CASCADUSA: CASCADED SPECULATIVE DECODING FOR EFFICIENT LARGE LANGUAGE MODEL SERVING

Yunzhong Xiao [*1] **Yifei Wang** [*1] **Xi Mao** [*1]

ABSTRACT

This paper introduces "Cascaded Speculative Decoding," an innovative architecture designed to enhance the efficiency of serving Large Language Models (LLMs). Our proposed method extends the existing MEDUSA model made by (Cai et al., 2024) by introducing a cascading connection scheme among the de-coding heads, which incorporates causal relationships between successive tokens. By integrating multiple speculative decoding mechanisms into a cascaded framework, this approach accelerates the inference process without compromising the generative quality of the model. Comparative experiments demonstrate that Cascaded Speculative Decoding not only achieves faster inference speeds but also maintains high accuracy in token prediction even in a heavily constrained training environment where both the GPU computational power and the disk storage space are limited. The result underscores our models' potential to significantly improve LLM applicability in various computational environments. Our findings suggest that this method could be an advancement in the deployment and operational efficiency of LLMs across diverse domains.

1 INTRODUCTION

Large Language Models (LLMs) have experienced significant growth in recent years, showcasing impressive capabilities in generating human-like language texts across various domains, such as question answering (Brown et al., 2020). the vast number of parameters in LLMs leads to a substantial demand for computational resources, posing challenges in serving LLMs economically. The sequential nature of autoregressive LLMs requires each token to be predicted one at a time, which significantly limits the throughput and scalability of LLM deployments. Speculative decoding has been introduced as a promising solution to enhance inference efficiency. Instead of decoding tokens sequentially, speculative decoding utilizes a smaller model to predict LLM outputs, which are then validated by the LLM. This approach enables the LLM to decode multiple tokens simultaneously, thereby accelerating the inference speed (Leviathan et al., 2023). This technique reduces the number of sequential operations, thereby accelerating the inference process and reducing latency.

Recent advancements in speculative decoding techniques have shown potential in various applications of LLMs, including natural language processing, code generation, and automated content creation. For instance, the MEDUSA framework introduces additional decoding heads that enable parallel token prediction, significantly reducing inference time while maintaining high accuracy and minimal computational overhead (Cai et al., 2024). Similarly, the EAGLE approach rethinks feature uncertainty in speculative decoding, proposing a method that surpasses traditional decoding efficiencies by integrating model certainty into the decoding process (Li et al., 2024).

Further research by (Xia et al., 2024) provides a comprehensive survey of speculative decoding techniques, highlighting the pivotal role of fine-tuned feedforward neural networks (FFNs) on top of Transformer decoders to enhance LLM inference. These studies collectively underscore the importance of speculative decoding in addressing the latency issues inherent in sequential decoding processes. Moreover, the introduction of lookahead decoding strategies as explored by (Fu et al., 2024) presents an innovative approach where the decoder anticipates future tokens without the need for auxiliary models, thus streamlining the decoding process and enhancing efficiency.

Building upon these foundational works, our research introduces a novel approach termed "Cascaded Speculative Decoding," which layers multiple speculative decoding mechanisms to further boost the inference speed without compromising the model's generative quality. This method leverages the inherent parallelism in speculative decoding by sequentially layering predictions and validations, allowing

[*]Equal contribution [1]Information Networking Institute, Carnegie Mellon University. Correspondence to: Yunzhong Xiao <shawnxiao@cmu.edu>, Yifei Wang <yifeiw3@cmu.edu>, Xi Mao <xim@cmu.edu>.

Proceedings of the 4^{th} MLSys Conference, San Jose, CA, USA, 2021.

for a more granular control of prediction accuracy and computational expense. The cascaded approach aims to refine the balance between speed and accuracy, addressing the trade-offs identified in earlier implementations of speculative decoding. Through comparative studies, we aim to explore the power of Cascaded Speculative Decoding in real-world LLM applications. Furthermore, we will propose new variants of Cascaded Speculative Decoding with better accuracy without compromising training or inference efficiency.

2 Problem and Deliverables

Despite the existence of various strategies for speculative decoding (Xia et al., 2024), designing an accurate and efficient decoding strategy remains an open question. This ongoing challenge provides the impetus for our investigation into alternative and potentially more effective decoding frameworks. Our study is primarily focused on the MEDUSA model proposed by (Cai et al., 2024), examining its potential and limitations in real-world scenarios, and explore several modifications to the original MEDUSA model to see any improvement in accuracy or efficiency.

