ViSpec: Accelerating Vision-Language Models with Vision-Aware Speculative Decoding

Jialiang Kang[1,2]* **Han Shu**[2] **Wenshuo Li**[2] **Yingjie Zhai**[2] **Xinghao Chen**[2]†
[1]Peking University
[2]Huawei Noah's Ark Lab
jkang@stu.pku.edu.cn
{han.shu,liwenshuo,zhaiyingjie,xinghao.chen}@huawei.com

Abstract

Speculative decoding is a widely adopted technique for accelerating inference in large language models (LLMs), yet its application to vision-language models (VLMs) remains underexplored, with existing methods achieving only modest speedups ($< 1.5\times$). This gap is increasingly significant as multimodal capabilities become central to large-scale models. We hypothesize that large VLMs can effectively filter redundant image information layer by layer without compromising textual comprehension, whereas smaller draft models struggle to do so. To address this, we introduce **Vision-Aware Speculative Decoding (ViSpec)**, a novel framework tailored for VLMs. ViSpec employs a lightweight vision adaptor module to compress image tokens into a compact representation, which is seamlessly integrated into the draft model's attention mechanism while preserving original image positional information. Additionally, we extract a global feature vector for each input image and augment all subsequent text tokens with this feature to enhance multimodal coherence. To overcome the scarcity of multimodal datasets with long assistant responses, we curate a specialized training dataset by repurposing existing datasets and generating extended outputs using the target VLM with modified prompts. Our training strategy mitigates the risk of the draft model exploiting direct access to the target model's hidden states, which could otherwise lead to shortcut learning when training solely on target model outputs. Extensive experiments validate ViSpec, achieving, to our knowledge, the first substantial speedup in VLM speculative decoding. Code is available at https://github.com/KangJialiang/ViSpec.

1 Introduction

The success of large language models (LLMs) has spurred the development of vision-language models (VLMs) capable of processing and generating content from both visual and textual inputs. Recent VLMs, such as LLaVA-NeXT (LLaVA-1.6) [15] and Qwen2.5-VL [1], demonstrate impressive performance in tasks including image captioning, visual question answering, and multimodal dialogue. However, as VLMs increase in scale and complexity, their inference times grow substantially, posing significant challenges for practical deployment.

Speculative decoding [13] has proven effective in accelerating LLM inference by employing a smaller, faster draft model to propose candidate token sequences, which the larger target model verifies in parallel. Correct predictions from the draft model enable the target model to skip costly autoregressive computations, resulting in significant speedups. While speculative decoding is well-established for

*Work done during an internship at Huawei Noah's Ark Lab.
†Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

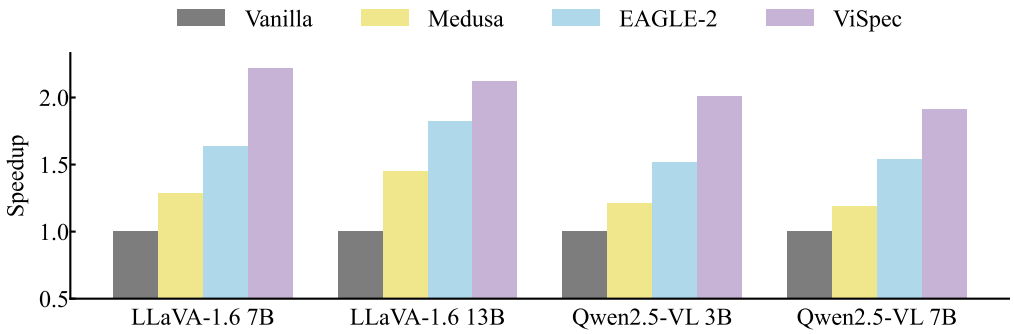

Figure 1: Speedup ratios of various methods at temperature = 0, evaluated on the GQA test set using four VLMs: LLaVA-v1.6-Vicuna-7B, LLaVA-v1.6-Vicuna-13B, Qwen2.5-VL-3B-Instruct, and Qwen2.5-VL-7B-Instruct.

LLMs, its application to VLMs remains underexplored, with prior approaches [8, 12] achieving only marginal speedups. We attribute this limitation to fundamental differences between textual and visual data. Text, honed over centuries, is abstract and information-dense, whereas images, despite their visual richness, often contain considerable redundancy. Consequently, small draft models struggle to extract pertinent visual information while preserving textual coherence in VLMs.

To address this challenge, we propose **Vision-Aware Speculative Decoding (ViSpec)**, a novel speculative decoding framework designed specifically for VLMs. ViSpec incorporates a lightweight vision adaptor module to compress numerous image tokens into a compact, informative representation. These compressed tokens are seamlessly integrated into the draft model's attention layers, retaining the original image's positional information. Furthermore, drawing inspiration from target-aware feature injection in EAGLE [19, 18, 20], we extract a global feature vector for each input image and augment all subsequent text tokens with this feature until the next image is encountered. This mechanism equips the draft model with robust global visual context, enhancing prediction accuracy.

A significant obstacle in developing speculative decoding for VLMs is the scarcity of large-scale, publicly available multimodal datasets with extended assistant responses. To overcome this, we repurpose existing datasets by modifying prompts and leveraging the target VLM to generate long responses, thereby creating synthetic training data. Although the draft model could potentially exploit access to the target model's hidden states during training, the randomness in the target model's sampling strategy and our adoption of multi-token prediction, inspired by DeepSeek [21], effectively mitigate this risk.

Our experiments demonstrate that ViSpec significantly outperforms existing speculative decoding methods for VLMs, achieving substantial speedups without compromising generation quality. To our knowledge, this work represents the first meaningful acceleration of VLM inference through speculative decoding.

Our main contributions are as follows:

- We introduce Vision-Aware Speculative Decoding (ViSpec), a speculative decoding framework tailored for VLMs.
- We propose dual integration mechanisms—attention integration and feature augmentation—to enable a small draft model to efficiently incorporate visual context.
- We develop a training strategy that extends existing vision-language datasets to include long-response tasks, leveraging multi-token prediction.
- We empirically validate ViSpec on four popular VLMs, achieving notable speed improvements and establishing the first practical acceleration in this domain.

