Towards Fast Multilingual LLM Inference: Speculative Decoding and Specialized Drafters

Anonymous ACL submission

Abstract

Large language models (LLMs) have revolutionized natural language processing and broadened their applicability across diverse commercial applications. However, the deployment of these models is constrained by high inference time in multilingual settings. To mitigate this challenge, this paper explores a training recipe of an assistant model in speculative decoding, which are leveraged to draft and-then its future tokens are verified by the target LLM. We show that language-specific draft models, optimized through a targeted *pretrain-and-finetune* strategy, substantially brings a speedup of inference time compared to the previous methods. We validate these models across various languages in inference time, out-of-domain speedup, and GPT-4o evaluation.

1 Introduction

Large language models (LLMs) such as Gemini (Team et al., 2023), GPT (Achiam et al., 2023), and Llama (Touvron et al., 2023a) have remarkable success across various natural language processing tasks. Their deployment in commercial settings has expanded to include applications such as coding assistance, writing support, conversational interfaces, and tools for search (Reid et al., 2024). Despite their potential, the practical deployment of these models is often limited by prohibitively high inference time, particularly in multilingual contexts (Ahia et al., 2023). For example, character-level and byte-level models exhibit encoding length discrepancies exceeding fourfold for certain language pairs, resulting in significant disparities in cost and inference time available to different language communities (Petrov et al., 2024). These challenges present substantial huddles to scalable and cost-efficient applications of LLMs.

Speculative decoding, utilizing assistant models, has emerged as a promising strategy to improve LLM inference efficiency (Leviathan et al., 2023; Chen et al., 2023; Xia et al., 2024), inspired by speculative execution (Burton, 1985). This method drafts potential future tokens by leveraging a smaller model for the initial predictions. In parallel, these tokens are verified by the target LLM, ensuring only outputs aligned with the target LLM's predictions are accepted. Recent efforts are focused on aligning these initial predictions with the target LLM's outputs (Liu et al., 2023; Zhou et al., 2023). This involves advancing the training methods and modifying the architectural design of drafters (Miao et al., 2024; Li et al., 2024).

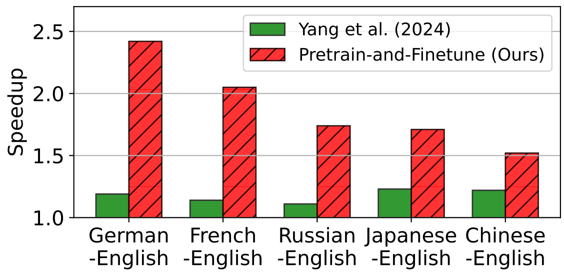


Figure 1: Speedup ratio[1] relative to the standard autoregressive greedy decoding on various multilingual datasets. Target model is Vicuna 7B v1.3 and the drafter is Vicuna 68M. Speculative greedy sampling is implemented with the drafter by Yang et al. (2024) (green) and our specialized drafter (*pretrain-and-finetune*) (red).

Although speculative decoding has garnered considerable hype recently, the adaptation of this approach to multilingual scenarios common in real-world applications remains largely unexplored. Prevailing methods (Cai et al., 2024; Li et al., 2024; Yang et al., 2024) use small drafters simply trained on datasets such as ShareGPT (ShareGPT, 2023) which is often used for instruction tuning of LLMs to learn a pattern of target LLM's language modeling. However, our investigations reveal that such approach are insufficient for multilingual translation (Figure 1). This research also raises concerns

[1]Evaluated on a single RTX3090 GPU with a batch size 1.

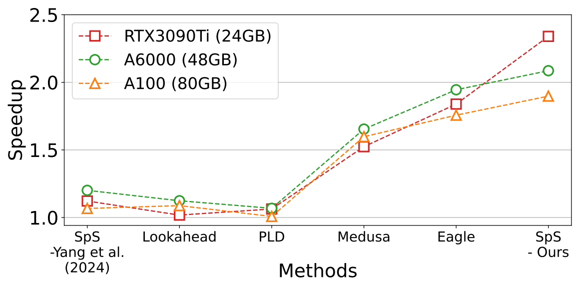


Figure 2: Speedup comparison of various speculative decoding methods on WMT16 De-En dataset (Bojar et al., 2016) with greedy settings ($T = 0.0$) across various hardwares. Target model is Vicuna-7B.

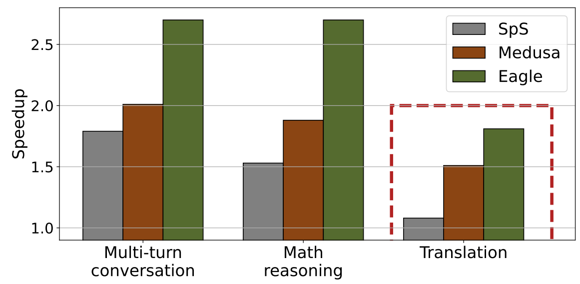


Figure 3: Speedup[2] comparison across categories containing multi-turn conversation (MT-Bench) (Zheng et al., 2024), math reasoning (GSM8K) (Cobbe et al., 2021), and translation (WMT16 De-En). Target model is Vicuna-7B with speculative greedy sampling.

regarding the capacity of such small drafters with simple tuning to comprehend the nuances of all target languages, thus questioning the feasibility of employing such models for universal speculative decoding. This paper aims to shed light upon the behaviors of drafters in speculative decoding within multilingual tasks and to explore their efficacy. Our contributions are as follows:

- We demonstrate that the strategy of *pretrain-and-finetune* significantly improves the alignment of drafter models, achieving the highest speedup ratio among the baselines (Figure 2).
- We find that the speedup ratio increases as the number of tokens specific to the target task used in training increases. This speedup is logarithmically proportional to the scale of token count in drafter training.
- In multilingual translation, we observe that input languages consistent with the training set result in notable speedup, whereas outputs aligned with the training domain do not necessarily lead to improved performance. Additionally, our results are corroborated by GPT-4o judgment scores and qualitative analyses.

