Token Pruning in Multimodal Large Language Models: Are We Solving the Right Problem?

Zichen Wen[1,2,3*] **Yifeng Gao**[1,2*] **Weijia Li**[4,3] **Conghui He**[3†] **Linfeng Zhang**[1,2†]
[1]Shanghai Jiao Tong University [2]EPIC Lab, SJTU
[3]Shanghai AI Laboratory [4]Sun Yat-sen University
zichen.wen@outlook.com, heconghui@pjlab.org.cn, zhanglinfeng@sjtu.edu.cn

Abstract

Multimodal large language models (MLLMs) have shown remarkable performance for cross-modal understanding and generation, yet still suffer from severe inference costs. Recently, abundant works have been proposed to solve this problem with token pruning, which identifies the redundant tokens in MLLMs and then prunes them to reduce the computation and KV storage costs, leading to significant acceleration without training. While these methods claim efficiency gains, critical questions about their fundamental design and evaluation remain unanswered: *Why do many existing approaches underperform even compared to naive random token selection? Are attention-based scoring sufficient for reliably identifying redundant tokens? Is language information really helpful during token pruning? What makes a good trade-off between token importance and duplication? Are current evaluation protocols comprehensive and unbiased?* The ignorance of previous research on these problems hinders the long-term development of token pruning. In this paper, we answer these questions one by one, providing insights into the design of future token pruning methods.

1 Introduction

Multi-modal language models (MLLMs) (Huang et al., 2023; Driess et al., 2023; Liu et al., 2024c; Bai et al., 2023), especially the vision-language models have demonstrated powerful effectiveness in various tasks. However, the extremely high computational and storage costs have limited the application of MLLMs in real-time applications, which is caused by not only the enormous parameters inherited from LLMs but also a large number of tokens from the large visual information such as high-resolution images and multi-frame videos.

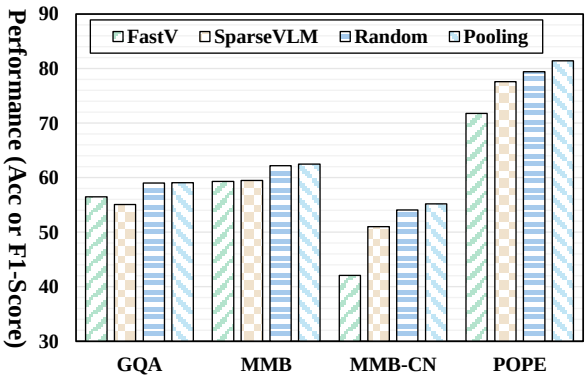


Figure 1: **Comparison between FastV, SparseVLM, and naive baselines.** On several common datasets, the performance of FastV and SparseVLM is even worse than random token dropping and pooling.

To solve this problem, abundant efforts have been made in **token pruning** (Chen et al., 2024; Zhang et al., 2024c; Liu et al., 2024d, 2025c,a), which aims to reduce the number of input tokens in MLLMs. Usually, token pruning methods first introduce a carefully-designed criterion to measure the importance of a vision token, and then prune the redundant tokens, or merge the redundant tokens into fewer tokens. As a result, the following computation of the pruned tokens or the merged tokens can be removed or reduced, bringing efficiency in both computation and storage. For instance, some recent works show that more than 70% tokens can be pruned with a tolerant loss in accuracy (Chen et al., 2024). Most attractively, thanks to the natural ability of MLLMs to process tokens in different lengths, token pruning can be applied to most existing MLLMs with no need for additional training, and thus attracts great attention from both academic researchers and industrial developers.

However, despite the popularity of token pruning, numerous foundational questions have long been overlooked and remain largely unexplored, giving rise to several surprising phenomena. For instance, Figure 1 demonstrates the comparison between two classical token pruning methods in-

*Equal Contribution.
†Corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2025, pages 15537–15549

cluding FastV (Chen et al., 2024) and SparseVLM (Zhang et al., 2024c), and two naive baselines, including random token selection and direct average pooling on tokens. **Surprisingly, the two baselines outperform the two well-designed token pruning methods in most benchmarks by a clear margin**. This counterintuitive phenomenon may demonstrate that the current understanding of so-called important tokens is far away from the truth. Unfortunately, most recent works just focus on purchasing higher performance, while ignoring these questions, which may hinder the long-term development of token pruning.

In this paper, we have conducted massive experiments and analyses to dive into the fundamental problems of token pruning, with the main takeaways as follows.

- Attention-based token selection methods suffer from position bias, where vision tokens in the later positions are more likely to be retained. Reducing position bias in these methods can benefit their performance.
- Language information is helpful in token pruning only when a given task strongly correlates with the language information.
- Both the importance and uniqueness (low similarity) of tokens have a significant influence on the performance of token pruning and their influence varies from different tasks.
- FLOPs and the number of retained tokens are unreliable metrics for token pruning methods. Compatibility with hardware has a significant influence on real acceleration performance.
- Training-aware token pruning which directly merges tokens in spatially adjacent positions may bring more benefits than carefully-designed training-aware pruning.

We hope that this paper can provide insights into the future design of token pruning, and correct the long-neglected evaluation issues in this field.

2 Related Work

2.1 Multimodal Large Language Models

The remarkable success of large language models (LLMs) (Radford et al., 2019; Brown et al., 2020; Wen et al., 2024) has spurred a growing trend of extending their advanced reasoning capabilities to multi-modal tasks, leading to the development of vision-language models (VLMs) (Huang et al., 2023; Driess et al., 2023; Liu et al., 2024c; Bai et al., 2023). These VLMs typically consist of a visual encoder (Radford et al., 2021) that serializes input image representations and an LLM responsible for text generation. To enable the LLM to process visual inputs, an alignment module is employed to bridge the gap between visual and textual modalities. This module can take various forms, such as an MLP projector or a more complex query-based network. While this integration allows the LLM to gain visual perception, it also introduces significant computational challenges due to the long sequences of visual tokens.

Moreover, existing VLMs often exhibit limitations, such as visual shortcomings or hallucinations, which hinder their performance. Efforts to enhance VLM capabilities by increasing input image resolution have further exacerbated computational demands. For instance, encoding higher-resolution images results in a substantial increase in the number of visual tokens. A model like LLaVA-1.5 (Liu et al., 2024a) generates 576 visual tokens for a single image, while its successor, LLaVA-NeXT (Liu et al., 2024b), produces up to 2880 tokens at double the resolution, far exceeding the length of typical textual prompts. Optimizing the inference efficiency of VLMs is thus a critical task to facilitate their deployment in real-world scenarios with limited computational resources.