2 Related Work

2.1 Speculative Decoding

Speculative decoding [13] accelerates inference in LLMs by utilizing a smaller draft model to propose candidate token sequences, which the target model verifies in parallel. This method achieves speedups of 3–4$\times$ on text-only tasks while preserving output fidelity [13]. Subsequent advancements have refined this approach. Self-speculative decoding [36] derives the draft model from the target model, minimizing training overhead through shared parameters. On-the-fly adaptation methods, such as SwiftDecode [33], dynamically adjust the draft model during inference to adapt to varying input distributions, enhancing robustness across tasks. Specifier [25] employs an ensemble of small models for parallel draft generation, increasing prediction diversity and speedup. Cascade Speculative Drafting [5] uses a sequence of draft models with increasing complexity to balance speed and accuracy, achieving up to 3.5$\times$ speedup on large-scale LLMs. Medusa [2] integrates multiple decoding heads into the target model's architecture, eliminating the need for a separate draft model while maintaining comparable performance. The EAGLE series [19, 18, 20] improves draft predictions by injecting target-aware hidden states, aligning the draft model closely with the target model's output distribution. A concurrent work, EAGLE-3 [20], adopts a multi-token prediction strategy termed *training-time test*, though its performance gains depend heavily on scaling training data, with limited improvements when dataset size is fixed. Additionally, SpecTr [29] optimizes token acceptance rates using optimal transport, while REST [10] enhances draft predictions with retrieval-based external knowledge, excelling in knowledge-intensive tasks. Recent surveys [34] offer detailed analyses of speculative decoding techniques, discussing their trade-offs and open challenges.

2.2 Vision-Language Models

Vision-language models (VLMs) integrate visual and textual inputs to address tasks such as image captioning, visual question answering, and multimodal dialogue. Recent advancements have led to state-of-the-art models like LLaVA-NeXT [15] and Qwen2.5-VL [1], which combine powerful vision encoders, such as CLIP [26], with large-scale LLMs to achieve superior performance across diverse applications. Models like BLIP-2 [16] and MiniGPT-4 [38] leverage pretrained vision and language components with learnable interfaces to bridge modality gaps, enabling efficient multimodal processing. However, the computational complexity of processing high-dimensional image inputs, coupled with the autoregressive nature of text generation, results in significant inference latency, posing challenges for real-time deployment. Efforts to address these issues include efficient vision encoders, such as those proposed in EVA-CLIP [28], and optimized training strategies that reduce memory overhead. Despite these advances, inference efficiency remains a critical bottleneck, motivating the exploration of speculative decoding for VLMs.

2.3 Speculative Decoding for Vision-Language Models

The application of speculative decoding to VLMs is an emerging area with limited prior work. The only notable effort, by [8], applied speculative decoding to LLaVA-7B using a small language-only draft model, achieving up to 1.5$\times$ speedup. Their experiments with a small VLM draft model incorporating an image encoder yielded only marginal gains, highlighting the challenge of effectively processing visual information in the draft model due to the high redundancy and computational complexity of image inputs. These limitations highlight the need for specialized frameworks that can effectively integrate visual and textual information in draft models while maintaining high prediction accuracy. Our proposed ViSpec framework addresses these challenges by introducing vision-aware mechanisms to enhance the draft model's ability to process multimodal inputs efficiently.

3 Preliminaries

3.1 Speculative Decoding

Speculative decoding [3, 13, 25, 29] is a lossless acceleration technique for LLMs that alternates between a drafting stage and a verification stage to expedite autoregressive decoding. Let t_i denote the i-th token in a sequence, and let $T_{a:b} = \{t_a, t_{a+1}, \ldots, t_b\}$ represent a token sequence. Given a prefix $T_{1:j}$, speculative decoding proceeds as follows: in the drafting stage, a lightweight draft

model autoregressively generates a sequence of k tokens, $\hat{T}_{j+1:j+k}$, along with their probabilities $\hat{p}_{j+i}(\hat{t}_{j+i})$ for each token $\hat{t}_{j+i}$. In the verification stage, the target model evaluates $\hat{T}_{j+1:j+k}$, computing its own probabilities $p_{j+i}(\hat{t}_{j+i})$. Each draft token $\hat{t}_{j+i}$ is accepted with probability $\min\left(1, \frac{p_{j+i}(\hat{t}_{j+i})}{\hat{p}_{j+i}(\hat{t}_{j+i})}\right)$. If a token is rejected, a new token is sampled from the normalized distribution $\text{norm}\left(\max\left(0, p_{j+i}(\hat{t}_{j+i}) - \hat{p}_{j+i}(\hat{t}_{j+i})\right)\right)$, and subsequent draft tokens are discarded. This process ensures that the generated sequence maintains the same probability distribution as that produced by the target model without acceleration, preserving distributional consistency. We enhance this approach by adopting the context-aware dynamic draft tree from EAGLE-2 [18], an improvement over the draft tree in [25], which enables the draft model to generate multiple candidate tokens per position, facilitating more efficient exploration of the token space.

3.2 Vision-Language Models

Modern VLMs [1, 15, 16] typically extend a base large language model (LLM) by incorporating visual information through a vision encoder. Formally, given an input image I, a vision encoder $\mathcal{E}_v$ maps it to a sequence of visual embeddings $V_{1:r} = \mathcal{E}_v(I) \in \mathbb{R}^{r \times d}$, where r denotes the number of visual embeddings and d is the embedding dimension. Let $\mathcal{E}_t$ represent the text embedding layer of the LLM. For a multimodal input sequence comprising both visual and textual tokens, the joint input representation is constructed as $H_{1:n} = \mathcal{E}_t(T_{1:k}) \oplus V_{1:r} \oplus \mathcal{E}_t(T_{k+1:j})$, where $\oplus$ denotes sequence concatenation. The VLM processes this hybrid sequence autoregressively. Notably, the LLM architecture remains unchanged; the only modification is the inclusion of visual embeddings $V_{1:r}$ within the input sequence. Since the output space remains the text token vocabulary $\mathcal{V}$ and the autoregressive generation mechanism is preserved, speculative decoding methods designed for LLMs can, in principle, be directly applied to VLMs by treating visual embeddings as part of the input context. Formally, for any prefix containing visual embeddings $V_{1:r}$ and text tokens $T_{1:j}$, the speculative decoding procedure outlined in Sec. 3.1 remains valid, with probabilities p_{j+i} and $\hat{p}_{j+i}$ implicitly conditioned on $V_{1:r}$.