2 Method

2.1 Preliminaries: speculative decoding

Speculative decoding employs a draft-verify-accept paradigm for fast inference. This method leverages a simpler assistant model (M_p) to predict easy tokens, thereby addressing memory bandwidth constraints in LLM inference (Shazeer, 2019):

1. **Draft**: An assistant model M_p, which is less computationally intensive than the target LLM M_q, drafts the future tokens $\{x_{t_1}, \ldots, x_{t_K}\}$ based on the input sequence $x_1, \ldots, x_t$.
2. **Verify**: The target LLM M_q assesses each token x_{t_i} regarding whether it is aligned with its own predictions: $p_i = M_p(x_{t_i} | x_1, \ldots, x_t, x_{t_1}, \ldots, x_{t_{i-1}})$, $q_i = M_q(x_{t_i} | x_1, \ldots, x_t, x_{t_1}, \ldots, x_{t_{i-1}})$.
3. **Accept**: Tokens meeting the validation criteria (e.g., rejection sampling) aligned with M_q's outputs are retained. Tokens failing verification are either discarded or corrected, and the draft-verify cycle is repeated.

In this paper, the verification process employs rejection sampling (Leviathan et al., 2023; Li et al., 2024) when the temperature parameter is above zero to accept only tokens that closely match M_q's predictions. For greedy decoding with a temperature of zero, tokens are accepted if they are identical to M_q's predictions.

2.2 Motivation

Our evaluation of various speculative models, including SpS (Chen et al., 2023), Medusa (Cai et al., 2024), Eagle (Li et al., 2024), as shown in Figure 3, demonstrates that speedup ratios significantly differ by task domain. While these models excel in English tasks such as multi-turn conversations and mathematical reasoning, where they achieve notable speed improvements, they underperform in translation tasks (red dotted box in Figure 3). This result confirms that the effectiveness of these models is not universal but may highly language-specific. The consistent underperformance in translation tasks highlights a key weakness and drives our study towards developing specialized drafters.

[2] Evaluated on a single RTX3090 GPU with a batch size 1.

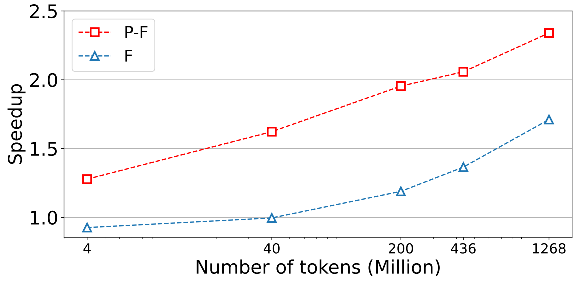


Figure 4: Speedup with speculative greedy sampling on the WMT16 De-En dataset as the training token for finetune (F) count varies, displayed on a logarithmic x-axis. ‘P-F’ represents our strategy and ‘F’ involves training solely on De-En without pretrain step (P).

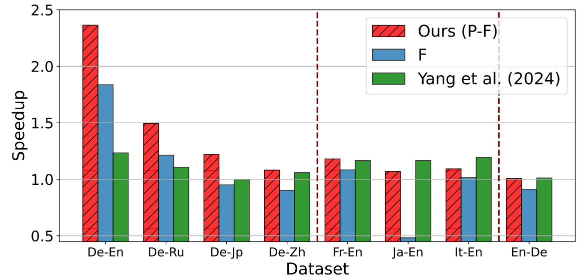


Figure 5: Speedup with speculative greedy sampling on various out-of-domain dataset as the drafters for ‘Ours (P-F)’ and ‘F’ are trained on WMT16 De-En dataset.

2.3 Training specialized assistant models

At the core of our approach is the recognition that smaller models, due to their inherent limited capacity, struggle to capture the diverse token distributions across languages. To address this challenge, we present specialized drafter models tailored to each language. Our strategy consists of:

1. **Pretrain (P)**: Assistant models are pretrained on a part of C4 (Raffel et al., 2019) and ShareGPT dataset (ShareGPT, 2023) for language modeling.

2. **Finetune (F)**: The models are finetuned on the target lingual task with instructions to refine their responses to non-English inputs.

While the practices of pretraining and finetuning are well-established paradigms in language model training, applying these steps to drafter models represents a novel adaptation within the field. Traditionally, assistnat models have been trained from scratch with little strategic rationale or clear justification for dataset selection.

Figure 4 shows that the *pretrain-and-finetune* strategy significantly boosts the speedup ratio as the number of training tokens increases. Our ‘P-F’ approach outperforms models that are only finetuned (F), and even surpasses the speedup rates by Yang et al. (2024), which stood at 1.12.