2.2 Visual Token Compression

Visual tokens are often significantly more numerous than text tokens, with higher spatial redundancy and lower information density. To address this issue, various methods have been proposed for reducing visual token counts in vision language models. For instance, some approaches modify model components, such as using context tokens in Q-Former (Li et al., 2024) or applying adaptive pooling at the patch level, but these typically require additional training and increase computational costs. Other techniques, like Token Merging (ToMe) (Bolya et al., 2023) and FastV (Chen et al., 2024), focus on reducing tokens without retraining by merging tokens or selecting important ones based on attention scores. SparseVLM (Zhang et al., 2024c) incorporates text guidance through cross-modal attention to refine token selection. However, these methods often overlook hardware acceleration compatibility and fail to account for token duplication alongside token importance. Furthermore, while token pruning has been extensively explored in natural language processing and computer vision to improve

inference efficiency, its application to VLMs remains under-explored. Existing pruning strategies, such as those in FastV and SparseVLM, rely on text-visual attention within large language models (LLMs) to evaluate token importance, which may not align well with actual visual token relevance.

3 Benchmarking

We begin by presenting the datasets, models, and pruning methods included in our study, along with the rationale behind the selection. Next, we outline the experimental setup and provide guidance on interpreting the results reported in our study. Finally, we analyze the findings, emphasizing notable patterns and offering insights that may inform future research in this area.

3.1 Models

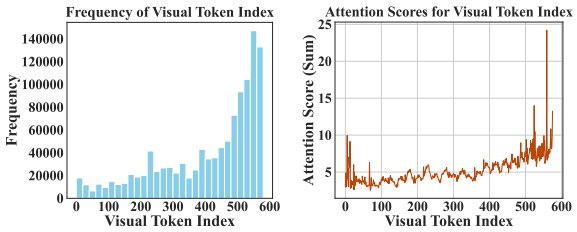
We selected several representative MLLMs, including LLaVA-1.5-7B and 13B (Liu et al., 2024a), LLaVA-Next-7B (Liu et al., 2024b), and Qwen2-VL (Wang et al., 2024) series (7B-Instruct and 72B-Instruct). LLaVA-1.5-7B integrates CLIP and LLaMA for vision-language alignment via end-to-end training, employing MLP connectors to fuse visual-text features for multimodal reasoning. LLaVA-Next-7B enhances data efficiency and inference robustness with dynamic resolution and hierarchical feature integration, improving fine-grained visual understanding. Qwen2-VL series excel in high-resolution input processing and instruction-following, supporting complex tasks like document analysis and cross-modal in-context learning through unified vision-language representations.

3.2 Datasets

To evaluate the impact of pruning on different tasks, we selected a diverse set of datasets, including visual understanding tasks including GQA (Hudson and Manning, 2019), MMBench (MMB) (Liu et al., 2025d), MME (Fu et al., 2023), POPE (Li et al., 2023), ScienceQA (Lu et al., 2022), VQA^{V2} (VQA V2) (Goyal et al., 2017) and VQA^{Text} (TextVQA) (Singh et al., 2019), grounding task RefCOCO (Yu et al., 2016; Mao et al., 2016) and object retrieval task Visual Haystack (Wu et al., 2025), . We briefly introduce these datasets in Table 9.

3.3 Token Pruning Method

To rigorously evaluate the properties of visual token pruning, we select three representative and high-performing methods: FastV (Chen et al., 2024), SparseVLM (Zhang et al., 2024c), and MustDrop (Liu et al., 2024d). FastV (Chen et al., 2024) optimizes computational efficiency by learning adaptive attention patterns in early layers and pruning low-attention visual tokens post-layer 2 of LLMs, effectively reducing redundancy. SparseVLM (Zhang et al., 2024c) introduces a text-guided, training-free pruning mechanism that leverages self-attention matrices between text and visual tokens to assess importance. It maximizes sparsity while preserving semantically relevant tokens without additional parameters or fine-tuning. MustDrop (Liu et al., 2024d) addresses token redundancy across the entire model lifecycle. It merges spatially similar tokens during vision encoding, employs text-guided dual-attention filtering in prefilling, and implements output-aware KV cache compression during decoding. This multi-stage approach ensures balanced retention of critical tokens while enhancing inference efficiency. These methods exemplify diverse strategies for token pruning, spanning adaptive attention, text-guided sparsity, and lifecycle-aware optimization.



(a) Token Frequency (b) Token Attention

Figure 2: **Analysis of the distribution of tokens and attention scores over the position of tokens.** Tokens with larger indexes are located at the bottom of images.

4 Token Pruning Revisited: Are Simple Methods Better?

When considering token pruning in multimodal large language models, two very basic methods naturally come to mind: random token pruning (hereafter referred to as **Random**) and token pooling (hereafter referred to as **Pooling**). Comparison with these two simple baselines is reliable evidence to demonstrate the significance of a well-designed token pruning method, yet has been ignored by most previous works. To address this gap, we investigated these two simple approaches in detail. Specifically, we conducted experiments on multiple widely-used benchmarks under pruning ratios of 75% and 87.5%, comparing Random and Pooling[1]

[1]In the experiments, Pooling specifically refers to applying a pooling operation to the visual tokens at the second layer of

Method	GQA	MMB	MMB-CN	MME	POPE	SQA	VQA^{Text}	VizWiz	Avg.
Upper Bound, 576 Tokens **(100%)**									
Vanilla	61.9	64.7	58.1	1862	85.9	69.5	58.2	50.0	100%
Retain 144 Tokens (↓ **75.0%**)									
Random	59.0	62.2	54.1	1736	79.4	67.8	51.7	**51.9**	95.0% (-5.0%)
Pooling	59.1	**62.5**	**55.2**	**1763**	**81.4**	69.1	53.4	**51.9**	96.4% (-3.6%)
Window FastV	**59.2**	59.3	51.0	1737	80.3	66.4	50.8	50.3	93.2% (-6.8%)
Vanilla FastV	56.5	59.3	42.1	1689	71.8	65.3	53.6	51.3	89.8% (-10.2%)
Reverse FastV	49.9	36.9	26.4	1239	59.8	60.9	36.9	48.4	70.8% (-29.2%)
SparseVLM	55.1	59.5	51.0	1711	77.6	**69.3**	**54.9**	51.4	93.5% (-6.5%)
Retain 64 Tokens (↓ **88.9%**)									
Random	**55.9**	**58.1**	**48.1**	1599	70.4	66.8	48.2	**51.6**	89.1% (-10.9%)
Pooling	54.2	56.0	46.0	1545	71.2	**67.2**	49.4	49.9	87.6% (-12.4%)
Window FastV	55.8	56.2	41.2	**1630**	72.6	66.3	47.8	50.0	87.2% (-12.8%)
Vanilla FastV	46.1	47.2	38.1	1255	58.6	64.9	47.8	50.8	78.2% (-21.8%)
Reverse FastV	44.6	24.0	15.7	1114	45.2	60.8	35.9	48.4	61.8% (-38.2%)
SparseVLM	52.7	56.2	46.1	1505	**75.1**	62.2	**51.8**	50.1	87.3% (-12.7%)

Table 1: **Performance Comparison of LLaVA-1.5-7B with Different Token Retention Strategies.** Reverse FastV is a variant of the FastV that retains tokens with the smallest attention scores.

