IN-BATCH ENSEMBLE DRAFTING: ROBUST SPECULATIVE DECODING FOR LVLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the success of Speculative Decoding (SD) in LLM inference acceleration, it largely remains unexplored for Large Vision Language Models (LVLMs), an advanced class of LLMs that can handle multimodal prompts consisting of text and image tokens. To bridge this gap, we first conduct a comprehensive benchmarking study, focusing on the effectiveness of various drafting methods. We observe that various drafting methods have their own advantages, and none of them consistently outperforms the others. Motivated by this observation, we propose **In-batch Ensemble Drafting (IbED)**, a simple yet effective SD method for LVLMs. IbED leverages multiple drafting methods without incurring much additional latency via batch inference and, compared to multimodal drafting, consistently demonstrates significant improvements in block efficiency, averaging 6% (with a maximum of 23%) across a wide range of datasets.

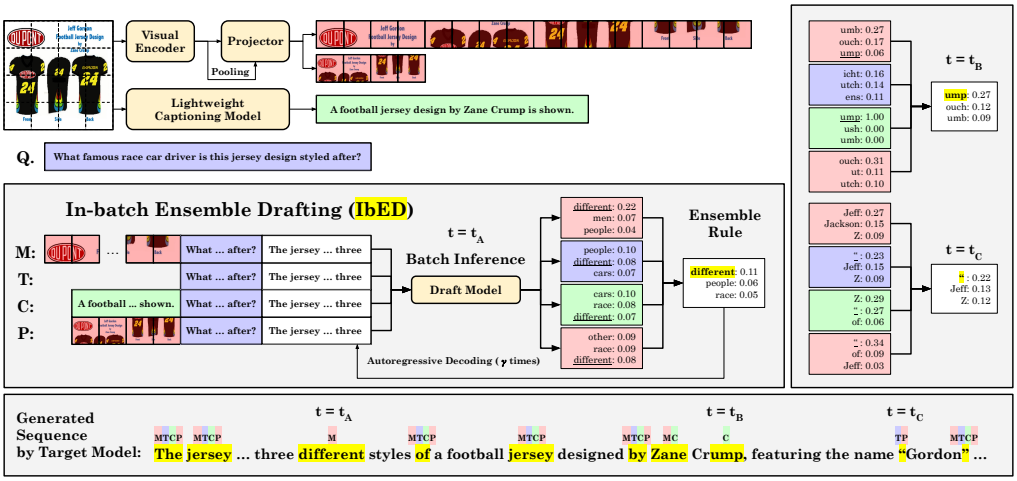


Figure 1: **Overview of In-batch Ensemble Drafting (IbED).** IbED ensembles four distinct drafting methods for LVLMs: multimodal drafting (M) and pooled multimodal drafting (P) which use direct image inputs, caption drafting (C) which uses image captions instead of images, and text-only drafting (T). These approaches are processed in parallel via batch inference. IbED effectively leverages various draft methods, achieving superior and robust performance across diverse scenarios. An example from the TextVQA dataset is shown with the real output of the IbED algorithm. Notably, the tokens accepted by IbED include all the tokens accepted by each method (M, T, C, P).

1 INTRODUCTION

Multimodal Large Language Models (MLLMs) Yin et al. (2024); Wu et al. (2023); Zhang et al. (2024a) are an advanced class of LLMs (Brown et al., 2020; Ouyang et al., 2022; Touvron et al., 2023) designed to process multiple modalities, such as images, audio, and video, alongside text. In particular, *Large Vision Language Models (LVLMs)* Chen et al. (2024c), which can handle prompts comprised of *text and images*—also known as Large Multimodal Models Li et al. (2024b); Jin et al. (2024); Song et al. (2023)—have attracted significant attention due to their unique applications.

As LVLMs are increasingly deployed, reducing their inference time has become a critical issue. LVLMs convert each image into several hundred tokens (Radford et al., 2021; Liu et al., 2023; 2024a), resulting in considerably higher inference cost. Therefore, accelerating LVLM inference is of substantial practical importance. Recently, methods like token pruning, layer skipping, and kv cache compression have been proposed to accelerate LVLM inference (Shang et al., 2024; Chen et al., 2024b; Lin et al., 2024; Liu et al., 2024c; Wan et al., 2024; McKinzie et al., 2024). While effective, these approximation techniques cannot fully preserve the original LVLM's output distribution. Moreover, they primarily reduce prompt processing time (prefilling stage) but have limited impact on response generation time (decoding stage), making them less effective for long outputs.

Speculative decoding (SD) (Leviathan et al., 2023; Chen et al., 2023) is an LLM inference acceleration technique that fully preserves the output distribution. While SD has been successful for LLM, SD for LVLMs has been far less explored, with the only notable exception by Gagrani et al. (2024). The authors successfully accelerated LVLM inference via SD for the first time and found that generating draft tokens using a small LVLM without input image tokens—relying only on text tokens—yielded comparable performance. Unfortunately, the authors did not provide a detailed analysis of this observation, underscoring the need for an in-depth study of SD for LVLM.

Motivated by this, we analyze SD for LVLM and propose an improved SD method for LVLMs. In Sec. 2, we present the first benchmark results of existing SD methods applied to LVLMs. To this end, we curate a benchmark dataset along with an out-of-distribution (OOD) dataset. Through extensive analysis, we reproduce the phenomenon observed by Gagrani et al. (2024) on a larger scale, showing that text-only and multimodal drafts each have their own advantages. In Sec. 3, we evaluate the effectiveness of alternative drafting methods such as image captioning or image pooling, observing that different methods have complementary advantages.

Leveraging these findings, in Sec. 4, we propose a new SD method for LVLMs, which we call *Inbatch Ensemble Drafting (IbED)* (Fig. 1). The key idea of IbED is very simple: we use multiple draft methods simultaneously via batch inference and use the ensemble of multiple probability distributions obtained from them. In the regime where batch inference with the draft model does not incur extra latency, this allows for the efficient use of various drafting methods. Note that unlike conventional ensemble learning, IbED does *not* require additional model parameters. We observe that IbED significantly improves the acceptance rate of draft tokens and enhances performance across diverse tasks and datasets, making it more robust. IbED achieves the best performance among all single drafting methods on each individual benchmark and OOD dataset, and demonstrates an average performance improvement of 6% (with a max of 23%) compared to multimodal drafting.

2 BENCHMARKING SD FOR LVLMS

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
LLaMA	68M	Text-only	2.06	1.75	1.83	2.23	1.95	2.48	2.13	1.76	1.72
LLaVA-1.5	68M	Multimodal	**2.24**	**2.12**	**2.26**	**2.39**	**2.34**	2.51	1.96	1.19	1.16
		Text-only	2.22	2.03	2.20	2.34	2.27	**2.77**	**2.34**	**2.05**	**2.05**
LLaVA-1.5	160M	Multimodal	**2.59**	**2.40**	**2.56**	**2.75**	**2.66**	2.77	2.23	1.29	1.27
		Text-only	2.49	2.18	2.40	2.70	2.63	**3.06**	**2.73**	**2.28**	**2.31**
LLaVA-OV	68M	Multimodal	**2.18**	**1.90**	**2.05**	2.31	**2.12**	2.31	2.15	1.77	1.74
		Text-only	**2.18**	1.86	**2.05**	**2.32**	2.11	**2.68**	**2.25**	**1.90**	**1.83**

Table 1: Block efficiency results of multimodal drafting and text-only drafting. **Bold** indicates the best results. While multimodal drafting performs better for the majority of the benchmark datasets, its performance drops noticeably on OOD datasets. In contrast, text-only drafting shows robustness to OOD datasets.

In this section, we systematically study speculative decoding for LVLMs, evaluating the performance of multimodal and text-only drafting across various benchmark datasets.

2.1 EXPERIMENT SETTINGS

Target and Draft Models We employ LLaVA-1.5 7B (Liu et al., 2024a) as our target model. We conducted benchmarks on draft models across i) model sizes: 68M, 160M, 290M, and ii) model

types: LLaVA-1.5, LLaVA-OV (Li et al., 2024a), LLaMA (see Appendix I.1 for more details). We set $\gamma = 5$ and perform greedy decoding with a maximum of 128 new tokens.

Benchmark Datasets Benchmark selection is crucial for performance evaluation, yet no established benchmark exists for LVLM SD. For SD, systems should maintain reliable speedup across an extensive range of tasks. To validate this, we carefully curated seven benchmark datasets encompassing both single-image and multi-image scenarios. We verified the response reliability of our target model for both scenarios (see Appendices G and K for benchmark details).

Out-of-Distribution (OOD) Scenarios We further evaluate performance under OOD scenarios. This is crucial for the deployment of LVLMs, which must maintain speedup even for OOD user queries caused by more diverse and variable-length inputs due to multiple images. As a notable instance of OOD scenarios for our baseline LLaVA-1.5, we include two datasets containing multi-image ($n = 5$) inputs (Li et al., 2019; Huang et al., 2016). Note that in evaluating performance of SD for production, the emphasis is on acceleration rather than the response quality itself.

Drafting Methods: Multimodal and Text-only The multimodal drafting follows the standard LVLM process. In contrast, text-only drafting, which was first explored in (Gagrani et al., 2024), uses only textual data as input for the draft model and follows the standard LLM process.

2.2 Experimental Results

Table 1 shows the block efficiency results of multimodal drafting and text-only drafting across various draft models and datasets. Comparing the block efficiency across model types and sizes, we observe improvements from LLaMA 68M to LLaVA-OV 68M, LLaVA-1.5 68M, and LLaVA-1.5 160M. The performance improves as draft model's distribution becomes closer to target distribution. Between drafting methods, multimodal drafting achieves higher block efficiency than text-only drafting on most benchmark datasets (5 out of 7), though text-only drafting demonstrates comparable overall performance. Fig. 1 illustrates the differences between these methods. Since target tokens can often be generated without visual context, text-only drafting attains comparable acceptance ratios. In cases requiring visual context, such as generating *"Zane,"* multimodal drafting succeeds where text-only drafting fails. Conversely, text-only drafting can outperform multimodal drafting when visual context is unnecessary, as it can focus solely on textual information.

OOD Datasets On OOD datasets, the performance trends of drafting methods are reversed. For LLaVA-1.5 68M and 160M, multimodal drafting's block efficiency approaches 1, making SD slower than the standard autoregressive decoding. In contrast, LLaVA-OV shows valid performance in multimodal drafting due to its specialization in multi-image processing. However, it still exhibits block efficiency lower than 2 and notably performs worse than text-only drafting. This highlights the limited robustness of multimodal drafting, particularly when compared to text-only drafting.

