LLMC+: Benchmarking Vision-Language Model Compression with a plug-and-play Toolkit

Chengtao Lv[1, 3*], **Bilang Zhang**[2, 3*], **Yang Yong**[3], **Ruihao Gong**[2, 3†], **Yushi Huang**[3, 4*], **Shiqiao Gu**[3], **Jiajun Wu**[2], **Yumeng Shi**[1], **Jinyang Guo**[2], **Wenya Wang**[1†]

[1]Nanyang Technological University
[2]Beihang University
[3]SenseTime Research
[4]Hong Kong University of Science and Technology

Abstract

Large Vision-Language Models (VLMs) exhibit impressive multi-modal capabilities but suffer from prohibitive computational and memory demands, due to their long visual token sequences and massive parameter sizes. To address these issues, recent works have proposed training-free compression methods. However, existing efforts often suffer from three major limitations: (1) Current approaches do not decompose techniques into comparable modules, hindering fair evaluation across spatial and temporal redundancy. (2) Evaluation confined to simple single-turn tasks, failing to reflect performance in realistic scenarios. (3) Isolated use of individual compression techniques, without exploring their joint potential. To overcome these gaps, we introduce LLMC+, a comprehensive VLM compression benchmark with a versatile, plug-and-play toolkit. LLMC+ supports over 20 algorithms across five representative VLM families and enables systematic study of token-level and model-level compression. Our benchmark reveals that: (1) Spatial and temporal redundancies demand distinct technical strategies. (2) Token reduction methods degrade significantly in multi-turn dialogue and detail-sensitive tasks. (3) Combining token and model compression achieves extreme compression with minimal performance loss. We believe LLMC+ will facilitate fair evaluation and inspire future research in efficient VLM.

Code — https://github.com/ModelTC/LightCompress

1 Introduction

Recently, Large Language Models (LMMs) (Touvron et al. 2023; Liu et al. 2024a; Brown et al. 2020) have achieved rapid advancements in Natural Language Processing (NLP), which has become a significant milestone in the AI revolution. This breakthrough has quickly extended to vision modalities: mainstream Vision Language Models (VLMs) (Liu et al. 2023, 2024b; Wang et al. 2024a; Chen et al. 2024b) typically encode visual inputs into tokens and unify multiple modalities within a shared embedding space, demonstrating strong visual-language understanding and generation capabilities in various tasks (Singh et al. 2019; Antol et al. 2015; Hudson and Manning 2019).

However, their remarkable capabilities can be largely attributed to two aspects: 1) The number of visual tokens often reaches hundreds or even thousands, dominating the input. For example, images in LLaVA-NeXT (Li et al. 2024a) are converted into 2,880 tokens. While in video streams or high-resolution scenarios, the number of tokens increases dramatically, further intensifying the computational costs. 2) VLMs have massive memory footprints (*e.g.*, billion-scale parameters). Some large-scale VLMs, such as Qwen2.5-VL-72B (Wang et al. 2024a), consume approximately 140GB of memory for storage, becoming a major GPU memory bottleneck during inference. These two issues constrain their widespread application on resource-limited devices.

To effectively mitigate intractable computational and memory overhead, several training-free compression works have been proposed successively, which can be briefly classified into two fields: 1) *token-level compression*. These methods typically reduce less salient visual tokens through token reduction (Bolya et al. 2022; Chen et al. 2024a). 2) *model-level compression*. Their primary objective is to squeeze model weights through techniques like quantization (Gong et al. 2025b), network pruning (Sun et al. 2023), and low-rank factorization (Wang et al. 2024b).

Nevertheless, three worrisome problems still appear in current training-free compression research for VLMs. First, existing methods often target different types of redundancy (*e.g.*, spatial or temporal) with distinct technical dimensions, leading to a lack of fair comparison and in-depth analysis across these dimensions. Second, these methods are limited to evaluation on general single-turn VQA tasks, lacking comprehensive assessments on challenging and practical tasks. Third, they typically rely on a single compression measure, without exploring the risks and potential of joint multiple compression strategies.