Method	GQA	MMB	MMB-CN	MME	POPE	SQA	VQA^{Text}	VizWiz	Avg.
Upper Bound, 576 Tokens **(100%)**									
Vanilla	63.3	68.9	62.3	1818	85.9	72.8	61.3	56.6	100%
Retain 144 Tokens (↓ **75.0%**)									
Random	60.3	**65.9**	**58.3**	**1767**	80.6	**71.4**	54.3	57.6	95.5% (-4.5%)
Pooling	**60.6**	65.5	57.7	1742	**83.6**	71.3	**56.3**	56.6	95.8% (-4.2%)
Window FastV	59.5	65.5	57.7	1674	82.8	57.2	48.0	**60.9**	91.8% (-8.2%)
Vanilla FastV	57.7	53.9	46.8	1633	79.3	57.0	51.0	60.3	86.9% (-13.1%)
SparseVLM	57.9	63.8	55.8	1704	81.1	69.9	43.9	56.3	91.1% (-8.9%)
Retain 64 Tokens (↓ **88.9%**)									
Random	**57.5**	**62.6**	54.4	**1681**	73.8	70.9	50.6	57.5	91.1% (-8.9%)
Pooling	55.4	58.3	51.4	1552	74.0	**72.0**	**52.3**	51.4	87.7% (-12.3%)
Window FastV	56.7	58.3	51.4	1599	**76.5**	55.4	43.2	**59.5**	85.7% (-14.3%)
Vanilla FastV	53.7	50.9	42.1	1567	69.3	56.8	47.1	59.2	81.6% (-18.4%)
SparseVLM	50.6	61.3	**54.8**	1402	65.0	69.0	22.7	54.5	79.7% (-20.3%)

Table 2: **Performance Comparison of LLaVA-1.5-13B with Different Token Retention Strategies.**

Method	GQA	MME	POPE	SQA	VQA^{Text}	Avg.
Upper Bound, 576 Tokens **(100%)**						
Vanilla	65.3	2521	87.4	91.6	82.8	100%
Retain 144 Tokens (↓ **75.0%**)						
Random	63.9	**2476**	87.1	85.7	74.0	95.7% (-4.3%)
Pooling	63.1	2463	86.9	**86.9**	75.1	95.9% (-4.1%)
Window FastV	**64.2**	2445	**88.5**	85.9	75.4	96.3% (-3.7%)
Vanilla FastV	56.5	2219	80.9	85.3	**75.6**	90.3% (-9.7%)
Retain 64 Tokens (↓ **88.9%**)						
Random	**61.9**	**2394**	85.5	79.3	64.5	90.4% (-9.6%)
Pooling	**61.9**	2391	84.3	81.1	65.9	90.9% (-9.1%)
Window FastV	**61.9**	2377	**85.6**	**83.8**	65.8	91.6% (-8.4%)
Vanilla FastV	55.7	2089	78.7	83.3	**66.8**	86.0% (-14.0%)

Table 3: **Performance Comparison of Qwen2-VL-72B with Different Token Retention Strategies.**

with several recent token pruning methods (*e.g.*, FastV and SparseVLM). As shown in Table 1, surprisingly, Random and Pooling outperformed carefully designed methods on nearly 2/3 benchmarks.

the language model. Please refer to the specific implementation in Algorithm 4.

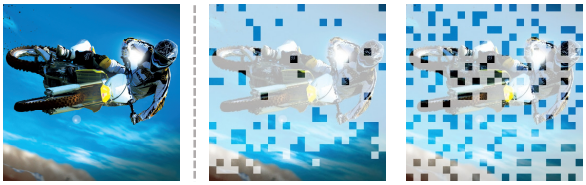
When scaling up to larger models, the experimental results of LLaVA-1.5-13B[2] and Qwen2-VL-72B[3] in Tables 2 and 3 also demonstrate the superior performance of the simple methods Random and Pooling. These surprising results shocked us since its inferior performance compared with random selection may demonstrate that we are on the wrong road toward the ideal token pruning methods.

4.1 Token Distribution: Spatial Uniformity Outperforms Position Bias

We further explored the underlying reasons behind this phenomenon. Taking FastV (Chen et al., 2024) as an example, this method leverages the attention scores assigned to visual tokens by the last token

[2] https://huggingface.co/liuhaotian/llava-v1.5-13b

[3] https://huggingface.co/Qwen/Qwen2-VL-72B-Instruct



Original Image (a) Vanilla FastV (b) Window FastV

Figure 3: **Sparse Visualization of Vanilla FastV and Window FastV with 25% Retained Visual Tokens.**

to evaluate the importance of each visual token, which may introduce the basis for token pruning.

Using 8,910 samples from the POPE dataset, we conducted a statistical analysis of the visual tokens retained by FastV. As illustrated in Figure 2, tokens located toward the end of the visual token sequence were assigned significantly higher attention scores and were retained far more frequently than tokens in other positions. This indicates that methods relying on attention scores to select visual tokens inherently suffer from a severe position bias during token reduction. In contrast, tokens retained by Random or Pooling exhibit a naturally uniform spatial distribution. We argue that this spatial uniformity may be the key reason why some existing methods underperform Random and Pooling.

4.2 Validating the Hypothesis: From Position Bias to Spatial Uniformity

To validate our hypothesis, we proposed a modification to FastV, introducing a variant called Window FastV. Specifically, we incorporated a sliding window mechanism into the original FastV framework. Within each window, a predetermined reduction ratio and window size were used to select a fixed number of visual tokens. For the specific implementation of Window FastV, please refer to Algorithm 3 in Appendix B. Compared to Vanilla FastV, Window FastV ensures the spatial uniformity of the retained tokens, as shown in Figure 3.

