TABED: Test-Time Adaptive Ensemble Drafting for Robust Speculative Decoding in LVLMs

Anonymous ACL submission

Abstract

Speculative decoding (SD) has proven effective in accelerating LLM inference by quickly generating draft tokens and verifying them in parallel. However, SD remains largely unexplored for Large Vision Language Models (LVLMs), advanced LLMs capable of processing both image and text prompts. To address this gap, we first benchmark existing drafting methods for LVLMs across diverse scenarios and observe that methods using small draft models show scenario-specific performance fluctuations. Motivated by these findings, we propose Test-time Adaptive Batched Ensemble Drafting (TABED), which dynamically ensembles multiple drafts obtained via batch inference by leveraging measurable deviations from past ground truths available in the SD setting. Across diverse input scenarios, TABED achieves an average robust expected walltime speedup of 1.74x compared to standard decoding and a 5% improvement over individual drafting methods, though it does not incur additional training costs (i.e., training-free) and keeps ensembling costs negligible by sharing model parameters. To further enhance its extensibility, we also explore incorporating alternative drafting methods using image pooling and captioning. Our method maintains seamless compatibility with existing LVLM acceleration techniques, and we open-source custom-trained draft LVLMs to ensure reproducibility.

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), also known as Large Multimodal Models (Li et al., 2024b; Jin et al., 2024; Song et al., 2023), specialize in handling prompts comprised of *text and images*. These models have attracted significant attention due to their unique applications, including multimodal chatbots, visual question answering (VQA), and augmented reality (AR) (OpenAI, 2023; Anthropic, 2024; Gemini Team Google: Anil et al., 2023).

As LVLMs are increasingly deployed, reducing their inference time has become a critical challenge. In addition to the standard LLM autoregressive decoding process, LVLMs must (1) pre-process each image in the input into several hundred *image tokens* (Radford et al., 2021; Liu et al., 2023, 2024a) and (2) process both text and image tokens, resulting in considerably higher inference time.

Recently, methods like token pruning, layer skipping, and key-value cache compression have been proposed to accelerate LVLM inference (Shang et al., 2024; Chen et al., 2024b; Lin et al., 2024; Liu et al., 2024e; Wan et al., 2024; McKinzie et al., 2024). While effective, these approximation techniques cannot preserve the original LVLM's output distribution. Moreover, they primarily reduce prompt processing time (i.e., the prefilling stage) and have less impact on response generation time (i.e., the decoding stage).

In contrast, Speculative Decoding (SD) is an acceleration technique for LLM inference that fully preserves the output distribution (Leviathan et al., 2023; Chen et al., 2023). SD *speculates* a specified number of draft tokens and then uses the original target model to *verify* these tokens in parallel. For LLM inference, various methods have been proposed to enhance each component in SD (Xia et al., 2024). To tailor and further exploit their specific approach, these methods often use additional training, either for a separate lightweight draft model (Miao et al., 2023; Kim et al., 2023; Zhou et al., 2024) or an added component within the target model (Cai et al., 2024; Li et al., 2024d; Zhang et al., 2024b). These approaches signifi-

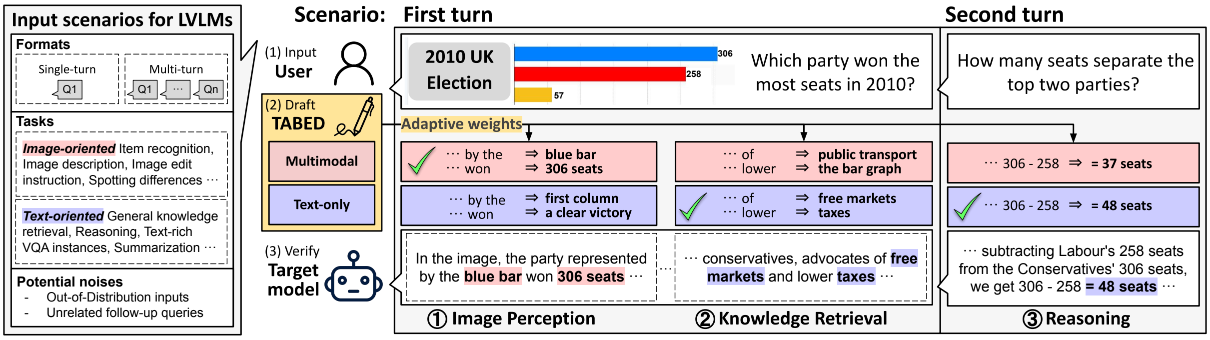


Figure 1: **Overview of TABED.** LVLMs must handle a wide range of input scenarios, including various combinations of formats, tasks, and potential noise. To effectively accelerate target LVLMs using SD, different drafting methods are required for intra-response and inter-response cases. For example, in the first turn, the drafting method needs to identify the graphical information the user is referring to based on the input image, and then perform text-based general knowledge retrieval. (① to ②). In the second turn, it performs mathematical reasoning using the accumulated text context (③). Existing single drafting methods with small models, whether Multimodal or Text-only, show fluctuating performance across scenarios (Section 4). **TABED** addresses this by dynamically ensembling multiple drafts using past ground truths in SD, thereby achieving robust speedup across diverse scenarios (Section 5) and can be further extended with additional drafting methods (Section 6). See Fig. 2 for details.

cantly improve performance, but often at the cost of additional training or reduced compatibility with other acceleration methods.

Unlike LLM inference, SD for LVLMs has been far less explored, with the only notable exception by Gagrani et al. (2024). They were the first to successfully accelerate LVLM inference via SD using a text-only drafting method (i.e., the drafting relied solely on the text tokens and ignored the image tokens)—an observation that is both intriguing and counterintuitive. However, the authors evaluate their method on a few datasets with single-turn VQA tasks, limiting insight into when the method excels without image modality.

To assess whether existing drafting methods (Gagrani et al., 2024) consistently accelerate LVLM inference via SD across diverse real-world scenarios, we first benchmark them at scale on seven benchmark datasets and two out-of-distribution (OOD) datasets using interactions with various types of multi-turn instructions. As a result, none of the existing drafting methods, whether multimodal or text-only, effectively handles the full range of input scenarios when using a small draft model. Consequently, selecting a single drafting method in advance is a nontrivial challenge. This raises the question: *"Can we develop a method that integrates multiple draftings to ensure robust performance across all input scenarios?"*

To address this, we present Test-time Adaptive Batched Ensemble Drafting (TABED), which (1) obtains multiple drafts simultaneously through batch inference while sharing the same model parameters and (2) dynamically weights the drafts based on their differences from previous ground truth labels after verification by the target model. This approach is entirely training-free and incurs no additional cost during inference, making it generalizable to a wider range of drafting scenarios. To further leverage this flexibility, we explore incorporating alternative drafting strategies, including image pooling and captioning, into our method.

As a result, TABED consistently shows the best or second-best performance across all combinations of scenarios, with an average robust expected walltime speedup of 1.74x and over 5% improvement compared to existing drafting strategies, which often exhibit scenario-specific performance fluctuations. Further extending drafting strategies with complementary advantages also leads to enhanced performance. This is particularly significant, as this improvement achieved solely through better utilization of a small draft model as is. As it does not incur additional costs in training or inference, it remains compatible with other LVLM acceleration techniques, including those targeting the prefilling stage or other types of SD. To support LVLM SD research and ensure reproducibility, we open-source our code and draft LVLMs.

In summary, main contributions of our work are:

- We benchmark existing drafting methods and find scenario-specific performance fluctuations across various input scenarios for LVLM SD (Sec. 4).

- We propose TABED, which achieves superior and robust performance across diverse scenarios (Sec. 5), and further explore and incorporate new drafting methods (Sec. 6).

- We open-source our code and the custom-trained draft model to support the emerging field of LVLM SD and ensure reproducibility.

2 Related Work

2.1 Large Vision Language Models

LVLMs Frontier proprietary LVLMs (OpenAI, 2023; Anthropic, 2024; Gemini Team Google: Anil et al., 2023) exhibit state-of-the-art performance across multiple modalities beyond just text. Meanwhile, open-source models such as 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 for 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, enabling 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. (2024e); 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.