3 Exploring Drafting Methods for LVLMs

Since multimodal drafting does not consistently outperform text-only drafting in the preceding analysis, we investigate whether multimodal drafting can be improved. We introduce and benchmark two alternative multimodal drafting schemes, *pooled multimodal drafting* and *caption drafting*.

3.1 Pooled Multimodal Drafting

What if we include image information in the draft model but compress its context? Previous works (Shang et al., 2024; Chen et al., 2024b) show that although image tokens are more numerous than text tokens, their importance is relatively sparse. We compress image information during inference using average pooling with a 2×2 kernel, reducing the number of visual tokens from 576 to 144.

Experimental Results (Table 2, Appendix E) For benchmark datasets, pooled multimodal drafting performs slightly worse than multimodal drafting. However, they still outperform text-only drafting for 4 out of 7 datasets. This indicates that even the pooled visual tokens exhibit a certain

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
LLaMA	68M	T	2.06	1.75	1.83	2.23	1.95	**2.48**	2.13	1.76	1.72
		C	**2.12**	**1.80**	**1.86**	**2.26**	**1.98**	2.46	2.13	1.77	1.72
		TC	2.11	**1.80**	**1.86**	**2.26**	**1.98**	**2.48**	**2.15**	**1.78**	**1.75**
LLaVA-1.5	68M	M	2.24	2.12	2.26	2.39	2.34	2.51	1.96	1.19	1.16
		T	2.22	2.03	2.20	2.34	2.27	2.77	2.34	2.05	2.05
		C	2.28	2.08	2.24	2.41	2.31	2.77	2.36	2.08	2.10
		P	2.23	2.08	2.26	2.36	2.23	2.73	2.27	2.07	2.09
		MT	2.26	2.13	2.27	2.39	2.40	2.75	2.37	1.94	1.91
		MC	**2.30**	**2.17**	**2.29**	**2.42**	2.39	2.74	2.35	1.99	1.93
		MTC	2.29	2.15	2.28	2.41	**2.41**	**2.79**	2.40	2.08	2.06
		MTP	2.26	2.13	2.27	2.39	2.39	2.77	2.37	2.02	2.06
		MTCP	2.28	2.16	2.28	2.41	2.40	2.78	2.39	2.10	2.13
		MTCP*	2.28	2.16	2.28	2.41	2.39	**2.79**	**2.42**	**2.13**	**2.16**

Table 2: Block efficiency results of multimodal drafting (M), text-only drafting (T), caption drafting (C), pooled multimodal drafting (P), and IbED (**TC** and **MT**∼**MTCP**). **MTCP*** indicate the use of test-time adaptive ensemble weights. **Bold** and underline indicate the best and the second-best results. Results of IbED are highlighted when they show the best performance among all constituent methods (e.g., when MT performs best compared to M and T). Notably, MTCP and MTCP* are highlighted for all benchmark and OOD datasets.

level of image awareness. For OOD datasets, however, pooled multimodal drafting shows significantly better performance than multimodal drafting and even better than text-only drafting. Pooling visual tokens from 576 to 144 reduces over 2K tokens for OOD datasets, enhancing the robustness.

3.2 Caption Drafting

We examine whether injecting image information into a text-only draft model improves block efficiency without direct image input. One simple way to map images to text is through captions. We employ a lightweight image captioning model to generate captions for each image, using these captions as input for the draft model instead of the images themselves. Details are in Appendix F.

Experimental Results (Table 2, Appendix F.3) For LLaMA as a draft model, textual input is essential for image comprehension since it lacks image perception. Caption drafting shows the best performance compared to text-only drafting across 6 out of 7 benchmark datasets and all OOD datasets. For LLaVA-1.5, caption drafting achieves the best performance compared to text-only drafting across all benchmark and OOD datasets. Compared to multimodal drafting, caption-based drafting shows comparable performance on benchmark datasets and significantly better performance on OOD datasets, indicating that caption drafting exhibits strong robustness similar to text-only drafting. Fig. 1 shows a case where caption drafting surpasses multimodal drafting in image comprehension by extracting specific details like *"Zane Crump."*

4 In-batch Ensemble Drafting

Sections 2 and 3 reveal that while the draft model has limitations, its effectiveness improves when tailoring drafting methods to input scenarios, as each method offers unique advantages. However, predicting the best method for a given scenario remains challenging. These findings lead us to ask: Can we design a drafting method that combines these complementary strengths?

4.1 Proposed Method (Algorithm 1)

We propose In-batch Ensemble Drafting (IbED), a novel drafting method tailored for LVLMs. Unlike typical ensemble learning, IbED shares model parameters across different drafting methods, generating diverse outputs through batch inference. The distributions are then ensembled to sample the next token for the draft candidate. IbED offers advantages in three aspects. i) **Computation:** IbED avoids additional training costs and keeps ensembling costs low due to the draft model's small size. ii) **Latency:** Batch inference latency is nearly identical to single inference. iii) **Performance:** Due to their limited capacity, draft models gain substantially from ensemble learning.

We use four drafting types: multimodal (M), text-only (T), caption (C), and pooled multimodal (P). We apply a simple weighted averaging ensemble method. To show effectiveness without hyperparameter tuning, we use equal weights (1:1, 1:1:1, and 1:1:1:1).

Algorithm 1 In-batch Ensemble Drafting (IbED)

Parameter: Prefix list $X = [x^{(1)}, ..., x^{(m)}]$, Weight list $W = [w^{(1)}, ..., w^{(m)}]$
▷ $x^{(i)}$ can be $(x_{\text{img}}, x_{\text{txt}})$, $(x_{\text{caption}}, x_{\text{txt}})$, $(\texttt{None}, x_{\text{txt}})$, ...
Input: Batch sequence $b_{<t} \coloneqq [(x^{(1)}, y_{<t}), ..., (x^{(m)}, y_{<t})]$
Output: γ draft tokens $y_t, ..., y_{t+\gamma-1}$ and ensembled probabilities $q_t, ..., q_{t+\gamma-1}$

procedure IBED($b_{<t}; \gamma, W$)
  for $i \leftarrow 0$ **to** $\gamma - 1$ **do**
    $[q_{t+i}^{(1)}, ..., q_{t+i}^{(m)}] \leftarrow \text{BATCHINFERENCE}(b_{<t+i})$
    $q_{t+i} \leftarrow \text{WEIGHTEDMEAN}([q_{t+i}^{(1)}, ..., q_{t+i}^{(m)}]; W)$
    $y_{t+i} \leftarrow \text{SAMPLE}(q_{t+i})$
  end for
  return $[y_t, ..., y_{t+\gamma-1}], [q_t, ..., q_{t+\gamma-1}]$
end procedure

Test-Time Adaptive Ensemble Weights Fixed equal weights work in ensemble learning, but test-time adaptive weights improve performance further. Since test-time adaptation methods are designed for real-world testing scenarios, they assume that only input data can be utilized without access to the corresponding ground truth labels (Wang et al., 2021; 2022). For SD, however, the target model verifies each block generated from draft model. This information can be leveraged when drafting restarts from step t, allowing for the dynamic adjustment of each drafting method's influence based on its performance. To achieve this, we introduce the weight list $W_t = [w_t^{(1)}, \ldots, w_t^{(m)}]$ at each timestep t when a new drafting block begins (Algorithm 1). This weight list W_t is used throughout the current drafting block of γ tokens (i.e., from timestep t to $t + \gamma - 1$). Each weight $w_t^{(i)}$ reflects the reliability of the drafting method $q^{(i)}$ based on its past performance, quantified by the accumulated error over the previous l steps of the window as $e_t^{(i)} = \sum_{t'=t-l}^{t-1} D_{\text{KL}}\big(p(\cdot \mid x, y_{<t'}) \parallel q^{(i)}(\cdot \mid x, y_{<t'})\big)$, where D_{KL} is the KL divergence between target p and $q^{(i)}$ at each of the previous l steps within the window $t' \in [t-l, t-1]$. We then compute the weights as $w_t^{(i)} = (1/e_t^{(i)})/(\sum_{j=1}^{m} 1/e_t^{(j)})$. Drafting methods with lower accumulated errors, indicating closer alignment to the target, receive higher weights.

4.2 Experimental Results (Table 2, Appendix D)

In comparison to single drafting, IbED demonstrates superior block efficiency across most datasets, exhibiting not only improved average performance but also consistent enhancement across all datasets. For LLaMA, TC is highlighted for 6 out of 7 benchmark datasets. For LLaVA-1.5, MT and MC are highlighted for 6 out of 7 and 5 out of 7 datasets respectively. When ensembling three or four drafting methods, MTC, MTP, MTCP are highlighted for all benchmark datasets. Fig. 1 shows how IbED attains superior performance by integrating the strengths of drafting methods.

OOD Datasets IbED matches or outperforms single drafting methods, even with a significantly weaker M. Despite M having only half the block efficiency compared to T, C, and P in MTCP, an ensemble with equal weighting achieved the best performance among all methods (highlighted for all OOD datasets). This finding highlights IbED's robustness, even without weight adjustments for M's reduced OOD efficiency. Furthermore, MTCP* (with test-time adaptive ensemble weights) significantly improves OOD performance by distinguishing between strong and weak drafting methods, optimally assigning weights, and enhancing robustness.

Summary MT and MTP show superior performance compared to M, T, and P, and their simple implementation demonstrates the practical usability of IbED. Moreover, adding caption drafting (MC, MTC, and MTCP) further improves performance. Notably, MTCP and MTCP* consistently show the best block efficiency compared to M, T, C, and P across all benchmark and OOD datasets.

5 Conclusion

We analyze the fundamentals of SD for LVLMs through an extensive benchmark evaluation and investigate various drafting methods that remained unexplored. We introduce IbED, which combines probability distributions from multiple drafting methods, and achieve consistent performance improvements across diverse tasks, including OOD tasks, demonstrating the robustness of IbED.

REFERENCES

Anthropic. The Claude 3 Model Family: Opus, Sonnet, Haiku. 2024. URL https://api.semanticscholar.org/CorpusID:268232499.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Yingshan Chang, Mridu Narang, Hisami Suzuki, Guihong Cao, Jianfeng Gao, and Yonatan Bisk. Webqa: Multihop and multimodal qa. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16495–16504, 2022.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Jian Chen, Vashisth Tiwari, Ranajoy Sadhukhan, Zhuoming Chen, Jinyuan Shi, Ian En-Hsu Yen, and Beidi Chen. MagicDec: Breaking the Latency-Throughput Tradeoff for Long Context Generation with Speculative Decoding. *arXiv preprint arXiv:2408.11049*, 2024a.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. *arXiv preprint arXiv:2403.06764*, 2024b.

Yangyi Chen, Karan Sikka, Michael Cogswell, Heng Ji, and Ajay Divakaran. Dress: Instructing large vision-language models to align and interact with humans via natural language feedback, 2024c. URL https://arxiv.org/abs/2311.10081.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, 2019.

Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pp. 1–15. Springer, 2000.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Yao Fu. Challenges in Deploying Long-Context Transformers: A Theoretical Peak Performance Analysis. *arXiv preprint arXiv:2405.08944*, 2024.

Mukul Gagrani, Raghavv Goel, Wonseok Jeon, Junyoung Park, Mingu Lee, and Christopher Lott. On Speculative Decoding for Multimodal Large Language Models. *arXiv preprint arXiv:2404.08856*, 2024.

Mudasir A Ganaie, Minghui Hu, Ashwani Kumar Malik, Muhammad Tanveer, and Ponnuthurai N Suganthan. Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 115:105151, 2022.

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

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, et al. HallusionBench: an advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14375–14385, 2024.

Ting-Hao Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. Visual storytelling. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pp. 1233–1239, 2016.

Doohyuk Jang, Sihwan Park, June Yong Yang, Yeonsung Jung, Jihun Yun, Souvik Kundu, Sung-Yub Kim, and Eunho Yang. Lantern: Accelerating visual autoregressive models with relaxed speculative decoding. *arXiv preprint arXiv:2410.03355*, 2024.

Harsh Jhamtani and Taylor Berg-Kirkpatrick. Learning to Describe Differences Between Pairs of Similar Images. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Yizhang Jin, Jian Li, Yexin Liu, Tianjun Gu, Kai Wu, Zhengkai Jiang, Muyang He, Bo Zhao, Xin Tan, Zhenye Gan, Yabiao Wang, Chengjie Wang, and Lizhuang Ma. Efficient Multimodal Large Language Models: A Survey, 2024. URL https://arxiv.org/abs/2405.10739.

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

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-OneVision: Easy Visual Task Transfer. *arXiv preprint arXiv:2408.03326*, 2024a.

Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. LLaVA-NeXT-Interleave: Tackling Multi-image, Video, and 3D in Large Multimodal Models. *arXiv preprint arXiv:2407.07895*, 2024b.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pp. 12888–12900. PMLR, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

Yitong Li, Zhe Gan, Yelong Shen, Jingjing Liu, Yu Cheng, Yuexin Wu, Lawrence Carin, David Carlson, and Jianfeng Gao. Storygan: A sequential conditional gan for story visualization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6329–6338, 2019.

Zhihang Lin, Mingbao Lin, Luxi Lin, and Rongrong Ji. Boosting Multimodal Large Language Models with Visual Tokens Withdrawal for Rapid Inference. *arXiv preprint arXiv:2405.05803*, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. In *Advances in Neural Information Processing Systems*, 2023.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26296–26306, 2024a.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xucheng Yin, Cheng lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: On the hidden mystery of ocr in large multimodal models, 2024b. URL https://arxiv.org/abs/2305.07895.

Zuyan Liu, Benlin Liu, Jiahui Wang, Yuhao Dong, Guangyi Chen, Yongming Rao, Ranjay Krishna, and Jiwen Lu. Efficient Inference of Vision Instruction-Following Models with Elastic Cache. *arXiv preprint arXiv:2407.18121*, 2024c.

Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A Benchmark for Question Answering about Charts with Visual and Logical Reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2263–2279, 2022.

Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Floris Weers, et al. Mm1: Methods, analysis & insights from multimodal llm pre-training. *arXiv preprint arXiv:2403.09611*, 2024.

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

OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=TG8KACxEON.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *arXiv preprint arXiv:2403.15388*, 2024.

Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. Textcaps: a dataset for image captioning with reading comprehension, 2020. URL https://arxiv.org/abs/2003.12462.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8317–8326, 2019.

Shezheng Song, Xiaopeng Li, Shasha Li, Shan Zhao, Jie Yu, Jun Ma, Xiaoguang Mao, and Weimin Zhang. How to bridge the gap between modalities: A comprehensive survey on multimodal large language model, 2023. URL https://arxiv.org/abs/2311.07594.

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

Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. SpecTr: Fast Speculative Decoding via Optimal Transport, 2024b. URL https://arxiv.org/abs/2310.15141.

Yao Teng, Han Shi, Xian Liu, Xuefei Ning, Guohao Dai, Yu Wang, Zhenguo Li, and Xihui Liu. Accelerating auto-regressive text-to-image generation with training-free speculative jacobi decoding. *arXiv preprint arXiv:2410.01699*, 2024.

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

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

Zhongwei Wan, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. LOOK-M: Look-Once Optimization in KV Cache for Efficient Multimodal Long-Context Inference. *arXiv preprint arXiv:2406.18139*, 2024.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021.

Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7201–7211, 2022.

Jiayang Wu, Wensheng Gan, Zefeng Chen, Shicheng Wan, and Philip S. Yu. Multimodal Large Language Models: A Survey, 2023. URL https://arxiv.org/abs/2311.13165.

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4818–4829, 2024.

Sen Yang, Shujian Huang, Xinyu Dai, and Jiajun Chen. Multi-Candidate Speculative Decoding. *arXiv preprint arXiv:2401.06706*, 2024.

Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A Survey on Multimodal Large Language Models, 2024. URL https://arxiv.org/abs/2306.13549.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid Loss for Language Image Pre-Training, 2023. URL https://arxiv.org/abs/2303.15343.

Duzhen Zhang, Yahan Yu, Jiahua Dong, Chenxing Li, Dan Su, Chenhui Chu, and Dong Yu. MM-LLMs: Recent Advances in MultiModal Large Language Models, 2024a. URL https://arxiv.org/abs/2401.13601.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. *Advances in Neural Information Processing Systems*, 36, 2024b.

Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. DistillSpec: Improving Speculative Decoding via Knowledge Distillation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=rsY6J3ZaTF.

Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. CRC press, 2012.

Appendix

We structure our supplementary material as follows. We review the related work in Appendix A and present the preliminaries in Appendix B. In Appendix C, we evaluate three draft models other than the primary draft model. In Appendix D, we evaluate IbED's performance under different ensemble weightings. In Appendix E, we present the full results of pooled multimodal drafting for both fine-tuned and non-fine-tuned draft models across various pooling rates. In Appendix F, we evaluate different models and prompts for captioning. In Appendix G, we present quantitative evaluations of and qualitative samples produced by our target model. In Appendix H, we supply the experiments for the Remarks in the main text. In Appendix I, we introduce the training details. In Appendix J, we describe the prompt formats for each dataset and drafting method. In Appendix K, we describe each dataset in detail.

A Related Work

A.1 Large Vision Language Models

LVLMs Frontier proprietary LVLMs (OpenAI, 2023; Anthropic, 2024; Gemini Team Google: Anil et al., 2023) demonstrate state-of-the-art performance across multimodalities beyond just text. Meanwhile, open-source models like the LLaVA series (Liu et al., 2023; 2024a; Li et al., 2024b;a) and LLaMA 3.2 (Dubey et al., 2024) are also rapidly advancing. While various methods exist for embedding image inputs (Yin et al., 2024; Jin et al., 2024), one of the most prominent approaches, LLaVA, employs an off-the-shelf vision encoder (Radford et al., 2021; Zhai et al., 2023) and a trainable projector to convert each image into several hundred visual context tokens of an LLM.

Approximate Inference To address the inefficiency of handling visual tokens from images, several approaches have been proposed based on a common finding: only a sparse subset of the hundreds of visual tokens is important, allowing for reduced computational cost with minimal information loss. Shang et al. (2024); Chen et al. (2024b); Lin et al. (2024) dynamically prune significant visual tokens based on attention sparsity. Further focusing on reducing redundant key-value caches, Liu et al. (2024c); Wan et al. (2024) retain key-value vectors by merging or discarding less critical caches during inference. However, from a latency perspective, these approaches primarily benefit the prefilling stage while providing limited advantages for the decoding stage.

A.2 Speculative Decoding

SD for LLMs SD accelerates LLM inference using a small draft model while preserving the target model's output distribution (Leviathan et al., 2023; Chen et al., 2023). To improve the drafting phase, various efforts have been made, including generating multiple draft candidates (Miao et al., 2023; Sun et al., 2024b; Yang et al., 2024), and fine-tuning the draft model with knowledge distillation (Zhou et al., 2024). Some studies address cases with exceptionally long prefill lengths (e.g., 100k), which significantly affect decoding efficiency (Sun et al., 2024a; Chen et al., 2024a).

SD for LVLMs Gagrani et al. (2024) is the only prior work that studied SD for LVLMs.[1] They introduced text-only drafting and claiming its performance is comparable to multimodal drafting. However, their benchmark results and detailed analysis of each drafting were limited, and they did not address how to best use multimodal information for improved drafting. Furthermore, whether or not one can effectively use multiple drafting methods remains unclear.

A.3 Ensemble Learning

Ensemble learning (Dietterich, 2000; Ganaie et al., 2022) is a powerful technique that combines the predictions of multiple models to improve overall performance. While small models may individually suffer from limited capacity and high bias, their ensemble can reduce both the bias and variance in predictions (Zhou, 2012).

[1] Jang et al. (2024) and Teng et al. (2024) propose using SD for accelerating text-to-image generative models, which are different from LVLMs.

B PRELIMINARIES

B.1 THEORETICAL LATENCY OF TRANSFORMERS

The latency bottlenecks in Transformers Vaswani et al. (2017) are classified as compute-bound or memory-bound. Compute-bound operations are limited by processing speed, which include matrix multiplication and attention. Memory-bound scenarios arise when available memory becomes a limiting factor, often due to large model weights or long input sequences. The bottleneck in effect depends on the inference phase, model architecture, hardware specifications, and other factors.

Prefilling Stage Since prefilling requires parallel computations for a large number of tokens, it is compute-bound, leading to significant increases in latency as the prefill length grows. In the case of LVLMs, the proportion of visual tokens within the prefill length is significantly large. Therefore, addressing the redundancy of visual tokens is essential for cost-efficient prefilling (Shang et al., 2024; Chen et al., 2024b; Lin et al., 2024).