Dataset with self-distillation The training dataset for our assistant models is generated through self-distillation from the target LLM, ensuring alignment with its outputs (Kim and Rush, 2016; Zhou et al., 2023; Cai et al., 2024). To capture the full range of the target’s output variability, we generate multiple responses at a range of temperatures—{0.0, 0.3, 0.7, 1.0}.

3 Experiment

3.1 Experimental setup

Models We utilize Vicuna 7B (Chiang et al., 2023), Gemma-Instruct 7B (Team et al., 2024), and Llama2-chat (Touvron et al., 2023b) as target LLMs. The drafter models employed include Vicuna 68M (Yang et al., 2024), a custom Gemma 250M drafter and Llama 68M (Miao et al., 2024). Training configurations are outlined in Appendix F.

Number of drafts For speculative sampling (SpS), we use a single draft candidate (Chen et al., 2023). In contrast, Medusa and Eagle models are evaluated using multiple drafts via tree-attention mechanism by following their original settings.

Training and evaluation Training datasets for each target model are generated via self-distillation and comprise five datasets: German (De)→English (En), French (Fr)→En, Russian (Ru)→En, Japanese (Ja)→En and Chinese (Zh)→En, each with 4 million (M) conversations (∼ 1.3 billion (B) tokens) sourced from WMT14 Fr-En (Bojar et al., 2014), WMT16 De-En, and Ru-En (Bojar et al., 2016), and JParaCrawl-v3.0 (Morishita et al., 2022). Evaluations are conducted using a single NVIDIA 3090 GPU, under both greedy settings (T=0.0) and for diversity at T=1.0 with three different seeds. The details are in Appendix F.

3.2 Main result

Overall Table 1 shows that our specialized drafter (*pretrain-and-finetune*) for targeted languages demonstrates superior performance across multiple translation tasks, recording the highest speedup in both deterministic (T=0.0) and diverse (T=1.0) settings. At T=0.0, our model outperforms all competitors with an average speedup ratio of

Table 1: Speedup comparison of different methods for Vicuna 7B v1.3. Results are provided for T=0.0 and T=1.0 across various translation tasks. For our approach, each drafter is finetuned with the corresponding dataset.

Method	T=0.0						T=1.0					
	De→En	Fr→En	Ru→En	Ja→En	Zh→En	Avg	De→En	Fr→En	Ru→En	Ja→En	Zh→En	Avg
Sps - Yang et al. (2024)	$1.19_{\pm0.06}$	$1.14_{\pm0.05}$	$1.11_{\pm0.04}$	$1.23_{\pm0.03}$	$1.22_{\pm0.00}$	$1.18_{\pm0.04}$	$1.07_{\pm0.03}$	$1.06_{\pm0.02}$	$1.04_{\pm0.01}$	$1.15_{\pm0.02}$	$1.11_{\pm0.02}$	$1.09_{\pm0.02}$
Lookahead (Fu et al., 2024)	$1.03_{\pm0.01}$	$1.01_{\pm0.02}$	$0.98_{\pm0.01}$	$1.00_{\pm0.01}$	$0.96_{\pm0.00}$	$1.00_{\pm0.01}$	$1.03_{\pm0.03}$	$1.04_{\pm0.03}$	$0.99_{\pm0.00}$	$0.98_{\pm0.05}$	$0.98_{\pm0.00}$	$1.01_{\pm0.02}$
PLD (Saxena, 2023)	$1.13_{\pm0.06}$	$1.05_{\pm0.04}$	$1.03_{\pm0.00}$	$1.09_{\pm0.05}$	$0.99_{\pm0.07}$	$1.06_{\pm0.05}$	-	-	-	-	-	-
Medusa (Cai et al., 2024)	$1.58_{\pm0.05}$	$1.57_{\pm0.01}$	$1.52_{\pm0.01}$	$1.55_{\pm0.01}$	$1.43_{\pm0.00}$	$1.53_{\pm0.02}$	$1.61_{\pm0.03}$	$1.69_{\pm0.01}$	$\mathbf{1.62_{\pm0.00}}$	$\mathbf{1.72_{\pm0.01}}$	$\mathbf{1.60_{\pm0.01}}$	$1.65_{\pm0.01}$
Eagle (Li et al., 2024)	$1.90_{\pm0.05}$	$1.88_{\pm0.00}$	$1.67_{\pm0.05}$	$\mathbf{1.88_{\pm0.01}}$	$\mathbf{1.75_{\pm0.01}}$	$1.81_{\pm0.02}$	$1.57_{\pm0.00}$	$1.61_{\pm0.01}$	$1.45_{\pm0.02}$	$1.63_{\pm0.01}$	$1.51_{\pm0.03}$	$1.55_{\pm0.01}$
Sps - *pretrain-and-finetune* (Ours)	$\mathbf{2.42_{\pm0.02}}$	$\mathbf{2.05_{\pm0.04}}$	$\mathbf{1.74_{\pm0.02}}$	$1.71_{\pm0.01}$	$1.52_{\pm0.01}$	$\mathbf{1.89_{\pm0.02}}$	$\mathbf{1.99_{\pm0.01}}$	$\mathbf{1.86_{\pm0.03}}$	$1.58_{\pm0.00}$	$1.67_{\pm0.01}$	$1.44_{\pm0.00}$	$\mathbf{1.71_{\pm0.01}}$

Table 2: Examples of speculative decoding on WMT16 De-En dataset. Black indicates standard decoded output and magenta indicates accepted draft tokens.

