Emanuele Rodolà. 2023. Accelerating transformer inference for translation via parallel decoding. *arXiv preprint arXiv:2305.10427*.

Apoorv Saxena. 2023. Prompt lookup decoding.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low

precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8815–8821.

Benjamin Spector and Chris Re. 2023. Accelerating llm inference with staged speculative decoding. *arXiv preprint arXiv:2308.04623*.

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

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

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. *arXiv preprint arXiv:2106.04970*.

Ziteng Sun, Jae Hun Ro, Ahmad Beirami, and Ananda Theertha Suresh. 2024b. Optimal block-level draft verification for accelerating speculative decoding. *arXiv preprint arXiv:2403.10444*.

Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. 2024c. Spectr: Fast speculative decoding via optimal transport. *Advances in Neural Information Processing Systems*, 36.

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

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

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

Neeraj Varshney, Agneet Chatterjee, Mihir Parmar, and Chitta Baral. 2023. Accelerating llm inference by enabling intermediate layer decoding. *arXiv preprint arXiv:2310.18581*.

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. *Preprint*, arXiv:2401.07851.

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

Seongjun Yang, Gibbeum Lee, Jaewoong Cho, Dimitris Papailiopoulos, and Kangwook Lee. 2023b. Predictive pipelined decoding: A compute-latency trade-off for exact llm decoding. *arXiv preprint arXiv:2307.05908*.

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

Ali Hadi Zadeh, Isak Edo, Omar Mohamed Awad, and Andreas Moshovos. 2020. Gobo: Quantizing attention-based nlp models for low latency and energy efficient inference. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 811–824. IEEE.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pages 36–39. IEEE.

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

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

Weilin Zhao, Yuxiang Huang, Xu Han, Chaojun Xiao, Zhiyuan Liu, and Maosong Sun. 2024. Ouroboros: Speculative decoding with large model enhanced drafting. *arXiv preprint arXiv:2402.13720*.

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. *arXiv preprint arXiv:2306.05685*.

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

A Implementation Details

Vanilla: We use models from the Huggingface.transformers library with the PyTorch backend and pre-allocated KV cache. Other methods also use these models as their base.

(Standard) Speculative Sampling: We use the assisted generation feature from the HuggingFace Transformers library.

PLD, Lookahead, Medusa, and Hydra: We use the default settings and the officially released weights.

EAGLE: Vicuna and LLaMA2-Chat draft models use the officially released weights, while LLaMA3-Instruct is trained using the ShareGPT dataset (consistent with Medusa and Hydra).

EAGLE-2: For the 7B (8B), 13B, and 70B original LLMs, we set the total number of draft tokens to 60, 50, and 48, respectively, with a draft tree depth of 6, and select 10 nodes during the expansion phase.