To this end, this paper presents LLMC+, a VLM compression benchmark with a versatile toolkit covering *token-level* and *model-level* compression. Specifically, LLMC+ supports over 20 compression algorithms and five different families of VLMs. 1) Based on LLMC+, we introduce a novel token reduction taxonomy specifically for handling

*Work done during internships at SenseTime Research.
†Corresponding authors.

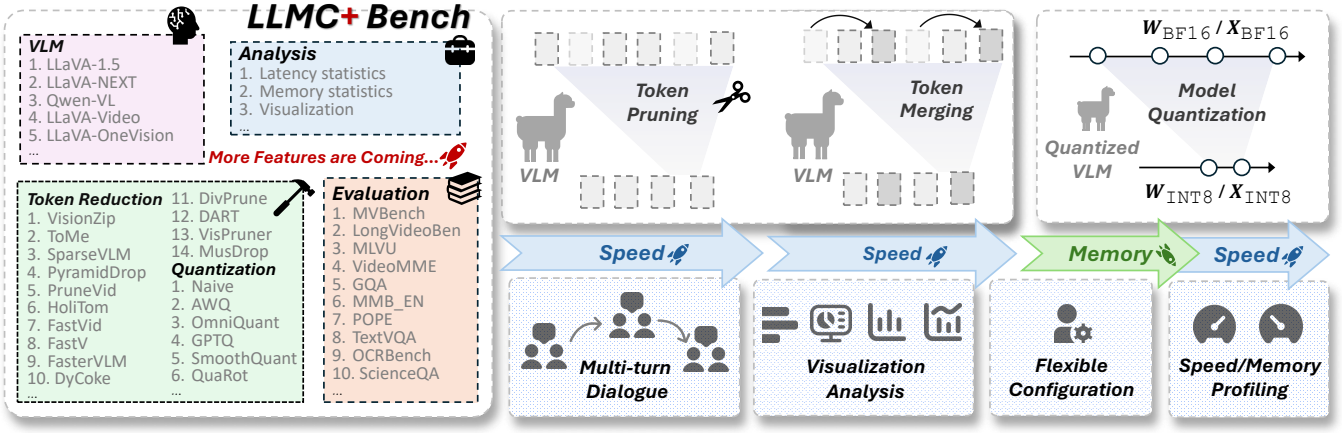

Figure 1: Illustration of our proposed powerful toolkit, LLMC+. Due to its high flexibility and versatility, we build a VLM compression benchmark upon it and conduct an in-depth analysis.

spatial and temporal redundancy. We further distill their core technical dimensions, covering metrics (*attention-based vs. similarity-based*), solutions (*merge vs. prune*), and video segmentation strategies (*fixed vs. dynamic*). Extensive experiments are conducted along these dimensions, accompanied by in-depth analysis. 2) Besides, we evaluate VLMs on practical task scenarios, such as multi-turn dialogue and detail-sensitive tasks like OCR and DocVQA, to uncover potential risks introduced by compression. 3) Finally, we combine token reduction and quantization to achieve extreme compression. We deploy quantized kernels on hardware to validate real acceleration and memory saving. Our benchmark reveals that 1) Spatial and temporal redundancy require distinct core strategies to be handled effectively. 2) Token reduction methods suffer from non-trivial performance degradation on practical tasks. 3) Combining multiple techniques enables extreme compression with accuracy guarantees.

We emphasize that the contributions of our LLMC+ can be summarized as follows:

- *Versatile Toolkit.* LLMC+ is the **first** plug-and-play compression toolkit specifically designed for VLMs, supporting over 20 compression algorithms across five different families of VLMs with flexible configuration.
- *Modular Comparison.* We construct a comprehensive taxonomy for token reduction that comprises all core technical modules, and conduct systematic evaluations for each module to ensure fair comparison.
- *Practical Evaluation.* Our findings highlight flaws in current evaluation practices and advocate for integrating our proposed practical evaluations into future evaluation standards.
- *Best Practice.* By following the modular guidelines proposed in this paper and combining token reduction with quantization, we achieve extreme compression.

2 LLMC+ Implementation

To enable comprehensive and fair comparison, we develop a versatile compression toolkit for VLMs. We highlight several key features in Fig. 1.