2.2 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 to study SD for LVLMs. They introduced text-only drafting, claiming its performance is comparable to multimodal drafting. However, their benchmark results were limited, and they provided little detailed analysis of each drafting method. Additionally, 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. Jang et al. (2024) and Teng et al. (2024) propose SD methods to improve text-to-image generative models, distinct from LVLM approaches.

2.3 Test-time Adaptation

To enhance model robustness against distribution shifts during test time, various generalization and adaptation techniques have been proposed (Liang et al., 2025). In situations where data arriving during test time lacks ground truth labels, these techniques update models through explicit gradient steps by formulating loss using pseudo labeling (Hardt and Sun, 2024; Wang et al., 2022), self-supervision (Sun et al., 2020; Krause et al., 2018), or normalization statistics (Schneider et al., 2020; Wang et al., 2021), thereby causing inevitable delays during test time. In contrast, SD allows the use of ground truth information obtained through verification for test-time adaptation.

3 Preliminaries

3.1 Theoretical Latency of Transformers

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 let $T(B, S, \gamma)$ represent the time to verify γ tokens in parallel. Under moderate S (*e.g.*, $S \leq$ 3k) 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 F.

3.2 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 $\text{norm}(\max(0, p(y) - q(y)))$.[1]

Block Efficiency and Walltime Speedup Given an 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 $T_q(B, S, 1)$ denote the time required for M_p and M_q to decode a single token, and let $T_p(B, S, \gamma)$ 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. The expected walltime speedup of SD can be expressed as the *token rate ratio:*

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

We first empirically measure the *draft-to-target latency ratio* $\frac{T_q}{T_p}$ and then use the speedup calculated by Eq. (1) to evaluate the performance of SD. See Appendix G for measurement details.

4 Benchmarking SD for LVLMs

In this section, we systematically benchmark speculative decoding for LVLMs by evaluating existing drafting methods, both multimodal and text-only, across a variety of input scenarios.

4.1 Experiment Settings

Target and Draft Models We employ 7B and 13B models from the LLaVA-1.5 (Liu et al., 2024a) and LLaVA-NeXT (Liu et al., 2024b) series, both of which are widely adopted public LVLMs, as our target models. We then use three distinct variants of the draft model, considering different model sizes and training strategies to ensure comprehensive benchmarking. Due to the lack of sufficiently small LVLMs that ensure a low *draft-to-target latency ratio* (represented by $\frac{T_q}{T_p}$ in Eq. (1)), we select the small public LLaMA models of 68M and 160M (Miao et al., 2023) as base models for training. We then train the draft model using different training methodologies, specifically utilizing the procedures for LLaVA-1.5, as well as LLaVA-OV (Li et al., 2024a), which are specialized in multi-image processing (see Appendix H.1 for training details).

To summarize, we benchmarked draft models by size (68M, 160M) and type (LLaVA-1.5, LLaVA-OV), and target models by size (7B, 13B) and type (LLaVA-1.5, LLaVA-NeXT). We particularly focused on the results of LLaVA-1.5 68M, which has the smallest model size and an aligned representation with the target model. We observe that the draft model perceives multimodality in representative multimodal tasks (see Appendix H.2).

Input Scenarios Systems using SD must deliver consistent performance gains across diverse real-world scenarios, especially due to their application with LVLMs in practical settings. To examine this requirement, we initially curated seven benchmark datasets due to the lack of pre-existing benchmarks specifically for LVLM SD.

[1] When the prefix $(x, y_{<t})$ is clear from the context, we use $p(y)$ and $q(y)$ to denote $p(y_t|x, y_{<t})$ and $q(y_t|x, y_{<t})$, respectively.

Draft Model			Benchmark Datasets (First Turn)								OOD Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	PSV	VIST
LLaVA-1.5	68M	M	2.28	2.15	2.56	2.21	2.19	1.96	2.34	2.24	1.19	1.16
		T	2.19	2.08	2.31	2.16	2.23	2.34	2.27	2.23	2.05	2.05
		MT	2.25	2.15	2.47	2.21	2.31	2.37	2.4	2.31	1.94	1.91
		MT*	2.26	2.16	2.52	2.21	2.29	2.39	2.36	2.31	2.02	2.04
Draft Model			Benchmark Datasets (Second Turn)								NLP Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	NQ	GSM8K
LLaVA-1.5	68M	M	2.1	1.96	2.78	2.18	1.61	1.53	1.83	2.00	1.98	2.25
		T	2.32	2.23	2.91	2.56	1.87	2.01	2.08	2.28	2.03	2.30
		MT	2.29	2.24	2.93	2.54	1.81	1.91	2.02	2.25	2.02	2.28
		MT*	2.29	2.23	2.93	2.56	1.85	1.99	2.05	2.27	2.03	2.29

Table 1: Block efficiency results for various drafting methods are presented: existing methods (multimodal (M) and text-only (T)), **MT** ensembling multiple drafts obtained through batched inference with static equal weights, and **MT***, which dynamically weights M and T through test-time adaptation. Across all combinations of turn-taking, datasets (both benchmark and OOD), image inclusion, and contextual relatedness, MT* consistently achieves either the best or second-best performance, while M and T vary by scenario. Even when ranked second, its performance is close to the best, unlike the larger gaps seen with other methods. In OOD scenarios where some drafting methods fail, MT* significantly outperforms MT by discriminating between multiple methods. See Appendix A for full results across various sizes and types of draft and target models.

These datasets include tasks involving both single-image (Liu et al., 2023; Mathew et al., 2021; Li et al., 2023b; Yu et al., 2023) and multi-image scenarios (Tan et al., 2019; Zhang et al., 2024c; Jhamtani and Berg-Kirkpatrick, 2018). To further challenge the system's ability to handle unexpected and varied queries while maintaining consistent performance, we added two additional datasets featuring five images per query (Li et al., 2019; Huang et al., 2016), serving as notable Out-of-Distribution (OOD) cases. Moreover, we extend the evaluation from single-turn to multi-turn scenarios using benchmark datasets that include various types of follow-up queries, including those dependent on prior responses (e.g., follow-up requests with images from the same dataset or text-only tasks from multi-turn benchmarks for LVLMs (Liu et al., 2024c)) and distinct text-only reasoning tasks (Kwiatkowski et al., 2019; Cobbe et al., 2021).

Drafting Methods: Multimodal and Text-only The multimodal drafting method follows the standard LVLM process, receiving both images and text as input. In contrast, the text-only drafting method, first explored in Gagrani et al. (2024), uses only textual data, adhering to the standard LLM process. For all experiments in this paper, including this section, we set $\gamma = 5$ and perform greedy decoding with a maximum of 128 new tokens.

4.2 Experimental Results

Table 1 presents the block efficiency results for multimodal (M) and text-only (T) drafting methods. Across all scenarios, neither drafting method consistently outperformed the other. Notably, the multimodal drafting method demonstrated higher block efficiency than the text-only approach in single-turn scenarios across most benchmark datasets. However, the text-only drafting performed comparably or slightly sub-optimally overall in single-turn situations, often surpassing the multimodal drafting in subsequent turns–particularly in tasks dependent on prior responses and independent reasoning tasks, and significantly in OOD cases. This trend continued regardless of the model size or multi-image awareness capabilities. Existing drafting methods, both multimodal and text-only, show scenario-specific performance fluctuations with draft models small enough for SD, with neither method consistently outperforming the other across various input scenarios for LVLM inference. Before execution, it is challenging to determine in advance which method is superior, and even if known, resolving inconsistencies with a single drafting method is difficult. See Appendix A for full results across various sizes and types of draft and target models.

No one-size-fits-all among existing drafting methods for LVLM input scenarios.

5 Test-time Adaptive Batched Ensemble Drafting

For LVLM SD, selecting a single drafting method in advance for a small draft model is challenging. In this context, ensemble learning offers a promising solution, as it is often used to reduce both bias and variance in predictions (Dietterich, 2000; Ganaie et al., 2022), particularly for models with limited capacity (Zhou, 2012).