Decoding Stage Predicting one next token is usually *not compute-bound*, and per-token decoding latency remains approximately constant unless the context length is very large. Thus, one can verify multiple next tokens in parallel (either for a sequence with multiple draft tokens given as input or for multiple sequences in a batch) with minimal impact on latency (Chen et al., 2024a; Fu, 2024). Following Chen et al. (2024a), for a given batch size B and a sequence length S, let $T(B, S, 1)$ denote the time to decode a single token and $T(B, S, \gamma)$ the time to verify γ tokens in parallel. Under moderate S (*e.g.*, $S \leq 3$k) and sufficiently small B (*e.g.*, $B \leq 4$) and γ (*e.g.*, $\gamma \leq 10$), the decoding phase displays the following observations (Chen et al., 2024a; Fu, 2024), where $\Delta T = T_{\text{max}} - T_{\text{min}}$ denotes the maximum time difference across the varying parameter in each remark:

Remark 1. *For given B and S, regardless of γ, $T(B, S, \gamma)$ remains approximately constant (*e.g.*, $|\Delta T/T| < 0.05$).*

Remark 2. *For a given B, regardless of S, $T(B, S, 1)$ remains approximately constant (*e.g.*, $|\Delta T/T| < 0.05$).*

Remark 3. *For a given S, regardless of B, $T(B, S, 1)$ remains approximately constant (*e.g.*, $|\Delta T/T| < 0.05$).*

Note that the magnitude of the relative difference $|\Delta T/T|$ depends on various factors, such as model architecture, model size, and hardware specifications. We empirically demonstrate Remarks 1 to 3 in Appendix H.

B.2 SPECULATIVE DECODING

Algorithm Following (Leviathan et al., 2023; Zhou et al., 2024), let M_p be the target model whose inference we aim to accelerate, and let M_q be the draft model for the same task. For a given prefix x, generated sequence $y_{<t}$, chunk length γ, and $n = 0, \ldots, \gamma - 1$, the following steps are repeated until either an <EOS> token is accepted or the maximum sequence length is reached:

1. The *Drafting Phase*, where M_q sequentially generates γ draft tokens from $q(y_{t+n}|x, y_{<t+n})$.
2. The *Verification Phase*, where M_p reviews these draft tokens in parallel, comparing them to $p(y_{t+n}|x, y_{<t+n})$.
3. For sampling, each token y_{t+n} is sequentially accepted with probability $\min\left(1, \frac{p(y_{t+n}|x, y_{<t+n})}{q(y_{t+n}|x, y_{<t+n})}\right)$. If any token is rejected before the end of the block, subsequent tokens are discarded, and the rejected token is resampled from the adjusted distribution $\text{norm}(\max(0, p(y) - q(y)))$.[2]

Block Efficiency and Wall-clock Time Improvement Given input, the *block efficiency* $\tau_{p,q}(\gamma)$ is defined as the expected number of accepted tokens per block. Let $T_p(B, S, 1)$ and $T_q(B, S, 1)$

[2]Whenever the prefix $(x, y_{<t})$ is clear from the context, we'll use $p(y)$ and $q(y)$ to denote $p(y_t|x, y_{<t})$ and $q(y_t|x, y_{<t})$, respectively.

denote the time required for M_p and M_q, to decode a single token, and $T_p(B, S, \gamma)$ denote the time required for M_p to verify γ tokens in parallel. For brevity, we use the simplified notations T_p, T_q, and $T_p(\gamma)$, omitting B and S. The required time per block in SD, denoted as T_{SD}, can be approximated as $T_{\text{SD}} = \gamma \cdot T_q + T_p(\gamma) \approx \gamma \cdot T_q + T_p$ by Remark 1. The *token rate* is defined as the number of tokens generated per unit time. SD's wall-clock time improvement can be expressed as the *token rate ratio:*

$$\frac{\text{Token rate (SD)}}{\text{Token rate (target)}} = \frac{\tau_{p,q}(\gamma)/T_{\text{SD}}}{1/T_p} \approx \frac{\tau_{p,q}(\gamma)}{\gamma \cdot \frac{T_q}{T_p} + 1} \tag{1}$$

Both the block efficiency $\tau_{p,q}(\gamma)$ and the *draft-to-target latency ratio* $\frac{T_q}{T_p}$ are determined by the choice of M_q, assuming M_p is fixed. Remarks 2 and 3 imply the following:

Remark 4. *For a given γ, regardless of B and S, $T_{SD}/T_p = \gamma \cdot \frac{T_q}{T_p} + 1$ remains nearly identical.* (e.g.*if we assume $T_q/T_p = 0.05$ and $\gamma = 5$, $|\Delta T_{SD}/T_{SD}| < 0.01$*).

Remark 4 shows that the wall-clock time improvement in Eq. (1) becomes proportional solely to the value of $\tau_{p,q}(\gamma)$, since its denominator $\gamma \cdot \frac{T_q}{T_p} + 1$ is constant. Moreover, when measuring the actual wall-clock time, precise performance comparison becomes challenging due to potential noise from various factors such as hardware variations. Therefore, we utilize block efficiency $\tau_{p,q}(\gamma)$ to accurately evaluate the performance of speculative decoding.

C IbED with Different Draft Models

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
LLaVA-1.5	160M	M	2.59	2.40	**2.56**	2.75	2.66	2.77	2.23	1.29	1.27
		T	2.49	2.18	2.40	2.70	2.63	3.06	2.73	2.28	2.31
		C	2.54	2.27	2.45	2.73	2.67	3.06	2.75	2.48	2.46
		P	2.55	2.29	2.51	2.74	2.77	3.06	2.78	**2.53**	**2.59**
		MT	2.59	2.38	2.55	2.75	2.78	3.00	2.74	2.18	2.15
		MC	**2.64**	**2.42**	2.55	**2.79**	2.80	3.03	2.73	2.31	2.25
		MTC	2.61	2.39	2.54	2.77	2.78	3.07	2.80	2.40	2.36
		MTP	2.58	2.37	2.53	2.75	2.81	3.03	2.79	2.44	2.42
		MTCP	2.59	2.38	2.53	2.77	**2.82**	**3.08**	**2.81**	2.50	2.49
LLaVA-1.5	290M	M	1.52	1.70	**1.97**	1.55	2.10	1.46	1.94	2.07	2.21
		T	1.52	1.60	1.83	1.54	2.02	1.43	1.98	1.90	1.94
		C	1.51	1.62	1.84	1.55	2.06	1.44	1.99	1.93	2.03
		P	1.53	1.67	1.93	1.57	2.12	1.43	2.05	**2.11**	2.23
		MT	**1.55**	1.69	**1.97**	1.57	2.15	1.47	**2.09**	2.06	2.17
		MC	**1.55**	**1.71**	**1.97**	**1.58**	2.15	**1.48**	**2.09**	2.10	**2.24**
		MTC	**1.55**	1.69	1.95	1.57	2.14	1.47	**2.09**	2.03	2.16
		MTP	**1.55**	1.70	**1.97**	**1.58**	2.15	1.46	2.08	**2.11**	2.22
		MTCP	1.54	1.68	1.94	1.57	**2.16**	1.46	**2.09**	2.07	2.20
LLaVA-OV	68M	M	2.18	1.90	2.05	2.31	2.12	2.31	2.15	1.77	1.74
		T	2.18	1.86	2.05	2.32	2.11	**2.68**	2.25	1.90	1.83
		C	2.20	1.93	2.07	2.33	2.11	2.66	2.26	1.86	1.83
		P	2.19	1.89	2.04	2.34	2.13	2.58	2.27	2.00	1.90
		MT	2.22	1.94	2.09	**2.35**	2.22	2.64	2.32	1.96	1.91
		MC	**2.24**	**1.99**	**2.11**	2.34	2.20	2.64	2.32	1.95	1.90
		MTC	**2.24**	1.97	2.10	**2.35**	2.22	2.67	**2.35**	1.98	1.92
		MTP	2.22	1.93	2.08	**2.35**	2.21	2.65	2.33	**2.01**	**1.96**
		MTCP	2.23	1.96	2.10	**2.35**	**2.23**	2.67	**2.35**	2.00	1.95

Table 3: Evaluation of two larger LLaVA-1.5 draft models (160M and 290M), obtained through standard visual instruction tuning, and a same-sized LLaVA-OV draft model (68M), obtained through multi-image-aware fine-tuning. Results of IbED are highlighted when they show the best performance among all constituent methods (e.g., when MT performs best compared to M and T).

In this section, we evaluate the performance of our IbED for three different choices of the draft model: LLaVA-1.5 160M and LLaVA-1.5 290M (both fine-tuned with the same recipe as LLaVA-1.5 68M), and LLaVA-OV (the same architecture as LLaVA-1.5 68M but fine-tuned with the multi-

image-aware OneVision recipe). The full results are presented in Table 3. For each model, most of the MTCP ensemble results are highlighted.

D IbED with Non-Uniform Weights

In this section, we investigate whether tuning the ensemble weights can enhance IbED. We consider sweeping over weights (Appendix D.1), test-time adaptive weights (Appendix D.2), and learnable weights (Appendix D.3).

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
LLaVA-1.5	68M	M	2.24	2.12	2.26	2.39	2.34	2.51	1.96	1.19	1.16
		T	2.22	2.03	2.20	2.34	2.27	2.77	2.34	2.05	2.05
		C	2.28	2.08	2.24	2.41	2.31	2.77	2.36	2.08	2.10
		P	2.23	2.08	2.26	2.36	2.23	2.73	2.27	2.07	2.09
		MT (1:1)	2.26	2.13	2.27	2.39	2.40	2.75	2.37	1.94	1.91
		MT (2:1)	2.26	2.13	2.29	2.39	2.40	2.69	2.29	1.76	1.69
		MT (3:1)	2.26	2.13	2.28	2.40	2.38	2.65	2.21	1.63	1.54
		MT (4:1)	2.26	2.14	2.28	2.40	2.36	2.64	2.16	1.54	1.45
		MC (1:1)	**2.30**	**2.17**	2.29	**2.42**	2.39	2.74	2.35	1.99	1.93
		MC (2:1)	**2.30**	**2.17**	**2.30**	2.41	2.39	2.70	2.27	1.80	1.71
		MC (3:1)	2.29	2.16	2.29	2.40	2.38	2.66	2.20	1.66	1.56
		MC (4:1)	2.28	2.16	2.29	2.40	2.37	2.64	2.16	1.56	1.46
		MTC (1:1:1)	2.29	2.15	2.28	2.41	**2.41**	**2.79**	**2.40**	2.08	2.06
		MTC (2:1:1)	2.29	**2.17**	2.29	**2.42**	**2.41**	2.73	2.38	1.99	1.96
		MTC (3:1:1)	2.28	**2.17**	2.29	2.41	2.39	2.69	2.34	1.90	1.83
		MTC (4:1:1)	2.28	2.16	2.29	2.40	2.39	2.67	2.29	1.80	1.71
		MTP (1:1:1)	2.26	2.13	2.27	2.39	2.39	2.77	2.37	2.02	2.06
		MTP (2:1:1)	2.26	2.15	2.27	2.40	2.39	2.74	2.33	1.94	1.93
		MTP (3:1:1)	2.26	2.15	2.28	2.40	2.38	2.70	2.29	1.85	1.80
		MTP (4:1:1)	2.26	2.15	2.29	2.40	2.38	2.68	2.25	1.76	1.70
		MTCP (1:1:1:1)	2.28	2.16	2.28	2.41	2.40	2.78	2.39	**2.10**	**2.13**
		MTCP (2:1:1:1)	2.28	**2.17**	2.29	2.41	2.40	2.76	2.37	1.90	1.81
		MTCP (3:1:1:1)	2.28	2.16	2.29	2.40	2.40	2.75	2.35	1.79	1.70
		MTCP (4:1:1:1)	2.28	2.16	2.29	2.40	2.39	2.72	2.33	1.71	1.62

Table 4: Block efficiency results of IbED for various weights. Results of IbED are highlighted when they show the best performance among all constituent methods (e.g., when MT performs best compared to M and T). The ensemble results are consistently highlighted across the weight variations.