Our research is structured around several key deliverables:

- **Comprehensive Review and Evaluation**: We aim to synthesize and critically assess the existing MEDUSA framework. This review will cover its implementations and performance metrics, serving as a foundational element for identifying potential areas of improvement.
- **Development of Enhanced MEDUSA Variants**: Inspired by the original MEDUSA framework, we propose to develop new variants that could offer improved decoding accuracy and efficiency. These variants will be designed to reduce the trade-offs between speed and accuracy typically observed in speculative decoding.
- **Fine-tuning of Proposed Variants**: We will conduct extensive training and fine-tuning of our models use real-world datasets within a consistent and powerful computational environment. We will do the training with a Bridges-2 machine equipped with a NVIDIA V100 GPU located in the Pittsburgh Supercomputing Center
- **Performance Metrics Evaluation**: A key component of our research will be the rigorous evaluation of performance metrics such as decoding latency and the average number of tokens that cascaded small models successfully predict. We will use a set of baseline LLMs to benchmark against our cascading approach and collect these metrics, which will provide quantitative evidence of the improvements our modifications bring to the speculative decoding process.

The outcomes of this research will not only contribute to the academic community by providing insights into speculative decoding dynamics, but also offer practical value by enhancing the operational efficiency of LLMs in various applications.

3 Related Work

Speculative decoding has emerged as a promising paradigm to accelerate large language model (LLM) inference in recent years. The motivation behind this is the nature of LLM auto-regressive inference, which is bottle-necked by the memory instead of computational operations (Xia et al., 2024). This process leads to the under-utilization of the computation accelerators.

3.1 Overview

To address this issue, Stern et al. introduced Blockwise Decoding (Stern et al., 2018) to leverage idle computational resources to enhance parallelism. The Draft-then-Verify paradigm was firstly proposed. In 2022, SpecDec (Xia et al., 2023) was invented and achieved around 5x speedup over auto-regressive decoding by utilizing an independent drafter.

An intuitive way for further optimizing a single drafter would be using cascaded drafters. Cascaded Speculative Drafting (Chen et al., 2024), which incorporates vertical cascade and horizontal cascade was invented.

- **Vertical Cascade** recursively uses smaller models to draft for larger models, eliminating the need for auto-regressive generation in neural models
- **Horizontal Cascade** allocates smaller, faster draft models to generate tokens that are less likely to be accepted later in the drafting process

Experiments demonstrate that CS Drafting achieves up to an 81% additional speedup over standard speculative decoding while maintaining the same output distribution as the target LLM.

3.2 Limitations

- **Model selection** Most of the speculative decoding techniques relies on the availability of suitable draft models of varying sizes, finding the compatible set of models usually requires human selection and time consuming
- **Memory Usage** Although drafting reduced inference latency, but the use of draft models in the cascaded fashion will increase the overall memory footprint. In resource-constrained environment such as on-device LLM serving, deploying the system could be challenging.

3.3 Prior works

Some more recent works are trying to address those limitations. We found those works below particularly valuable for our tasks.

MEDUSA incorporated additional decoding heads that predict multiple subsequent tokens simultaneously (Cai et al., 2024). By employing a tree-based attention mechanism, MEDUSA manages to verify multiple candidate continuations in parallel during each decoding step, thereby reducing the total number of decoding steps and minimizing latency. This setup enhances computational efficiency by leveraging parallel processing with minimal impact on single-step latency. MEDUSA offers two fine-tuning protocols: MEDUSA-1, which fine-tunes on top of a frozen backbone LLM, and MEDUSA-2, which involves joint training with the backbone LLM for improved prediction accuracy and greater speedup. The method proves effective across various model sizes, demonstrating significant speedups without sacrificing generation quality, and can be seamlessly integrated into existing LLM systems.