2.1 Various Algorithms and Models

LLMC+ supports a wide range of compression schemes, which can be broadly categorized into two fields. The first category includes 15 token reduction algorithms that concentrate on accelerating inference speed. The second category comprises model compression methods, covering 6 quantization algorithms, including both weight-only and weight-activation quantization. Moreover, LLMC+ integrates VLMs from different families, ranging from traditional image-based VLMs to video-oriented ones, including LLaVA-1.5 (Liu et al. 2023), LLaVA-NeXT (Li et al. 2024a), Qwen2.5-VL (Bai et al. 2025), Video-LLaVA (Lin et al. 2023), and LLaVA-OneVision (Li et al. 2024a).

2.2 Flexible Configuration

The modular design of LLMC+ facilitates modality-aware compression (*e.g.*, Vision Tower and LLM), seamless integration of diverse compression techniques (*e.g.*, token reduction and quantization), multi-turn dialogues, and convenient configuration for latency/memory profiling and visualization. LLMC+ enables developers to perform customized analysis and compression tailored to their specific needs.

2.3 Benchmarks

LLMC+ is integrated with LMMs-Eval (Zhang et al. 2024a) for evaluation. For image tasks, we conduct experiments on nine widely used image understanding benchmarks, including visual question answering benchmarks such as GQA (Hudson and Manning 2019), ScienceQA (Saikh et al. 2022), TextVQA (Singh et al. 2019), and VizWiz (Bigham et al. 2010), as well as multi-modal reasoning benchmarks such as MMBench (Liu et al. 2024d), MME (Fu et al. 2023), POPE (Li et al. 2023), OCRBench (Liu et al. 2024e), and DocVQA (Tito, Karatzas, and Valveny 2023). For video tasks, we select four standard video understanding benchmarks: MVBench (Li et al. 2024b), LongVideoBench (Wu et al. 2024), MLVU (Zhou et al. 2024), and VideoMME (Fu et al. 2025). These benchmarks cover complex scenarios and

Method	Venue	Vision		Language		Progressive
		Prune	Merge	Prune	Merge	
ToMe	ICLR 2023	✗	S	✗	✗	✓
FastV	ECCV 2024	✗	✗	A	✗	✗
SparseVLM	ICML 2025	✗	✗	A	S	✓
PDrop	CVPR 2025	✗	✗	A	✗	✓
VisionZip	CVPR 2025	A	S	✗	✗	✗
VisPruner	ICCV 2025	A+S	✗	✗	✗	✗
DART	EMNLP 2025	✗	✗	S	✗	✗
DivPrune	CVPR 2025	S	✗	✗	✗	✗
MustDrop	arXiv 2024	A	S	A	✗	✗
HoliTom	NeurIPS 2025	✗	S	✗	S	✗

Table 1: Solutions for spatial redundancy. "A" and "S" denote attention- and similarity-based metrics, respectively.

Method	Venue	Segment	Prune	Merge
DyCoke	CVPR 2025	F	✓	✗
PruneVid	ACL 2025	D	✗	✓
FastVID	NeurIPS 2025	D	✗	✓
HoliTom	NeurIPS 2025	D	✗	✓

Table 2: Solutions for temporal redundancy. "F" and "D" denote Fixed and Dynamic segmentation strategies.

varying durations, enabling a comprehensive evaluation of effectiveness and generalization ability for these methods.

3 Dive into Token Reduction

In this section, we first present our taxonomy for addressing spatial and temporal redundancy (Sec. 3.1), which comprises two core technical aspects, respectively. For spatial redundancy, we conduct experiments in the vision (Sec. 3.2) and language components (Sec. 3.3) separately. For temporal redundancy, we distill a two-step pipeline and evaluate key techniques at each step (Sec. 3.4).

3.1 Taxonomy

Token reduction primarily aims to eliminate redundancy in visual inputs. While images exhibit *spatial redundancy*, videos additionally contain *temporal redundancy*.

First, we present a taxonomy of methods designed to address spatial redundancy, as demonstrated in Tab. 1. Based on different perspectives, we divide them into the following categories:

- *Attention-based vs. Similarity-based metric*. Attention-based metrics typically rely on the question prompt or the [CLS] token, while similarity-based metrics measure the pairwise distance between tokens.
- *Prune vs. Merge.* Pruning discards insignificant visual tokens, whereas merging fuses them into other tokens.

