AASD: Accelerate Inference by Aligning Speculative Decoding in Multimodal Large Language Models

Chaoqun Yang
Tsinghua University

Yuanda Zhang
Tsinghua University

Xiying Huang
Tsinghua University

Abstract

Multimodal Large Language Models (MLLMs) have achieved notable success in visual instruction tuning, yet their inference is time-consuming due to the autoregressive decoding of Large Language Model (LLM) backbone. Traditional methods for accelerating inference, including model compression and migration from language model acceleration, often compromise output quality or face challenges in effectively integrating multimodal features. To address these issues, we propose **AASD**, a novel framework for **Accelerating inference with refined KV Cache** and **Aligning speculative decoding in MLLMs**. Our approach leverages the target model's cached Key-Value (KV) pairs to extract vital information for generating draft tokens, enabling efficient speculative decoding. To reduce the computational burden associated with long multimodal token sequences, we introduce a KV Projector to compress the KV Cache while maintaining representational fidelity. Additionally, we design a Target-Draft Attention mechanism that optimizes the alignment between the draft and target models, achieving the benefits of real inference scenarios with minimal computational overhead. Extensive experiments on mainstream MLLMs demonstrate that our method achieves up to a 2× inference speedup without sacrificing accuracy. This study not only provides an effective and lightweight solution for accelerating MLLM inference but also introduces a novel alignment strategy for speculative decoding in multimodal contexts, laying a strong foundation for future research in efficient MLLMs. Code is availiable at `https://anonymous.4open.science/r/ASD-F571`.

1 Introduction

The rapid development of Multimodal Large Language Models (MLLMs) is exerting a profound impact on the whole world. These models have demonstrated remarkable capabilities in various domains, including visual understanding, question answering, logical reasoning *etc.* [1, 3, 31, 32], and they are expected to further drive artificial intelligence toward Artificial General Intelligence. However, despite the widespread recognition of MLLMs' accuracy and versatility, inference speed remains a significant challenge in practical applications. Current multimodal models generally rely on autoregressive decoding, which, while ensuring coherent outputs, greatly limits inference speed. As model size and complexity increase, so do the computational cost and resource demands associated with their inference [10, 35]. So how to accelerate the inference of the MLLMs is an important problem, which has both academic value and practical significance.

At present, research focusing on accelerating the inference of MLLMs remains relatively scarce. Algorithmic research primarily falls into three categories (Figure 1):

- Traditional model compression techniques: such as distillation [28, 34], quantization [16, 23], and pruning [30, 45] aim to reduce model size and computational burden but encounter issues with fidelity loss and implementation complexity.

Preprint. Under review.

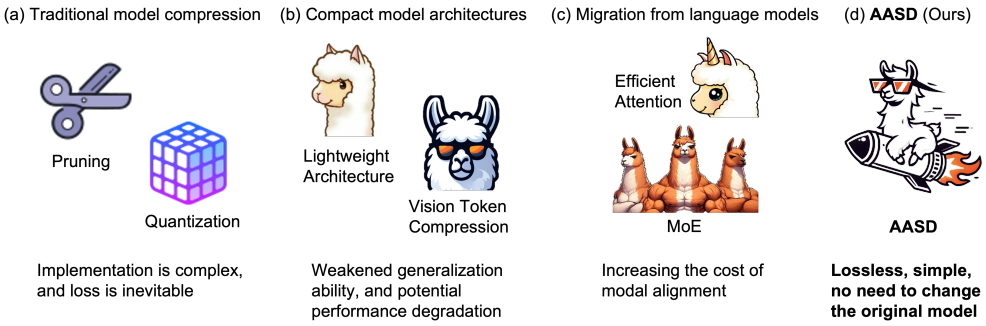


Figure 1: Comparison of mainstream methods and ours.

- Compact architectures: leveraging lightweight models [27, 40, 44] and efficient encoders such as vision token compression [33, 37]; however, they may sacrifice multimodal alignment and overall performance.
- Adaptation of LLM acceleration techniques to MLLMs: such as efficient attention [6, 27] and MoE [19], offers potential solutions but faces challenges in integrating visual features.

Most of these methods tend to compromise output quality, which is unacceptable for many precision-demanding applications. Recently, Speculative Decoding (SD) has emerged as a promising approach for accelerating inference without compromising accuracy, and it has shown effective utility in (Natural Language Processing) NLP tasks [11, 36, 43]. Recent work has attempted to migrate this technique to multimodal scenarios. Gagrani et al. (2024) proposes the speculative decoding with a language-only draft model to improve inference efficiency. However, a language-only draft model struggles to align closely with MLLMs, leading to limited acceleration benefits that fail to meet practical demands. On the other hand, speculative decoding usually requires a small model that has a high acceptance rate and low inference time to achieve an acceleration effect [36, 43]. This requirement is relatively easy to fulfill in NLP tasks but poses a significant challenge in multimodal contexts, where tokens are longer and encode more extensive information. Smaller models with fewer parameters struggle to capture these rich multimodal representations, hindering their ability to effectively align with the target model. Conversely, using a larger draft model to better capture this information adds computational overhead, thereby increasing inference latency.

To address these challenges, we proposed **AASD**: **A**ccelerate the inference with refined KV Cache and **A**lign **S**peculative **D**ecoding in Multimodal Large Language Models:

(1) Accelerate the inference with refined KV Cache. Typical speculative decoding uses the *draft-then-verify* paradigm. During inference, the draft model first generates multiple draft tokens, which are then verified in parallel by the target model in a single call. We observed that during inference, the target model caches Key-Value (KV) pairs, which contain significant information about the target distribution. Leveraging this insight, we designed a speculating module that utilizes the last layer's KV Cache from the target model to extract pertinent information for generating draft tokens. Considering that the Key and Value of multimodal tokens are relatively long, which increases the inference costs and the training difficulty of the speculating module, we introduced the KV Projector. This projector is designed to compress information from the multimodal KV Cache and then calculate Cross Attention with the Query from the speculating module.