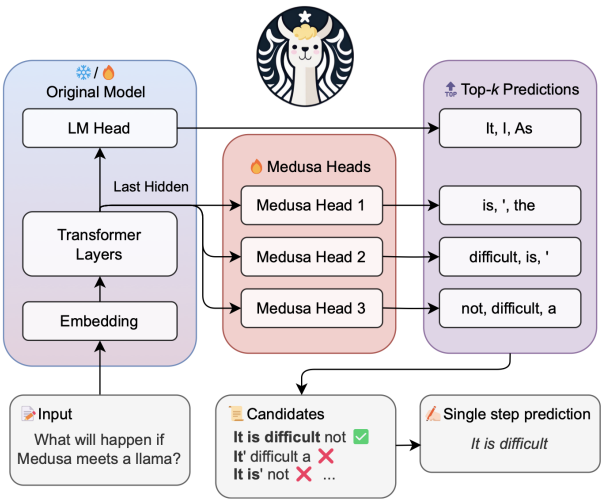

Figure 1. Model Architecture of Medusa (Cai et al., 2024)

Speculative Streaming accelerates large language model (LLM) inference without the need for auxiliary draft models (Bhendawade et al., 2024). By integrating multi-stream attention into the target model, the method facilitates simultaneous speculation and verification of future n-grams, significantly simplifying the system while reducing computational overhead. Speculative Streaming merges verification and speculation tasks within a single streamlined process, enhancing decoding speeds by 1.8 to 3.1 times across various tasks, without compromising generation quality. It achieves these improvements with drastically fewer parameters compared to previous methods, making it particularly advantageous for deployment on resource-constrained devices.

Recursive Speculative Decoding (RSD) is a tree-based speculative decoding method for accelerating inference in large language models (Jeon et al., 2024). RSD enhances diversity by constructing draft-token trees through sampling without replacement, utilizing two distinct approaches: RSD with Constant branching factors (RSD-C) and RSD with Stochastic Beam Search (RSD-S). The method employs recursive rejection sampling to verify and refine the draft tokens generated by a smaller, less computationally demanding model, aligning them with the target LLM's probability distributions. Experimental results demonstrate that RSD consistently outperforms existing speculative decoding methods, particularly in scenarios with fixed computational budgets, which is critical for resource-bounded devices.

4 Proposed Model Architecture

4.1 Medusa Architecture Review

In this study, we extend the Medusa architecture, which incorporates multiple decoding heads capable of predicting tokens at different positions in a sequence (Cai et al., 2024). These decoding heads are integrated into the training process alongside the backbone Large Language Model (LLM). The hidden states from the backbone LLM are transformed via an $d \times d$ dimensional adapter layer, which then feeds into a Medusa head to generate a probability distribution across the vocabulary space. Both the Medusa head and the adapter layer are concurrently trained with a frozen backbone LLM, leveraging the pre-learned representations of the base model to enhance fine-tuning efficiency.

4.2 Cascadusa Model Overview

The foundational architecture of Medusa facilitates sequential token prediction through parallel processing but does not account for potential interdependencies among the decoding heads. To address this, we propose a novel architecture termed "Cascadusa," which enhances the original design by introducing a cascading connection scheme among the decoding heads. Specifically, each head in Cascadusa receives from the output of the preceding head's adapter layer, augmented with a residual connection. This design intuitively incorporates causal relationships between successive tokens and allows each decoding head to adjust its predictions based on the output of its predecessors.

Additionally, we introduce a variant of this architecture, "Cascadusa-var," which positions itself between the original Medusa and Cascadusa by modifying the nature of the residual connections. This variant aims to highlight the impact of the cascaded architecture on the accuracy of token predictions by establishing a baseline comparison with the

original Medusa setup.