Similarly, we categorize methods (see Tab. 2) addressing temporal redundancy into the following types:

- *Fixed vs. Dynamic segmentation*. Fixed segmentation partitions frames into segments of equal length, whereas dynamic segmentation generates segments that may vary in length.
- *Prune vs. Merge.* Similar to the above, but primarily applied to consecutive frames.

3.2 Token Reduction in Vision Tower

We report the detailed results of token reduction schemes for Vision Tower in Tab. 3. It is worth noting that for each row, like "VisionZip PA" indicates the use of an **A**ttention-based metric to **P**rune the unimportant tokens. We have several key observations: (1) *Prune-based methods generally outperform Merge-based ones in Vision Tower.* For example, three prune-based methods significantly outperform "ToMe MS" by a large margin. (2) *Similarity-based and Attention-based metrics, such as VisionZip PA (Yang et al. 2025) and DivPrune PS (Alvar et al. 2025), exhibit only trivial differences in accuracy for most settings.* In detail, [CLS] attention serves as a strong indicator of token importance within the encoder and has demonstrated superior performance compared to other methods, as evidenced by its adoption in several previous works (Yang et al. 2025; Zhang et al. 2024b; Liu et al. 2024c). However, for some advanced Vision Towers that cannot obtain [CLS] attention, such as SigLIP (Zhai et al. 2023). We believe there are two possible solutions. The first is to apply a similarity-based token pruning method, such as DivPrune PS (Alvar et al. 2025). The second is to reintegrate the SigLIP head into the model to generate [CLS] token, which incurs negligible computational overhead.

Besides, some methods apply token reduction in the shallow layers of Vision Tower. For instance, MustDrop MS (Liu et al. 2024c) performs spatial merge with sliding windows in the first layer of Vision Tower, while ToMe MS (Bolya et al. 2022) adopts a progressive token reduction strategy, where tokens are gradually pruned layer by layer during forward inference. To compare the impact of applying these methods at different depths, we measure their performance when applied to either shallow or deep layers of Vision Tower. Tab. 4 shows that applying token reduction to shallow layers leads to significant accuracy degradation, while offering slight improvements in prefill time. Therefore, *we recommend performing token reduction at the last layer of Vision Tower for a better trade-off between efficiency and accuracy.* However, these methods are not applicable to models like Qwen2.5-VL (Bai et al. 2025), because they occur before the model's built-in spatial token merger module.

3.3 Token Reduction in LLM

In contrast to text-agnostic token reduction methods applied within Vision Tower, token reduction within the LLM often leverages the question prompt as guidance for identifying important tokens. These methods typically perform token reduction in the shallow layers of the LLM. For a fair comparison, we implement all methods in our taxonomy (Tab. 1) at 5-th layer of the LLM during assessment. *The effectiveness of methods in LLM varies substantially*

[1]The seven benchmarks include GQA, MMB_EN, MME, POPE, TextVQA, VizWiz_VQA, and ScienceQA. Detailed results are provided in the **Appendix**.