4 Method

4.1 Overcoming Redundancy: Image Embedding Compression

In speculative decoding, the draft model is typically a smaller, shallower version of the target model. We demonstrate that a single-layer Transformer-based draft model is fundamentally limited in processing long, redundant sequences, particularly when redundant image patches (e.g., uniform color blocks) dominate the input.

Consider a sequence of $R+1$ image and text embeddings $e_1, \ldots, e_{R+1} \in \mathbb{R}^d$, where R embeddings are identical, i.e., $e_{r_1} = \ldots = e_{r_R} = s$, and a unique token at position u has embedding $e_u = t$. The Transformer has a single self-attention layer with weight matrices $W_q, W_k, W_v \in \mathbb{R}^{d \times d}$. Ignoring positional encoding and attention scaling for simplicity, the output at position i is:

$$y_i = \sum_{j=1}^{R+1} \alpha_{ij} v_j, \quad \text{where} \quad \alpha_{ij} = \frac{\exp\left(q_i k_j^\top\right)}{\sum_{k=1}^{R+1} \exp\left(q_i k_k^\top\right)}, \tag{1}$$

with $q_i = W_q e_i$, $k_j = W_k e_j$, and $v_j = W_v e_j$. For the R redundant tokens, we have:

$$q_i k_r^\top = (W_q e_i)(W_k e_r)^\top = W_q e_i s^\top W_k^\top, \tag{2}$$

which is identical across all redundant tokens. As R increases, the attention weight to the unique token becomes:

$$\alpha_{iu} = \frac{\exp(B)}{R \exp(A) + \exp(B)}, \tag{3}$$

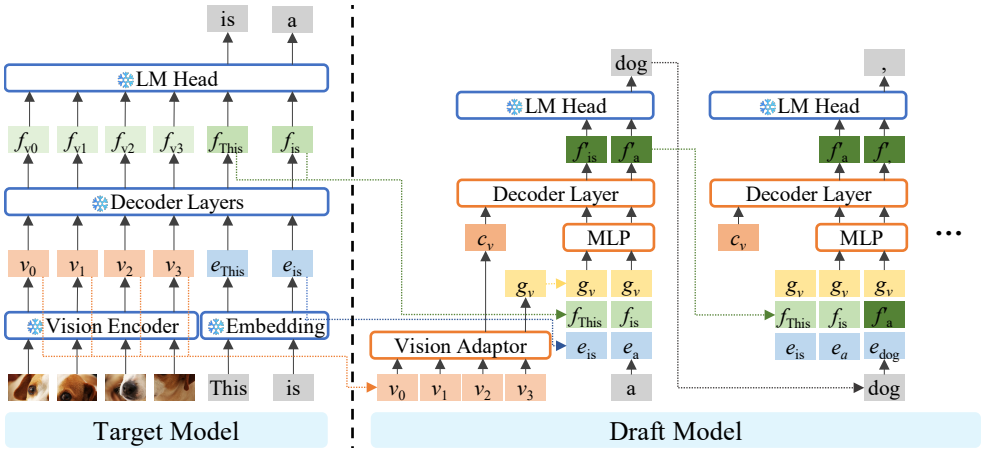

Figure 2: Overview of the ViSpec framework. Given an input image and text prompt, ViSpec compresses image tokens using a lightweight vision adaptor to produce a small set of visual tokens. These tokens are prepended to the text input and fed into the draft model's attention mechanism. A global visual feature vector, extracted from the compressed image tokens, is injected into the draft model's text generation process. The figure illustrates two decoding steps of the draft model, where f denotes the target model's last-layer hidden state, f' the draft model's last-layer hidden state, v visual embeddings, e text embeddings, c compressed image tokens, and g the global visual feature vector.

where $A = W_q e_i s^\top W_k^\top$ is the score for redundant tokens, and $B = W_q e_i t^\top W_k^\top$ is the score for the unique token. As $R \to \infty$, the denominator is dominated by $R\exp(A)$, causing $\alpha_{iu} \to 0$. Meanwhile, $\alpha_{ir} \to \frac{1}{R}$ for each redundant token, so the output approximates:

$$y_i \approx \sum_{m=1}^{R} \frac{v_{r_m}}{R} = W_v s, \tag{4}$$

effectively averaging over the redundant tokens and neglecting the unique token. Furthermore, it has been proven theoretically that a $K+1$ layer network is required to handle a nesting complexity of K [30], indicating that shallow draft models struggle to extract useful information from long, redundant image embeddings, thus constraining their effectiveness in speculative decoding for VLMs.

The limitations of shallow draft models in processing multimodal sequences necessitate a specialized approach for speculative decoding in VLMs. Drawing on insights from [22], which emphasize the critical role of draft token generation speed in achieving end-to-end speedup, we propose a lightweight Q-Former-inspired [16] vision adaptor (see Fig. 3). This module utilizes a lightweight Transformer encoder with a fixed set of learnable query vectors. The visual features extracted from the input image serve as key and value inputs to the Transformer's attention layers, while the learnable query vectors function as queries. Through this attention mechanism, each query vector selectively attends to relevant portions of the visual features, condensing them into a small set of compact feature vectors. These vectors, significantly fewer than the original embeddings, act as compressed visual embeddings. They are seamlessly integrated into the draft model's attention mechanism, preserving the positional information of the original image by maintaining relative spatial locations. By splitting the input into a concise image sequence and a compressed-image-plus-text sequence, we improve the draft model's efficiency in handling long multimodal sequences.

4.2 Addressing Lost-in-the-Middle: Global Visual Feature Integration

While image embedding compression can condense visual tokens into a compact sequence amidst a series of text tokens, this approach poses challenges in speculative decoding, which prioritizes long assistant responses. The *lost in the middle* effect [23], particularly pronounced in shallow models such as our draft model, causes performance degradation when critical visual information is situated in the middle of long contexts, leading to a U-shaped performance curve.