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
LLaVA-1.5	68M	M	2.24	2.12	2.26	2.39	2.34	2.51	1.96	1.19	1.16
		T	2.22	2.03	2.20	2.34	2.27	2.77	2.34	2.05	2.05
		C	2.28	2.08	2.24	2.41	2.31	2.77	2.36	2.08	2.10
		P	2.23	2.08	2.26	2.36	2.23	2.73	2.27	2.07	2.09
		MTCP (1:1:1:1)	2.28	**2.16**	2.28	2.41	**2.40**	2.78	2.39	2.10	2.13
		MTCP (kld-1)	**2.29**	**2.16**	2.28	**2.42**	2.39	2.78	**2.42**	2.12	2.15
		MTCP (kld-4)	**2.29**	**2.16**	2.28	**2.42**	2.39	2.78	**2.42**	2.12	**2.17**
		MTCP (kld-16)	2.28	**2.16**	2.28	2.41	2.38	**2.79**	**2.42**	2.12	**2.17**
		MTCP (kld-full)	2.28	**2.16**	2.28	2.41	2.39	**2.79**	**2.42**	**2.13**	2.16
		MTCP (tvd-1)	**2.29**	**2.16**	2.28	**2.42**	2.39	2.78	2.41	2.11	2.14
		MTCP (tvd-4)	2.28	**2.16**	**2.29**	**2.42**	2.39	2.78	2.41	2.12	2.15
		MTCP (tvd-16)	2.28	**2.16**	2.28	2.41	2.39	2.78	**2.42**	2.12	2.15
		MTCP (tvd-full)	2.28	**2.16**	2.28	2.41	2.39	**2.79**	**2.42**	2.12	2.15

Table 5: Block efficiency results of IbED with test-time adaptive ensemble weights. For example, MTCP (kld-4) indicates that the test-time adaptive ensemble weights were computed using KLD with a window size of 4. Results of IbED are highlighted when they show the best performance among all constituent methods (e.g., when MT performs best compared to M and T).

D.1 Sweeping over Ensemble Weights

As a first step, we evaluate IbED over a fixed set of ensemble weights. Specifically, we vary the weight of multimodal drafting (M) from 1 to 4, while keeping the other weights at unity. As shown

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
		M	2.24	2.12	2.26	2.39	2.34	2.51	1.96	1.19	1.16
		T	2.22	2.03	2.20	2.34	2.27	2.77	2.34	2.05	2.05
		C	2.28	2.08	2.24	2.41	2.31	2.77	2.36	2.08	2.10
		P	2.23	2.08	2.26	2.36	2.23	2.73	2.27	2.07	2.09
		MTCP (1:1:1:1)	2.28	2.16	2.28	2.41	**2.40**	2.78	2.39	2.10	2.13
		MTCP (kld-1-0)	2.26	2.14	2.27	**2.42**	2.31	2.72	2.33	2.04	2.07
		MTCP (kld-1-0.25)	2.28	2.16	2.28	**2.42**	2.37	2.76	2.39	2.10	2.14
		MTCP (kld-1-0.5)	**2.29**	2.16	**2.29**	**2.42**	2.38	2.78	2.40	2.12	2.15
		MTCP (kld-1-0.75)	**2.29**	2.16	2.28	**2.42**	2.38	2.78	2.40	2.12	2.15
		MTCP (kld-1-1)	**2.29**	2.16	2.28	**2.42**	2.39	2.78	**2.42**	2.12	2.15
		MTCP (kld-4-0)	2.27	2.12	2.27	2.40	2.32	2.72	2.37	2.08	2.06
		MTCP (kld-4-0.25)	2.28	**2.17**	**2.29**	**2.42**	2.37	2.76	2.40	2.12	2.14
LLaVA-1.5	68M	MTCP (kld-4-0.5)	2.28	**2.17**	2.28	**2.42**	2.39	2.78	2.40	2.12	2.16
		MTCP (kld-4-0.75)	**2.29**	**2.17**	2.28	**2.42**	2.39	2.78	2.40	2.12	2.15
		MTCP (kld-4-1)	**2.29**	2.16	2.28	**2.42**	2.39	2.78	**2.42**	2.12	**2.17**
		MTCP (kld-16-0)	2.26	2.15	2.26	2.41	2.31	2.72	2.36	2.08	2.08
		MTCP (kld-16-0.25)	**2.29**	**2.17**	2.28	**2.42**	2.38	2.78	2.40	**2.13**	2.15
		MTCP (kld-16-0.5)	**2.29**	2.16	2.28	**2.42**	2.39	**2.79**	2.40	**2.13**	2.16
		MTCP (kld-16-0.75)	2.28	**2.17**	2.28	2.41	2.39	**2.79**	2.40	**2.13**	2.16
		MTCP (kld-16-1)	2.28	2.16	2.28	2.41	2.38	**2.79**	**2.42**	2.12	**2.17**
		MTCP (kld-full-0)	2.25	2.14	2.27	2.41	2.31	2.72	2.35	2.08	2.08
		MTCP (kld-full-0.25)	**2.29**	2.16	2.28	**2.42**	2.37	**2.79**	2.39	**2.13**	2.15
		MTCP (kld-full-0.5)	2.28	2.16	2.28	**2.42**	2.39	**2.79**	2.40	**2.13**	2.16
		MTCP (kld-full-0.75)	2.28	**2.17**	2.28	**2.42**	2.39	**2.79**	2.40	**2.13**	2.16
		MTCP (kld-full-1)	2.28	2.16	2.28	2.41	2.39	**2.79**	**2.42**	**2.13**	2.16

Table 6: Block efficiency results of IbED with test-time adaptive ensemble weights. For example, MTCP (kld-4-0.5) indicates that the test-time adaptive ensemble weights were computed using KLD with a window size of 4 and a temperature of 0.5. Results of IbED are highlighted when they show the best performance among all constituent methods (e.g., when MT performs best compared to M and T).

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
		M	2.24	2.12	2.26	2.39	2.34	2.51	1.96	1.19	1.16
		T	2.22	2.03	2.20	2.34	2.27	2.77	2.34	2.05	2.05
		C	2.28	2.08	2.24	2.41	2.31	2.77	2.36	2.08	2.10
		P	2.23	2.08	2.26	2.36	2.23	2.73	2.27	2.07	2.09
LLaVA-1.5	68M	MTCP (1:1:1:1)	2.28	2.16	2.28	2.41	2.40	2.78	2.39	2.10	2.13
		MTCP (logit-nll)	**2.29**	2.16	**2.29**	**2.42**	2.39	2.79	2.38	2.08	2.11
		MTCP (logit-kld)	**2.29**	2.16	2.27	2.41	2.39	**2.80**	2.38	2.09	2.12
		MTCP (probability-nll)	**2.29**	2.14	2.26	**2.42**	2.31	2.79	2.39	1.52	1.59
		MTCP (probability-kld)	**2.29**	**2.18**	**2.29**	2.41	**2.42**	**2.80**	**2.40**	**2.13**	**2.15**

Table 7: Block efficiency results of IbED with learnable ensemble weights. Results of IbED are highlighted when they show the best performance among all constituent methods (e.g., when MT performs best compared to M and T). MTCP (probability-kld) consistently demonstrates the best performance among the various ensemble weights tested.

in Table 4, IbED demonstrates consistent performance across the weights from 1 to 4. The uniform choice without any prior knowledge—1:1:1:1—performs moderately well.

D.2 Test-Time Adaptive Ensemble Weights

We then present the experimental results for test-time adaptive ensemble weights. Specifically, we examine the effects of using Kullback-Leibler Divergence (KLD) and Total Variation Distance (TVD) while varying the window size l for calculating $e_t^{(i)}$ (Table 5), and adjusting the temperature during weight normalization (Table 6). To apply temperature τ to the weight W, we use $e^{\log(W)/\tau}$. For example, MTCP (kld-4-0.5) indicates that the test-time adaptive ensemble weights were computed using KLD, a window size of 4, and a temperature of 0.5. Based on the results, we selected MTCP (kld-full-1) as our baseline.

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
LLaVA-1.5	68M	M	0.392	0.502	0.577	0.348	0.393	0.178	0.117	0.007	0.005
		T	0.316	0.305	0.341	0.307	0.227	0.500	0.365	0.215	0.281
		C	0.258	0.157	0.079	0.336	0.297	0.146	0.314	0.224	0.260
		P	0.034	0.036	0.003	0.009	0.083	0.176	0.204	0.554	0.454

Table 8: Learned ensemble weights for MTCP (probability-kld) in Table 7. Different weights are assigned to each of the M, T, C, and P, depending on the dataset characteristics. The weights are normalized using the softmax function.

D.3 Learnable Ensemble Weights

Beyond the use of non-learnable weights, we further investigate whether adapting weights through learning for each draft can improve IbED. To explore this, we trained parameters corresponding to the number of drafts (e.g., four parameters for the MTCP ensemble), varying the space and label type used for ensembling. During inference with speculative decoding, we performed inference by linearly combining each draft using the pre-trained parameters. To train these parameters, we prepared 200 samples for each dataset that are entirely separate from the inference stage. For each sample x, we obtained two types of labels using the target model M_p: (1) hard labels, which are sequences sampled from $p(y|x)$, and (2) soft labels, which is probability defined as $p(\cdot|x, y_{<t})$ for each timestep t.

When ensembling in the probability space, it is analogous to the case where the weight list W in Algorithm 1 is not fixed at equal proportions but is instead a learned parameter normalized via Softmax. Conversely, when ensembling in the logit space, the logits from the target model M_p are linearly combined using the learned parameters and W (which is not normalized), and the ensembled probabilities are obtained by applying softmax to these combined logits.

Consequently, Table 7 demonstrates that learning with soft labels outperformed hard labels, and ensembling in the probability space yielded better results than ensembling in the logit space. Notably, Table 8 examines the normalized weights W in the context of soft labels and ensembling in the probability space. It shows that the magnitude of each weight tends to be proportional to the block efficiency of each drafting in Table 7. This indicates that the learned weights can effectively filter out drafts with suboptimal block efficiency.