(2) Align speculative decoding in Multimodal Large Language Models. Utilizing the target model's KV Cache during inference allows the draft model to learn representations closer to those of the target model, but it poses a challenge for training. When generating the *k*-th draft token at the *i*-th step during inference, it requires access to the first *i-1* KV pairs from the target model and the *i*-th to *(i+k-1)*-th KV pairs from the draft model. Replicating this scenario exactly in training is complex and costly. A regular lower triangular causal mask cannot achieve this goal, and accurately mirroring inference conditions would incur $O(n^2)$ computational and memory costs, which is a tough problem. To overcome this, we developed the Target-Draft Attention (T-D Attn), an optimized mechanism that achieves the effects of real inference with minimal additional computation, thereby addressing the challenge of aligning training with the inference scenario in a computationally efficient manner.

Our extensive experiments on various mainstream MLLMs validate that the proposed method achieves an effective acceleration for inference (around 2× speedup). Our work not only provides a lightweight acceleration solution for efficient MLLM inference, but also introduces a novel approach for aligning SD with multimodal contexts, establishing a solid foundation for future research in efficient MLLM inference. The main contributions of this study are summarized as follows: ***i)*** Lightweight inference acceleration method: A lightweight, efficient inference acceleration framework specifically designed for MLLMs, achieving significant speed improvements while maintaining model accuracy, providing a feasible solution for real-time MLLM applications. ***ii)*** Innovative multimodal alignment strategy: we proposed an innovative strategy to align SD with multimodal contexts, enabling the model to effectively leverage the synergy between visual and textual information in multimodal input scenarios, further enhancing decoding efficiency and information fusion. ***iii)*** Strong Empirical Evidence of Acceleration: Through extensive empirical validation, the proposed method achieved up to 2× speedup in inference, demonstrating its high efficiency and robustness in practical applications.

2 Related Work

2.1 Accelerating Inference of Multimodal Large Language Models

Multimodal Large Language Models (MLLMs) are large language models (LLMs) that integrate multimodal capabilities, enabling them to receive, interpret, and generate outputs based on multimodal information [41]. These models are engineered to tackle complex tasks, including image description, visual question answering (VQA), multimodal translation, and instruction-following by jointly processing and generating natural language along with visual and other perceptual data.

A typical MLLM architecture comprises three core components: a multimodal encoder, a LLM backbone, and a multimodal connector to coordinate multiple modalities [9]. The input to an MLLM often includes raw multimodal data, such as images, audio, and text, which are first processed by modality-specific encoders and transformed into a standardized format suitable for the model's processing. The model's output depends on the task at hand, for instance, in VQA, the output may be a text answer to a visual query, while in image description, the output is a textual description of the image content.

MLLMs usually require significant computational resources for both training and inference. When handling high-resolution images or performing complex reasoning tasks, the inference latency can be considerable, which poses challenges for real-time applications [17]. As a result, accelerating MLLM inference has become an active research area, with key approaches including: ***i)*** Vision Token Compression. Techniques such as token pruning [14, 42] and the design of novel vision-language bridging modules [2, 8] aim to reduce the number of visual tokens, enhancing computational efficiency. ***ii)*** Bypassing Visual Tokens. Speculative decoding methods enable selective bypassing of image tokens and associated processing components [7], which reduces computational load by withdrawing visual tokens at specific layers [21]. ***iii)*** Hardware Acceleration. Solutions such as the development of elastic cache systems have been proposed to improve the inference efficiency of instruction-following multimodal models [24].

2.2 Speculative Decoding

Speculative Decoding uses the idea of *draft-then-verify* to fully leverage the parallel processing capabilities of GPU [18]. Specifically, this method initially utilizes a draft model, typically a small and rapidly executing model, to generate multiple draft tokens. Subsequently, the target model verifies these draft tokens in a single, parallel operation, reducing the number of auto-regressive decoding steps of the target model, thereby accelerating the overall inference process [36]. The speedup provided by speculative decoding mainly depends on the acceptance of the draft tokens [18, 38]. In other words, the alignment of the draft model and the target model largely determines the upper limit of acceleration.

3 Method

3.1 Preliminaries

Below we use lowercase letters for vectors and uppercase letters for matrices except $\boldsymbol{X}$, which represents model inputs. For ease of reading, all vectors default to row vectors for ease of reading and the representation of the partitioned matrix does not clearly distinguish the column and column directions, such as $K_n = (K_{1\sim i}, K_{i+1\sim n})$, where $K_n \in R^{n\times d}$, $K_{1\sim i} \in R^{i\times d}$ and $K_{i+1\sim n} \in R^{i\times d}$.

3.1.1 Multimodal Large Language Models

A typical MLLM comprises 3 modules: a modality encoder to encode visual inputs, a connector to map the visual encodings to text embeddings, and a LLM backbone to generate response. In inference, inputting an image $\boldsymbol{X}_I$ and a prompt text $\boldsymbol{X}_T$, the modality encoder converts the image $\boldsymbol{X}_I$ to visual encodings, which is then transformed into embeddings E_I in the text embedding space. Meanwhile, LLM's encoder converts the prompt text into text embeddings E_T. And then LLM backbone autoregressively generates tokens using these embeddings. The whole process can be expressed by the following formula:

$$x \sim p(x|\boldsymbol{X}_I, \boldsymbol{X}_T) = M(\boldsymbol{X}_I, \boldsymbol{X}_T) \tag{1}$$

where x denotes the token sampled from the probability space p generated by the model M conditioning on the input image $\boldsymbol{X}_I$ and prompt text $\boldsymbol{X}_T$.