Input
Translate German to English: So wie er gestartet ist , wird es nicht lange dauern , bis er auf der „ Pferd des Jahres “ -Schau ist – und ich bin mir sicher , dass er gut abschneiden wird.
SpS with a drafter by Yang et al. (2024)
As he started, it won't take long until he's on the "Horse of the Year" show, and I'm sure he'll do well.
Eagle (Li et al., 2024)
As he started, it won't take long until he's on the "Horse of the Year" show, and I'm sure he'll do well.
SpS with our specialized drafter (*pretrain-and-finetune*)
As he started, it won't take long until he's on the "Horse of the Year" show, and I'm sure he'll do well.

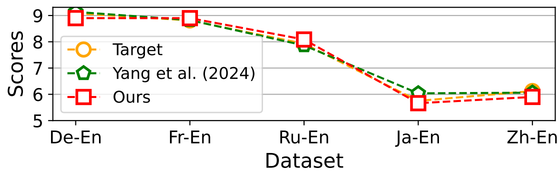


Figure 6: GPT-4o judgment scores following the Zheng et al. (2024) on various multilingual translation dataset. The score is evaluated random sampling with T=1.0.

Table 3: Ablations with speedup as the training stages continue on WMT19 Zh→En.

Target LLM - Drafter	P	+ F
Gemma-Instruct 7B - Gemma 250M	$0.92_{\pm0.01}$	$1.04_{\pm0.02}$
Llama2-chat 7B - Llama 68M	$1.47_{\pm0.00}$	$1.95_{\pm0.01}$

1.89. Similarly, at T=1.0, it maintains robust performance with an overall speedup ratio of 1.71.

Speedup on out-of-domain translation tasks As Figure 5 shows, our analysis reveals variability applying the drafter, trained on the WMT16 De-En dataset, across diverse translation pairs. Speedups are consistently higher when translating from German to other languages, highlighting the importance of input domain consistency with the training data. Conversely, translations involving non-German languages with English and English-German pairings show limited gains. This result emphasizes that effective speculation relies more on matching the translation task's input domain with the training data than on the output domain.

Qualitative analysis on responses Table 2 provides examples of speculative inference on the WMT16 De-En dataset. Both Eagle and our method incorporate a significant number of accepted tokens from drafts. However, our model achieves this with $\sim 75\%$ fewer parameters, leading to reduced latency and faster inference time (Table 1). Speculation typically takes place at critical junctions of the sentence such as transitions between clauses and phrases.

GPT-4o judgment analysis Figure 6 depicts the GPT-4o judgment scores (Zheng et al., 2024) generated using a temperature of 1.0. Our drafter closely matches the target Vicuna LLM across multiple datasets. The setup and further results are in Appendix F and Appendix G.

Ablation study Table 3 presents the ablation results, illustrating the progressive impact of the *pretrain-and-finetune* approach on the performance of Gemma and Llama2-chat models.

4 Conclusion

This paper has demonstrated that the *pretrain-and-finetune* strategy for training drafters significantly enhances speedup ratio relative to standard autoregressive decoding in multilingual translation tasks. This gain grows logarithmically with the increase in the number of training tokens. Supported by qualitative analysis, out-of-domain analysis, and GPT-4o evaluation, this strategy substantially outperforms the state-of-the-art methods in various language pairs. Our study uncovers approaches to maximize the benefits from drafter models, thereby setting a new benchmark in this area.

Limitations

Despite the improvement, our approach, requiring separate drafters for each language, introduces complexities in deployment, especially in multilingual settings. For instance, in environments where multiple languages are frequently interchanged, such as multinational corporations or global customer service platforms, the lack of an automated drafter selection system could hinder operational efficiency. Currently, our study focuses on independent drafters; however, examining systems that utilize interdependent models, similar to Eagle and Medusa, might offer insights into more interesting strategies. Additionally, while our findings are promising for translation tasks, expanding this methodology to other multilingual applications, like real-time multilingual generation or summarization, is essential to understand its broader applicability and uncover additional constraints.

This work primarily presents no direct ethical concerns. Further discussions are detailed in Appendix B and Appendix H.

References

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

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David R Mortensen, Noah A Smith, and Yulia Tsvetkov. 2023. Do all languages cost the same? tokenization in the era of commercial language models. *arXiv preprint arXiv:2305.13707*.

Sangmin Bae, Jongwoo Ko, Hwanjun Song, and Se-Young Yun. 2023. Fast and robust early-exiting framework for autoregressive language models with synchronized parallel decoding. *arXiv preprint arXiv:2310.05424*.

Nikhil Bhendawade, Irina Belousova, Qichen Fu, Henry Mason, Mohammad Rastegari, and Mahyar Najibi. 2024. Speculative streaming: Fast llm inference without auxiliary models. *arXiv preprint arXiv:2402.11131*.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.

Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation (wmt16). In *First conference on machine translation*, pages 131–198. Association for Computational Linguistics.

F Warren Burton. 1985. Speculative computation, parallelism, and functional programming. *IEEE Transactions on Computers*, 100(12):1190–1193.

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. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.

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*.

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*.

Yimin Fan, Yaobo Liang, Alexandre Muzio, Hany Hassan, Houqiang Li, Ming Zhou, and Nan Duan. 2021. Discovering representation sprachbund for multilingual pre-training. *arXiv preprint arXiv:2109.00271*.