E Full results of Pooled Multimodal Drafting

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
LLaVA 1.5	68M	Multimodal	2.24	**2.12**	**2.26**	**2.39**	2.34	2.51	1.96	1.19	1.16
		Text-only	2.22	2.03	2.20	2.34	2.27	2.77	2.34	2.05	2.05
		Pool (144)	2.23	2.08	**2.26**	2.36	2.23	2.73	2.27	2.07	2.09
		Pool (144, ft)	**2.26**	2.09	**2.26**	**2.39**	2.38	**2.80**	2.33	**2.27**	**2.27**
		Pool (36)	2.17	2.01	2.21	2.32	2.20	2.73	2.25	2.05	2.06
		Pool (36, ft)	2.22	2.06	2.25	2.38	2.36	2.78	2.34	2.19	2.23
		Pool (9)	2.20	2.03	2.21	2.34	2.25	2.74	2.30	2.06	2.08
		Pool (9, ft)	2.23	2.05	2.22	2.37	2.37	2.79	**2.35**	2.18	2.21
		Pool (1)	2.23	2.03	2.23	2.37	2.25	2.76	2.34	2.06	2.07
		Pool (1, ft)	2.23	2.06	2.21	2.37	**2.39**	2.78	**2.35**	2.21	2.22

Table 9: Block efficiency results with different pooling rates. Applying image pooling during both fine-tuning and inference (denoted by Pool(n, ft)) improves performance compared to applying it during inference only (denoted by Pool(n)).

In this section, we provide the full results for pooled multimodal drafting with various pooling rates. The notation Pool (n) denotes that n represents the number of visual tokens remaining after the pooling operation. Based on the results obtained from searching across values of n (144, 36, 9, 1), we have selected Pool (144) as our default configuration for pooled multimodal drafting. To further investigate whether it helps to apply the image token pooling during fine-tuning, we separately train draft models for each of n and report the results, which are marked as Pool(n, ft).

Table 9 presents the block efficiency results for the fine-tuned and non-fine-tuned draft models across various pooling methods. The results demonstrate that the block efficiency of the fine-tuned model is higher than that of the non-fine-tuned model.

F Details and full results for Caption Drafting

Model	Type	Latency (s)		
		$n = 1$	$n = 2$	$n = 5$
Target LVLM (prefilling)	LLaVA-1.5 7B	0.112	0.207	0.540
Image Captioning	BLIP	**0.054**	**0.055**	**0.074**
	Florence-2	0.105	0.149	0.292

Table 10: Latency analysis of image captioning models. BLIP and Florence-2 captioning latencies are lower than the target LVLM's prefilling latency. Parallel processing can therefore hide captioning latency without affecting time to first token.

In this section, we describe various types of lightweight image captioning models that can be used for caption drafting (Appendix F.1). We then demonstrate that captioning model inference completes earlier than the target model's prefilling by analyzing the captioning model's latency (Appendix F.2). Lastly, we present full results of the block efficiency when each captioning model is utilized (Appendix F.3).

F.1 Captioning Models

BLIP (Li et al., 2022) A vision-language model trained on bootstrapped synthetic captions. It uses a visual transformer and the text encoder of BERT Devlin et al. (2019) to separately encode image and text.

`https://huggingface.co/Salesforce/blip-image-captioning-base`

BLIP-2 (Li et al., 2023) A vision-language model using a frozen off-the-shelf image encoder and LLM. A querying transformer trained using boostrapped data is included for cross-modal alignment.

`https://huggingface.co/Salesforce/blip2-opt-2.7b`

Florence-2 (Xiao et al., 2024) A vision-language model that is instruction-trained for a variety of tasks. Its architecture consists of a single sequence-to-sequence transformer and a vision encoder.

`https://huggingface.co/microsoft/Florence-2-large-ft`

F.2 Latency Analysis

It is important to ensure that the captioning model runs fast enough so that it does not delay drafting. In this line, we measure in Table 10 the time taken by the two captioning models, BLIP and Florence-2, to generate captions. The results demonstrate captioning completes earlier than target model's prefilling.

F.3 Additional Experimental Results

The default caption model utilized in our study is Florence-2, which also supports the generation of detailed captions. However, the latency associated with generating detailed captions is longer compared to default captions. Hence, we report the results obtained using the detailed captions from Florence-2, denoted as more detailed captions (MDC), as well as the results with the default captions (C). We additionally evaluate the performance of other off-the-shelf image captioning models such as BLIP and BLIP-2.

Table 11 presents the block efficiency results for the various image captioning models, evaluated under single-method and ensemble drafting scenarios. All captioning models are effective in improving

the block efficiency. Florence-2 shows the best results and thus is our default choice. Between the captioning strategies for Florence-2, the more detailed caption (F-2-MDC) shows slightly higher block efficiency than the default caption (F-2-C). The efficacy of ensemble drafting reproduces for all captioning models, as evidenced by highlighting on most of the MTCP ensemble results. Note that the block efficiency is higher in the ensemble result using detailed captions (F-2-MDC) compared to the case with default captions (F-2-C).

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	Method	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB	PSV	VIST
LLaVA-1.5	68M	M	2.24	2.12	2.26	2.39	2.34	2.51	1.96	1.19	1.16
		T	2.22	2.03	2.20	2.34	2.27	2.77	2.34	2.05	2.05
		C (F-2-C)	2.28	2.08	2.24	2.41	2.31	2.77	2.36	2.08	2.10
		C (F-2-MDC)	2.27	2.11	2.26	2.44	2.28	2.74	2.36	2.10	2.11
		C (BLIP)	2.23	2.02	2.23	2.40	2.28	2.75	2.37	**2.12**	2.10
		C (BLIP-2)	2.25	2.07	2.23	2.37	2.30	2.78	2.37	2.09	2.12
		MC (F-2-C)	2.30	**2.17**	2.29	2.42	2.39	2.74	2.35	1.99	1.93
		MC (F-2-MDC)	**2.31**	**2.17**	**2.30**	**2.46**	2.38	2.73	2.34	1.99	1.96
		MC (BLIP)	2.27	2.11	2.28	2.42	2.37	2.72	2.35	2.00	1.94
		MC (BLIP-2)	2.28	2.15	2.28	2.40	2.39	2.75	2.34	1.98	1.95
		MTC (F-2-C)	2.29	2.15	2.28	2.41	**2.41**	2.79	2.40	2.08	2.06
		MTC (F-2-MDC)	2.29	2.15	2.29	2.44	2.40	2.79	**2.41**	2.09	2.08
		MTC (BLIP)	2.26	2.11	2.27	2.41	2.38	2.78	2.39	2.05	2.05
		MTC (BLIP-2)	2.26	2.14	2.28	2.40	2.39	**2.81**	2.40	2.06	2.06
		MTCP (F-2-C)	2.28	2.16	2.28	2.41	2.40	2.78	2.39	2.10	2.13
		MTCP (F-2-MDC)	2.29	2.15	2.29	2.43	2.39	2.79	2.40	**2.12**	**2.15**
		MTCP (BLIP)	2.26	2.11	2.28	2.41	2.39	2.78	2.39	2.08	2.13
		MTCP (BLIP-2)	2.26	2.14	2.28	2.40	2.39	2.80	2.39	2.09	2.13

Table 11: Evaluations of block efficiency for various image captioning models and prompts under single-method and ensemble drafting scenarios. Results of IbED are highlighted when they show the best performance among all constituent methods (e.g., when MC performs best compared to M and C). Overall, Florence-2 demonstrates slightly better efficacy than BLIP and BLIP-2. The efficacy of Florence-2 captions can often be improved by the use of the more detailed captions (MDC) prompt compared to the default captions (C) prompt. Note that the benefits in block efficiency must be carefully weighed against the extra latency incurred by the generation of longer captions.

G Evaluation of Target Model

	Benchmark Datasets						
Model	ChartQA	TextVQA	VQAv2	HBench	Spot	WebQA	MB
GPT-4o	**50.0**	**73.6**	69.7	**66.0**	**19.1**	58.6	**10.5**
GPT-4o-mini	46.4	72.2	65.4	58.0	12.0	53.7	9.1
LLaVA-1.5 7B	14.3	52.9	**69.9**	27.0	13.9	**59.3**	9.6
LLaVA-1.5 7B (text-only)	3.6	9.0	38.0	20.0	11.1	48.9	8.4

Table 12: Evaluation results for the target model on benchmark datasets. Our target model, LLaVA-1.5 7B, demonstrates performance comparable to GPT-4o-mini, while GPT-4o achieves the best or second-best performance across all tasks. The injection of multimodal information in the input scenarios of the benchmark datasets benefits the target model, LLaVA-1.5 7B.

In this section, we present both quantitative results (Appendix G.1) and qualitative samples (Appendix G.2) produced by LLaVA-1.5 7B, which serves as the target model in our experimental setting.

G.1 Quantitaive Evaluation

We assess the output quality of LLaVA-1.5 7B using benchmark datasets, as shown in Table 12. The results demonstrate that the quality of LLaVA-1.5 7B is comparable to GPT-4o-mini, while GPT-4o achieves the best or second-best performance across all tasks. In all input scenarios of the benchmark datasets, the injection of multimodal information benefits the target model, LLaVA-1.5 7B, from the perspective of output quality.

G.2 Generated Samples

ChartQA [Fig. 5a]

Q: Which party won the most seats in 2010?

A: In the image, there is a chart showing the number of seats won by each party in the 2010 election. The chart indicates that the Conservative Party won the most seats with 366, followed by the Liberal Democrats with 57 seats, and the Labour Party with 295 seats. Therefore, the Conservative Party won the most seats in the 2010 election.

TextVQA [Fig. 5b]

Q: Does this restaurant serve wine?

A: Yes, the restaurant serves wine. The image shows a restaurant with a sign that reads "Zin Restaurant & Wine Bar." This suggests that the restaurant offers wine as part of its menu, and it is likely that the establishment serves a variety of wines to its customers.

VQAv2 [Fig. 5c]

Q: Is the lady excited to eat her food?

A: Yes, the lady appears to be excited to eat her food. In the image, she is smiling and holding a hot dog, which suggests that she is looking forward to enjoying her meal. Her body language and facial expression convey her enthusiasm and anticipation for the food.

HallusionBench [Fig. 5d]

Q: Is the lady excited to eat her food?

A: Yes, Switzerland is the leading importing country of gold, silverware, and jewelry with the highest import value in 2018. In the image, Switzerland is shown as the country with the highest import value in 2018, which indicates that it is the leading importing country of gold, silverware, and jewelry. This is supported by the fact that Switzerland is the only country with a value greater than $100 million for its gold, silverware, and jewelry imports.

Spot The Difference [Fig. 5e]

Q: What has changed from the first image to the second?