In this section, we introduce Test-time Adaptive Batched Ensemble Drafting (TABED), a method that applies ensemble learning with dynamic weight adaptation to each of the drafts obtained via batch inference. It is fully training-free, highly extensible, and delivers robust performance across diverse input scenarios. Fig. 2 provides an overview of TABED.

5.1 Proposed Method

Unlike typical test-time scenarios where incoming data lacks ground truth labels (Wang et al., 2021, 2022), SD allows access to both hard labels (i.e., $y_{<t}$) and soft labels (i.e., $p(y_{t'})$) for $t' < t$ after verifying all steps prior to t using the target model M_p. When drafting is performed in step t, this information can be leveraged to dynamically adjust ensemble weights w_t, controlling each drafting method's influence based on its past performance. This approach can enhance ensemble learning by effectively combining probabilities $[q_t^{(1)}, \dots, q_t^{(m)}]$ from m drafting methods, ensuring the resulting ensembled distribution $q(\cdot|x, y_{<t}; w_t)$ closely aligns with the target distribution $p(\cdot|x, y_{<t})$. For instance, the system can adjust the weight assigned to the multimodal drafting method by recognizing varying needs for visual context for a specific sample x and timestep t.

Ensemble learning via Batched Inference To obtain multiple predictions from draft model M_q, we utilize batched inference tailored for LVLMs. At each decoding timestep t to $t + \gamma - 1$, all m drafting methods share the parameters of M_q, and their distributions are ensembled to sample the next token (Algorithm 1). We employ a weighted averaging ensemble method, sampling a token from the ensembled distribution to continue drafting. This process incurs no additional costs during training (i.e., training-free) and keeps ensembling costs negligible (Eq. (1), Remark 3), unlike typical ensemble learning, which often requires extra parameters or models to generate multiple predictions.

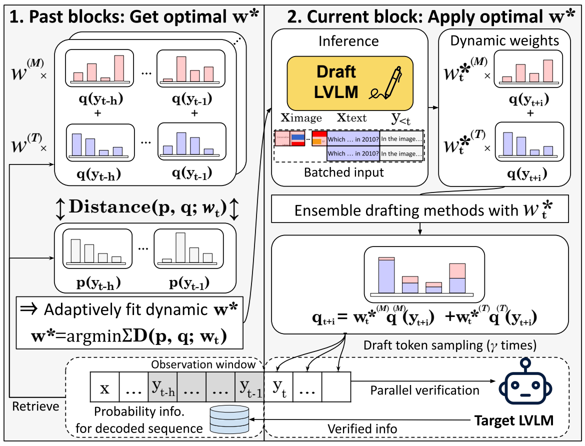


Figure 2: TABED predicts the optimal weight, w^*, based on the deviation (i.e., distance) of past drafting blocks from the ground truth obtained through the verification in SD. It then dynamically ensembles multiple drafts obtained via batch inference. See Algorithm 1 for further details.

Test-time Adaptive Ensemble Weights To effectively combine drafting methods based on varying needs, we explore test-time adaptation to dynamically weight each method. Specifically, at the beginning of each new drafting block, we sample a list of ensemble weights $W_t = [w_t^1, \dots, w_t^n]$ using the weight sampling policy $S_W(t)$ (Algorithm 1). For each $w_t^j \in \mathbb{R}^m$, we compute ensembled draft probabilities $q_{t'}^j = q(\cdot|x, y_{<t'}; w_t^j)$ over the past time step window $t' \in [t-h, t-1]$. We then select the optimal w_t^* from W_t to be used at timestep t by utilizing hard labels $y_{<t}$ (by maximizing the number of token matches sampled from each $q_{t'}^j$), or soft labels $p(y_{t'})$ (by minimizing the accumulated error e_t^j over previous steps t'), or both.

$$e_t^j = \sum_{t'} D_{\mathrm{KL}}\left(p(\cdot \mid x, y_{<t'}) \,\|\, q(\cdot \mid x, y_{<t'}; w_t^j)\right)$$

where D_{KL} is the KL divergence between p and $q_{t'}^j$ at each of the previous steps t'. This weight w_t^* is used throughout the current drafting block of γ tokens (i.e., from timestep t to $t + \gamma - 1$).

Drafting methods with higher weights indicate closer alignment to the target model. In our experiments, we also explored the effects of employing various weight sampling policies S_W, varying the window size h for past time steps, and using Total Variation Distance (TVD) instead of KLD.

SD enables test-time adaptation using ground truth, obtained through verification.

Algorithm 1 TABED

Parameter: M_q, Prefix $X = [x^{(1)}, ..., x^{(m)}]$ for m drafting methods, Weight sampling policy S_W, local window length h

▷ W-AVG stands for Weighted Average

▷ D stands for Distance

Input: Batch $b_{<t} := [(x^{(1)}, y_{<t}), ..., (x^{(m)}, y_{<t})]$

Output: γ draft tokens $y_t, ..., y_{t+\gamma-1}$ and ensembled probabilities $q_t, ..., q_{t+\gamma-1}$

```
procedure TABED(b_{<t}; γ, S_W)
    W_t = [w_t^1, ..., w_t^n] ~ S_W(t)
    w_t^* ← arg min_j Σ_{t'∈[t−h,t−1]} D(p_{t'}, q_{t'}^j)
    for i ← 0 to γ − 1 do
        [q_{t+i}^{(1)}, ..., q_{t+i}^{(m)}] ← M_q(b_{<t+i})
        q_{t+i} ← W-AVG([q_{t+i}^{(1)}, ..., q_{t+i}^{(m)}]; w_t^*)
        y_{t+i} ← SAMPLE(q_{t+i})
    end for
    return [y_t, ..., y_{t+γ−1}], [q_t, ..., q_{t+γ−1}]
end procedure
```

5.2 Experimental Results

Table 1 presents the block efficiency results for TABED, which is denoted by MT*. MT* consistently achieved either the best or second-best performance across all input scenarios. These scenarios include turn-taking, image inclusion, prior context relevance, and both benchmark and OOD datasets. Even when MT* ranked second, its performance remained close to the best, unlike the larger performance gaps observed for other methods. Without incurring additional costs during training or inference, this resulted in an average robust expected walltime speedup of 1.74x compared to standard decoding and a 5% improvement compared to existing single drafting methods M and T, when averaged across all benchmark datasets and turns. These consistent gains highlight TABED's ability to effectively prioritize stronger drafting methods by dynamically assigning optimal weights, even when individual methods exhibit scenario-specific performance fluctuations. See Appendix B for full experimental results with varying sampling policy S_W, window size h, and distance D.

Fig. 3 provides an in-depth view of how the adaptive weights w^* predicted by TABED change during the decoding phase. It illustrates the weights employed for drafting methods M, T, MT, and MT*, as the decoding step progresses. The ensemble weights are shaded if the token sampled from the resulting ensembled distribution is accepted by the target model. By dynamically adjusting the ensemble weights, TABED more effectively explores the shaded region for acceptance compared to other methods. Specifically, compared to MT, which employs simpler ensembling with a static equal weight ratio, MT* with dynamic weights excels in cases requiring effective discrimination between multiple drafting methods (Fig. 3). This explains the significant performance gap between MT and MT* in OOD scenarios.