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*.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & faster large language models via multi-token prediction. *arXiv preprint arXiv:2404.19737*.

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

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Taehyeon Kim, Ananda Theertha Suresh, Kishore Papineni, Michael Riley, Sanjiv Kumar, and Adrian Benton. 2024. Exploring and improving drafts in blockwise parallel decoding. *Preprint*, arXiv:2404.09221.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. 2024. Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*.

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.

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

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joey Gonzalez. 2020. Train big, then compress: Rethinking model size for efficient training and inference of transformers. In *International Conference on machine learning*, pages 5958–5968. PMLR.

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

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, et al. 2024. Specinfer: Accelerating large language model serving with tree-based speculative inference and verification. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, pages 932–949.

Makoto Morishita, Katsuki Chousa, Jun Suzuki, and Masaaki Nagata. 2022. Jparacrawl v3. 0: A large-scale english-japanese parallel corpus. *arXiv preprint arXiv:2202.12607*.

OpenAI. 2024. Hello GPT-4o. Accessed: Insert the current date.

Jiayi Pan. 2023. Tiny-vicuna 1b. https://huggingface.co/Jiayi-Pan/Tiny-Vicuna-1B.

David A Patterson. 2004. Latency lags bandwith. *Communications of the ACM*, 47(10):71–75.

Aleksandar Petrov, Emanuele La Malfa, Philip Torr, and Adel Bibi. 2024. Language model tokenizers introduce unfairness between languages. *Advances in Neural Information Processing Systems*, 36.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Apoorv Saxena. 2023. Prompt lookup decoding.

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

ShareGPT. 2023. Sharegpt: Vicuna unfiltered dataset. https://huggingface.co/datasets/Aeala/ShareGPT_Vicuna_unfiltered. Accessed: 2024.

Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*.

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

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

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

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. 2024. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*.

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

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

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

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*.

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

Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. 2023. Distillspec: Improving speculative decoding via knowledge distillation. *arXiv preprint arXiv:2310.08461*.

A Overview of appendix

This appendix provides supplementary material that expands on the main contents. Each section is designed to complement the research presented:

- Appendix B: **Broader impact** - Examines the wider implications of our findings on speculative decoding.
- Appendix C: **Future work** - Outlines possible directions for future research, building upon the current study's findings to explore new avenues and improvements.
- Appendix D: **Related works** - Provides a comprehensive review of literature and previous research that relate to the speculative decoding techniques discussed in the paper.
- Appendix E: **Algorithm** - Details the algorithms used in the speculative decoding processes, providing pseudocode and explanations to support reproducibility.
- Appendix F: **Implementation details** - Offers an in-depth look at the practical implementation of the speculative decoding methods, including baselines, self-distillation, training, and GPT-4o evaluation.
- Appendix G: **Additional experimental results** - Presents extra experimental data and analyses that were not included in the main sections due to space constraints.
- Appendix H: **Discussions** - Engages in discussions on results, such as foundational beliefs that underpin our research approach, the number of drafts used, and drafter size.

Each appendix is intended to provide clarity and additional context to the research.

B Broader impact

Implementing language-specific drafters significantly enhances the speed of large language models tailored to diverse linguistic environments. For instance, a system could leverage heuristic analysis of input prompt token distributions to automatically select an optimal drafter, streamlining processing efficiency. Moreover, if a user interface allows individuals to choose their preferred language, the system can instantly apply the corresponding drafter, thereby accelerating response times considerably.

Such advancements not only reduce computational load but also enrich the user experience by providing rapid and culturally relevant responses in multilingual contexts.

C Future work

Future projects will explore broadening the scope of our speculative decoding framework to cover general multi-task environments, extending beyond multilingual translation to include varied domains such as legal and medical text processing. A significant challenge lies in developing an efficient method for selecting the appropriate drafter among multiple options when direct user input is unavailable or when inputs consist of mixed languages. This issue becomes more complex as the ambiguity of language indicators increases. To alleviate this, designing an advanced router capable of intelligently assigning tasks to the most suitable drafter based on the nature of the input presents a promising direction. Training such a router involves leveraging advanced techniques to understand and predict the optimal drafter based on contextual representations. This approach aims to improve the overall efficiency and accuracy of language model applications across diverse and dynamically changing content landscapes.

D Related works

D.1 Speculative decoding

Speculative decoding, advancing from blockwise parallel decoding introduced by Stern et al. (2018), adopts a draft-then-verify paradigm to enhance LLM inference efficiency. This method addresses latency issues in autoregressive decoding, which stem from the extensive memory transfers required for each token generation, leading to computational underutilization (Xia et al., 2024; Patterson, 2004). To further advance this paradigm, Leviathan et al. (2023) and Chen et al. (2023) introduced speculative decoding and sampling, which includes the lossless acceleration of various sampling methods. These methods utilize smaller models from the same series, such as T5-small, to accelerate inference for larger counterparts like T5-XXL without additional training.

Recent advancements in speculative decoding, exemplified by models like EAGLE (Li et al., 2024) and Medusa (Cai et al., 2024), have significantly refined the efficiency of LLMs by integrating lightweight feedforward neural network (FFN) heads directly into their architecture. These FFN heads facilitate the early drafting of token sequences, enhancing throughput and reducing latency. Similarly, approaches such as the self-speculative model (Zhang et al., 2023) and Elhoushi et al. (2024) incorporate early exiting and layer skipping strategies, allowing for a reduction in computational load by prematurely terminating decoding processes or bypassing less impactful neural layers. Another line of research explores the blockwise parallel language models with multiple softmax heads pretrained from scratch presented by Stern et al. (2018) by either refining its drafts (Kim et al., 2024) or scaling up the model size (Gloeckle et al., 2024).