Model	Method	Type	Acc.	Rel.	Acc.	Rel.	Acc.	Rel.
			Upper Bound, 576 Tokens (100%)					
	Vanilla	-	64.3	100%	64.3	100%	64.3	100%
			192 Tokens (↓ 66.7%)		*128 Tokens (↓ 77.8%)*		*64 Tokens (↓ 88.9%)*	
	VisionZip (Yang et al. 2025)	PA	62.9	97.9%	62.1	96.7%	60.5	94.1%
	VisPruner (Zhang et al. 2024b)	PS	62.5	97.2%	61.3	95.4%	59.7	92.8%
	DivPrune (Alvar et al. 2025)	PS	62.9	97.8%	62.4	97.1%	60.9	94.7%
LLaVA-1.5-7B (Liu et al. 2024b)	ToMe (Bolya et al. 2022)	MS	61.6	95.8%	61.0	94.9%	59.2	92.1%
	VisionZip (Yang et al. 2025)	MS	60.8	94.5%	59.4	92.4%	56.3	87.5%
	MustDrop (Liu et al. 2024c)	MS	61.7	95.9%	59.3	92.3%	52.0	80.9%
	VisionZip (Yang et al. 2025)	PA+MS	62.9	97.9%	62.2	96.7%	60.4	93.9%
			Upper Bound, ∼2880 Tokens (100%)					
	Vanilla	-	68.6	100%	68.6	100%	68.6	100%
			∼640 Tokens (↓ 66.7%)		*∼320 Tokens (↓ 77.8%)*		*∼160 Tokens (↓ 88.9%)*	
	VisionZip (Yang et al. 2025)	PA	66.8	97.3%	64.9	94.6%	62.2	90.6%
	VisPruner (Zhang et al. 2024b)	PS	64.9	94.6%	62.7	91.4%	59.6	86.8%
LLaVA-NeXT-7B (**?**)	DivPrune (Alvar et al. 2025)	PS	65.6	95.5%	63.8	93.0%	62.3	90.8%
	VisionZip (Yang et al. 2025)	MS	62.7	91.4%	60.1	87.6%	56.1	81.8%
	MustDrop (Liu et al. 2024c)	MS	64.9	94.5%	61.3	89.2%	-	-
			Upper Bound, 100% Tokens					
	Vanilla	-	79.3	100%	79.3	100%	79.3	100%
			33.3% Tokens (↓ 66.7%)		*22.2% Tokens (↓ 77.8%)*		*11.1% Tokens (↓ 88.9%)*	
	VisionZip (Yang et al. 2025)	PA	77.9	98.2%	75.9	95.7%	70.7	89.2%
	VisPruner (Zhang et al. 2024b)	PS	76.8	96.8%	74.4	93.9%	68.8	86.7%
Qwen2.5-VL-7B (Bai et al. 2025)	DivPrune (Alvar et al. 2025)	PS	77.3	97.4%	75.6	95.3%	71.5	90.2%
	VisionZip (Yang et al. 2025)	MS	76.9	97.0%	74.6	94.1%	68.3	86.1%

Table 3: Average performance of different token reduction methods in Vision Tower across seven benchmarks[1].

Method	Type	Acc.	Rel.	Prefill
Upper Bound, 576 Tokens (100%)				
LLaVA-1.5-7B (Liu et al. 2024b)	-	64.3	100%	27.3 ms
Retain 192 Tokens in Average (↓ 66.7%)				
ToMe* (Bolya et al. 2022)	MS	53.5	83.2%	16.6 ms
ToMe† (Bolya et al. 2022)	MS	62.4	97.0%	16.6 ms
MustDrop* (Liu et al. 2024c)	MS	58.8	91.4%	16.6 ms
MustDrop† (Liu et al. 2024c)	MS	61.7	95.9%	16.6 ms
Retain 128 Tokens in Average (↓ 77.8%)				
ToMe* (Bolya et al. 2022)	MS	45.3	70.5%	15.8 ms
ToMe† (Bolya et al. 2022)	MS	60.3	93.8%	16.0 ms
MustDrop* (Liu et al. 2024c)	MS	54.2	84.3%	16.0 ms
MustDrop† (Liu et al. 2024c)	MS	59.3	92.3%	16.2 ms

Table 4: Results of token reduction at different layers in Vision Tower. * and † denote merging in the first and last layer.

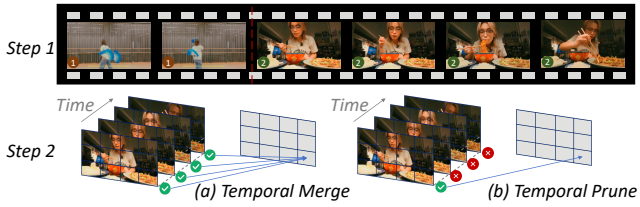

Figure 2: The pipeline of removing temporary redundancy in the two steps.

across model families. For instance, within the LLaVA family, prune-based approaches (i.e., FastV PA (Chen et al. 2024a), SparseVLM PA (Zhang et al. 2024c), and DART PS (Wen et al. 2025)) consistently outperform merge-based schemes (e.g., HoliTom MS (Shao et al. 2025)). In contrast, for Qwen2.5-VL, HoliTom MS (Shao et al. 2025) demonstrates competitive performance. Token reduction methods applied within the LLM typically introduce additional computational overhead but tend to achieve better performance compared to those applied in Vision Tower. For instance, DART PS (Wen et al. 2025) can maintain nearly lossless performance on LLaVA-1.5-7B even when preserving only one-third of the visual tokens. *This also highlights a key limitation of attention-based metrics within the LLM*. They tend to assign higher attention scores to visual tokens that are spatially closer to text tokens (Zhang et al. 2024b) (See in the Appendix). Therefore, it is necessary to reconsider the use of attention-based metrics for token reduction in LLM.