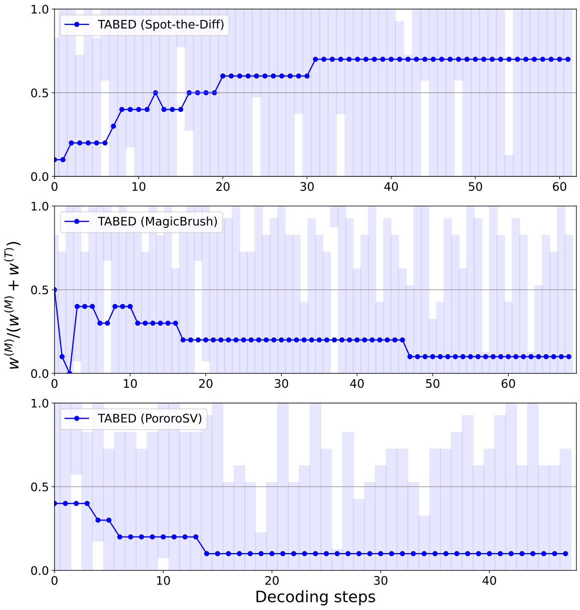


Figure 3: Qualitative samples of ensemble weights and the acceptance of sampled tokens from the resulting ensembled distribution across datasets. The y-axis denotes the proportion of $w^{(M)}$ relative to $w^{(M)} + w^{(T)}$, and the x-axis indicates decoding steps. In comparison to other single drafting methods, M and T ($y = 1$ and 0), or the static ensembling method MT ($y = 0.5$), adaptive ensemble weights w^* predicted by **TABED** (the blue graph) effectively explore the region shaded, representing the weights that lead to token acceptance by the target model. See Appendix C for more examples.

> *Ensemble drafting method achieves robustness and speedup, and covers diverse scenarios including OOD cases by employing dynamic weights.*

6 Exploring LVLM Drafting Candidates

Since image tokens in LVLMs have relatively sparse importance than text tokens, several methods have been proposed to process image tokens and reduce computational cost (Shang et al., 2024; Chen et al., 2024b). In this section, we benchmark two alternative drafting methods for LVLMs—pooled multimodal and caption-based—which process image information differently. We then extend TABED to include these alternatives. This seamless integration improves generalization (Zhou, 2012) without additional training costs and keeps ensembling costs negligible by sharing model parameters.

6.1 Pooled Multimodal Drafting

To condense sparsely important image tokens while preserving the 2D spatial structure, we apply average pooling during inference just before the projector transforms them into the text embedding space. Specifically, we use a 2×2 pooling kernel, reducing the number of visual tokens from 576 to 144 in our default configuration.

6.2 Caption Drafting

To convert sparsely important image tokens into textual descriptions, we employ a lightweight image captioning model (Li et al., 2022, 2023a; Xiao et al., 2024). The captioning model runs only once, in parallel with the target model's prefilling stage, making its overhead negligible. Any minor delays, such as those caused by hardware variations, are amortized over the entire decoding process (Fig. 9). We use Florence as the default captioning model (see Appendix D for further details).

6.3 Experimental Results

Table 2 presents the block efficiency results averaged across datasets in each category for pooled-multimodal drafting (P), caption-based drafting (C), static ensemble drafting that integrates these into existing methods (MTCP), and dynamic ensemble drafting with adaptive weights (MTCP*).

First, the differently processed visual information from P and C adds image awareness to the drafting process. By supplementing T, C retains T's robustness and perform better on both standard and OOD datasets.

Second, MTCP*, which uses dynamic weights, consistently achieves the highest speedup across all benchmark and OOD datasets, outperforming single drafting (M, T, C, and P) and static ensemble drafting (MTCP) methods in both the first and second turns. Even static ensemble methods that incorporate more diverse drafting methods demonstrate stronger generalization, while dynamically weighting these methods further improves performance and better aligns the ensemble distribution

Draft Model			Benchmark		OOD
Type	Size	Method	First	Second	Avg.
LLaVA-1.5	68M	M	2.24	2.00	1.18
		T	2.23	2.28	2.05
		C	2.29	2.30	2.09
		P	2.23	2.25	2.08
		MTCP	2.32	2.31	2.12
		MTCP*	2.32	2.32	2.13

Table 2: Speedup results are presented for the single drafting methods M, T, caption-based (C), and pooled-multimodal (P), as well as the ensemble methods MTCP and MTCP*. The ensembling benefit from combining multiple drafting methods, with TABED consistently achieving the highest speedup in both the first and second turns across all benchmark and OOD datasets. Speedup is averaged within each dataset category. See Appendix A for full results.

with the target distribution, all without incurring additional costs. See Appendix A for full results and details.

Integrating more drafting methods yields more robustness and speedup.

7 Conclusion

We benchmark existing drafting methods for LVLMs and observe scenario-specific performance fluctuations with small draft models. To address this, we propose Test-time Adaptive Batched Ensemble Drafting (TABED), which dynamically ensembles multiple drafts via batch inference using measurable deviations from past ground truths in the SD setting. TABED consistently outperforms individual drafting methods and delivers robust speedup. We also integrate image pooling and captioning as alternative drafting methods, finding that static ensembling improves generalization and dynamic weighting further enhances performance.

8 Limitations and Future Work

While we focused on a single draft candidate and a single verification scheme, TABED could benefit from integration with multiple draft candidates (Miao et al., 2023; Yang et al., 2024; Cai et al., 2024) for further speedup, as our method is orthogonal to and fully compatible with these approaches. We also believe that TABED is applicable to other MLLMs (e.g., those handling audio and text tokens (Fu et al., 2024)).

References

Anthropic. 2024. The Claude 3 Model Family: Opus, Sonnet, Haiku.

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

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*.

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

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2024b. 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*.

Yangyi Chen, Karan Sikka, Michael Cogswell, Heng Ji, and Ajay Divakaran. 2024c. Dress: Instructing large vision-language models to align and interact with humans via natural language feedback. *Preprint*, arXiv:2311.10081.

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*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *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)*, pages 4171–4186.

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

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

Chaoyou Fu, Haojia Lin, Zuwei Long, Yunhang Shen, Meng Zhao, Yifan Zhang, Xiong Wang, Di Yin, Long Ma, Xiawu Zheng, et al. 2024. Vita: Towards open-source interactive omni multimodal llm. *arXiv preprint arXiv:2408.05211*.

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

Mukul Gagrani, Raghavv Goel, Wonseok Jeon, Junyoung Park, Mingu Lee, and Christopher Lott. 2024. On speculative decoding for multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8285–8289.

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

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. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Moritz Hardt and Yu Sun. 2024. Test-time training on nearest neighbors for large language models. *Preprint*, arXiv:2305.18466.

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

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

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

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. 2024. Efficient Multimodal Large Language Models: A Survey. *Preprint*, arXiv:2405.10739.

Sehoon Kim, Karttikeya Mangalam, Suhong Moon, Jitendra Malik, Michael W. Mahoney, Amir Gholami, and Kurt Keutzer. 2023. Speculative decoding with big little decoder. *Preprint*, arXiv:2302.07863.

Ben Krause, Emmanuel Kahembwe, Iain Murray, and Steve Renals. 2018. Dynamic evaluation of neural sequence models. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2766–2775. PMLR.

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

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

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

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

Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. 2024c. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. *Preprint*, arXiv:2407.07895.

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

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

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023b. Evaluating Object Hallucination in Large Vision-Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 292–305.

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

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

Jian Liang, Ran He, and Tieniu Tan. 2025. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, 133(1):31–64.

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

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

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024b. Llava-next: Improved reasoning, ocr, and world knowledge.

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

Shuo Liu, Kaining Ying, Hao Zhang, Yue Yang, Yuqi Lin, Tianle Zhang, Chuanhao Li, Yu Qiao, Ping Luo, Wenqi Shao, and Kaipeng Zhang. 2024c. Convbench: A multi-turn conversation evaluation benchmark with hierarchical capability for large vision-language models. *Preprint*, arXiv:2403.20194.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xucheng Yin, Cheng lin Liu, Lianwen Jin, and Xiang Bai. 2024d. Ocrbench: On the hidden mystery of ocr in large multimodal models. *Preprint*, arXiv:2305.07895.

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

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

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209.

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

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

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

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. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

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. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. 2020. Improving robustness against common corruptions by covariate shift adaptation. *Advances in neural information processing systems*, 33:11539–11551.

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

Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. 2020. Textcaps: a dataset for image captioning with reading comprehension. *Preprint*, arXiv:2003.12462.