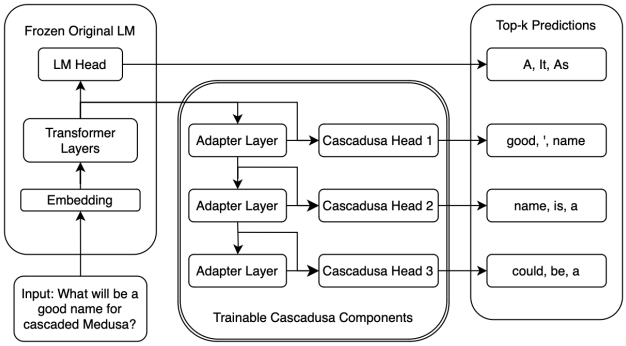

Figure 2. Cascadusa Architecture

4.3 Frozen Large Language Model

For our experiments, we utilize the Vicuna 7B model (Chiang et al., 2023), a derivative of the Llama 2 architecture (Touvron et al., 2023), which has been specifically fine-tuned on the ShareGPT dataset containing conversational data from ChatGPT 3.5. The Vicuna 7B v1.5 model is distinguished within the open-source community for its robust performance as a conversational assistant and supports extensive contextual interactions with a token capacity of up to 16,000. In our architecture, the final hidden layer of Vicuna 7B serves as the foundational input for the Cascadusa adapter layers. This model remains static during experiments to ensure that enhancements in token prediction accuracy can be attributed to the trainable components of Cascadusa without the confounding effects of backbone model variability. Training the Cascadusa components against this frozen backbone employs cross-entropy loss, wherein the predictive accuracy progressively decreases as the sequence of decoding heads increases, a phenomenon consistent with the increasing complexity of predicting further along the token sequence.

4.4 Trainable Cascadusa Components

The Cascadusa model comprises two principal trainable components: the residual adapter layers and the Cascadusa heads. Each component is specifically designed to refine the prediction capabilities of the system through structured learning and parameter adaptation.

The definition of the k-th head prediction is given by:

$$p_t^k = \text{softmax}\left(W_2^k \cdot \left(\text{SiLU}\left(W_1^k \cdot h_{k-1}\right) + h_{k-1}\right)\right) \quad (1)$$

where W_1^k is the adapter layer, W_2^k is the Cascadusa head, h_{k-1} is the hidden state of the k-th head's adapter layer and h_0 is the last hidden state of the LM.

4.4.1 *Residual Adapter Layer*

The residual adapter layers in Cascadusa act as pivotal transformation units within the decoding pipeline. For the initial decoding head, this layer adapts the last hidden states of the Vicuna 7B backbone into a suitable form for the first Cascadusa head. Subsequent adapter layers are tasked with transforming the output from the previous head's adapter layer, thus perpetuating the flow of processed information across the sequence of decoding heads. This architecture allows each subsequent head to build upon the modified representations of its predecessors, facilitated by a residual connection that helps maintain the integrity of information throughout the sequence.

4.4.2 *Cascadusa Head*

Each Cascadusa head is a sophisticated component initialized with parameters from the corresponding head of the original language model. This strategic initialization leverages pre-trained efficiencies for immediate relevance while allowing incremental adjustments through training. The primary function of each Cascadusa head is to convert the enhanced hidden states, outputted by the adapter layer, into a probability distribution over the vocabulary.

4.5 Cascadusa-var

While the Cascadusa architecture demonstrates enhanced prediction accuracy for the second and third decoding heads, it exhibits a slight decrement in accuracy for the initial head. To explore the impact of cascading dependencies on prediction accuracy, we introduce Cascadusa-var. This variant modifies the standard Cascadusa model by substituting its intra-head residual connections with connections that link directly back to the original language model's last hidden state. This adjustment aims to mitigate the potential adverse effects on the initial decoding performance while maintaining the benefits of cascaded processing. Cascadusa-var thus serves as an intermediary architecture, combining elements from both the original Medusa and Cascadusa frameworks, to balance the integration depth with prediction fidelity.

The definition of the k-th head prediction is given by:

$$p_t^k = \text{softmax}\left(W_2^k \cdot \left(\text{SiLU}\left(W_1^k \cdot h_{k-1}\right) + h_t\right)\right) \quad (2)$$

5 EXPERIMENT

This section describes our experiment environment setup, model training and evaluation, and experimental results.

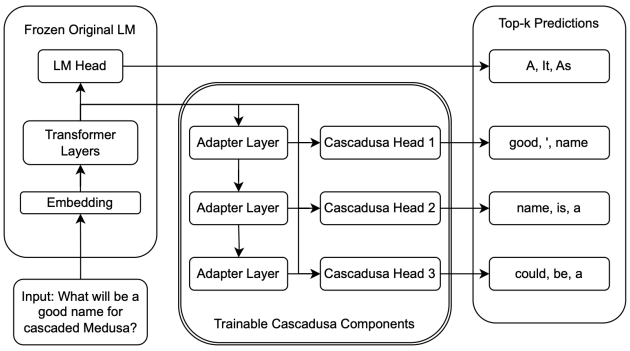

Figure 3. Cascadusa-var Architecture

5.1 Experimental Setup

5.1.1 Platform

All experiments were conducted on the Pittsburgh Supercomputing Center (PSC)'s Bridges 2 machines. For model training, we allocated a GPU session equipped with two Nvidia V100-32GB GPUs, along with 500GB of shared storage space and 490GB of DRAM. During model evaluation, we utilized a single GPU node featuring one Nvidia V100-32GB GPU, while maintaining the other resources consistent with the training session.