3.1.2 Speculative Decoding

Speculative decoding is a lossless approach to accelerate the inference of autoregressive models, adopting a *draft-then-verify* paradigm. To expedite inference of a large model, commonly referred to as the *target model* (M_T), this method employs a small auxiliary model, referred to as the *draft model* (M_D), which is typically obtained through fine-tuning or distillation.

3.1.3 KV Cache

The core of LLM is the attention computing mechanism:

$$\boldsymbol{O} = \boldsymbol{A}\boldsymbol{V} = Softmax(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}})\boldsymbol{V} \tag{2}$$

where $\boldsymbol{O} \in R^{n\times d}$ is the attention output, $\boldsymbol{A} \in R^{n\times n}$ is the attention weight, $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} \in R^{n\times d}$ correspond to query states, key states, and value states respectively. n denotes the sequence length, and d denotes the hidden dimension. Typically, the computation of $\boldsymbol{A}$ incorporates a lower triangular matrix known as a causal mask to prevent subsequent tokens from attending to preceding tokens, thereby maintaining the autoregressive property of the model. For ease of reading, we use simplified symbols ($\boldsymbol{Q}\boldsymbol{K}^T$) to represent $Softmax(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}})$ below.

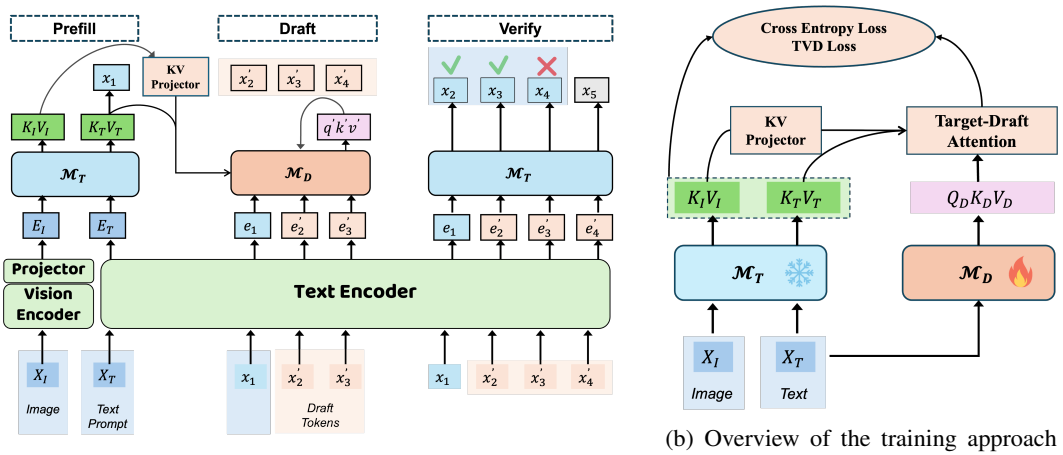
Due to the nature of autoregressive generation, where each computation step utilizes the $\boldsymbol{K}$ and $\boldsymbol{V}$ representations of preceding tokens, it is common practice to cache $\boldsymbol{K}$ and $\boldsymbol{V}$ at each inference step for subsequent use. This caching strategy is crucial for maintaining the efficiency of the generation process. The procedure in MLLMs can be detailed as follows:

(1) Prefilling stage. An input image $\boldsymbol{X}_I$ token sequence $\boldsymbol{X}_T$ is processed to compute the attention output. Concurrently, the key representations $\boldsymbol{K} = (\boldsymbol{K}_I, \boldsymbol{K}_T)$ and value representations $\boldsymbol{V} = (\boldsymbol{V}_I, \boldsymbol{V}_T)$ derived from this sequence are cached. This initial computation establishes the foundational context for the autoregressive generation process.

(2) Decoding stage. For each subsequent token input x, its corresponding query $\boldsymbol{q}$, key $\boldsymbol{k}$, and value $\boldsymbol{v}$ are computed. These are then concatenated with the previously cached $\boldsymbol{K}$ and $\boldsymbol{V}$ to calculate the attention outputs. That is $\boldsymbol{o} = (\boldsymbol{q}\tilde{\boldsymbol{K}})\tilde{\boldsymbol{V}}$, where $\tilde{\boldsymbol{K}} = Concat(\boldsymbol{K}, \boldsymbol{k}), \tilde{\boldsymbol{V}} = Concat(\boldsymbol{V}, \boldsymbol{v})$.

3.2 Accelerated Inference with Refined KV Cache

Our inference framework, shown in Figure 2a, addresses the unique challenges of applying Speculative Decoding (SD) in multimodal tasks. Unlike NLP tasks, achieving satisfactory performance with a



(a) Overview of the inference framework (**Accelerated Inference with Refined KV Cache**).

(b) Overview of the training approach (**Aligned Training by Target-Draft Attention**).

Figure 2: Overview of our method.

smaller draft model in multimodal settings is significantly more difficult. Current relatively compact models, such as KarmaVLM with a 0.4B vision encoder and a 0.5B LLM backbone [29], DeepSeek-VL with a 0.4B vision encoder and a 1.3B LLM backbone [25], and MiniCPM-V 2.0 with a 0.4B vision encoder and a 2.7B LLM backbone [40], still have a huge number of parameters. Moreover, it has been pointed out that a small MLLM as a draft model brings limited acceleration gain to SD, sometimes even worse than a draft model with language-only LLM [7]. This is most likely because learning directly from multimodal data is too difficult for small models. This motivated us to design **AASD**: An approach that accelerate the inference by effectively refining KV Cache generated by the target model to better align the draft model outputs with the target model. We structure our method in two key stages: Prefilling and Decoding.