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

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

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. 2020. Test-time training with self-supervision for generalization under distribution shifts. *Preprint*, arXiv:1909.13231.

Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. 2024b. SpecTr: Fast Speculative Decoding via Optimal Transport. *Preprint*, arXiv:2310.15141.

Hao Tan, Franck Dernoncourt, Zhe Lin, Trung Bui, and Mohit Bansal. 2019. Expressing Visual Relationships via Language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1873–1883.

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

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

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

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

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

Jiayang Wu, Wensheng Gan, Zefeng Chen, Shicheng Wan, and Philip S. Yu. 2023. Multimodal Large Language Models: A Survey. *Preprint*, arXiv:2311.13165.

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

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. 2024. 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*, pages 4818–4829.

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

Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. 2024. A Survey on Multimodal Large Language Models. *Preprint*, arXiv:2306.13549.

Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. 2023. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid Loss for Language Image Pre-Training. *Preprint*, arXiv:2303.15343.

Duzhen Zhang, Yahan Yu, Jiahua Dong, Chenxing Li, Dan Su, Chenhui Chu, and Dong Yu. 2024a. MM-LLMs: Recent Advances in MultiModal Large Language Models. *Preprint*, arXiv:2401.13601.

Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. 2024b. Draft & verify: Lossless large language model acceleration via self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 11263–11282. Association for Computational Linguistics.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. 2024c. Magicbrush: A manually annotated dataset for instruction-guided image editing. *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. 2024. DistillSpec: Improving Speculative Decoding via Knowledge Distillation. In *The Twelfth International Conference on Learning Representations*.

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

A Comprehensive Benchmarking Across Draft and Target Models

In this section, we present the full results corresponding to the main experiments shown in Tables 1 and 2 for all drafting methods considered in this paper, including both single (multimodal and text-only) and ensemble approaches (MT* and MTCP*), across various sizes and types of draft and target models. Specifically, we benchmark draft models (LLaVA-1.5 68M, LLaVA-1.5 160M, LLaVA-OV 68M) and target models (LLaVA-1.5 7B, LLaVA-1.5 13B, LLaVA-NeXT 7B, LLaVA-NeXT 13B). As shown in Tables 10 to 14, the findings from Sections 4 to 6 hold across different combinations of draft and target models.

B Implementation Details of TABED

In this section, we examine two key factors influencing the dynamic ensembling approach: the observation window length h used for computing adaptive weights (Appendix B.1), and the weight sampling policy S_w that determines how ensemble weights are selected in our experiments (Appendix B.2).

B.1 Observation Window Length h

The responsiveness of the dynamic weights predicted by TABED can be controlled by the observation window length h. In Algorithm 1, we introduce an observation window of length h for the drafting method to compute dynamic weights based on information within this window. By varying h, we further examine how this approach addresses the inherent nature of autoregressive decoding, which relies solely on past context for next-token prediction. For both benchmark and OOD datasets, we explore the effect of varying h by setting it to 1, 4, 16, and to include all previous tokens. As shown in Table 3, there is no clear winner among the different window lengths h. Therefore, by default, using all previous information (i.e., ALL) before the current timestep is a reasonable choice.

B.2 Weight Sampling Policy S_w

There are numerous variants of TABED, obtained by varying the weight sampling policy S_w in Algorithm 1. Here, we describe the representative policy used in our experiments throughout the paper. For S_W in MT*, we sample $W_t = [w_t^j]$, where $w_t^j = [1 - \frac{j}{n}, \frac{j}{n}] \in \mathbb{R}^m$ and $n = 10$, to succinctly represent linear combinations with non-negative integer coefficients. For MTCP*, we sample $W_t = [w_t]$, where $w_t = \text{softmax}([1/e_t^{(1)}, \ldots, 1/e_t^{(m)}]; \tau) \in \mathbb{R}^m$ using soft labels, temperature τ, and $n = 1$, offering an efficient alternative as m increases. We set temperature $\tau = 1$ by default.

C More Qualitative Samples of Dynamic Ensemble Weights

In this section, we provide further qualitative examples of dynamic ensemble weights. In Fig. 4, the y-axis denotes the proportion of $w^{(M)}$ relative to $w^{(M)} + w^{(T)}$, and the x-axis indicates decoding steps. In comparison to other single drafting methods, M and T ($y = 1$ and 0), or the static ensembling method MT ($y = 0.5$), adaptive ensemble weights w^* predicted by TABED (the blue graph) effectively explore the region shaded, representing the weights that lead to token acceptance by the target model.

D Details for Caption Drafting

In this section, we describe various types of lightweight image captioning models that can be used for caption drafting (Appendix D.1). We then demonstrate that captioning model inference completes earlier than the target model's prefilling by analyzing the captioning model's latency (Appendix D.2).

D.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., 2023a) 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.

Draft Model			Benchmark Datasets							OOD Datasets	
Type	Size	h	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	PSV	VIST
LLaVA 1.5	68M	1	2.26	2.15	2.54	2.21	2.30	2.36	2.35	2.00	2.02
		4	2.25	2.16	2.54	2.20	2.29	2.40	2.36	2.05	2.05
		16	2.26	2.17	2.53	2.21	2.31	2.42	2.36	2.04	2.05
		ALL	2.26	2.16	2.52	2.21	2.29	2.39	2.36	2.02	2.04

Table 3: Experimental results with different observation window length h, which constrain the number of previous timesteps used to compute adaptive weights. $h = 1$ indicates that the weights are based only on the ground truth from the previous decoding step.

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 4: 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.

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

D.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 4 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.

E Evaluation of Target Model

Model	Datasets	
	Spot-the-Diff	MagicBrush
GPT-4o	19.1	10.5
GPT-4o-mini	12.0	9.1
LLaVA-1.5 7B	13.9	9.6
LLaVA-1.5 7B (text-only)	11.1	8.4

Table 5: Evaluation results for the target model on benchmark datasets. Our target model, LLaVA-1.5 7B, demonstrates performance comparable to GPT-4o-mini. 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 E.1) and qualitative samples (Appendix E.2) produced by LLaVA-1.5 7B, which serves as the target model in our experimental setting.

E.1 Quantitaive Evaluation

We assess the output quality of LLaVA-1.5 7B using benchmark datasets, as shown in Table 5. The results demonstrate that the quality of LLaVA-1.5 7B is comparable to GPT-4o-mini.The injection of multimodal information benefits the target model, LLaVA-1.5 7B, from the perspective of output quality.

E.2 Generated Samples

Spot The Difference [Fig. 8e]

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.

MagicBrush [Fig. 8f]

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. 8g]

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. 8h]

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.

F Empirical Validation of Remarks 1-3

In this section, we validate the Remarks from Sec. 3.1 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 Remarks 2 and 3, respectively. All experiments are conducted on an A100 80GB GPU using the fp16 data type for the models. 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:

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.*, $|\Delta T/T| < 0.05$).*

Fig. 5 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 $S \leq 3$k, 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.

Fig. 6 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%.

G Empirical measurements of draft-to-target latency ratio T_q/T_p

In this section, we present empirical measurements of the draft-to-target latency ratio (T_q/T_p) across well-known LVLM benchmarks (Li et al., 2024c). By providing detailed results on improvements in wall-clock time, we show that SD performance is proportional to block efficiency, as we empirically validate in Remarks 1 to 3. This allows us to calculate the expected speedup by inputting the draft-to-target latency ratio into Eq. (1). All measurements in Table 6 and averaged across datasets and performed on an Nvidia A6000 48GB GPU.

Target Model Size	Draft Model Size	draft-to-target latency ratio T_q/T_p
7B	68M	0.063
	160M	0.206
13B	68M	**0.042**
	160M	0.137

Table 6: Empirical measurements of the draft-to-target latency ratio (T_q/T_p), covering all model sizes.

H 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 H.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 H.2).