5.1.2 Dataset Generation

We generated our training data following Medusa's self-distillation approach. Initially, we obtained a list of seed conversations (i.e., prompts) from ShareGPT, which were subsequently input into the Vicuna model to generate corresponding replies. Notably, to expedite model training, we downscaled the size of the seed dataset by 80

5.2 Model Training

Our proposed model was trained employing a method akin to that of Medusa. However, due to our constrained GPU resources, we tailored Medusa's training method to alleviate memory requirements and align with our system specifications:

- We opted for Vicuna-7B, the smallest model within the Vicuna series, as our base model to mitigate GPU memory demands. Despite the fact that the LLM's parameters remain fixed during training, selecting a smaller model significantly curtails GPU memory usage, facilitating training with a larger parameter count.
- Given that the flash attention library **flash-attn** (Dao et al., 2022) lacks support for V100 GPU, we disabled this feature during model training.
- To further diminish memory consumption, we loaded the base LLM model in 8-bit quantized form.

Furthermore, while the original Medusa was trained on a single Nvidia A100 GPU, our training pipeline, even with the aforementioned adjustments, failed to execute on a single V100 GPU. Consequently, we employed Huggingface's **accelerate** library to train our model across two GPUs, leveraging Deepspeed's ZeRO-1 optimization. Theoretically, the training process could leverage up to 4 V100 GPUs simultaneously (constrained by PSC's policy). However, the request for four GPUs at once substantially extends the wait time for resource allocation, making it an impractical choice.

Before training our proposed model, to ensure fair comparison, we initially trained the original Medusa model with three additional heads, employing the aforementioned configurations and our self-distillation datasets as a baseline model. Subsequently, we proceeded to train our Cascadusa model and its variant under the same training settings.

5.3 Evaluation

5.3.1 Evaluation Metrics

To assess the effectiveness of our proposed model, we conduct a comparative analysis involving the baseline model (as detailed in the preceding section), our Cascadusa model, and its variants. We primarily employ two sets of criteria for evaluation: (1) Comparison of the top-1 accuracy of each prediction head. (2) Comparison of the speedup ratio achieved by each model.

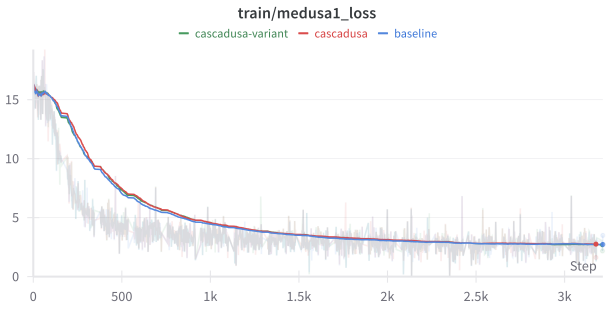

Figure 4. Visualization of losses during model training. The image is created via **wandb**. The transparent lines indicate original data, while the solid lines show the losses after smoothing.

5.3.2 Experimental Results

Initially, we visualize the loss of Medusa heads during model training, as illustrated in Figure 4. It is evident that our proposed models converge at a similar rate to the original Medusa model. We then compare the top-1 to top-5

Table 1. Comparison of Metrics between Baseline, Cascadusa, and Cascadusa Variant