D.2 Inference acceleration of LLM

As LLMs continue to evolve rapidly, enhancing their inference speed has become a focal area of research. Traditional techniques such as knowledge distillation (Gu et al., 2023; Ko et al., 2024), model compression (Li et al., 2020), and quantization (Xiao et al., 2023) aim to optimize these models but often require extensive training adjustments or significant architectural modifications. More recent strategies have shifted towards applying early exiting mechanisms, particularly within series like T5 (Schuster et al., 2022; Bae et al., 2023) and decoder-only architectures (Varshney et al., 2023), to streamline inference processes. Although early exiting can significantly hasten model responses by truncating computational sequences, this method typically introduces a trade-off with performance degradation (Schuster et al., 2022).

E Algorithm: speculative sampling

By referring to Chen et al. (2023), Algorithm 1 demonstrates the speculative sampling process. Initiating with an initial prompt, an assistant model is utilized to generate multiple prospective continuations at each step, which are concurrently verified against the target LLM's predictions.

Each candidate token's acceptance probability is calculated based on the target LLM's relative confidence compared to the assistant model's suggestion (i.e., rejection sampling). If a value, randomly drawn from a uniform distribution, falls below this threshold, the token is accepted and incorporated into the ongoing sequence. If not, the algorithm recalibrates, adjusting the speculative path by directly sampling from the differences in predictions,

Algorithm 1: Speculative sampling

input : Target LLM $\mathcal{M}_q$, a small assistant model $\mathcal{M}_p$, initial prompt sequence $x_1, \ldots, x_t$ and target sequence length T.

Initialize $t \leftarrow 1$
while $t < T$ **do**
  for $k \leftarrow 1, \ldots, K$ **do**
    $x_{t_k} \sim \mathcal{M}_p(x|x_1, \ldots, x_t, x_{t_1}, \ldots, x_{t_{k-1}})$
  end for
  In parallel, compute $K + 1$ sets of logits drafts $x_{t_1}, \ldots, x_{t_K}$ with the target LLM $\mathcal{M}_q$:
$$\mathcal{M}_q(x|x_1, \ldots, x_t), \mathcal{M}_q(x|x_1, \ldots, x_t, x_{t_1}), \ldots, \mathcal{M}_q(x|x_1, \ldots, x_t, x_{t_1}, \ldots, x_{t_K})$$
  for $j \leftarrow 1, \ldots, K$ **do**
    Sample $r \sim U[0, 1]$ from a uniform distribution
    if $r < \min(1, \frac{\mathcal{M}_q(x|x_1,\ldots,x_{t+j-1})}{\mathcal{M}_p(x|x_1,\ldots,x_{t+j-1})})$ **then**
      Set $x_{t+j} \leftarrow x_{t_j}$ and $t \leftarrow t + 1$
    else
      Sample $x_{t+j} \sim (\mathcal{M}_q(x|x_1, \ldots, x_{t+j-1}) - \mathcal{M}_p(x|x_1, \ldots, x_{t+j-1}))_+$ and exit for loop.
    end if
  end for
  If all tokens $x_{t+1}, \ldots, x_{t+K}$ are accepted, sample extra token $x_{t+K+1} \sim \mathcal{M}_q(x|x_1, \ldots, x_t, x_{t+K})$ and set $t \leftarrow t + 1$
end while

enhancing accuracy and contextual relevance.

F Implementation details

F.1 Baselines

Following the Spec-Bench settings (Xia et al., 2024), we have selected 5 speculative decoding methods, all open-source and rigorously tested for reliability. Each method represents a unique approach to improving LLM inference speeds:

1. **SpS** (Chen et al., 2023): SpS employs a smaller LM from the same model series as the drafter. In the verification, this method corrects the last token with residual probability if the token is rejected.

2. **Medusa** (Cai et al., 2024) and **Eagle** (Li et al., 2024): Both methods enhance the target LLM by integrating additional lightweight FFN heads. These heads are designed to efficiently draft potential token sequences depending on the penultimate representations from the target LLM.

3. **Lookahead** (Fu et al., 2024): This method appends multiple special tokens to the end of the input prompt. These tokens are used for parallel drafting, with the resultant drafts transformed into n-gram candidates for efficient prediction.

4. **PLD** (Saxena, 2023): Serving as the practical code implementation of Yang et al. (2023), PLD selects text spans directly from the input to serve as drafts, optimizing the relevance and accuracy of the initial predictions.

F.2 Self-distillation

We follow the self-distillation pipeline as described by Cai et al. (2024). Initially, a public dataset, such as WMT 16 De-En, is selected as the training dataset. The target model's responses are then generated using the OpenAI API server, with input prompts derived directly from the training dataset.

Install prerequisites For software dependencies, CUDA 12.1 and PyTorch 2.1.2 are required. To start the server, install the necessary dependencies:

```
vllm==0.4.0, openai==0.28.0
```

Use of vLLM We utilize the vLLM library for self-distillation, executing the following command:

```
python -m
vllm.entrypoints.openai.api_server
--model lmsys/vicuna-7b-v1.3
--port 8000 --max-model-len 2048
```

Input prompt For instance, when self-distillation the WMT14 Fr-En dataset using the

Table 4: Custom Gemma 250M model configuration.