H.1 Details of Training

LLaVA-1.5 (Liu et al., 2024a) The process for developing draft models with the LLaVA-1.5 (68M, 160M) training recipe was divided into two stages: pre-training and instruction fine-tuning (IFT). Pre-training focused on training the projector while the parameters of the LLM and vision encoder were frozen. During the IFT stage, visual instruction tuning was used to teach the LLM to follow multi-modal instructions. The vision encoder remained frozen throughout both stages. The hyperparameters used for each stage are described in Table 7. 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.

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 7: Details of hyperparameters used in LLaVA-1.5 training

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 were aligned with the word embedding space of LLMs. High-quality knowledge learning balanced computational efficiency with the integration of new knowledge into LVLMs. Visual instruction tuning consisted of two phases: (i) Single-Image Training, where the model learned to perform visual tasks using instructions from single images, and (ii) OneVision Training, where the model learned 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 was updated, whereas all components including LLM were updated in the following three 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 8, 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.

H.2 Evaluation Results

Table 9 presents the evaluation results of our primary draft model, LLaVA-1.5 68M, on the OCRBench (Liu et al., 2024d) 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.

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 8: Details of hyperparameters used in LLaVA-OV training

Fig. 7 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 9: 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 multi-modality, the injection of multimodal information benefits the custom-trained draft model.

I 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 I.1). We then provide details on replacing image tokens in text-only and caption drafting (Appendix I.2).

I.1 Prompt Formats for Each Dataset

We use the following prompt formats for their 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.

LLaVA-Bench (In-the-Wild) <*s*> *USER: <image> For the following question, provide a detailed explanation of your reasoning leading to the answer. [QUESTION] ASSISTANT:*

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

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

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

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

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

I.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.

J Details of Each Dataset

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

J.1 Benchmark Datasets

LLaVA-Bench (In-the-Wild) (Liu et al., 2023) A dataset for comparing the performance of vision-language models against state-of-the-art proprietary models. Each prompt is provided with an image, a caption, and a reference answer from text-only GPT-4. Prompt styles include question answering, image description, and complex reasoning. In total, the dataset contains 60 unique prompts and 24 unique images.

`https://huggingface.co/datasets/lmms-lab/llava-bench-in-the-wild`

DocVQA (Mathew et al., 2021) A dataset designed for visual question answering on document images, comprising 50,000 questions over 12,000+ diverse document images.

`https://huggingface.co/datasets/lmms-lab/LMMs-Eval-Lite`

POPE (Li et al., 2023b) A multimodal question answering dataset that asks binary (yes or no) questions about whether certain objects are present in an image. The subset used for evaluation in our work contains 100 pairs of images and questions.

`https://huggingface.co/datasets/lmms-lab/LMMs-Eval-Lite`

MMVet (Yu et al., 2023) A benchmark designed to evaluate large multimodal models on complex tasks, focusing on the integration of six core vision-language capabilities: recognition, OCR, knowledge, language generation, spatial awareness, and math.

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

Spot the Difference (Jhamtani and 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`

IEdit (Tan et al., 2019) A dataset to train models to describe the relationship between images via editing instructions. The subset used for evaluation in our work contains 100 image pairs of a source image and a target image, accompanied by instructions on how to transform the source image into the target.

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

MagicBrush (Zhang et al., 2024c) 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`

J.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`

J.3 Time Analysis of LVLM Inference Stages

To analyze how the number of input images affects the LVLM inference time, we select ChartQA (Masry et al., 2022), Spot the Difference (Jhamtani and 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. 9 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.

Draft Model			Benchmark Datasets (First Turn)								OOD Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	PSV	VIST
LLaVA-1.5	68M	M	2.28	2.15	2.56	2.21	2.19	1.96	2.34	2.24	1.19	1.16
		T	2.19	2.08	2.31	2.16	2.23	2.34	2.27	2.23	2.05	2.05
		C	2.22	2.15	2.50	2.17	2.29	2.36	2.31	2.29	2.08	2.10
		P	2.22	2.08	2.42	2.17	2.22	2.27	2.23	2.23	2.07	2.09
		MT	2.25	2.15	2.47	2.21	2.31	2.37	2.40	2.31	1.94	1.91
		MT*	2.26	2.16	2.52	2.21	2.29	2.39	2.36	2.31	2.02	2.04
		MTCP	2.27	2.15	2.49	2.22	2.32	2.39	2.40	2.32	2.10	2.13
		MTCP*	2.29	2.17	2.51	2.22	2.30	2.40	2.35	2.32	2.11	2.14
Draft Model			Benchmark Datasets (Second Turn)								NLP Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	NQ	GSM8K
LLaVA-1.5	68M	M	2.1	1.96	2.78	2.18	1.61	1.53	1.83	2.00	1.98	2.25
		T	2.32	2.23	2.91	2.56	1.87	2.01	2.08	2.28	2.03	2.30
		C	2.28	2.29	2.94	2.58	1.87	2.01	2.10	2.30	2.03	2.29
		P	2.28	2.09	2.95	2.49	1.86	2.00	2.08	2.25	2.01	2.26
		MT	2.29	2.24	2.93	2.54	1.81	1.91	2.02	2.25	2.02	2.28
		MT*	2.29	2.23	2.93	2.56	1.85	1.99	2.05	2.27	2.03	2.29
		MTCP	2.32	2.29	3.03	2.59	1.86	2.00	2.08	2.31	2.02	2.27
		MTCP*	2.34	2.31	3.04	2.59	1.86	2.01	2.09	2.32	2.02	2.28

Table 10: Full experimental results with LLaVA-1.5 68M (draft model) and LLaVA-1.5 7B (target model)

Draft Model			Benchmark Datasets (First Turn)								OOD Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	PSV	VIST
LLaVA-1.5	68M	M	2.01	1.96	2.58	2.14	1.89	1.77	2.05	2.06	1.19	1.15
		T	1.97	1.92	2.27	2.11	1.85	2.12	2.03	2.04	1.79	1.90
		C	2.00	1.97	2.48	2.14	1.92	2.16	2.04	2.10	1.78	1.93
		P	2.00	1.90	2.41	2.11	1.87	2.05	1.97	2.04	1.78	1.92
		MT	2.02	1.97	2.47	2.14	1.94	2.14	2.08	2.11	1.75	1.80
		MT*	2.02	1.97	2.50	2.14	1.94	2.15	2.07	2.11	1.79	1.89
		MTCP	2.02	1.99	2.50	2.15	1.94	2.18	2.09	2.12	1.81	1.93
		MTCP*	2.03	1.99	2.52	2.15	1.94	2.19	2.08	2.13	1.82	1.95
Draft Model			Benchmark Datasets (Second Turn)								NLP Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	NQ	GSM8K
LLaVA-1.5	68M	M	1.88	1.88	2.78	2.08	1.52	1.43	1.67	1.89	1.94	2.12
		T	2.05	2.09	2.84	2.45	1.76	1.83	1.93	2.14	1.99	2.17
		C	2.07	2.12	2.92	2.48	1.75	1.84	1.94	2.16	1.99	2.17
		P	2.04	1.97	2.89	2.39	1.75	1.82	1.93	2.11	1.96	2.13
		MT	2.03	2.08	2.88	2.41	1.69	1.75	1.86	2.10	1.97	2.15
		MT*	2.04	2.10	2.88	2.42	1.71	1.78	1.87	2.11	1.97	2.15
		MTCP	2.06	2.13	2.95	2.46	1.74	1.82	1.94	2.16	1.98	2.16
		MTCP*	2.07	2.14	2.97	2.47	1.75	1.83	1.95	2.17	1.98	2.16

Table 11: Full experimental results with LLaVA-1.5 68M (draft model) and LLaVA-NeXT 7B (target model)