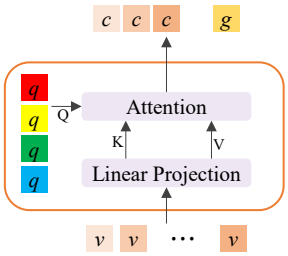

Figure 3: Architecture of the vision adaptor module. A compact Transformer encoder with fixed learnable query vectors q processes input visual embeddings v through an attention layer, yielding a small set of compressed image tokens c and a single global visual feature g.

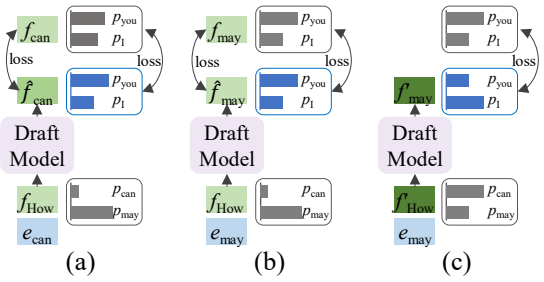

Figure 4: Comparison of training procedures: (a) EAGLE training, (b) training with greedy target model responses without multi-token prediction, and (c) ViSpec training. Here, e denotes input embeddings, f represents target model hidden states, $\hat{f}$ indicates EAGLE draft model hidden states, f' denotes ViSpec draft model hidden states, and p signifies token probabilities.

Although compressed visual embeddings provide a compact representation of images for the draft model, they may not fully capture the holistic visual context. As discussed in Sec. 4.1, simply increasing the number of image tokens is suboptimal, as shallow draft models lack the capacity to effectively attend to lengthy, redundant sequences. Moreover, as generated text sequences lengthen, image tokens become increasingly obscured within the text, exacerbating the *lost in the middle* effect [23] and undermining the draft model's ability to maintain consistent visual grounding. This often results in reduced coherence between the generated text and the input image. To address these challenges, we propose extracting a global feature vector from the input image and integrating it into each subsequent text token, ensuring persistent access to global visual context throughout the text generation process.

We derive the global feature vector from the final output of the vision adaptor module. This vector is transformed and incorporated into the hidden states of all subsequent text tokens in the draft model. Formally, at each text position t, we compute the augmented hidden state f_t^{aug} as:

$$f_t^{\text{aug}} = f_t + W_g g, \tag{5}$$

where f_t is the original hidden state, g denotes the global visual feature vector, and W_g is a learned projection matrix. This architectural enhancement equips the draft model with continuous visual context, enhancing its ability to generate accurate speculative tokens that maintain strong alignment with the input image across extended generation sequences.

4.3 Dataset Generation and Training

Training an effective draft model for speculative decoding requires a large, diverse dataset of high-quality target model outputs. For VLMs, this necessitates multimodal datasets with extended assistant responses, which are scarce in the public domain. To address this, we propose a novel data generation strategy that repurposes existing multimodal datasets, even those lacking long responses. We modify prompts in datasets such as visual question answering or image captioning to elicit longer, more descriptive responses from the target VLM. For instance, in visual question answering, we rephrase simple questions to request detailed explanations or reasoning. Similarly, for image captioning, we prompt the VLM to produce elaborate descriptions. This approach yields a robust synthetic training dataset without requiring manual annotation.

A potential concern is that the draft model might overfit to the target model's outputs by exploiting its hidden states during training. However, the randomness in the target model's sampling strategy, driven by a temperature parameter, and the adoption of multi-token prediction, as proposed by DeepSeek [21], mitigate this risk. We illustrate this in Fig. 4. In (a) EAGLE's text-only training, the target model is likely to generate "may" instead of "can" when the ground truth is "can." Here, the target model's hidden state f_{How} acts as noisy augmentation, enabling the draft model to learn corrective behavior, as f_{can} is conditioned on both f_{How} and the embedding e_{can}. In (b) training with

greedy target model responses without multi-token prediction, the draft model's inputs (e.g., f_{How}, e_{may}) exhibit a one-to-one correspondence, and the supervision f_{may} is conditioned solely on f_{How} (ignoring previous inputs, as f already contains most of the information), effectively reducing to a single-step Medusa [2]. In (c) ViSpec's training procedure, we avoid this issue by using sampling to disrupt one-to-one correspondences between hidden states f and embeddings e. Additionally, we incorporate the draft model's own hidden states f' as input, which serve a similar corrective role as in (a) when the target model deviates from the ground truth, allowing the draft model to learn self-correction without manually crafted datasets. We optimize the loss:

$$L = \text{CrossEntropy}(p_i, \hat{p}_i), \tag{6}$$

where p_i and $\hat{p}_i$ denote the target model's and draft model's probabilities for the i-th token, respectively.

5 Experiments

5.1 Experimental Setup

Hardware. All experiments are conducted on a single GPU. Draft models are trained using 8x GPUs.

Models. We evaluate our proposed **Vision-Aware Speculative Decoding (ViSpec)** framework on four open-source vision-language models: LLaVA-v1.6-Vicuna-7B [15], LLaVA-v1.6-Vicuna-13B [15], Qwen2.5-VL-3B-Instruct [1], and Qwen2.5-VL-7B-Instruct [1]. We use their official weights and configurations from the Hugging Face Transformers library [32].

Baselines. We compare ViSpec against two established speculative decoding frameworks originally designed for language models: Medusa [2] and EAGLE-2 [18]. To adapt these frameworks for VLMs, we modify their input pipelines to process image patch embeddings from the VLM's original vision encoder, enabling the draft models to generate speculative tokens conditioned on both visual and textual contexts. This adaptation is feasible as both Medusa and EAGLE-2 rely on general token prediction mechanisms that are theoretically compatible with multimodal sequences, provided the draft model can handle visual inputs.