After comparing prune and merge algorithms in both the vision and language components, we further investigate whether combining token prune and merge can lead to improved performance. Recent works (Yang et al. 2025; Zhang et al. 2024c) usually follow a two-step pipeline to integrate prune and merge: important tokens are first preserved through pruning, and then the remaining less informative contextual tokens are merged. Therefore, we select a representative algorithm from each of the vision and language parts for our experiments. As shown in Tab. 3 and Tab. 5, the standalone prune method and the combination (*i.e.*, PA+MS) are compared. It can be observed that in most cases, introducing merge in addition to pruning does not significantly improve the model's performance, and in some settings, it even leads to slightly lower accuracy compared to prune alone.

Model	Method	Type	Acc.	Rel.	Acc.	Rel.	Acc.	Rel.
			Upper Bound, 576 Tokens (100%)					
	Vanilla	-	64.3	100%	64.3	100%	64.3	100%
			192 Tokens (↓ 66.7%)		*128 Tokens (↓ 77.8%)*		*64 Tokens (↓ 88.9%)*	
LLaVA-1.5-7B (Liu et al. 2024b)	FastV (Chen et al. 2024a)	PA	62.4	97.1%	60.9	94.7%	56.8	88.3%
	SparseVLM (Zhang et al. 2024c)	PA	63.3	98.5%	62.4	97.1%	60.0	93.3%
	DART (Wen et al. 2025)	PS	63.7	99.0%	62.9	97.9%	61.4	95.5%
	HoliTom (Shao et al. 2025)	MS	61.5	95.6%	60.1	93.5%	57.5	89.4%
	SparseVLM (Zhang et al. 2024c)	PA+MS	63.2	98.2%	62.6	97.3%	60.5	94.0%
			Upper Bound, ∼2880 Tokens (100%)					
	Vanilla	-	68.6	100%	68.6	100%	68.6	100%
			∼640 Tokens (↓ 66.7%)		*∼320 Tokens (↓ 77.8%)*		*∼160 Tokens (↓ 88.9%)*	
LLaVA-NeXT-7B (**?**)	FastV (Chen et al. 2024a)	PA	65.8	95.9%	61.7	89.9%	56.0	81.5%
	SparseVLM (Zhang et al. 2024c)	PA	66.9	97.5%	64.8	94.5%	61.3	89.2%
	DART (Wen et al. 2025)	PS	67.2	98.0%	65.2	95.0%	62.1	90.5%
	HoliTom (Shao et al. 2025)	MS	64.7	94.2%	61.5	89.6%	54.6	79.6%
			Upper Bound, 100% Tokens					
	Vanilla	-	79.3	100%	79.3	100%	79.3	100%
			33.3% Tokens (↓ 66.7%)		*22.2% Tokens (↓ 77.8%)*		*11.1% Tokens (↓ 88.9%)*	
Qwen2.5-VL-7B (Bai et al. 2025)	FastV (Chen et al. 2024a)	PA	75.9	95.7%	72.5	91.4%	64.5	81.3%
	SparseVLM (Zhang et al. 2024c)	PA	77.6	97.9%	76.1	95.9%	72.0	90.7%
	DART (Wen et al. 2025)	PS	76.6	96.6%	74.6	94.1%	69.2	87.3%
	HoliTom (Shao et al. 2025)	MS	77.2	97.4%	75.0	94.6%	69.8	88.1%

Table 5: Average performance of different token reduction methods in LLM across seven benchmarks.