Draft Model			Benchmark Datasets (First Turn)								OOD Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	PSV	VIST
LLaVA-1.5	160M	M	2.29	2.29	3.06	2.44	2.17	2.04	2.26	2.36	1.23	1.24
		T	2.25	2.19	2.56	2.36	2.20	2.41	2.24	2.32	1.96	2.08
		C	2.26	2.26	2.87	2.42	2.26	2.46	2.29	2.40	2.05	2.19
		P	2.29	2.23	2.94	2.42	2.27	2.49	2.32	2.42	2.08	2.20
		MT	2.31	2.30	2.95	2.44	2.24	2.46	2.32	2.43	1.89	1.97
		MT*	2.26	2.31	3.01	2.44	2.19	2.49	2.31	2.43	1.95	2.07
		MTCP	2.31	2.30	3.02	2.44	2.30	2.54	2.34	2.46	2.06	2.20
		MTCP*	2.31	2.31	3.06	2.45	2.31	2.54	2.33	2.47	2.07	2.22
Draft Model			Benchmark Datasets (Second Turn)								NLP Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	NQ	GSM8K
LLaVA-1.5	160M	M	2.15	2.14	3.17	2.45	1.74	1.6	1.94	2.17	2.28	2.48
		T	2.37	2.31	3.04	2.72	2.11	2.14	2.33	2.43	2.3	2.55
		C	2.40	2.38	3.11	2.75	2.12	2.16	2.33	2.46	2.29	2.56
		P	2.39	2.35	3.25	2.76	2.08	2.13	2.33	2.47	2.29	2.51
		MT	2.37	2.37	3.21	2.77	2.04	2.06	2.23	2.44	2.29	2.54
		MT*	2.34	2.36	3.21	2.75	2.11	2.18	2.27	2.46	2.30	2.53
		MTCP	2.42	2.38	3.22	2.78	2.08	2.13	2.30	2.47	2.30	2.54
		MTCP*	2.42	2.39	3.22	2.79	2.09	2.14	2.31	2.48	2.30	2.54

Table 12: Full experimental results with LLaVA-1.5 160M (draft model) and LLaVA-NeXT 7B (target model)

Draft Model			Benchmark Datasets (First Turn)								OOD Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	PSV	VIST
LLaVA-1.5	160M	M	1.96	1.93	2.22	2.10	1.91	1.98	1.98	2.01	1.65	1.73
		T	1.99	1.91	2.02	2.10	1.86	2.24	1.99	2.02	1.81	1.86
		C	2.00	1.96	2.17	2.13	1.92	2.20	1.98	2.05	1.79	1.84
		P	1.99	1.91	2.13	2.10	1.96	2.19	2.02	2.04	1.87	1.91
		MT	2.01	1.95	2.21	2.14	1.93	2.22	2.02	2.07	1.84	1.89
		MT*	2.01	1.96	2.23	2.13	1.96	2.27	1.99	2.08	1.83	1.88
		MTCP	2.02	1.97	2.28	2.14	1.97	2.26	2.08	2.10	1.88	1.93
		MTCP*	2.02	1.97	2.29	2.14	1.97	2.27	2.07	2.10	1.88	1.93
Draft Model			Benchmark Datasets (Second Turn)								NLP Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	NQ	GSM8K
LLaVA-1.5	160M	M	1.97	1.96	2.37	2.11	1.63	1.64	1.77	1.92	1.87	2.16
		T	2.06	2.08	2.57	2.39	1.79	1.86	1.96	2.10	1.97	2.23
		C	2.07	2.12	2.67	2.43	1.79	1.86	1.94	2.13	1.96	2.23
		P	2.04	2.09	2.62	2.44	1.78	1.85	1.93	2.11	1.94	2.23
		MT	2.07	2.10	2.60	2.41	1.75	1.81	1.90	2.09	1.95	2.23
		MT*	2.07	2.10	2.60	2.42	1.76	1.83	1.91	2.10	1.95	2.23
		MTCP	2.09	2.14	2.71	2.45	1.78	1.86	1.93	2.14	1.96	2.24
		MTCP*	2.09	2.14	2.69	2.46	1.79	1.86	1.93	2.14	1.96	2.24

Table 13: Full experimental results with LLaVA-OV 68M (draft model) and LLaVA-NeXT 7B (target model)

Draft Model			Benchmark Datasets (First Turn)								OOD Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	PSV	VIST
LLaVA-1.5	68M	M	2.01	1.97	2.44	2.04	1.87	1.75	1.95	2.00	1.18	1.15
		T	1.98	1.93	2.23	2.02	1.86	2.07	1.94	2.00	1.73	1.89
		C	1.98	1.96	2.40	2.05	1.89	2.09	1.92	2.04	1.74	1.90
		P	1.97	1.91	2.35	2.02	1.87	1.98	1.87	2.00	1.74	1.90
		MT	2.00	1.97	2.36	2.06	1.91	2.07	1.99	2.05	1.69	1.79
		MT*	2.02	1.94	2.39	2.06	1.89	2.07	1.97	2.05	1.73	1.86
		MTCP	2.00	1.99	2.39	2.06	1.93	2.11	2.00	2.07	1.75	1.91
		MTCP*	2.01	1.99	2.40	2.06	1.93	2.11	1.99	2.07	1.75	1.93
Draft Model			Benchmark Datasets (Second Turn)								NLP Datasets	
Type	Size	Method	LLaVA-W	DocVQA	POPE	MMVet	IEdit	MB	Spot	Avg.	NQ	GSM8K
LLaVA-1.5	68M	M	1.85	1.82	2.73	2.02	1.52	1.43	1.72	1.87	1.97	2.18
		T	2.00	2.03	2.70	2.31	1.77	1.81	1.98	2.09	2.01	2.22
		C	2.03	2.08	2.77	2.35	1.77	1.81	2.00	2.12	2.01	2.23
		P	2.01	1.91	2.75	2.27	1.76	1.79	1.98	2.07	1.99	2.19
		MT	2.02	2.02	2.74	2.29	1.69	1.73	1.91	2.06	1.99	2.20
		MT*	2.01	2.01	2.79	2.30	1.74	1.78	1.95	2.08	2.00	2.22
		MTCP	2.04	2.07	2.82	2.32	1.74	1.79	1.97	2.11	2.00	2.21
		MTCP*	2.04	2.07	2.82	2.32	1.75	1.79	1.98	2.11	2.00	2.22

Table 14: Full experimental results with LLaVA-1.5 68M (draft model) and LLaVA-NeXT 13B (target model)