Metric	Baseline	Cascadusa	Change (%)	Cascadusa-var	Change (%)
Medusa1 Top1	0.58833	0.5835	-0.82%	0.58712	-0.21%
Medusa1 Top2	0.72113	0.70986	-1.56%	0.71791	-0.45%
Medusa1 Top3	0.77425	0.77062	-0.47%	0.77223	-0.26%
Medusa1 Top4	0.81247	0.80443	-0.99%	0.80765	-0.59%
Medusa1 Top5	0.82897	0.82817	-0.10%	0.82777	-0.14%
Medusa2 Top1	0.34326	0.3505	2.11%	0.34728	1.17%
Medusa2 Top2	0.46278	0.47445	2.52%	0.4672	0.96%
Medusa2 Top3	0.54286	0.5497	1.26%	0.54044	-0.45%
Medusa2 Top4	0.5839	0.59557	2.00%	0.58592	0.35%
Medusa2 Top5	0.62294	0.6338	1.75%	0.62294	0.00%
Medusa3 Top1	0.22133	0.22777	2.91%	0.22575	2.00%
Medusa3 Top2	0.31549	0.32716	3.70%	0.32113	1.79%
Medusa3 Top3	0.37988	0.39557	4.13%	0.38551	1.48%
Medusa3 Top4	0.42897	0.44386	3.47%	0.43139	0.56%
Medusa3 Top5	0.4668	0.47847	2.50%	0.46398	-0.61%

accuracies of all three medusa heads between three models, as shown in Table 1.

It can be seen from the table that, although our proposed model's head 1 have slightly lower accuracy compared with the baseline model, we have significantly improved the prediction accuracy of the second and third heads. Additionally, the model Cascadusa-var demonstrates intermediate performance between the Medusa model and Cascadusa.

We additionally evaluate the speedup ratio of our models compared to the original Medusa Model, as depicted in Table 2. Specifically, we execute all three models on the same prompts for 50 iterations and compute their average number of tokens generated per minute.

Table 2. Comparison of Tokens/sec and Speedup

Model	Tokens/sec	Speedup (%)
Baseline	43.983	0%
Cascadusa	45.109	2.6%
Cascadusa-var	44.023	0.1%

5.4 Limitations

Resource constraints, particularly concerning GPU type and memory, severely restrict the design space for our model structure. Additionally, due to limited disk quota and time constraints for this project, we had to significantly curtail the amount of data used for model training. Consequently, our model exhibits only marginal improvements over the original Medusa model. Given access to ample hardware resources, there is potential for substantial enhancements in both the prediction accuracy of Medusa heads and the overall speedup of the base model.

6 Conclusion

In this paper, we introduced Cascaded Speculative Decoding (Cascadusa), an architecture that enhances the efficiency of LLM serving. Our research demonstrated that Cascadusa accelerates LLM inference speeds while maintaining or improving prediction accuracy by integrating multiple speculative decoding mechanisms into a cascaded framework. This innovative approach demonstrates improvements in inference speed and accuracy. Our results advocate for the potential of Cascadusa in practical applications and highlight its contributions to the speculative decoding field. Future work will aim to optimize this architecture further and explore its applicability across various domains. By advancing speculative decoding strategies, we strive to make LLMs more accessible and effective for a broader range of computational environments, ultimately pushing the boundaries of machine learning system capabilities.

References

Bhendawade, N., Belousova, I., Fu, Q., Mason, H., Rastegari, M., and Najibi, M. Speculative streaming: Fast llm inference without auxiliary models, 2024.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Cai, T., Li, Y., Geng, Z., Peng, H., Lee, J. D., Chen, D., and Dao, T. Medusa: Simple llm inference acceleration framework with multiple decoding heads, 2024.

Chen, Z., Yang, X., Lin, J., Sun, C., Chang, K. C.-C., and

Huang, J. Cascade speculative drafting for even faster llm inference, 2024.

Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., Stoica, I., and Xing, E. P. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

Fu, Y., Bailis, P., Stoica, I., and Zhang, H. Break the sequential dependency of llm inference using lookahead decoding, 2024.

Jeon, W., Gagrani, M., Goel, R., Park, J., Lee, M., and Lott, C. Recursive speculative decoding: Accelerating llm inference via sampling without replacement, 2024.

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

Li, Y., Wei, F., Zhang, C., and Zhang, H. Eagle: Speculative sampling requires rethinking feature uncertainty, 2024.

Stern, M., Shazeer, N., and Uszkoreit, J. Blockwise parallel decoding for deep autoregressive models, 2018.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models, 2023.

Xia, H., Ge, T., Wang, P., Chen, S.-Q., Wei, F., and Sui, Z. Speculative decoding: Exploiting speculative execution for accelerating seq2seq generation, 2023.

Xia, H., Yang, Z., Dong, Q., Wang, P., Li, Y., Ge, T., Liu, T., Li, W., and Sui, Z. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*, 2024.