A: The image shows a city street with a grassy area in the middle of the road. In the first image, there are several cars parked along the street, and a few people are walking around. The street appears to be relatively empty, with no traffic or significant activity. In the second image, the street is bustling with activity. There are numerous cars parked along the street, and several people are walking around. The scene is more lively and dynamic, with a higher level of activity compared to the first image.

WebQA [Fig. 5f]

Q: Does the High Line park run through buildings?

A: Yes, the High Line park runs through buildings, as it is an elevated park built on an old rail line in the city. In the image, the park is situated under a building, showcasing the integration of urban green spaces with the city's architecture.

MagicBrush [Fig. 5g]

Q: Please provide the image edit instruction that can transfrom the source image to the target image.

A: To transform the source image into the target image, follow these steps: 1. Remove the red-haired girl from the scene, as she is not part of the target image. 2. Replace the girl with a back pack, which should be placed on the stairs. 3. Add a bus in the background, making sure it is yellow and occupies the entire width of the image. 4. Place a person in the background, standing near the bus, to give a sense of scale and context. 5. Adjust the lighting and shadows to create a more realistic scene.

PororoSV [Fig. 5h]

Q: Taking the stories from the initial images into account, can you conclude the story using the elements of the final image?

A: The image features a group of cartoon characters, including a penguin, a fox, and a pink character, standing in a snowy environment. The characters are positioned in various locations, with some standing closer to the foreground and others further back. The scene appears to be a part of a video game or animated series, as the characters are interacting with each other and their surroundings.

VIST [Fig. 5i]

Q: With the narratives paired with the initial images, how would you conclude the story using the last picture?

A: The image captures a thrilling moment during a volley ball game, with a woman leaping into the air to hit the ball. She is wearing a purple shirt and is in the process of jumping to make contact with the sports ball. The scene is filled with excitement and anticipation as the players on the court watch her performance. The volley ball game is taking place on a court, and the players are actively engaged in the match.

H Empirical Validation of Remarks

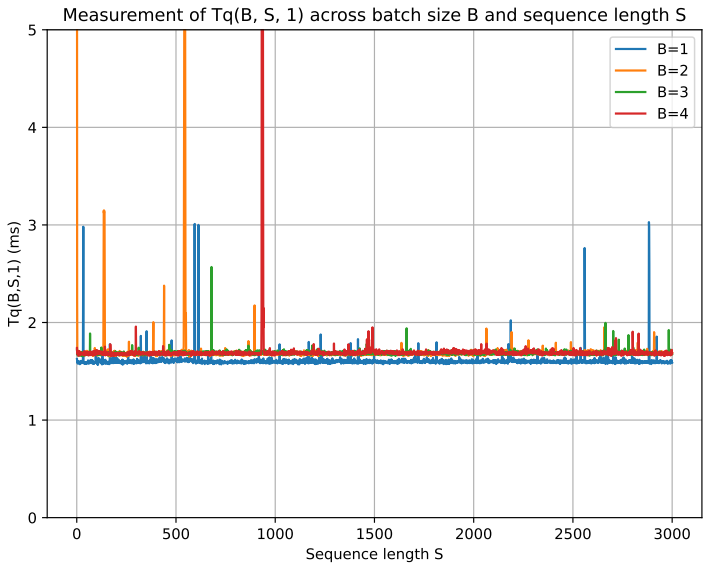


Figure 2: Empirical demonstration of Remarks 2 and 3.

In this section, we validate the Remarks from Appendix B.1 with empirical timing measurements. Focusing on the case where the remarks apply to speculative decoding settings, we use LLaVA-1.5 7B and LLaVA-1.5 68M to measure $T_p(B, S, \gamma)$ for Remark 1 and $T_q(B, S, 1)$ for Remark 2 and Remark 3, respectively. All experiments are conducted on an A100 80GB GPU using the fp16 data type for the models.

Fig. 2 shows $T_q(B, S, 1)$ in milliseconds for sequence lengths up to 3k for each batch size $B \in \{1, 2, 3, 4\}$. For moderate sequence lengths $S \leq$ 3k, T_q varies by no more than 5% for each B, which supports Remark 2. Similarly, when comparing different Bs with a fixed S, T_q varies by no more than 5%, which supports Remark 3.

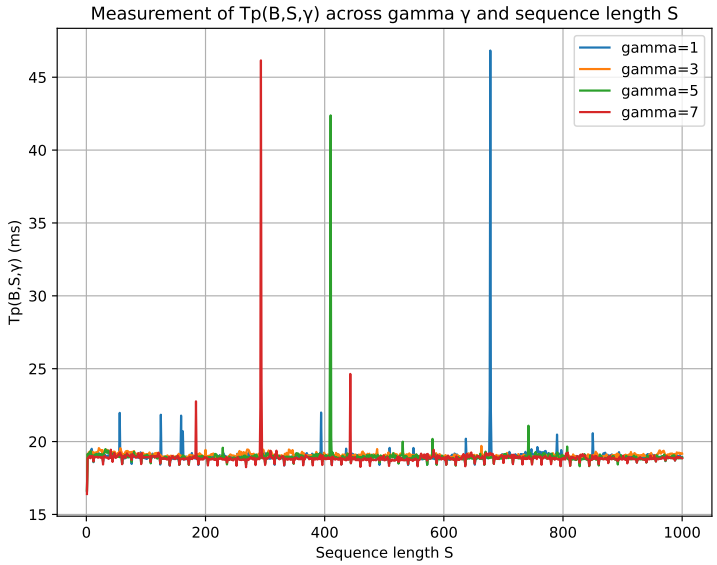


Figure 3: Empirical demonstration of Remark 1.

Fig. 3 shows $T_p(B, S, \gamma)$ in milliseconds for each $\gamma \in \{1, 3, 5, 7\}$. We test the case of B=1, which aligns with our experimental settings where the target model always performs inference on a single batch. Over the values of γ considered, T_p varies by no more than 5%.

I Training and Evaluation of Draft Models

In this section, we present a more detailed overview of our custom training procedure for the draft models (Appendix I.1). We then evaluate our primary draft model, LLaVA-1.5 68M, on multimodal tasks to ensure it has the capability to properly perceive multimodality, and we provide some qualitative samples from the draft model (Appendix I.2).

I.1 Details of Training

LLaVA-1.5 (Liu et al., 2024a) The process for developing draft models with LLaVA-1.5 (68M, 160M, and 290M) training recipe was divided into two stages: pre-training and instruction fine-tuning (IFT). Pre-training focuses on training the projector while the parameters of the LLM and vision encoder are frozen. During the IFT stage, visual instruction tuning is used to teach the LLM to follow multimodal instructions. The vision encoder remains frozen throughout both stages. The hyperparameters used for each stage are described in Table 13. We trained the draft model using datasets curated by the original author of LLaVA-1.5. For more training details, see https://github.com/haotian-liu/LLaVA/tree/main.

LLaVA-OneVision (Li et al., 2024a) The development of draft models using the LLaVA-OneVision (LLaVA-OV) training recipe was divided into three stages: language-image alignment, high-quality knowledge learning, and visual instruction tuning. In the language-image alignment stage, visual features are aligned with the word embedding space of LLMs. High-quality knowledge learning balances computational efficiency with the integration of new knowledge into LVLMs. Visual instruction tuning consists of two phases: (i) Single-Image Training, where the model learns to perform visual tasks using instructions from single images, and (ii) OneVision Training, where the model learns to execute multi-image visual tasks using a blend of video, single-image, and multi-image data. During the language-image alignment stage, only the projector for aligning visual features is updated, whereas all components including LLM are updated in the following three

Hyperparameter	Value
Training Epochs	1
Batch Size	256
Learning Rate (LR)	1e-3
LR Schedule Type	Cosine
Warm-up Ratio	0.03
Weight Decay	0.0

(a) Pretraining stage

Hyperparameter	Value
Training Epochs	1
Batch Size	128
Learning Rate (LR)	2e-5
LR Schedule Type	Cosine
Warm-up Ratio	0.03
Weight Decay	0.0

(b) Instruction fine-tuning stage

Table 13: Details of hyperparameters used in LLaVA-1.5 training

stages. We trained the draft model using datasets curated by the original author of LLaVA-OV (Li et al., 2024a). The hyperparameters used for each stage are described in Table 14, and the learning rate for the vision encoder is one-fifth of that for the LLM across all stages. For more details, visit https://github.com/LLaVA-VL/LLaVA-NeXT.

Hyperparameter	Value
Training Epochs	1
Batch Size	512
Learning Rate (LR)	1e-3
LR Schedule Type	Cosine
Warm-up Ratio	0.03
Weight Decay	0.0

(a) Image alignment stage

Hyperparameter	Value
Training Epochs	1
Batch Size	512
Learning Rate (LR)	1e-5
LR Schedule Type	Cosine
Warm-up Ratio	0.03
Weight Decay	0.0

(b) High-quality knowledge learning stage

Hyperparameter	Value
Training Epochs	1
Batch Size	512
Learning Rate (LR)	1e-5
LR Schedule Type	Cosine
Warm-up Ratio	0.03
Weight Decay	0.0

(c) Visual instruction tuning stage: Single-image training

Hyperparameter	Value
Training Epochs	1
Batch Size	512
Learning Rate (LR)	1e-5
LR Schedule Type	Cosine
Warm-up Ratio	0.03
Weight Decay	0.0

(d) Visual instruction tuning stage: OneVision training

Table 14: Details of hyperparameters used in LLaVA-OV training

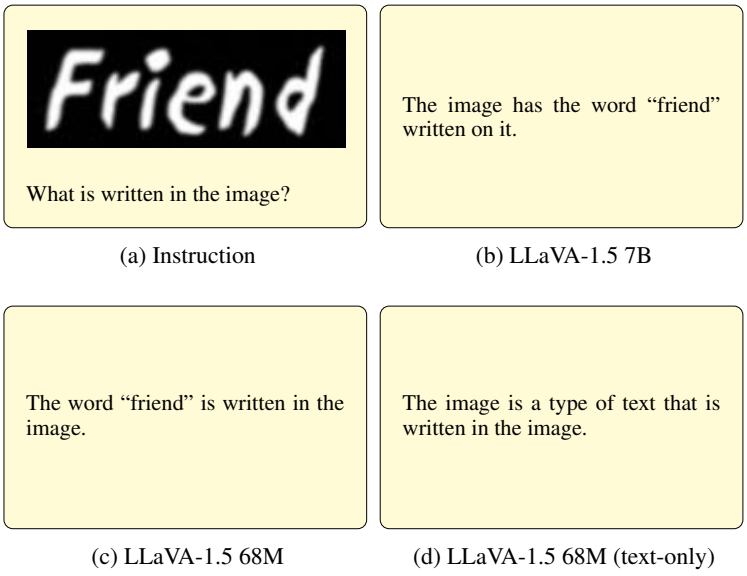
I.2 Evaluation Results

Table 15 presents the evaluation results of our primary draft model, LLaVA-1.5 68M, on OCR-Bench (Liu et al., 2024b) and TextCaps (Sidorov et al., 2020) datasets. We assess the output quality of the draft model with and without image inputs and compare the results with those of the target model, LLaVA-1.5 7B. In terms of output quality, the draft model with image inputs consistently outperforms the one without, illustrating that the injection of multimodal information benefits the custom-trained draft model.