Training Datasets. We train the draft models for both baselines and ViSpec using a two-stage process. Initially, all draft models are trained on the ShareGPT dataset, comprising 68,000 dialogue iterations, to establish a robust text-based foundation. For multimodal training, we fine-tune the baseline draft models (Medusa and EAGLE-2) on 68,000 samples randomly selected from the LLaVA Visual Instruct Pretrain LCS dataset [15], enabling them to process visual inputs. For ViSpec, we augment this dataset with synthetic long assistant responses generated using the target VLM, as described in Sec. 4.3.

Tasks. We evaluate performance on eight diverse multimodal benchmarks: ScienceQA (SQA) [24], MM-Vet [35], MME [7], TextVQA [27], COCO Captions (COCO Caps) [4], VizWiz [9], GQA [11], and SEED-Bench [14]. These datasets cover tasks such as visual question answering, image captioning, and multimodal evaluation. To ensure generalizability, we use consistent model weights across all tasks without task-specific fine-tuning. Following [8], we design prompts to elicit long, detailed responses from the models.

Metrics. As ViSpec employs strict speculative decoding to preserve the target model's generation quality, quality evaluation is unnecessary. We focus on acceleration performance, measured using the following metrics:

- **Average Acceptance Length τ:** The average number of tokens accepted from the draft model per drafting-verification cycle.
- **Speedup Ratio:** The ratio of inference time for standard autoregressive decoding to that for different speculative decoding methods.

5.2 Comparison with Baselines

Table 1 and Figure 1 present a comprehensive evaluation of ViSpec's acceleration performance compared to Medusa [2] and EAGLE-2 [18] across multiple vision-language models and tasks. The

Table 1: Speedup ratios and average acceptance lengths τ for different methods. For a fair comparison, we do not relax the draft token acceptance condition of Medusa under non-greedy settings as proposed in the original paper; instead, we adopt the same acceptance condition as EAGLE-2. Speedup ratios are computed based on the average time required to generate each token.

		SQA		MM-Vet		TextVQA		MME		COCO Caps		VizWiz		GQA		SEED-Bench		Avg.	
Model	Method	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup
Temperature=0																			
LLaVA-1.6 7B	Medusa	0.72	1.41x	0.73	1.42x	0.77	1.46x	0.70	1.41x	0.66	1.61x	0.76	1.38x	0.73	1.29x	0.72	1.38x	0.72	1.42x
	EAGLE-2	2.48	2.14x	0.63	1.48x	0.63	1.25x	1.25	1.68x	1.24	1.80x	1.15	1.40x	1.74	1.64x	1.40	1.59x	1.31	1.62x
	ViSpec	**2.86**	**2.37x**	**2.83**	**2.52x**	**2.95**	**2.90x**	**2.84**	**2.55x**	**3.30**	**3.22x**	**3.16**	**2.67x**	**2.88**	**2.22x**	**3.03**	**2.22x**	**2.98**	**2.58x**
LLaVA-1.6 13B	Medusa	0.84	1.61x	0.80	1.47x	0.89	1.51x	0.79	1.47x	0.75	1.48x	0.81	1.45x	0.85	1.45x	0.82	1.40x	0.82	1.48x
	EAGLE-2	2.02	2.12x	1.64	1.59x	1.71	1.91x	1.81	1.85x	1.83	2.01x	1.98	1.90x	2.10	1.82x	2.03	1.66x	1.89	1.86x
	ViSpec	**2.76**	**2.57x**	**2.73**	**2.34x**	**2.78**	**2.43x**	**2.78**	**2.36x**	**3.18**	**2.82x**	**2.93**	**2.26x**	**2.95**	**2.12x**	**3.04**	**2.16x**	**2.89**	**2.38x**
Qwen2.5-VL 3B	Medusa	0.57	1.07x	0.60	1.12x	0.66	1.08x	0.59	1.12x	0.62	1.21x	0.60	1.16x	0.65	1.21x	0.61	1.15x	0.61	1.14x
	EAGLE-2	1.18	1.41x	1.03	1.30x	0.98	1.26x	1.07	1.38x	1.40	1.60x	1.11	1.32x	1.39	1.52x	1.11	1.32x	1.16	1.39x
	ViSpec	**1.99**	**1.87x**	**2.13**	**1.81x**	**2.15**	**1.85x**	**1.96**	**1.82x**	**2.37**	**2.15x**	**2.22**	**1.71x**	**2.28**	**2.01x**	**2.37**	**1.78x**	**2.19**	**1.87x**
Qwen2.5-VL 7B	Medusa	0.60	1.13x	0.59	1.06x	0.58	1.05x	0.59	1.19x	0.61	1.11x	0.59	1.09x	0.64	1.19x	0.62	1.05x	0.60	1.11x
	EAGLE-2	1.40	1.49x	1.19	1.36x	1.14	1.23x	1.29	1.54x	1.46	1.50x	1.27	1.20x	1.53	1.54x	1.42	1.32x	1.34	1.40x
	ViSpec	**2.19**	**1.84x**	**2.16**	**1.74x**	**2.21**	**1.72x**	**2.15**	**1.96x**	**2.27**	**1.99x**	**2.31**	**1.71x**	**2.30**	**1.91x**	**2.34**	**1.55x**	**2.24**	**1.80x**
Temperature=1																			
LLaVA-1.6 7B	Medusa	0.58	1.36x	0.58	1.37x	0.57	1.32x	0.56	1.35x	0.58	1.67x	0.57	1.29x	0.60	1.19x	0.59	1.32x	0.58	1.36x
	EAGLE-2	1.78	2.17x	0.51	1.34x	0.41	1.11x	1.02	1.53x	1.03	1.78x	0.77	1.32x	1.33	1.47x	0.98	1.57x	0.98	1.54x
	ViSpec	**2.06**	**2.20x**	**1.94**	**1.99x**	**1.78**	**1.93x**	**1.96**	**1.98x**	**2.36**	**3.05x**	**2.32**	**2.21x**	**2.11**	**1.83x**	**2.16**	**1.94x**	**2.09**	**2.14x**
LLaVA-1.6 13B	Medusa	0.68	1.41x	0.67	1.44x	0.66	1.42x	0.66	1.40x	0.67	1.40x	0.64	1.37x	0.70	1.37x	0.68	1.37x	0.67	1.40x
	EAGLE-2	1.51	1.98x	1.29	1.73x	1.26	1.72x	1.45	1.78x	1.54	1.83x	1.46	1.72x	1.64	1.73x	1.60	1.79x	1.47	1.79x
	ViSpec	**2.02**	**2.25x**	**1.98**	**2.15x**	**1.90**	**2.08x**	**2.07**	**2.08x**	**2.43**	**2.39x**	**2.04**	**2.01x**	**2.19**	**2.03x**	**2.22**	**2.07x**	**2.11**	**2.13x**
Qwen2.5-VL 3B	Medusa	0.52	1.02x	0.48	1.02x	0.46	0.99x	0.46	1.02x	0.51	1.03x	0.46	0.99x	0.55	1.13x	0.49	1.03x	0.49	1.03x
	EAGLE-2	0.92	1.25x	0.70	1.19x	0.70	1.06x	0.84	1.26x	0.97	1.28x	0.84	1.19x	1.02	1.31x	0.86	1.16x	0.86	1.21x
	ViSpec	**1.49**	**1.49x**	**1.23**	**1.39x**	**1.32**	**1.38x**	**1.45**	**1.58x**	**1.42**	**1.50x**	**1.39**	**1.43x**	**1.49**	**1.59x**	**1.55**	**1.42x**	**1.42**	**1.47x**
Qwen2.5-VL 7B	Medusa	0.56	1.05x	0.51	0.95x	0.49	0.96x	0.51	1.02x	0.52	1.00x	0.50	1.02x	0.53	1.02x	0.53	1.02x	0.52	1.01x
	EAGLE-2	1.19	1.52x	0.92	1.19x	0.88	1.08x	1.00	1.23x	1.08	1.22x	0.94	1.13x	1.11	1.32x	1.04	1.19x	1.02	1.18x
	ViSpec	**1.82**	**1.62x**	**1.57**	**1.47x**	**1.51**	**1.37x**	**1.61**	**1.49x**	**1.63**	**1.50x**	**1.88**	**1.53x**	**1.61**	**1.56x**	**1.70**	**1.38x**	**1.66**	**1.49x**