(1) Prefilling stage. During the Prefilling stage, the target model processes the input image $\boldsymbol{X}_I$ and prompt text $\boldsymbol{X}_T$ to generate the initial KV Cache $\boldsymbol{K}_I, \boldsymbol{V}_I, \boldsymbol{K}_T, \boldsymbol{V}_T$ and the probability distribution over the next token. Although these KV Caches contain valuable information, the image KV Cache tends to be particularly large, creating both a learning barrier and a computational burden for the draft model. To address this, we introduce two feature projectors that compress the target model's KV Cache, enabling the draft model to better align with the target model during draft generation. Mathematically, this compression is represented as:

$$\boldsymbol{K}_I^* = \boldsymbol{W}_K \boldsymbol{K_I}, \boldsymbol{V}_I^* = \boldsymbol{W}_V \boldsymbol{V_I} \tag{3}$$

where

$$\boldsymbol{W}_K, \boldsymbol{W}_V \in R^{k \times n}, \boldsymbol{K}_I, \boldsymbol{V}_I \in R^{n \times d}, \boldsymbol{K}^*, \boldsymbol{V}^* \in R^{k \times d}$$

This operation compresses the lengthy sequence of visual key and value states into k tokens, here set to $k = 64$, effectively reducing approximately 90% of the redundant information. The target model then passes the compressed KV pairs, $\boldsymbol{K}_I^*$, $\boldsymbol{V}_I^*$, $\boldsymbol{K}_T$, and $\boldsymbol{V}_T$, to the draft model to generate the draft tokens.

(2) Decoding stage.

(2.1) Draft process. First, the draft model builds upon foregoing KV Cache generated by the target model to produce a sequence of candidate tokens. Specially, the draft model M_D receives token $\boldsymbol{x}_i$ from the target model and computes corresponding $\boldsymbol{q}_i', \boldsymbol{k}_i', \boldsymbol{v}_i'$, which are then combined with the target model's key states $\boldsymbol{K}_{i-1} = \boldsymbol{k}_{\leq i-1}$ and value states $\boldsymbol{V}_{i-1} = \boldsymbol{v}_{\leq i-1}$ up to the previous token, namely

$$\hat{\boldsymbol{K}}_i = (\boldsymbol{K}_{i-1},\ \boldsymbol{k}_i'), \hat{\boldsymbol{V}}_i = (\boldsymbol{V}_{i-1}, \boldsymbol{v}_i') \tag{4}$$

The draft model computes the attention using $\boldsymbol{q}_i', \hat{\boldsymbol{K}}_i, \hat{\boldsymbol{V}}_i$, and generated next draft token $\boldsymbol{x}_{i+1}'$. Then draft model continues to generate subsequent tokens by incorporating the newly generated tokens into the KV Cache, namely

$$x_{i+k+1}' = M_D(\boldsymbol{q'}_{i+k}, \hat{\boldsymbol{K}}_{i+k}, \hat{\boldsymbol{V}}_{i+k}) \tag{5}$$

where

$$\hat{\boldsymbol{K}}_{i+k} = (\boldsymbol{K}_{i-1}, \boldsymbol{K}'_{i\sim i+k}), \hat{\boldsymbol{V}}_{i+k} = (\boldsymbol{V}_{i-1}, \boldsymbol{V}'_{i\sim i+k})$$

(2.2) Verify process. The Verify process involves the target model validating the draft tokens. This validation is done by once forwarding on draft tokens of the target model, which computes the true token probabilities.

The target model receives the sequence of draft tokens $\boldsymbol{X}'_{i\sim i+\gamma} = (x'_{i+1}, \cdots, x'_{i+\gamma})$, and computes the true token probabilities $\boldsymbol{P}_{i\sim i+\gamma} = (\boldsymbol{p}_{i+1}, \cdots, \boldsymbol{p}_{i+\gamma})$, where

$$\boldsymbol{p}_{i+k+1} = M_T(x_{i+k}, \boldsymbol{K}_{i+k-1}, \boldsymbol{V}_{i+k-1}) \tag{6}$$

. Then get true tokens $\boldsymbol{X}_{i\sim\gamma} = (x_i, \cdots, x_{i+\lambda})$ and additional next token $x_{i+\gamma+1}$, by strictly match when doing greed decoding or speculative sampling when doing sampling decoding.

The process is recursively repeated until the entire generation is complete.

3.3 Aligned Training by Target-Draft Attention

The method outlined above offers improvements in inference efficiency but introduces challenges in training. A critical issue is the discrepancy between the training and inference phases. If we rely solely on the target model's KV Cache during training, a gap emerges during inference because the draft model's KV Cache also plays a role. Utilizing both the target and draft model's KV Caches during training is not straightforward. A naive approach would be to directly incorporate the target KV Cache into the training process, replacing the last KV with that from the draft model. However, due to the mask attention's matrix operations, this approach only accounts for the generation of the last draft token and fails to consider the preceding tokens. To address this, we designed Target-Draft Attention to simulate the actual inference scenario more accurately. To implement Target-Draft Attention, we modify the training procedure to blend the KV Caches from both the target and draft models, ensuring that the draft model learns to align with the target model's context while still leveraging its own contributions. Figure 2b shows our training framework.