We evaluated both Vanilla FastV and Window FastV across eight benchmarks. As shown in Table 1, under the setting where 75% of visual tokens are reduced, Window FastV exhibits an average performance drop that is **3.4%** less than that of Vanilla FastV. When adopting a more aggressive reduction ratio (↓ 88.9%), this gap widens to **9%**. These results not only validate our hypothesis but also inspire us to consider strategies that encourage the spatial uniformity of retained tokens when designing token pruning methods.

To further investigate the impact of token pruning on spatial position understanding, we selected

Method	RefCOCO			RefCOCO+			RefCOCOg		Avg.
	TestA	TestB	Val	TestA	TestB	Val	Test	Val	
Vanilla	72.3	51.5	73.3	66.3	29.7	68.3	49.5	51.5	100%
SparseVLMs	4.0	6.9	4.0	2.9	5.9	0.9	7.9	5.9	4.8% (↓ 95.2%)
Vanilla FastV	20.8	13.9	27.7	17.8	8.9	26.7	19.8	14.9	18.8% (↓ 81.2%)
Window FastV	22.8	22.8	25.7	18.8	7.9	18.8	20.8	23.8	20.2% (↓ 79.8%)
Random	22.8	27.7	32.7	17.8	13.9	32.7	20.8	16.8	23.2% (↓ 76.8%)
Pooling	34.7	17.8	26.7	23.8	17.8	24.8	14.9	20.8	22.7% (↓ 77.3%)

Table 4: **Performance Comparison on RefCOCO Series Grounding Tasks**. Evaluation is based on Precision@1, with a reduction ratio of 77.8%.

the RefCOCO (Yu et al., 2016) dataset, which requires the MLLM to generate a bounding box for a specified object phrase within an image. We consider this dataset to be an effective atomic benchmark for evaluating the spatial understanding capabilities of MLLMs. Our evaluation criterion is that a prediction is considered correct if the Intersection over Union (IoU) between the predicted bounding box and the ground truth area exceeds 0.5. As shown in Table 4, compared to conventional tasks, various token pruning methods exhibit a significant degradation in performance when applied to precise object localization. Notably, there is a marked difference between globally uniform attention-based pruning methods (e.g., Window FastV, or even naive approaches like Random and Pooling) and spatially non-uniform strategies (FastV, SparseVLM). This indicates that current token pruning techniques, particularly those that are spatially non-uniform, still possess substantial limitations in comprehending the spatial positioning of objects within images.

Summary 1. *The position bias in the distribution of retained visual tokens is a key factor affecting the performance of some existing token pruning methods. This insight suggests that ensuring the spatial uniformity of retained tokens should be an important consideration when designing token pruning strategies.*

5 Language in Visual Token Pruning: When and Why Does Language Matter?

Token pruning methods for multimodal models can be broadly categorized into two types: those guided by textual information (e.g., FastV (Chen et al., 2024), SparseVLM (Zhang et al., 2024c), MustDrop (Liu et al., 2024d)) and those that rely solely on visual information (e.g., FasterVLM (Zhang et al., 2024b)). While both approaches achieve comparable performance on common benchmarks, however, we hypothesize: Could it be that the im-

portance of language information is not evident simply because there has been a lack of testing on tasks where language information is especially critical? To validate our hypothesis, we select a typical scenario: Visual Haystack.

5.1 Visual Token Pruning in Strongly Text-Guided Tasks

Tasks such as Visual Haystack (Wu et al., 2025) (needle-in-a-haystack task on visual scenario) are inherently text-driven. In Visual Haystack task, the MLLM needs to select an image from a set of confusing images with an anchor phrase, and determine whether an object matching a target textual description exists within the selected image. These tasks demand precise alignment between textual and visual modalities. To evaluate the impact of text-guided pruning, we conducted experiments using the LLaVA-1.5-7B model on the VH dataset.

Method	# Input Images (More images means harder to retrieve)				
	Oracle	2	3	5	10
LLaVA-1.5-7B	$86.46_{\pm 1.25}$	$70.04_{\pm 1.49}$	$66.18_{\pm 1.58}$	$58.29_{\pm 1.49}$	$53.47_{\pm 1.48}$
Reduction ratio 77.8%					
SparseVLM	$81.26_{\pm 1.11}$	$66.14_{\pm 1.54}$	$66.54_{\pm 1.33}$	$58.22_{\pm 1.51}$	$53.99_{\pm 1.65}$
FastV	$76.30_{\pm 1.36}$	$61.17_{\pm 1.56}$	$58.34_{\pm 1.61}$	$53.39_{\pm 1.51}$	$52.06_{\pm 1.63}$
FastV$_{\texttt{VIS}}$	$71.90_{\pm 1.58}$	$61.55_{\pm 1.46}$	$55.82_{\pm 1.49}$	$52.72_{\pm 1.63}$	$52.83_{\pm 1.54}$
Random	$75.15_{\pm 1.30}$	$62.14_{\pm 1.61}$	$55.59_{\pm 1.49}$	$51.26_{\pm 1.36}$	$50.76_{\pm 1.75}$

Table 5: **Performance comparison of different methods on Visual Haystack (VH)**. VH requires MLLMs to select an image from multiple images based on an anchor word and determine the existence of a target word object in the image. FastV$_{\texttt{VIS}}$ means FastV without language information guided.

To validate the importance of text guidance, we modified FastV to operate without textual information and denote it FastV$_{\texttt{VIS}}$. Originally, FastV calculates the importance of visual tokens based on the attention score with the last text token. FastV$_{\texttt{VIS}}$ computes with the last visual token instead, thereby eliminating the influence of text information while preserving the essence of the method. Our results in Table 5 show that this modificaiton FastV$_{\texttt{VIS}}$ reveals a significant drop in performance, confirming the importance of leveraging textual cues in strongly text-guided tasks. The comparison of different pruning methods also reveals that approaches utilizing textual information exhibit significantly better overall performance. It is noteworthy that SparseVLM, guided by text information, achieves a compression rate of 77.8% while maintaining nearly identical accuracy to the uncompressed model, particularly in scenarios with a higher number of confusing images.

However, there are also recent works methods (Zhang et al., 2024b; Liu et al., 2025b) that perform pruning solely in ViT without textual information and reports better performance than FastV and SparseVLM in common VQA benchmarks.

Therefore, for tasks with high reliance on language information, pruning strategies should be tailored to incorporate textual guidance effectively, and how to balance the use of linguistic information still requires further research.