table reports the average acceptance length τ and speedup ratios, calculated as the ratio of the average time required for standard autoregressive decoding to that of each method per token, under two temperature settings (0 and 1).

ViSpec consistently outperforms both Medusa and EAGLE-2 across all evaluated tasks and models, achieving the highest speedup ratios and τ values. For instance, at temperature 0 with LLaVA-v1.6-Vicuna-7B, ViSpec achieves a speedup of $2.90\times$ on TextVQA, surpassing EAGLE-2 ($1.25\times$) and Medusa ($1.46\times$) by a wide margin. Similarly, with LLaVA-v1.6-Vicuna-13B at temperature 0, ViSpec delivers a speedup of $2.57\times$ on ScienceQA, compared to EAGLE-2 ($2.12\times$) and Medusa ($1.61\times$). At temperature 1, ViSpec maintains its advantage, achieving a speedup of $2.25\times$ on ScienceQA with LLaVA-v1.6-Vicuna-13B, outperforming EAGLE-2 ($1.98\times$) and Medusa ($1.41\times$). These results underscore ViSpec's superior acceleration capabilities, with speedup ratios ranging from $1.37\times$ to $3.22\times$, compared to EAGLE-2 ($1.06\times$ to $2.17\times$) and Medusa ($0.95\times$ to $1.67\times$).

ViSpec demonstrates robust performance and generalizability across a diverse set of tasks, with notably high acceptance lengths on TextVQA, VizWiz, SEED-Bench, and COCO Captions. This suggests that its vision-aware approach effectively handles the varied sequential patterns inherent in these tasks. In contrast, the performance of EAGLE-2 and Medusa is more task-dependent. While they perform adequately on tasks like ScienceQA, they struggle on others, such as TextVQA and MM-Vet, particularly when compared to ViSpec. This indicates that their general-purpose draft mechanisms may not adapt as effectively to the complexities of visual-linguistic sequences.

Performance also varies across model architectures. LLaVA-1.6 models generally achieve higher speedup ratios and acceptance lengths compared to Qwen2.5-VL models. Such differences can be

attributed to the significantly larger vocabulary sizes of Qwen models, potentially increasing the complexity of token prediction.

5.3 Ablation Studies

Impact of Compressed Image Embedding Count. We evaluate the effect of varying the number of compressed image embeddings from 1 to 64 on ViSpec's performance, with results shown in Tab. 2. When the number remains significantly smaller than the original thousands of image embeddings, increasing the count has minimal impact on the average acceptance length τ. However, it reduces the speedup ratio due to the increased computational load on the draft model during token generation. A single compressed image embedding adequately captures essential visual information, prompting us to adopt one compressed embedding in our final implementation.

Table 2: Impact of varying the number of compressed image embeddings on ViSpec's performance across three datasets, measured by average acceptance length τ and speedup ratio.

Image Embeddings	COCO Captions		GQA		MME	
	τ	Speedup	τ	Speedup	τ	Speedup
1	3.30	3.22x	2.88	2.22x	2.84	2.55x
4	3.24	3.24x	2.84	2.24x	2.74	2.35x
16	3.23	3.21x	2.84	2.20x	2.76	2.38x
64	3.25	2.71x	2.86	1.91x	2.76	2.42x

Table 3: Ablation study on the effectiveness of ViSpec's components across three datasets, measured by average acceptance length τ and speedup ratio, with EAGLE-2 as the baseline.

Components	COCO Captions		GQA		MME	
	τ	Speedup	τ	Speedup	τ	Speedup
baseline	1.24	1.80x	1.74	1.64x	1.25	1.68x
+image embedding compression	2.04	2.37x	2.15	1.92x	2.04	1.83x
+global visual injection	2.14	2.42x	2.25	2.03x	2.14	1.95x
+dataset generation	3.30	3.22x	2.88	2.22x	2.84	2.55x

Effectiveness of Each Component. We conduct an ablation study to assess the contribution of ViSpec's core components: image embedding compression, global visual feature injection, and dataset generation. Using EAGLE-2 [18] as the baseline, we report the average acceptance length and speedup ratio across the COCO Captions, GQA, and MME datasets, as shown in Tab. 3. Adding image embedding compression increases the speedup ratio by up to 30%, enabling the draft model to efficiently process visual information. Incorporating global visual feature injection further improves speedup by 7%, underscoring its role in maintaining persistent visual context and enhancing multi-modal coherence. The inclusion of dataset generation yields an additional 30% speedup, equipping the draft model to handle extended multimodal sequences effectively. Together, these components synergistically enhance ViSpec's acceleration performance while ensuring robust performance across diverse tasks.