First let's recall the draft process of inference procedure. Assuming that now it will generate the *k*-th draft token

$$x'_{i+k+1} = M_D(\boldsymbol{q}'_{i+k}, \hat{\boldsymbol{K}}_{i+k}, \hat{\boldsymbol{V}}_{i+k}) \tag{7}$$

which involves the calculation of attention weights

$$\hat{\boldsymbol{a}}_{i+k} = (\boldsymbol{q'}_{i+k}\hat{\boldsymbol{K}}^T_{i+k}) \tag{8}$$

and subsequently, the cattention output

$$\hat{\boldsymbol{o}}_{i+k} = \hat{\boldsymbol{a}}_{i+k}\hat{\boldsymbol{V}}_{i+k} = (\boldsymbol{q'}_{i+k}\hat{\boldsymbol{K}}^T_{i+k})\hat{\boldsymbol{V}}_{i+k} \tag{9}$$

In training, we have the inputs $\boldsymbol{K}_n, \boldsymbol{V}_n, \boldsymbol{Q'}_n, \boldsymbol{K'}_n, \boldsymbol{V'}_n$. For $\gamma = 1$, a natural idea to do this is to replace $\boldsymbol{k}_n, \boldsymbol{v}_n$ in $\boldsymbol{K}_n, \boldsymbol{V}_n$ with $\boldsymbol{k'}_n, \boldsymbol{v'}_n$ from the draft model, but it yields $(\boldsymbol{Q'}_n\hat{\boldsymbol{K}}^T_n)\hat{\boldsymbol{V}}_n$, where $\boldsymbol{Q'}_n\hat{\boldsymbol{K}}^T_n = [\boldsymbol{Q'}_{n-1}\hat{\boldsymbol{K}}^T_n, \boldsymbol{q'}_n\hat{\boldsymbol{K}}^T_n]$ After masking, the result becomes $[\boldsymbol{Q'}_{n-1}\boldsymbol{K}^T_{n-1}, \boldsymbol{q'}_n\hat{\boldsymbol{K}}^T_n]$, where only the last vector $\boldsymbol{q'}_n\hat{\boldsymbol{K}}^T_n$ is what we need. To achieve accurate inference reproduction, construction n sets of qKV pairs $\{(\boldsymbol{q'}_i, \hat{\boldsymbol{K}}_i, \hat{\boldsymbol{V}}_i), i = 1, \cdots, n\}$ is necessary, but this approach imposes substantial memory overhead and inefficient resource usage.

So we need to optimize the calculation. From alignment purposes, we want to end up with

$$\hat{\boldsymbol{O}}_n = \{(\boldsymbol{q'}_i\hat{\boldsymbol{K}}^T_i)\hat{\boldsymbol{V}}_i, i = 1, \cdots, n\} \tag{10}$$

when the number of generated tokens is 1. To generate *(k+1)*-th token, we have

$$x'_{i+k+1} = M_D(\boldsymbol{q'}_{i+k}, \hat{\boldsymbol{K}}_{i+k}, \hat{\boldsymbol{V}}_{i+k}) \tag{11}$$

where

$$\hat{\boldsymbol{K}}_{i+k} = (\boldsymbol{K}_{i-1}, \boldsymbol{K'}_{i\sim i+k}), \hat{\boldsymbol{V}}_{i+k} = (\boldsymbol{V}_{i-1}, \boldsymbol{V'}_{i\sim i+k})$$

Because of the $softmax$, we must first compute the attention weights,

$$\hat{\boldsymbol{a}}_{i+k} = (\boldsymbol{q'}_{i+k}\hat{\boldsymbol{K}}^T_{i+k}) = (\boldsymbol{q'}_{i+k}\boldsymbol{K}^T_{i-1}, \boldsymbol{q'}_{i+k}\boldsymbol{K'}^T_{i\sim i+k}) \tag{12}$$

then compute

$$\begin{aligned}\hat{\boldsymbol{o}}_{i+k} &= \hat{\boldsymbol{a}}_{i+k}\hat{\boldsymbol{V}}_{i+k} \\ &= \hat{\boldsymbol{a}}_{i+k}(\boldsymbol{V}_{i-1}, \boldsymbol{V'}_{i\sim i+k}) \\ &= \hat{\boldsymbol{a}}_{i+k}\boldsymbol{V}_{i-1} + \hat{\boldsymbol{a}}_{i+k}\boldsymbol{V'}_{i\sim i+k}\end{aligned} \tag{13}$$

Here, $\hat{\boldsymbol{A}}_n = \{\hat{\boldsymbol{a}}_{i+k},\ i = 1, \cdots, n-k\}$ is formed by $\boldsymbol{Q'}_n\boldsymbol{K}_n$ and $(\boldsymbol{Q'}_n\boldsymbol{K'}_n)$, while $\hat{\boldsymbol{O}}_n$ is formed by $\boldsymbol{A'}_n\boldsymbol{V}_n$ and $\boldsymbol{A'}_n\boldsymbol{V'}_n$. To avoid redundant calculations, we precompute these matrices and index relevant values to construct the target matrix, which greatly reduced training time and memory overhead.

4 Experiments

4.1 Experimental setup

Models, datasets and tasks. We conducted experiments on LLaVA-7B and LLaVA-13B [22], encompassing the common sizes of current mainstream MLLMs. We evaluated our method across multiple tasks including mixed generic tasks (conversation, detailed description, and complex reasoning) on LLaVA-Bench In-the-Wild [22] dataset, Image captioning task on images from **coco** [20] dataset, and chain-of thought (CoT) reasoning on Science QA [26] dataset.