Configuration	Value
Activation function	GeLU (Hendrycks and Gimpel, 2016)
Hidden size	768
Intermediate size	6144
Number of attention heads	16
Number of hidden layers	2
Number of key-value heads	2
RMS epsilon	1e-06
Vocabulary size	256000

Vicuna7b v1.3 model, the input prompt consists of a system prompt and a user prompt. In the user prompt, we prepend "Translate French to English: ".

> A chat between a curious user and an artificial intelligence assistant. The assistant gives helpful, detailed, and polite answers to the user's questions. USER: Translate French to English: Madame la Présidente, c'est une motion de procédure. ASSISTANT:

F.3 Details on training setup

For the shared settings across all training drafters, we employ the Fastchat[3] framework. We utilize a cosine learning rate scheduler with a warmup ratio of 0.03 and the AdamW (Loshchilov and Hutter, 2017) optimizer. The drafter is trained using the 'P-F' strategy (ours) for 3 epochs, and using the 'F' strategy (without the pretraining step 'P') for 5 epochs to ensure sufficient learning. The model's maximum length is set to 2048 tokens. The training is conducted using 4 GPUs with a batch size of 2 per GPU.

For finetuning the Vicuna 68M drafter (Yang et al., 2024), the learning rate is set to 2e-5. Similarly, for finetuning the Llama 68M model (Miao et al., 2024), the learning rate is set to 3e-5.

As a drafter for Gemma-Instruct 7B model, we newly design a Gemma 250M model as a drafter (Table 4). We use the same training recipe with Vicuna 68M and Llama 68M.

F.4 Details on GPT-4o evaluation

We follow LLM-as-a-Judge framework (Zheng et al., 2024) to evaluate the model's answers. The GPT-4o model is utilized as a judge, which has greater performance on both English and non-English than GPT-4 Turbo (OpenAI, 2024). For **Single answer grading**, used prompt is followed:

> [System]
> You are a helpful assistant. Please act as an impartial judge and evaluate the quality of the response provided by an AI assistant to the user question displayed below. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of the response. Begin your evaluation by providing a short explanation. Be as objective as possible. After providing your explanation, you must rate the response on a scale of 1 to 10 by strictly following this format: "[[rating]]", for example: "Rating: [[5]]".
>
> [Question]
> {question}
>
> [The Start of Assistant's Answer]
> {answer}
> [The End of Assistant's Answer]

The detail implementation of LLM-as-a-judge is in the following GitHub repository[4].

G Additional experimental results

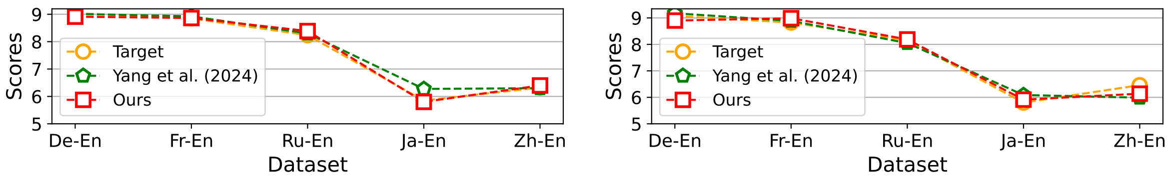
G.1 Out-of-domain speedup

Building on the findings discussed in the main body, this subsection further explores the speedup variations achieved by employing a drafter trained on each dataset across a range of translation tasks. Figure 8 depicts the speedup results using speculative greedy sampling for drafters trained on different datasets: Ru-En, Ja-En, and Zh-En.

Most observations align with those discussed in Section 3. Notably, drafters trained on the Ja-En (Figure 8 (b)) and Zh-En (Figure 8 (c)) datasets consistently outperform Yang et al. (2024)'s drafter, even on out-of-domain tasks. We hypothesize these into two folds. Firstly, this suggests that certain intrinsic properties of the Japanese and Chinese languages may improve the efficacy of speculative decoding when applied to unrelated language pairs, possibly due to specific syntactic or lexical features that are effectively captured during training. In another scenario, the target LLM does not work well on those tasks, and thus drafters are easier to catch

[3] https://github.com/lm-sys/FastChat/tree/main

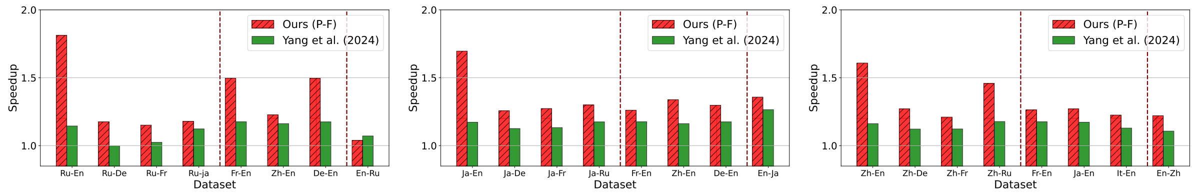
[4] https://github.com/lm-sys/FastChat/tree/main/fastchat/llm_judge



(a) T=0.8

(b) T=0.9

Figure 7: GPT-4o evaluation scores following the Zheng et al. (2024) on various multilingual translation dataset. Each figure denotes the score of random sampling with different temperature on the output whose target LLM is Vicuna 7B v1.3.