Summary 2. *Text-guided pruning improves performance in text-heavy tasks. Pruning methods should adapt to task needs.*

6 The α Dilemma: Importance vs. Redundancy in Token Pruning

In this section, we systematically analyze the fundamental tension in token pruning for multimodal large language models: *should we prioritize removing redundant tokens to preserve structural patterns, or eliminate less important tokens to maintain predictive capacity?*

6.1 Redundancy Criteria

This criterion adopts a *task-agnostic* perspective, focusing exclusively on input patterns. The core objective is to eliminate redundant tokens while preserving the input's *structural integrity* and minimizing information loss - analogous to finding the minimal sufficient statistics in information theory.

Through the lens of mutual information (Latham and Roudi, 2009), we formulate this as maximizing information preservation between original tokens $\mathbf{X}$ and retained tokens $\mathbf{X}'$:

$$\max_{\mathcal{P}} I(\mathbf{X}; \mathbf{X}') = \mathcal{H}(\mathbf{X}) - \mathcal{H}(\mathbf{X}|\mathbf{X}'), \quad (1)$$

where $\mathcal{P}$ denotes the pruning operator. This ensures the $\mathbf{X}'$ retains maximal dependence on $\mathbf{X}$ under length constraint $\|\mathbf{X}'\| = \|\mathbf{X}\| - \Delta L$. The formulation directly connects to the *compression phase* of the information bottleneck principle (Tishby et al., 2000), where $\mathcal{P}^*$ solves:

$$\mathcal{P}^* = \arg\min_{\mathcal{P}} \|\mathbf{X}'\| \quad \text{s.t. } I(\mathbf{X}; \mathbf{X}') \geq \gamma, \quad (2)$$

with γ as the minimal acceptable mutual information. This preserves structural patterns without task-specific considerations.

Benchmark	Vanilla	Balance between Importance and Redundancy α										
		0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
MME	1862	1707	1714	1711	1706	1707	1711	1702	1699	1680	1688	1689
POPE	85.9	82.8	82.6	82.4	82.4	81.9	81.6	80.9	79.7	77.9	75.6	71.8
SQA	69.5	64.8	65.2	65.2	65.1	65.1	65.3	65.3	65.2	65.5	65.7	65.3
VQA$^{\text{Text}}$	58.2	53.6	53.8	54.8	54.0	54.1	54.3	54.3	54.5	54.4	54.2	53.6

Table 6: Performance comparison under different α balancing importance and redundancy criteria.

6.2 Importance Criteria

In contrast, this *task-oriented* criterion explicitly considers the target output $\mathbf{Y}$. The goal shifts to preserving tokens critical for *prediction accuracy*, formalized through predictive sufficiency:

$$I(\mathbf{X}';\mathbf{Y}) \geq I(\mathbf{X};\mathbf{Y}) - \epsilon, \tag{3}$$

where ϵ is the tolerable information loss. Expanding via the chain rule:

$$\underbrace{I(\mathbf{X};\mathbf{Y})}_{\text{Original}} = \underbrace{I(\mathbf{X}';\mathbf{Y})}_{\text{Pruned}} + \underbrace{I(\mathbf{X}\setminus\mathbf{X}';\mathbf{Y}|\mathbf{X}')}_{\text{Discarded}}. \tag{4}$$

The bound $I(\mathbf{X}\setminus\mathbf{X}';\mathbf{Y}|\mathbf{X}') \leq \epsilon$ implies that discarded tokens provide negligible additional information about $\mathbf{Y}$ when conditioned on retained tokens. This captures the essence of importance - truly critical tokens contain non-decomposable predictive information.

The task dependence manifests in the information plane:

$$\mathcal{R}(\beta) = \max_{\mathbf{X}'} \left[I(\mathbf{X}';\mathbf{Y}) - \beta^{-1} I(\mathbf{X};\mathbf{X}') \right], \tag{5}$$

where β controls redundancy-importance tradeoff.

6.3 Empirical Validation of Adaptive Criteria Balancing

Building on Eq. 5, we implement an adaptive scoring mechanism with tunable parameter α:

$$\text{Score}(x_i) = \alpha \cdot \underbrace{I(x_i;\mathbf{Y}|x_{\setminus i})}_{\text{Predictive Criticality}} + (1-\alpha) \cdot \underbrace{[1 - I(x_i;\mathbf{X}_{\setminus i})]}_{\text{Pattern Uniqueness}}. \tag{6}$$

Here $I(x_i;\mathbf{Y}|x_{\setminus i})$ measures a token's unique predictive value, while $1 - I(x_i;\mathbf{X}_{\setminus i})$ quantifies its pattern distinctiveness.

Specifically, FastV is a typical token pruning method that follows the importance criterion by selecting important visual tokens based on the attention scores of the last token in the sequence. We modify this approach by introducing a redundancy criterion, which calculates the cosine similarity between each visual token and the last token to derive a similarity score[4]. Ultimately, the final score in Eq. 6 is obtained by balancing these two metrics with a parameter α. Our experiments results in Table 6 reveal two key insights:

- **Perception-Dominant Tasks** (MME, POPE) achieve peak performance at $\alpha = 0.1$ and 0.0, respectively, favoring redundancy-first pruning to maintain structural integrity ($\uparrow I(\mathbf{X};\mathbf{X}')$).
- **Knowledge-Intensive Tasks** (SQA, VQA$^{\text{Text}}$) achieve optimal performance with $\alpha = 0.8 \sim 0.9$, favoring importance-first pruning to enhance semantic coherence ($\uparrow I(\mathbf{X}';\mathbf{Y})$).

Summary 3. *Prune by task: Redundancy-first preserves structural fidelity for perception tasks, while importance-first prioritizes predictive power for knowledge reasoning.*

7 Limitations and Challenges in Token Pruning Evaluation

Token pruning has emerged as a promising technique to improve the efficiency of MLLMs. However, despite its potential, the evaluation of token pruning methods remains fraught with challenges. In this section, we critically examine two key issues that hinder the accurate and meaningful assessment of token pruning techniques: **(i)** the over-reliance on FLOPs as a proxy for speed gains, and **(ii)** the failure to account for training-aware compression in some advanced MLLMs. We argue that addressing these challenges is crucial for developing more robust and reliable token pruning approaches.

7.1 Beyond FLOPs: Shifting the Focus to Actual Latency Gains

Phenomenon. Many existing token pruning approaches tend to measure the speedup of their methods by calculating or estimating the reduction in FLOPs resulting from token reduction, or even directly using the token reduction ratio as a metric.