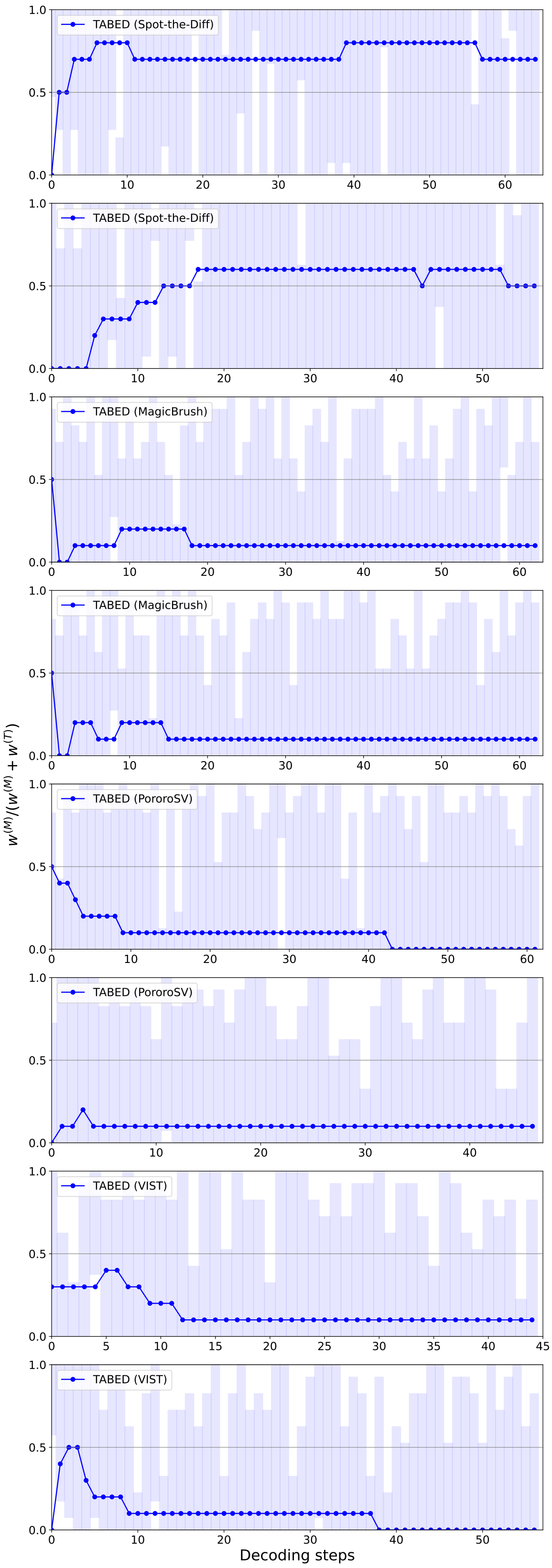


Figure 4: Additional qualitative samples of dynamic ensemble weights

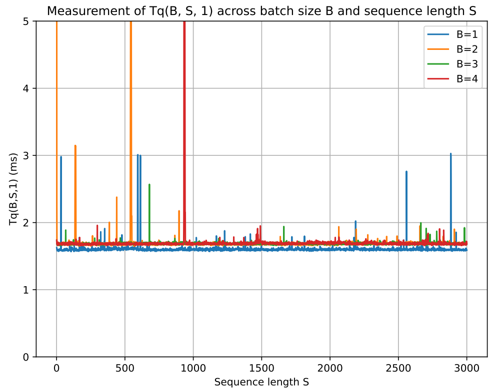


Figure 5: Empirical demonstration of Remarks 2 and 3.

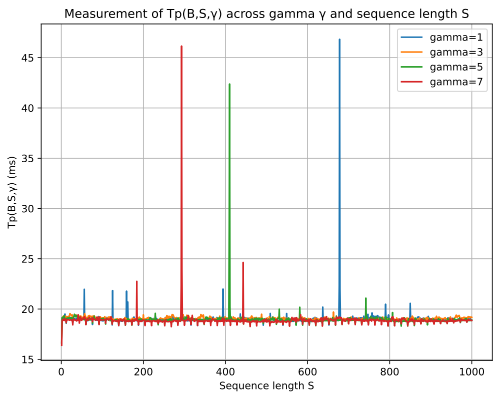
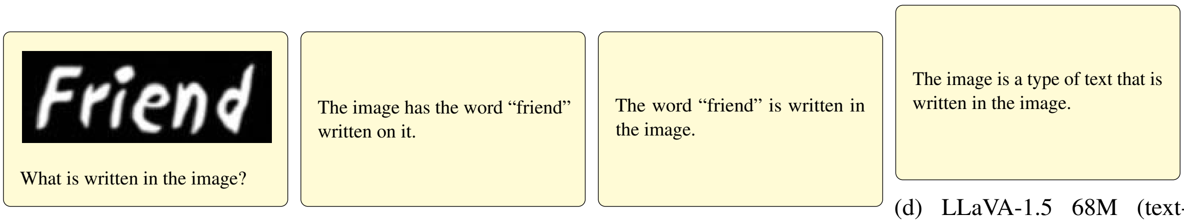
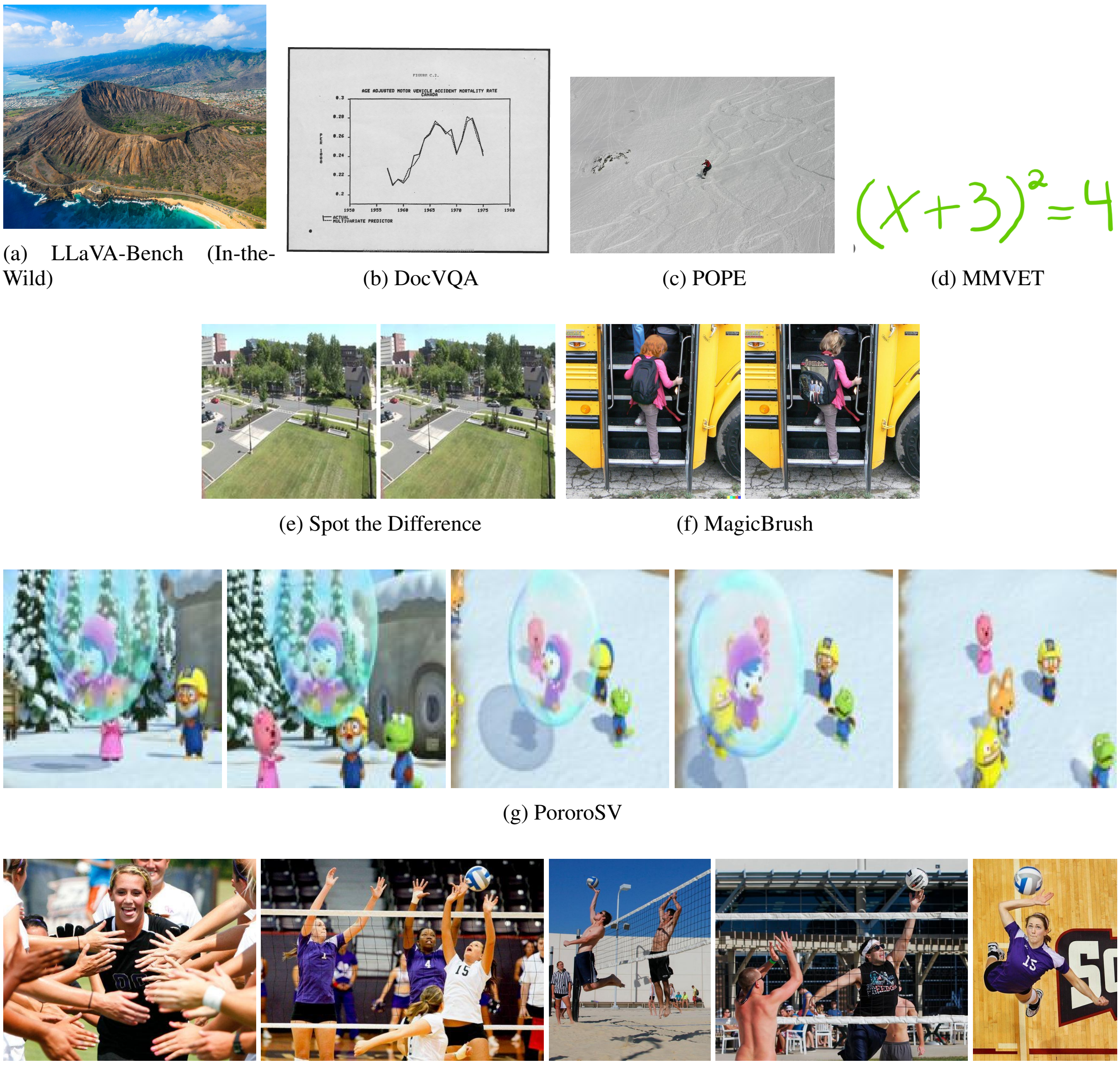


Figure 6: Empirical demonstration of Remark 1.



(a) Instruction (b) LLaVA-1.5 7B (c) LLaVA-1.5 68M (d) LLaVA-1.5 68M (text-only)

Figure 7: 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.



(a) LLaVA-Bench (In-the-Wild) (b) DocVQA (c) POPE (d) MMVET

(e) Spot the Difference (f) MagicBrush

(g) PororoSV

(h) VIST

Figure 8: Qualitative image samples of benchmark and OOD datasets. The corresponding questions and answers are presented in Appendix E.

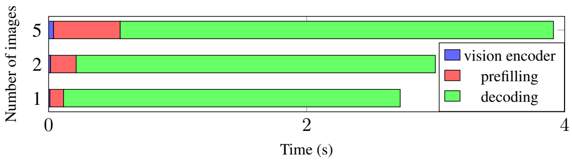


Figure 9: Inference time analysis for the LLaVA-1.5 7B model. Although the time for vision encoder and prefilling increases with the number of images, the decoding stage still dominates.