Fig. 4 presents qualitative samples from the OCRBench dataset. Both LLaVA-1.5 7B and 68M models provided accurate responses, whereas the text-only LLaVA-1.5 68M model failed to answer correctly due to its lack of image-processing capabilities.

	OCRBench	TextCaps	
Model	Accuracy	METEOR	ROUGE
LLaVA-1.5 7B	0.207	0.249	0.480
LLaVA-1.5 68M	0.048	0.133	0.254
LLaVA-1.5 68M (text-only)	0.014	0.064	0.132

Table 15: Evaluation results for the off-the-shelf target model and the custom-trained draft model on MLLM tasks. Since the draft model is trained to perceive multimodality, the injection of multimodal information benefits the custom-trained draft model.



(a) Instruction (b) LLaVA-1.5 7B

(c) LLaVA-1.5 68M (d) LLaVA-1.5 68M (text-only)

Figure 4: Qualitative evaluation samples from the OCRBench dataset by LLaVA-1.5 7B and 68M. Both the target (b) and the draft (c) models recognize the text “friend” written on the image by multimodal reasoning whereas the text-only model (d) fails, as expected.

J Prompts for Each Dataset and Drafting

In this section, we describe the formats of prompts used for inference on each dataset, including system prompts and how to organize prompts with text and image inputs (Appendix J.1). We then provide details on replacing image tokens in text-only and caption drafting (Appendix J.2).

J.1 Prompt Formats for Each Dataset

We use the following prompt formats for respective tasks. Based on the template for chat (*USER:* and *ASSISTANT:*), each system prompt is prepended with the start token *<s>*. The *<image>* token is used to represent image data within a prompt. *[QUESTION]* and *[CAPTION]* are placeholders denoting information unique to each sample of a dataset.

ChartQA *<s> USER: <image> For the following question, provide a detailed explanation of your reasoning leading to the answer. [QUESTION] ASSISTANT:*

TextVQA *<s> USER: <image> For the following question, provide a detailed explanation of your reasoning leading to the answer. [QUESTION] ASSISTANT:*

VQAv2 *<s> USER: <image> For the following question, provide a detailed explanation of your reasoning leading to the answer. [QUESTION] ASSISTANT:*

HallusionBench *<s> USER: <image> For the following question, provide a detailed explanation of your reasoning leading to the answer. [QUESTION] ASSISTANT:*

Spot The Difference *<s> USER: Explain the disparities between the first and second image. <image> <image> Difference: ASSISTANT:*

WebQA *<s> USER: Given the progression of the story with the first few images, can you write a fitting end considering the last image? <image> Image Caption #1: [CAPTION]. <image> Image Caption #2: [CAPTION]. Question: [QUESTION] Answer: ASSISTANT:*

MagicBrush *<s> USER: Please provide instructions for editing the source image to match the target image. Source Image: <image> Target Image: <image> Instruction: ASSISTANT:*

PororoSV *<s> USER: Given the progression of the story with the first few images, can you write a fitting end considering the last image? <image> Caption #1: [CAPTION] <image> Caption #2: [CAPTION]. <image> Caption #3: [CAPTION] <image> Caption #4: [CAPTION] <image> Caption #5: ASSISTANT:*

VIST *<s> USER: With the narratives paired with the initial images, how would you conclude the story using the last picture? <image> Caption #1: [CAPTION] <image> Caption #2: [CAPTION]. <image> Caption #3: [CAPTION] <image> Caption #4: [CAPTION] <image> Caption #5: ASSISTANT:*

J.2 Replacing Image tokens in Draftings

For text-only drafting, the *<image>* token is replaced by the escape character "\n". We experimented with several replacement methods: (1) tokenizing the string "<image>" into three tokens, and (2) retaining the special token <image> without replacing it with an image embedding. Method (2) resulted in very poor block efficiency, but method (1) showed comparable block efficiency. Our replacement approach is simple because it ensures that the prompt length remains consistent before and after replacement.

For caption drafting, the *<image>* token is replaced by a generated caption with a prefix. Specifically, after the lightweight captioning model generates a caption based on the image inputs in the sample, we prepend the string "image: " to the caption and replace the *<image>* token.

K Details of Each Dataset

In this section, we describe each of the curated datasets in benchmark (Appendix K.1) and OOD (Appendix K.2) datasets and provide links to them for convenience and reproducibility.

K.1 Benchmark Datasets

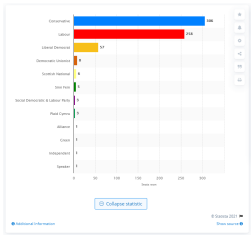
ChartQA (Masry et al., 2022) An image-text question answering dataset for testing visual comprehension of charts. The subset used for evaluation in our work contains 100 pairs of images and questions.

`https://huggingface.co/datasets/lmms-lab/ChartQA`

TextVQA (Singh et al., 2019) A visual question answering dataset that requires reading and reasoning about text within a provided image. The subset used for evaluation in our work contains 100 pairs of images and questions.

`https://huggingface.co/datasets/lmms-lab/textvqa`

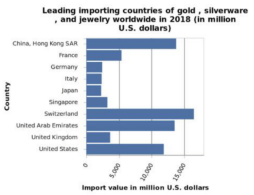
VQAv2 (Goyal et al., 2017) A visual question answering dataset that is well-balanced due to the inclusion of pairs of images/prompts that are similar but result in different answers. The subset used for evaluation in our work contains 100 pairs of images and questions.



(a) ChartQA

(b) TextVQA

(c) VQAv2



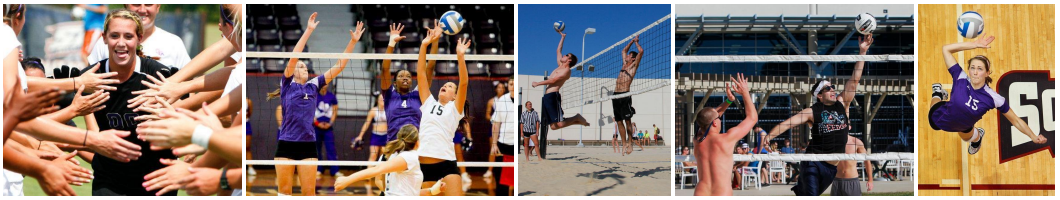
(d) HallusionBench

(e) Spot the Difference

(f) WebQA

(g) MagicBrush

(h) PororoSV



(i) VIST

Figure 5: Qualitative image samples of benchmark and OOD datasets. The corresponding questions and answers are presented in Appendix G.

https://huggingface.co/datasets/lmms-lab/VQAv2

HallusionBench (Guan et al., 2024) A dataset designed to measure the ability of large vision language models to reason despite hallucinations. The subset used for evaluation in our work contains 100 question and answer pairs.

https://huggingface.co/datasets/lmms-lab/HallusionBench

Spot the Difference (Jhamtani & Berg-Kirkpatrick, 2018) A dataset of crowd-sourced descriptions of differences between a pair of images. The subset used for evaluation in our work contains 100 annotated image pairs collected using individual frames of security-footage data.

https://huggingface.co/datasets/lmms-lab/LLaVA-NeXT-Interleave-Bench

WebQA (Chang et al., 2022) A visual question answering dataset requiring information retrieval and reasoning from both visual and text inputs. The subset used for evaluation in our work contains 100 QA pairs.

https://huggingface.co/datasets/lmms-lab/LLaVA-NeXT-Interleave-Bench

MagicBrush (Zhang et al., 2024b) A dataset for text-guided image editing containing manually annotated editing instructions to transform one real image into another. The subset used for evaluation in our work contains 100 triplets of a source image, a target image, and editing instructions.

https://huggingface.co/datasets/lmms-lab/LLaVA-NeXT-Interleave-Bench

K.2 OOD Datasets

Pororo-SV (Li et al., 2019) A dataset of stories each created by pairing 5 consecutive frames from the animated series *Pororo* with a text description. The subset used for evaluation in our work contains 100 stories.

https://huggingface.co/datasets/lmms-lab/LLaVA-NeXT-Interleave-Bench

VIST (Huang et al., 2016) A dataset of sequential images paired with three types of descriptions ranging from isolated factual descriptions to causal, narrative interpretations. The subset used for evaluation in our work contains 100 sequences of 3 images.

https://huggingface.co/datasets/lmms-lab/LLaVA-NeXT-Interleave-Bench

K.3 Time Analysis of LVLM Inference Stages

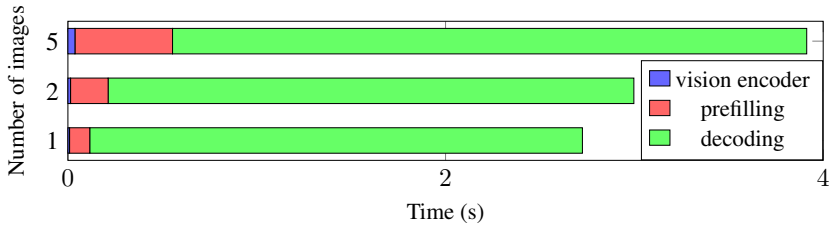


Figure 6: Inference time analysis for the LLaVA-1.5 7B model. Although the time for vision encoder and prefilling increases with the number of images, the decoding stage still dominates.

To analyze how the number of input images affect the LVLM inference time, we select ChartQA (Masry et al., 2022), Spot the Difference (Jhamtani & Berg-Kirkpatrick, 2018), and PororoSV (Li et al., 2019) datasets representing 1, 2, and 5 images with corresponding visual context lengths of 0.6k, 1.2k, and 3k, respectively. We visualize the generation time by component in Fig. 6 with 100 generated tokens for analysis, with actual average decoding lengths of 92, 117, and 88, respectively. The execution time of the *vision encoder* and *prefilling* stages increases in proportion with the number of input images, as each image is converted into several hundred context tokens. In contrast, the *decoding* stage shows little difference in execution time across varying numbers of input images, while dominating the total generation time. Hence, although reducing the number of visual tokens (Shang et al., 2024; Chen et al., 2024b; Lin et al., 2024) would significantly improve the efficiency of *vision encoder* and *prefilling* stages, it would have only marginal impact on the dominant *decoding* stage.