[4]Notably, since the similarity score and attention score are on different scales, we apply min-max normalization to both before computing the final score.

Methods	Tokens ↓	Latency ↓ (Min:Sec)	FLOPs ↓	KV Cache ↓ (MB)	POPE ↑ (F1-Score)
Vanilla LLaVA-Next-7B	2880	36:16	100%	1512.1	86.5
+ FastV	320	18:17	**12.8%**	168.0	78.3
+ SparseVLM	320	23:11	15.6%	168.0	82.3
+ MustDrop	320	23:40	11.5%	168.0	82.1

Table 7: Inference costs of the number of tokens, Total-Time, FLOPs, and KV Cache Memory.

However, can FLOPs or token reduction ratios truly reflect the actual acceleration achieved?

To investigate this question, we examined the speedup effects reported by several works. Our findings reveal that even when different methods exhibit identical or similar reduction ratios and FLOPs, their measured speeds can vary significantly. Table 7 presents the efficiency-related experimental results of these methods on LLaVA-Next-7B[5]. Specifically, under the same setting, SparseVLM's FLOPs are only **2.8%** higher than those of FastV, yet its latency is **26.8%** greater. This strongly suggests that relying on FLOPs to evaluate acceleration effects of proposed methods is inadequate. When assessing speed gains, it is imperative to *shift our focus to actual latency measurements*.

Reason. We also conducted a detailed analysis of the design intricacies of the three methods to uncover the underlying reasons for their performance differences. Specifically, FastV, SparseVLM, and MustDrop all fail to support the efficient Flash Attention operator (Dao et al., 2022; Dao, 2024), as they rely on the complete attention map to select visual tokens. However, FastV performs token pruning in only one layer of the language model, whereas the other two methods conduct pruning across four layers. This implies that, compared to FastV, these methods have more layers that are forced to use the traditional attention operator with $O(N^2)$ memory costs. This could be one of the key factors contributing to their slower speeds. Additionally, performing pruning layer by layer requires more complex operations to handle token selection. If the runtime overhead introduced during this stage becomes significant, it may offset the speed gains achieved by shortening the token sequence. Moreover, some of the transformer layers where these methods perform pruning are located deeper within the model. Pruning tokens in such deep layers may yield limited benefits, as the impact of token reduction diminishes at later stages of the network.

Appeal. This insight motivates us to consider the compatibility with efficient attention operators when designing token pruning methods. Additionally, it encourages us to implement the token pruning process as early as possible in the shallow layers using simpler approaches, avoiding the risk of excessive runtime overhead that could otherwise overshadow the intended acceleration benefits.

Summary 4. *(i) FLOPs are not a reliable metric for evaluating speed gains; greater emphasis should be placed on actual latency. (ii) We advocate for the implementation of token pruning in the shallow layers of MLLMs using simple or efficient operations, while ensuring compatibility with Flash Attention.*

7.2 The Overlooked Role of Training-Aware Compression in MLLMs

Method	GQA	MMB	MMB-CN	MME	POPE	SQA	VQA^{Text}	Avg.
Qwen2-VL-7B	*Upper Bound, All Tokens* **(100%)**							
Vanilla	62.2	80.5	81.2	2317	86.1	84.7	82.1	100%
Qwen2-VL-7B	*Token Reduction* (↓ **66.7%**)							
+ FastV	58.0	76.1	75.5	2130	82.1	80.0	77.3	94.0% (-6.0%)
+ FastV†	61.9	80.9	81.3	2296	86.2	84.6	81.7	99.8% (-0.2%)
Qwen2-VL-7B	*Token Reduction* (↓ **77.8%**)							
+ FastV	56.7	74.1	73.9	2031	79.2	78.3	72.0	91.0% (-8.0%)
+ FastV†	61.9	80.8	81.2	2300	86.1	86.4	81.4	100.0% (0.0%)
Qwen2-VL-7B	*Token Reduction* (↓ **88.9%**)							
+ FastV	51.9	70.1	65.2	1962	76.1	75.8	60.3	84.0% (-16.0%)
+ FastV†	61.9	81.1	81.0	2289	86.2	84.4	81.3	99.6% (-0.4%)

Table 8: Comparative Experiments on Qwen2-VL-7B.

In recent years, some of the latest MLLMs have adopted various advanced techniques during the training phase to enhance their efficiency. For instance, Qwen2-VL employs token merging strategy during training, consolidating four adjacent patches into a single visual token. Similarly, MiniCPM-V-2.6 incorporates a learnable query within its resetting module, mapping variable-length segment features into more compact representations.

This raises an intriguing question: *If MLLMs already implement training-aware compression techniques, should we take this into account when designing and evaluating token pruning methods for the inference stage?* Given that the visual tokens encoded by these models possess higher information density, removing the same number of visual tokens could result in greater information loss compared to traditional approaches.

To this end, we selected a representative MLLM that employs training-aware compression, Qwen2-VL-7B-Instruct[6] and conducted a series of experimental analyses. Specifically, we applied FastV in two sets of experiments: one disregarding the token compression performed during Qwen2-VL's

[5] https://huggingface.co/liuhaotian/llava-v1.6-vicuna-7b

[6] https://huggingface.co/Qwen/Qwen2-VL-7B-Instruct

training phase, and the other taking it into account: Let $\mathcal{P}$ denote the original number of image patches, and after processing with PatchMerger, the number of visual tokens $\mathcal{V}$ is:

$$\mathcal{V} = \frac{\mathcal{P}}{\text{TACR}}, \tag{7}$$

where TACR means training-aware compression ratio and the value is 4.

Finally, the Token Reduction Rate (TRR) can be formally defined as:

$$\text{TRR}(\text{FastV}^{\dagger}) \triangleq \underbrace{\text{TACR}}_{\text{Training-aware}} \times \underbrace{\text{TFRR}}_{\text{Training-free}} . \tag{8}$$

Surprisingly, as shown in Table 8, we find that when taking training-aware compression into account, the same token pruning method achieves performance on par with the vanilla model across multiple benchmarks, even under varying reduction ratios. This observation prompts us to reflect: *perhaps more research effort should be directed toward training-aware token compression techniques*. Even in cross-model comparisons, such as between LLaVA-1.5-7B (Vanilla FastV) in Table 1, which does not employ training-aware compression, and Qwen2-VL-7B-Instruct (FastV†), the latter clearly demonstrates less performance degradation.