(a) Drafter trained on Ru-En

(b) Drafter trained on Ja-En

(c) Drafter trained on Zh-En

Figure 8: Speedup with speculative greedy sampling with the same settings in Figure 5.

the target token distribution. More precisely, for instance, in Zh-Ru task, Vicuna 7B should translate the Chinese to Russian, but to English, and thus the speedup seems to happen for us due to English generation.

In the case of the Ru-En (Figure 8 (a)) trained drafter, translations from Russian to other languages generally surpass Yang et al. (2024)'s results. Interestingly, translations from French to English and German to English exhibit unexpectedly high speedups. This could hint at underlying linguistic similarities or shared grammatical structures between Russian, French, and German that the Ru-En drafter is particularly adept at handling, thereby facilitating more efficient speculative decoding. While Fan et al. (2021) demonstrates that Russian belongs to another cluster from En / Fr / De, perhaps our results provide a different perspective in lens of speculative decoding.

G.2 GPT-4o judgments

Figure 7 show additional GPT-4o evaluation scores for various multilingual translation datasets. The graphs display the comparative performance across different language pairs under two sampling conditions, at temperatures T=0.8 and T=0.9, respectively. Each data point reflects the quality of translations produced by the target model (orange circle), SpS with the instruction tuned model using ShareGPT (Yang et al., 2024) (green pentagon), and SpS with our specialized drafter (*pretrain-and-finetune*) (red square). For the red points, each drafter is trained with the corresponding dataset. For instance, when the red point specify De-En, it indicates that the drafter has been fine-tuned with the De-En dataset.

The results demonstrate negligible differences in quality among the three methods, underscoring the efficacy of speculative decoding in delivering translations with lossless quality. Both temperature settings show that our speculative decoding strategy closely matches the performance of the established target model across various language pairs. This consistent performance across different settings and language pairs illustrates that speculative decoding effectively maintains high-quality outputs without compromising accuracy due to increased randomness in sampling.

H Discussion

H.1 Why is pretrain-and-finetune better in small-size LM drafter?

Drafting in speculative decoding has been treated akin to n-gram prediction (Bhendawade et al., 2024), often relying on straightforward pretraining using datasets designed to replicate target LLM behaviors, such as the ShareGPT dataset (Yang et al., 2024). This approach posits that generating a limited sequence of future tokens suffices for speculative inference.

Contrary to this belief, our empirical result

Table 5: Speedup comparison of speculative greedy sampling across different drafter sizes on WMT16 De-En dataset.

Drafter	Vicuna 68M (Yang et al., 2024)	Vicuna 68M (*pretrain-and-finetune*; Ours)	Tiny-Vicuna 1B (Pan, 2023)
Speedup	1.19	**2.42**	0.75
Mean of accepted tokens	1.47	3.03	**3.06**

presents a different narrative. Figure 5 illustrates that even in seemingly straightforward translation tasks, such as from German to English, outcomes are not as effective. This suggests that drafting requires a broader array of language modeling capabilities to manage complex linguistic structures and context variations effectively.

Drafters, therefore, benefit significantly from a robust *pretrain-and-finetune* approach, where they are first exposed to a wide array of linguistic contexts and then finely tuned to specific tasks. This training regimen transforms them into compact, yet comprehensive, language models capable of handling diverse and challenging speculative decoding scenarios with better alignment.

H.2 Number of drafts

This study primarily explores the speculative decoding process utilizing a single draft. In contrast, advanced baseline methods such as EAGLE and Medusa deploy multiple drafts, leveraging tree-attention mechanisms to enrich draft selection. This technique allows for a broader exploration of multiple draft candidates at each decoding step, potentially increasing the rate and quality of accepted drafts.

Adapting our approach to incorporate multiple drafts with tree-attention could significantly enhance performance, suggesting an untapped potential in our method. Experimenting with this expanded setup could lead to notable improvements in the speculative sampling's effectiveness, particularly in increasing the mean number of high-quality tokens accepted per sequence. This prospect opens a critical path for future research, where deeper explorations could elevate the capabilities of our specialized drafters.

H.3 Is scaling up drafter size better for SpS?

Evaluating the efficacy of increasing drafter size reveals nuanced insights into speculative decoding performance. Table 5 compares three versions of drafters: the Vicuna 68M by Yang et al. (2024), our *pretrain-and-finetune* Vicuna 68M, and Tiny-Vicuna 1B (Pan, 2023)—a larger model with 1B parameters that has been instruction-tuned.

Despite Tiny-Vicuna 1B's substantial parameter count, it achieves a lower speedup of 0.75 compared to 2.34 by our optimized Vicuna 68M. Both models show similar mean accepted tokens, suggesting that increasing size does not proportionally enhance computational efficiency. This is due to speculative decoding's reliance on minimizing memory bottlenecks to exploit parallel computation effectively. Larger models like Tiny-Vicuna 1B exacerbate these bottlenecks, diminishing the potential speed gains from increased parallelism.

Conversely, our *pretrain-and-finetune* Vicuna 68M demonstrates that strategic training and optimization of a smaller model can achieve high efficiency and speed, highlighting the importance of model configuration over mere size increase. This balance between model size and computational dynamics is crucial for optimizing speculative decoding, suggesting that enhancing model capabilities through targeted training may be more effective than scaling size.