Metrics. Like other work about speculative decoding [5, 18, 39], we assess acceleration effects using the following metrics:

- *Walltime speedup ω:* The actual test speedup ratio relative to auto-regressive decoding.
- *Acceptance rate α:* The average of the ratio between the number of accepted tokens and the number of speculative tokens.
- *Block efficiency τ:* The average number of generated tokens per block (or target model forward) for a block size γ (or speculative steps).
- *Decoding speed δ:* The average number of generated tokens per second.

Table 1: Comparison with usual methods (the mean of different metrics across 3 datasets). Note: FT refers to the finetuned, and DT refers to the distilled.

Target Model	γ	Draft Model	ω	α	τ	δ
LLaVA-7B	3	FT-LLaMA	1.39	0.35	1.93	46.13
		DT-LLaMA	1.33	0.34	1.96	45.00
		FT-LLaVA	1.27	0.28	1.68	40.57
		DT-LLaVA	1.25	0.27	1.69	39.50
		Ours	**2.02**	**0.62**	**2.72**	**63.59**
LLaVA-7B	5	FT-LLaMA	1.37	0.34	2.55	42.77
		DT-LLaMA	1.37	0.34	2.54	43.71
		FT-LLaVA	1.21	0.28	2.22	38.35
		DT-LLaVA	1.21	0.28	2.21	38.34
		Ours	**2.06**	**0.62**	**3.92**	**65.02**
LLaVA-13B	3	FT-LLaMA	1.46	0.35	1.89	46.06
		DT-LLaMA	1.44	0.34	1.87	45.20
		FT-LLaVA	1.36	0.30	1.75	42.46
		DT-LLaVA	1.35	0.29	1.71	41.83
		Ours	**2.14**	**0.63**	**2.74**	**67.78**
LLaVA-13B	5	FT-LLaMA	1.44	0.35	2.60	45.29
		DT-LLaMA	1.44	0.35	2.61	45.66
		FT-LLaVA	1.32	0.30	2.35	42.20
		DT-LLaVA	1.31	0.29	2.37	41.64
		Ours	**2.24**	**0.62**	**3.99**	**70.45**

4.2 Main Results

4.2.1 Comparison with Usual Methods

Drawing on common practices from NLP tasks, we selected different small draft models as baselines. Since there are no small models suitable for rapid draft generation included in the LLaVA suite,

we trained a series of small models to support our experiments, including finetuned-LLaMA (FT-LLaMA in Table 1), distilled-LLaMA (DT-LLaMA), finetuned-LLaVA (FT-LLaVA), distilled-LLaVA (DT-LLaVA). All these small LLaMA model and the language model part of LLaVA is of 112M parameters. Specially, first, following Gagrani et al. (2024), we trained a small LLaMA-2 model from scratch (pretrained on RedPajama-Data-1T-Sample [4], then finetuned [12] on OIG-small-chip2 [15] and OpenAssistant [13], and distilled by seq-level distillationn. Furthermore, we use the small LLaMA model above as the language backbone of the LLaVA model. As shown in Table 1, our method exhibits superior performance across several key metrics, significantly surpassing traditional draft models. Overall, our model showed significant walltime speedup on multiple datasets such as LLava bench in the wild, coco caption, and SQA. For instance, with a $\gamma = 3$ configuration, it achieved a speedup of 2.02 (mean of 3 datasets), far exceeding other baseline methods by $45.3\% - 61.6\%$. This indicates our method's efficiency in reducing actual processing time relative to the target model. In terms of acceptance rate, our model also achieved outstanding results across datasets, notably reaching rates of 0.62, clearly outperforming baseline models like finetuned-LLaMA and distilled-LLaVA.

Our approach also excels in block efficiency, with the value up to 2.72, demonstrating robust capability in generating tokens. In decoding speed, our method stands out with speed reaching up to 64 token/s, exceeding the baseline by $37.8\% - 61.0\%$, indicating its suitability for real-time applications where quick response is crucial. In summary, the data in Table 1 convincingly demonstrates that our method, which integrates the strengths of enhanced speculative decoding and visual processing techniques, outperforming conventional methods across multiple performance metrics, thereby constituting an effective strategy for accelerating the inference of MLLMs.

Table 2: Ablation on vision projector (the mean of different metrics across 3 datasets).

Target Model	γ	Vsion Projector	ω	α	τ	δ
LLaVA-7B	3	w/o	1.64	0.49	2.33	51.48
		w/	$2.02^{\uparrow}$	$0.62^{\uparrow}$	$2.72^{\uparrow}$	$63.59^{\uparrow}$
LLaVA-7B	5	w/o	1.56	0.47	3.21	48.98
		w/	$2.06^{\uparrow}$	$0.62^{\uparrow}$	$3.92^{\uparrow}$	$65.02^{\uparrow}$
LLaVA-13B	3	w/o	1.72	0.49	2.30	54.27
		w/	$2.14^{\uparrow}$	$0.63^{\uparrow}$	$2.74^{\uparrow}$	$67.78^{\uparrow}$
LLaVA-13B	5	w/o	1.70	0.48	3.26	53.69
		w/	$2.24^{\uparrow}$	$0.62^{\uparrow}$	$3.99^{\uparrow}$	$70.45^{\uparrow}$

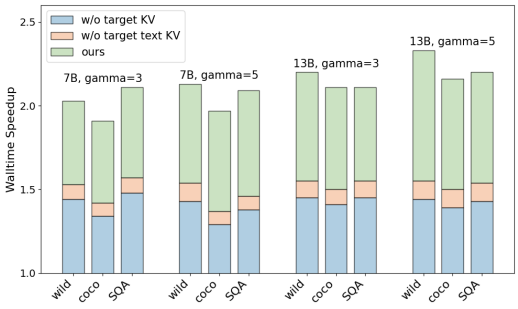


Figure 3: Ablation on target model's KV Cache.