Summary 5. *Training-aware token compression techniques deserve more research attention due to their potential for delivering superior performance guarantees.*

8 Conclusion

Our systematic investigation into token pruning for MLLMs reveals several critical yet overlooked issues. While existing methods prioritize attention-based scoring and language-guided strategies, we demonstrate that naive spatial uniformity, achieved through random selection or pooling, often outperforms complex designs due to inherent positional biases in visual tokens. Notably, the effectiveness of linguistic guidance depends on task alignment: it enhances performance in text-driven scenarios through cross-modal attention but risks degradation in vision-centric tasks. From an information-theoretic perspective, we shed light on the core principles of token pruning, *i.e.,* the pursuit of structural integrity versus prediction accuracy. Furthermore, we challenge the conventional reliance on FLOPs for efficiency evaluation, showing that latency serves as a more practical and meaningful metric. These findings provide a refined framework to guide the development of future token pruning methods, balancing simplicity, effectiveness, and task-specific adaptability.

9 Limitations

Our experiments and analyses have been primarily conducted on LLaVA, LLaVA-Next, and Qwen2-VL. While these multimodal large language models are highly representative, our exploration should be extended to a broader range of model architectures. Such an expansion would enable us to uncover more intriguing findings and gain more robust and comprehensive insights. Additionally, we should apply our analytical framework and experimental evaluations to models of varying sizes, ensuring that our conclusions are not only diverse but also applicable across different scales of architecture.

Acknowledgements

This work is supported by National Key R&D Program of China (2022ZD0160201).

References

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-VL: A frontier large vision-language model with versatile abilities. *arXiv:2308.12966*.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. 2023. Token merging: Your ViT but faster. In *International Conference on Learning Representations*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*.

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

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. 2023. Palm-e: An embodied multimodal language model. *Preprint*, arXiv:2303.03378.

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

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

Shaohan Huang, Li Dong, Wenhui Wang, Yaru Hao, Saksham Singhal, Shuming Ma, Tengchao Lv, Lei Cui, Owais Khan Mohammed, Barun Patra, Qiang Liu, Kriti Aggarwal, Zewen Chi, Johan Bjorck, Vishrav Chaudhary, Subhojit Som, Xia Song, and Furu Wei. 2023. Language is not all you need: Aligning perception with language models. *Preprint*, arXiv:2302.14045.

Drew A Hudson and Christopher D Manning. 2019. GQA: A new dataset for real-world visual reasoning and compositional question answering. *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Peter E Latham and Yasser Roudi. 2009. Mutual information. *Scholarpedia*, 4(1):1658.

Yanwei Li, Chengyao Wang, and Jiaya Jia. 2024. LLaMA-VID: An image is worth 2 tokens in large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Evaluating object hallucination in large vision-language models. *arXiv:2305.10355*.

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

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

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

Ting Liu, Liangtao Shi, Richang Hong, Yue Hu, Quanjun Yin, and Linfeng Zhang. 2024d. Multi-stage vision token dropping: Towards efficient multimodal large language model. *arXiv preprint arXiv:2411.10803*.

Xuyang Liu, Yiyu Wang, Junpeng Ma, and Linfeng Zhang. 2025a. Video compression commander: Plug-and-play inference acceleration for video large language models. *arXiv preprint arXiv:2505.14454*.

Xuyang Liu, Ziming Wang, Yuhang Han, Yingyao Wang, Jiale Yuan, Jun Song, Bo Zheng, Linfeng Zhang, Siteng Huang, and Honggang Chen. 2025b. Compression with global guidance: Towards training-free high-resolution mllms acceleration. *Preprint*, arXiv:2501.05179.

Xuyang Liu, Zichen Wen, Shaobo Wang, Junjie Chen, Zhishan Tao, Yubo Wang, Xiangqi Jin, Chang Zou, Yiyu Wang, Chenfei Liao, Xu Zheng, Honggang Chen, Weijia Li, Xuming Hu, Conghui He, and Linfeng Zhang. 2025c. Shifting ai efficiency from model-centric to data-centric compression. *arXiv preprint arXiv:2505.19147*.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. 2025d. Mmbench: Is your multi-modal model an all-around player? In *European Conference on Computer Vision*, pages 216–233. Springer.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. *Preprint*, arXiv:1511.02283.

Linke Ouyang, Yuan Qu, Hongbin Zhou, Jiawei Zhu, Rui Zhang, Qunshu Lin, Bin Wang, Zhiyuan Zhao, Man Jiang, Xiaomeng Zhao, Jin Shi, Fan Wu, Pei Chu, Minghao Liu, Zhenxiang Li, Chao Xu, Bo Zhang, Botian Shi, Zhongying Tu, and Conghui He. 2024. Omnidocbench: Benchmarking diverse pdf document parsing with comprehensive annotations. *Preprint*, arXiv:2412.07626.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Amanpreet Singh, Vivek Natarjan, Meet Shah, Yu Jiang, Xinlei Chen, Devi Parikh, and Marcus Rohrbach. 2019. Towards VQA models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8317–8326.

Naftali Tishby, Fernando C Pereira, and William Bialek. 2000. The information bottleneck method. *arXiv preprint physics/0004057*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Zichen Wen, Dadi Guo, and Huishuai Zhang. 2024. Aidbench: A benchmark for evaluating the authorship identification capability of large language models. *arXiv preprint arXiv:2411.13226*.

Tsung-Han Wu, Giscard Biamby, Jerome Quenum, Ritwik Gupta, Joseph E. Gonzalez, Trevor Darrell, and David Chan. 2025. Visual haystacks: A vision-centric needle-in-a-haystack benchmark. In *The Thirteenth International Conference on Learning Representations*.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. 2016. Modeling context in referring expressions. *Preprint*, arXiv:1608.00272.

Junyuan Zhang, Qintong Zhang, Bin Wang, Linke Ouyang, Zichen Wen, Ying Li, Ka-Ho Chow, Conghui He, and Wentao Zhang. 2024a. Ocr hinders rag: Evaluating the cascading impact of ocr on retrieval-augmented generation. *arXiv preprint arXiv:2412.02592*.

Qizhe Zhang, Aosong Cheng, Ming Lu, Zhiyong Zhuo, Minqi Wang, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. 2024b. [cls] attention is all you need for training-free visual token pruning: Make vlm inference faster. *Preprint*, arXiv:2412.01818.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. 2024c. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*.