Vision Adaptor Overheads. While the vision adaptor increases the draft model's parameter count, it theoretically reduces the prefill computation by processing fewer visual tokens. However, as draft models are small and efficient, we observe no statistically significant change in prefill latency (Tab. 4). The minor variations recorded are attributed to measurement noise.

Output Length vs. Speedup. Table 5 illustrates the relationship between the average output length and the achieved end-to-end speedup across various datasets. As expected, longer generation sequences generally yield higher speedup ratios, since they offer more opportunities for successful draft model predictions. Despite this trend, our method demonstrates robust performance, providing significant acceleration even on datasets characterized by shorter responses.

Table 4: Analysis of vision adaptor overheads during the prefill stage on the COCO Captions dataset.

Model	Params (M)		GFLOPs		Latency (s)	
	Base	+Adap	Base	+Adap	Base	+Adap
LLaVA-1.6 7B	367	451	956	179	0.227	0.231
LLaVA-1.6 13B	534	665	1460	279	0.334	0.334
Qwen2.5-VL 3B	404	425	57.3	18.3	0.002	0.004
Qwen2.5-VL 7B	826	890	172	55.5	0.018	0.016

Table 5: Relationship between output length and speedup ratio.

Dataset	Tokens	Speedup
GQA	46.25	2.22x
SEED-Bench	57.66	2.22x
SQA	74.07	2.37x
VizWiz	105.91	2.26x
MME	115.01	2.55x
MM-Vet	171.13	2.52x
COCO Caps	236.04	3.21x
TextVQA	353.58	2.90x

6 Conclusion

We introduce **Vision-Aware Speculative Decoding (ViSpec)**, the first framework to achieve significant acceleration for vision-language models (VLMs) through speculative decoding. By integrating compressed image embeddings, persistent global visual feature injection, and synthetic long-response dataset generation, ViSpec addresses key limitations in processing multimodal sequences with shallow draft models. Our experiments demonstrate speedups of up to $3.22\times$ across diverse VLMs and tasks, establishing ViSpec as a pioneering solution for multimodal inference acceleration. Despite this breakthrough, ViSpec's absolute speedup trails state-of-the-art text-only methods. We identify two primary avenues for improvement: first, curating higher-quality multimodal training datasets with greater conversational depth to enhance the draft model's predictive accuracy; second, optimizing vision encoder architectures, potentially via dynamic patch reduction or neural compression, to reduce visual processing overhead. These advancements, coupled with hardware-aware kernel optimizations, could bridge the performance gap between multimodal and text-only speculative decoding, enabling real-time deployment of advanced VLMs.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[2] Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D Lee, Deming Chen, and Tri Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads. In *International Conference on Machine Learning*, pages 5209–5235. PMLR, 2024.

[3] 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.

[4] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[5] Ziyi Chen, Xiaocong Yang, Jiacheng Lin, Chenkai Sun, Kevin Chang, and Jie Huang. Cascade speculative drafting for even faster llm inference. *Advances in Neural Information Processing Systems*, 37:86226–86242, 2024.

[6] Muhammad Iqbal Hasan Chowdhury, Kien Nguyen, Sridha Sridharan, and Clinton Fookes. Hierarchical relational attention for video question answering. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 599–603. IEEE, 2018.

[7] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023.

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

[9] Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3608–3617, 2018.

[10] Zhenyu He, Zexuan Zhong, Tianle Cai, Jason Lee, and Di He. Rest: Retrieval-based speculative decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1582–1595, 2024.

[11] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019.

[12] Minjae Lee, Wonjun Kang, Minghao Yan, Christian Classen, Hyung Il Koo, and Kangwook Lee. In-batch ensemble drafting: Toward fast and robust speculative decoding for multimodal language models. *OpenReview.net*, 2024.

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

[14] Bohao Li, Yuying Ge, Yixiao Ge, Guangzhi Wang, Rui Wang, Ruimao Zhang, and Ying Shan. Seed-bench: Benchmarking multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13299–13308, 2024.

[15] 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. *CoRR*, 2024.

[16] 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*, pages 19730–19742. PMLR, 2023.

[17] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206, 2024.

[18] Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle-2: Faster inference of language models with dynamic draft trees. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7421–7432, 2024.

[19] Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle: speculative sampling requires rethinking feature uncertainty. In *Proceedings of the 41st International Conference on Machine Learning*, pages 28935–28948, 2024.

[20] Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle-3: Scaling up inference acceleration of large language models via training-time test. *arXiv preprint arXiv:2503.01840*, 2025.

[21] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

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

[23] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12, 2024.

[24] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

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

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[27] 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*, pages 8317–8326, 2019.

[28] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*, 2023.

[29] Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. Spectr: Fast speculative decoding via optimal transport. *Advances in Neural Information Processing Systems*, 36:30222–30242, 2023.

[30] Mingze Wang and E Weinan. Understanding the expressive power and mechanisms of transformer for sequence modeling. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[31] Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, Wei Yu, and Dacheng Tao. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 7907–7915, 2025.

[32] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[33] Heming Xia, Yongqi Li, Jun Zhang, Cunxiao Du, and Wenjie Li. Swift: On-the-fly self-speculative decoding for llm inference acceleration. *arXiv preprint arXiv:2410.06916*, 2024.

[34] Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. In *ACL (Findings)*, 2024.

[35] Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. Mm-vet: Evaluating large multimodal models for integrated capabilities. In *International conference on machine learning*. PMLR, 2024.

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

[37] YiFan Zhang, Huanyu Zhang, Haochen Tian, Chaoyou Fu, Shuangqing Zhang, Junfei Wu, Feng Li, Kun Wang, Qingsong Wen, Zhang Zhang, et al. Mme-realworld: Could your multimodal llm challenge high-resolution real-world scenarios that are difficult for humans? In *The Thirteenth International Conference on Learning Representations*, 2025.