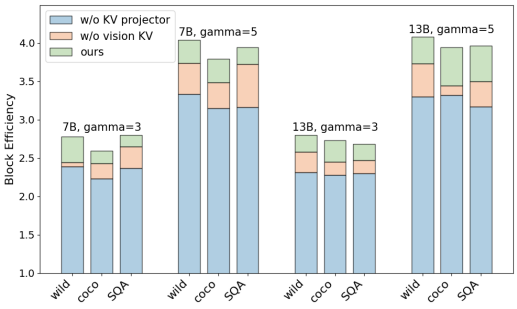


Figure 4: Ablation on vision KV Cache.

4.3 Ablation study

4.3.1 Effectiveness of Target Model's KV Cache

To evaluate the impact of using the KV Cache from the target model, we conducted experiments comparing the inference performance with and without using the target model's cache. KV Cache provides valuable contextual information accumulated in previous steps, which can be leveraged by the draft model in speculative decoding. By accessing the cached information directly from the target model, the draft model can achieve better alignment with the target's state, thus enhancing the acceptance rate and speedup.

As shown in Figure 3, our experimental results indicate that using the target model's KV Cache leads to a significant improvement in walltime speedup. Specifically, KV Cache of the target model is directly related to the final output results, enabling the draft model to make more accurate predictions. This reuse of context not only improves the overall coherence of generated sequences but also speeds up the speculative decoding process. In contrast, models without access to the KV Cache are prone to deviate from the target distribution, resulting in a lower acceptance rate and slower inference. Therefore, KV Cache from the target model plays a crucial role in improving the efficiency of speculative decoding in multimodal large language models.

4.3.2 Effectiveness of Vision KV Projector

To evaluate the impact of the Vision KV Projector module, we conducted experiments that compared the model's performance with and without this module. The Vision KV Projector is designed to selectively compress and extract relevant vision key-value pairs, enabling the model to efficiently leverage visual context in speculative decoding. By employing Vision KV Projector, our method enables the draft model to align more closely with the target model on visual tasks, thereby enhancing the effectiveness of speculative decoding.

As shown in Table 2, our experiments indicate that Vision KV Projector significantly improves both acceptance rate and walltime speedup. With Vision KV Projector enabled, the model demonstrates faster decoding, particularly in tasks with high visual complexity, as irrelevant visual data is filtered out. Additionally, we observed that the Vision KV Projector helps maintain coherence in multimodal contexts by preserving only the most relevant visual cues, allowing the model to align visual information more effectively with textual context. In contrast, without this module, the model processes a larger volume of visual data, resulting in slower decoding and reduced alignment between the draft model and the target model. In summary, Vision KV Projector enhances the efficiency of speculative decoding by focusing on essential visual elements. This approach not only optimizes resource usage but also improves the alignment between the draft model and the target model in multimodal tasks.

4.3.3 Do Vision Information Really Important?

To investigate the relative importance of vision and text information in the KV cache, we conducted experiments by selectively disabling either the image KV cache or the text KV cache. This comparison allows us to analyze the impact of each modality on the performance of the draft model during speculative decoding.

As shown in Figure 4, our experimental results reveal that disabling the text KV cache has a more substantial negative effect on block efficiency compared to disabling the image KV cache. Specifically, without the text KV cache, the model experiences a significant drop in block efficiency. This is because text information provides a critical sequential context that is essential for accurate token prediction and verification. In contrast, while image information contributes to the overall context, it does not require the same level of sequential dependency as text, making it relatively less crucial for maintaining the flow of decoding. In summary, text information plays a important role, but uncompressed visual information may impair the performance of the draft model. This finding emphasizes that in speculative decoding of multimodal large models, textual information is a necessity, while visual information is a bonus that needs to be effectively utilized to be effective.

5 Conclusion

In this study, we introduced an innovative approach to accelerate the inference processes of MLLMs, focusing on the synergistic integration and information reuse of image and text modalities within MLLMs. This method makes efficient use of KV Cache and aligns the draft model and the target model for speculative decoding in multimodal scenario, greatly increasing the inference speed of MLLMs. Our experimental results demonstrate that this approach can achieve up to a 2x speedup across a variety of multimodal tasks, from visual question answering to complex reasoning, fully validating the effectiveness and potential of our method.

References

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

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[3] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024.

[4] Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, 2023.

[5] Cunxiao Du, Jing Jiang, Xu Yuanchen, Jiawei Wu, Sicheng Yu, Yongqi Li, Shenggui Li, Kai Xu, Liqiang Nie, Zhaopeng Tu, et al. Glide with a cape: A low-hassle method to accelerate speculative decoding. *arXiv preprint arXiv:2402.02082*, 2024.

[6] Mohsen Fayyaz, Soroush Abbasi Koohpayegani, Farnoush Rezaei Jafari, Sunando Sengupta, Hamid Reza Vaezi Joze, Eric Sommerlade, Hamed Pirsiavash, and Jürgen Gall. Adaptive token sampling for efficient vision transformers. In *European Conference on Computer Vision*, pages 396–414. Springer, 2022.

[7] Mukul Gagrani, Raghavv Goel, Wonseok Jeon, Junyoung Park, Mingu Lee, and Christopher Lott. On speculative decoding for multimodal large language models. *arXiv preprint arXiv:2404.08856*, 2024.

[8] Ziyuan Huang, Kaixiang Ji, Biao Gong, Zhiwu Qing, Qinglong Zhang, Kecheng Zheng, Jian Wang, Jingdong Chen, and Ming Yang. Accelerating pre-training of multimodal llms via chain-of-sight. *arXiv preprint arXiv:2407.15819*, 2024.