A Future Works

In this work, we have conducted an in-depth exploration of a series of issues related to token pruning. These include the position bias problem inherent in methods based on attention scores, the guiding role of linguistic information during token pruning, the importance and redundancy of tokens, as well as certain limitations in the evaluation of token pruning methods. Looking ahead, we plan to further expand the scope of our research by considering whether token pruning or token merging should be prioritized in the context of token reduction. Additionally, we aim to evaluate and analyze various token reduction methods on more challenging OCR benchmarks (Zhang et al., 2024a; Ouyang et al., 2024), particularly datasets featuring rich-text OCR images. This future work will not only deepen our understanding of token reduction strategies but also provide valuable insights into their practical applications in complex scenarios.

B Algorithms

In this section, we present some core algorithms for the methods mentioned in the main text. Vanilla FastV (Algorithm 1) selects tokens with the highest attention scores for retention. Reverse FastV (Algorithm 2) modifies this strategy by selecting tokens with the lowest attention scores instead. Window FastV (Algorithm 3) introduces a spatially-aware token selection mechanism by dividing the image tokens into local windows and performing token selection within each window. Finally, Pooling (Algorithm 4) applies a pooling operation over token grids to retain a structured subset of tokens, ensuring spatial consistency.

C Dataset

In this section, we introduce the content of the datasets used, as well as the input and output formats in Table 9.

Algorithm 1 Vanilla FastV

Require: Input token sequence $X \in \mathbb{R}^{L \times d}$, image token range $[s, e]$, retention ratio r
Ensure: Compressed sequence representation X'

Initialize layer parameters $\{W_i\}_{i=1}^{N}$
for layer $l = 1$ **to** N **do**
  if $l = K - 1$ **then**
    Compute attention matrix A
    Record global attention scores $\alpha = \text{mean}(A)[s:e]$
  else if $l = K$ **then**
    Select top-k indices $I = \text{topk}(\alpha, \lfloor (e-s)r \rfloor)$
    Construct retention indices $\mathcal{I} = [0:s) \cup I \cup [e:L]$
    Compress sequence $X' = X[\mathcal{I}, :]$
    Update attention mask $M' = M[\mathcal{I}, \mathcal{I}]$
  else
    Regular Transformer computation
  end if
end for

Algorithm 2 Reverse FastV

Require: Input token sequence $X \in \mathbb{R}^{L \times d}$, image token range $[s, e]$, retention ratio r
Ensure: Compressed sequence representation X'

Initialize layer parameters $\{W_i\}_{i=1}^{N}$
for layer $l = 1$ **to** N **do**
  if $l = K - 1$ **then**
    Compute attention matrix A
    Record global attention scores $\alpha = \text{mean}(A)[s:e]$
    $\alpha = -\alpha$ ▷ Difference from Vanilla FastV
  else if $l = K$ **then**
    Select top-k indices $I = \text{topk}(-\alpha, \lfloor (e-s)r \rfloor)$
    Construct retention indices $\mathcal{I} = [0:s) \cup I \cup [e:L]$
    Compress sequence $X' = X[\mathcal{I}, :]$
    Update attention mask $M' = M[\mathcal{I}, \mathcal{I}]$
  else
    Regular Transformer computation
  end if
end for

Type	Dataset	Brief Description	Input	Output
Visual Understanding	VQA^{Text}	Rich textual viusal QA	Single image and question	Question Answer
	VQA V2	Open-ended viusal perception	Single image and question	Question Answer
	ScienceQA	Natural and social science QA	Single image and question	Question Answer
	POPE	Object hallucination evaluation	Single image and question	Question Answer
	GQA	Visual scene understanding	Single image and question	Question Answer
	MMBench	Perception and reasoning tasks	Single image and question	Question Answer
	MME	Perceptual ability evaluation	Single image and question	Question Answer
Object Recognition	Visual Haystack	Visual need-in-a-haystack	Multiple images, an image phrase and an object phrase	Existence of specified objects
Grounding	RefCOCO	Phrase object localizing	1 image and referring phrase of an object	Bounding box of the specified object

Table 9: The datasets we use for benchmarking.

Algorithm 3 Window FastV

Require: Input sequence $X \in \mathbb{R}^{L \times d}$, image region $\Omega = [s, e]$, window size (h, w)
Ensure: Compressed sequence representation X'

Initialize layer parameters $\{W_i\}_{i=1}^{N}$
for layer $l = 1$ **to** N **do**
  if $l = K - 1$ **then**
    Compute attention matrix A
    Record global attention scores $\alpha = \text{mean}(A)[s:e]$
  else if $l = K$ **then**
    Reshape the image region into a 2D grid $\Gamma \in \mathbb{R}^{h \times w}$
    Divide the grid into window patches $\{\mathcal{W}_{ij}\}_{i=1,j=1}^{m,n}$, where $\mathcal{W}_{ij} \subset \Gamma$
    for each window $\mathcal{W}_{ij}$ **do**
      Compute local attention scores $A_{ij} = \text{mean}(\alpha[\mathcal{W}_{ij}])$
      Select local top-k indices $I_{ij} = \text{topk}(A_{ij}, mn)$
      Convert local indices to global coordinates $\mathcal{G}_{ij} = \text{loc2glob}(I_{ij})$
    end for
    Aggregate all window indices $\mathcal{I} = \bigcup_{i,j} \mathcal{G}_{ij}$
    Construct the retained sequence: $X' = X\left[[0:s) \cup \mathcal{I} \cup [e:L], :\right]$
  else
    Regular Transformer computation
  end if
end for

Algorithm 4 Pooling

Require: Input sequence $X \in \mathbb{R}^{L \times d}$, image region $\Omega = [s, e]$, window size $a \times a$
Ensure: Compressed sequence representation X'

for layer $l = 1$ **to** N **do**
  if $l = K$ **then**
    Extract image tokens: $X_{img} = X[s : e, :]$
    Reshape into a 2D grid: $F \in \mathbb{R}^{h \times w \times d}$
      ▷ Where $h \times w = e - s$
    Perform window pooling:

$$\hat{F} = \text{Pool}(F, a, \rho) \in \mathbb{R}^{(h/a) \times (w/a) \times d}$$

    Construct index mapping:

$$\mathcal{M} = \{(i,j) \mapsto \underset{(p,q) \in \mathcal{W}_{ij}}{\text{argmax}} \ \|F[p,q,:]\|_1\}$$

    Build the retained index set:

$$\mathcal{I} = \{s + \mathcal{M}(k) | \forall k \in [1, (h/a)(w/a)]\}$$

    Generate the compressed sequence:

$$X' = X\left[[0:s) \cup \mathcal{I} \cup [e:L], :\right]$$

  else
    Regular Transformer computation
  end if
end for