[38] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction outline the proposed ViSpec framework, its contributions (vision-aware speculative decoding, image token compression, and dataset generation), and its scope (accelerating VLM inference).

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Sec. 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: See Sec. 4.1.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Sec. 5.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code will be made available upon acceptance, and we use only publicly available datasets.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: See Sec. 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: We fix the random seed for all experiments, so the results are deterministic.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Sec. 5.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research does not involve human subjects, crowdsourcing, or sensitive data. There is no indication of violating the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our work presents a technical advancement for accelerating inference in VLMs without modifying their outputs or capabilities. The acceleration method itself does not introduce new societal risks beyond those already present in the underlying models.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release any new models or datasets.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use only publicly available datasets, and we properly credit the original authors.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The answer NA means that the paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The paper uses VLMs (LLaVA, Qwen2.5-VL) to generate synthetic training data with long responses (Sec. 4.3). This usage is clearly described.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Implementation Details

Vanilla. We utilize models from the Hugging Face Transformers library with the PyTorch backend and pre-allocated KV cache. All other methods build upon these models.

Medusa. We implement a 1-layer, 3-head Medusa model, adhering to its default configuration. For training on both text-only and vision-language datasets, we use a learning rate of 3e-5, a batch size of 8, and the AdamW optimizer. The model is trained for 20 epochs with a 1-epoch warmup and linear learning rate decay. We set a maximum sequence length of 2048 for both dataset types. For inference, we adopt EAGLE-2's draft tree structure, configuring a total of 30 draft tokens, a tree depth of 3, and selecting 8 nodes during the expansion phase across all models and tasks.

EAGLE-2. We employ a 1-layer EAGLE-2 model, following its default settings. Training on text-only and vision-language datasets uses a learning rate of 3e-5, a batch size of 8, and the AdamW optimizer. The model is trained for 20 epochs with a 1-epoch warmup and linear learning rate decay, with a maximum sequence length of 2048 for both dataset types. For inference, we use EAGLE-2's draft tree with 30 draft tokens, a tree depth of 3, and 8 nodes selected during expansion, applied uniformly across all models and tasks.

ViSpec. We implement a single-layer draft model that mirrors a decoder layer of the target model. For training on text-only and vision-language datasets, we use a learning rate of 3e-6, a batch size of 8, and the AdamW optimizer. The model is trained for 20 epochs with a 1-epoch warmup and linear learning rate decay, supporting a maximum sequence length of 2048 for both dataset types. During inference, we adopt EAGLE-2's draft tree structure, configuring 30 draft tokens, a tree depth of 3, and selecting 8 nodes during expansion, applied consistently across all models and tasks.

Generation Prompts. For training dataset generation, we append the prompt "*Please answer with at least 1000 words.*" to each sample to elicit long responses. For inference, we use task-specific prompts to encourage detailed responses. For visual question answering (VQA) tasks, the prompt is: "*Please answer with an explanation.*" For optical character recognition (OCR) tasks, the prompt is: "*Perform an OCR task on the provided image. Extract the text accurately and provide a detailed explanation of the process. Ensure the response is comprehensive and well-structured.*" For captioning tasks, the prompt is: "*Provide a detailed description of the given image.*" For ScienceQA, we use its official chain-of-thought prompt to generate the answer, followed by the lecture and explanation (QCM→ALE).

B Additional Experiments

B.1 Experiments on High-Resolution Datasets

We conducted experiments on high-resolution datasets [31, 37], where ViSpec continues to demonstrate strong performance. Table 6 compares ViSpec against the EAGLE-2 baseline using both LLaVA-1.6 7B and Qwen2.5-VL 7B.

Table 6: Performance on high-resolution datasets, comparing average acceptance length τ and speedup ratio.

Model	Dataset	Method	τ	Speedup
LLaVA-1.6 7B	HR-Bench 4K	EAGLE-2	1.43	1.52x
		ViSpec	**2.86**	**1.93x**
	MME-RealWorld	EAGLE-2	1.42	1.75x
		ViSpec	**2.85**	**2.35x**
Qwen2.5-VL 7B	HR-Bench 4K	EAGLE-2	0.34	0.90x
		ViSpec	**2.16**	**1.29x**
	MME-RealWorld	EAGLE-2	0.52	0.95x
		ViSpec	**2.11**	**1.37x**

Notably, Qwen-VL does not cap its input image token count, resulting in a longer prefill time for high-resolution datasets. Since speculative decoding accelerates only the decoding stage, this extended

prefill duration reduces the overall speedup ratio. However, ViSpec's robust average acceptance length τ indicates that the decoding phase itself is still effectively accelerated.

B.2 Experiments with Temporal Data

In principle, ViSpec could be more effective for video inputs, as videos contain temporal redundancy in addition to the spatial redundancy found in static images. From an input processing standpoint, this task is not fundamentally different from handling image patches, as video inputs are typically processed as a sequence of frame embeddings. To test this hypothesis, we apply our draft model, which was trained exclusively on static image data, directly to video tasks without fine-tuning.

For this preliminary experiment, we compress each video frame into a single embedding, average their global features, and evaluate the Qwen2.5-VL 7B model on the MSVD-QA [6] and MVBench [17] datasets. MSVD-QA is a video question-answering task, while MVBench is a benchmark evaluating temporal understanding across 20 different tasks. We limit the input frames, as processing more would lengthen the prefill time, thereby reducing the speedup gained from the accelerated decoding stage. The results are presented in Tab. 7.

Table 7: Performance of Qwen2.5-VL 7B on video datasets, comparing average acceptance length τ and speedup ratio.

Dataset	Method	τ	Speedup
MSVD-QA	EAGLE-2	1.10	1.22x
	ViSpec	**2.16**	**1.46x**
MVBench	EAGLE-2	0.83	0.83x
	ViSpec	**2.09**	**1.32x**

The results demonstrate that ViSpec achieves a notable speedup even without video-specific training. Developing a dedicated framework optimized for video data remains a promising direction for future work.