[9] Yizhang Jin, Jian Li, Yexin Liu, Tianjun Gu, Kai Wu, Zhengkai Jiang, Muyang He, Bo Zhao, Xin Tan, Zhenye Gan, et al. Efficient multimodal large language models: A survey. *arXiv preprint arXiv:2405.10739*, 2024.

[10] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[11] Mahsa Khoshnoodi, Vinija Jain, Mingye Gao, Malavika Srikanth, and Aman Chadha. A comprehensive survey of accelerated generation techniques in large language models. *arXiv preprint arXiv:2405.13019*, 2024.

[12] Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, 2016.

[13] Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations-democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

[14] Denis Kuznedelev, Eldar Kurtić, Elias Frantar, and Dan Alistarh. Cap: Correlation-aware pruning for highly-accurate sparse vision models. *Advances in Neural Information Processing Systems*, 36, 2024.

[15] LAION.ai. Oig-small-chip2, 2023.

[16] Phuoc-Hoan Charles Le and Xinlin Li. Binaryvit: pushing binary vision transformers towards convolutional models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4664–4673, 2023.

[17] Yejin Lee, Anna Sun, Basil Hosmer, Bilge Acun, Can Balioglu, Changhan Wang, Charles David Hernandez, Christian Puhrsch, Daniel Haziza, Driss Guessous, et al. Characterizing and efficiently accelerating multimodal generation model inference. *arXiv preprint arXiv:2410.00215*, 2024.

[18] 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.

[19] Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Jinfa Huang, Junwu Zhang, Yatian Pang, Munan Ning, et al. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*, 2024.

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[21] Zhihang Lin, Mingbao Lin, Luxi Lin, and Rongrong Ji. Boosting multimodal large language models with visual tokens withdrawal for rapid inference. *arXiv preprint arXiv:2405.05803*, 2024.

[22] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[23] Yijiang Liu, Huanrui Yang, Zhen Dong, Kurt Keutzer, Li Du, and Shanghang Zhang. Noisyquant: Noisy bias-enhanced post-training activation quantization for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20321–20330, 2023.

[24] Zuyan Liu, Benlin Liu, Jiahui Wang, Yuhao Dong, Guangyi Chen, Yongming Rao, Ranjay Krishna, and Jiwen Lu. Efficient inference of vision instruction-following models with elastic cache. *arXiv preprint arXiv:2407.18121*, 2024.

[25] Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, et al. Deepseek-vl: towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*, 2024.

[26] Pan Lu, Swaroop Mishra, Tanglin 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. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

[27] Yanyuan Qiao, Zheng Yu, Longteng Guo, Sihan Chen, Zijia Zhao, Mingzhen Sun, Qi Wu, and Jing Liu. Vl-mamba: Exploring state space models for multimodal learning. *arXiv preprint arXiv:2403.13600*, 2024.

[28] Sucheng Ren, Zhengqi Gao, Tianyu Hua, Zihui Xue, Yonglong Tian, Shengfeng He, and Hang Zhao. Co-advise: Cross inductive bias distillation. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 16773–16782, 2022.

[29] swordlidev. Karmavlm, 2024.

[30] Ao Wang, Hui Chen, Zijia Lin, Sicheng Zhao, Jungong Han, and Guiguang Ding. Cait: Triple-win compression towards high accuracy, fast inference, and favorable transferability for vits. *arXiv preprint arXiv:2309.15755*, 2023.

[31] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[32] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[33] Yuxin Wen, Qingqing Cao, Qichen Fu, Sachin Mehta, and Mahyar Najibi. Efficient vision-language models by summarizing visual tokens into compact registers. *arXiv preprint arXiv:2410.14072*, 2024.

[34] Kan Wu, Jinnian Zhang, Houwen Peng, Mengchen Liu, Bin Xiao, Jianlong Fu, and Lu Yuan. Tinyvit: Fast pretraining distillation for small vision transformers. In *European conference on computer vision*, pages 68–85. Springer, 2022.

[35] Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*, 2024.

[36] 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. *arXiv preprint arXiv:2401.07851*, 2024.

[37] Ruyi Xu, Yuan Yao, Zonghao Guo, Junbo Cui, Zanlin Ni, Chunjiang Ge, Tat-Seng Chua, Zhiyuan Liu, Maosong Sun, and Gao Huang. Llava-uhd: an lmm perceiving any aspect ratio and high-resolution images. *arXiv preprint arXiv:2403.11703*, 2024.

[38] Minghao Yan, Saurabh Agarwal, and Shivaram Venkataraman. Decoding speculative decoding. *arXiv preprint arXiv:2402.01528*, 2024.

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

[40] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024.

[41] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*, 2023.

[42] Gaotong Yu, Yi Chen, and Jian Xu. Balancing performance and efficiency: A multimodal large language model pruning method based image text interaction. *arXiv preprint arXiv:2409.01162*, 2024.

[43] Chen Zhang, Zhuorui Liu, and Dawei Song. Beyond the speculative game: A survey of speculative execution in large language models. *arXiv preprint arXiv:2404.14897*, 2024.

[44] Baichuan Zhou, Ying Hu, Xi Weng, Junlong Jia, Jie Luo, Xien Liu, Ji Wu, and Lei Huang. Tinyllava: A framework of small-scale large multimodal models. *arXiv preprint arXiv:2402.14289*, 2024.

[45] Mingjian Zhu, Yehui Tang, and Kai Han. Vision transformer pruning. *arXiv preprint arXiv:2104.08500*, 2021.