Method	MR	RR	Segment	Prefill	MVBench	LongVideoBench	MLVU	VideoMME	Score	%
Duration					16 sec	1∼60 min	3∼120 min	1∼60 min		
Vanilla	-	100%	-	290.7ms	58.35	56.55	63.10	58.51	59.13	100.0%
Fixed	50%	56.2%	0 ms	164.7 ms	58.67	55.87	63.65	59.22	59.35	100.4%
PruneVid	50%	54.6%	3.3 ms	159.5 ms	58.45	55.80	62.76	58.70	58.93	99.7%
FastVID	50%	56.2%	2.5 ms	165.7 ms	58.35	54.90	62.98	59.00	58.81	99.5%
Fixed	80%	30.3%	0 ms	88.1 ms	57.98	54.37	60.79	58.18	57.83	97.8%
PruneVid	80%	27.7%	3.3 ms	82.5 ms	57.80	54.53	61.89	57.34	57.89	97.9%
FastVID	80%	30.3%	2.5 ms	88.1 ms	57.90	54.82	61.69	57.81	58.06	98.2%
HoliTom	-	52.4%	35.5 ms	149.0 ms	57.70	56.17	63.42	59.33	59.16	100.1%
HoliTom	-	68.9%	35.5 ms	199.6 ms	58.35	55.80	63.30	58.85	59.08	99.9%

Table 6: Comparison of four video segmentation methods on LLaVA-OneVision (Li et al. 2024a). **M**erge **R**ate (MR) denotes the intra-segment merge ratio, whereas **R**etention **R**ate (RR) measures the overall token retention for an input. This also implies that even with the same Merge Rate, different segment lengths will lead to varying Retention Rates.

3.4 From Image to Video

Although several token reduction methods for spatial redundancy have been discussed in the aforementioned sections, directly applying these solutions to video tasks overlooks a key characteristic of video data that necessitates a rethinking of traditional pruning strategies. Compared with images, videos naturally introduce an additional form of considerable redundancy: temporal redundancy. This redundancy primarily arises from similar tokens in adjacent frames, which fail to provide additional informative content. Therefore, recent studies (Tao et al. 2025; Fu et al. 2024; Shao et al. 2025; Huang, Zhou, and Han 2024; Shen et al. 2025) aim to reduce such redundancy.

Specifically, these approaches tailored for videos mainly follow a two-step paradigm (Fig. 2). In the first step, these algorithms partition a video into temporally ordered segments with high similarity, which often correspond to the same scene. In the second step, they reduce unimportant visual tokens within each segment while preserving informative ones. The remainder of this section explores how to perform these two steps to effectively reduce temporal redundancy.

First step: To thoroughly explore different segment partition strategies, we summarize four representative methods, covering DyCoke (Tao et al. 2025), FastVID (Shen et al. 2025), PruneVid (Huang, Zhou, and Han 2024), and HoliTom (Shao et al. 2025). To ensure a fair comparison, we apply a uniform merge strategy to highly similar tokens within each segment after partitioning. In addition to reporting performance under comparable compression ratios, we also provide the segmentation time and overall prefill time.

In Tab. 6, we report the detailed results of these schemes. When the merge rate is relatively low (around 50%), all

Method	MVBench	LongVideoBench	MLVU	VideoMME	Avg.
Duration	16 sec	1∼60 min	3∼120 min	1∼60 min	
DyCoke*	58.35	63.10	63.58	59.44	61.12
DyCoke†	58.23	63.10	62.99	58.74	60.77
PruneVid*	57.28	63.30	63.30	59.55	60.86
PruneVid†	56.58	63.31	63.31	59.40	60.65

Table 7: Comparison between Temporal Merge (superscript $*$) and Temporal Prune (superscript †).

methods achieve near-lossless performance, demonstrating the substantial temporal redundancy inherent in video tasks. As more tokens are merged, the fixed segmentation strategy exhibits a noticeable drop in accuracy. In contrast, dynamic segmentation methods such as PruneVid (Huang, Zhou, and Han 2024) and FastVID (Shen et al. 2025) achieve better performance while retaining the same or even fewer tokens. In terms of segmentation latency, HoliTom incurs an additional overhead of 35.5ms largely due to its use of dynamic programming to determine segment boundaries, which introduces an $\mathcal{O}(n^2)$ computational cost. In contrast, other dynamic segmentation methods (e.g., PruneVid, FastVID), require only 3.5ms and 2.5ms, making their latency negligible.

Second step: After obtaining high-quality segments, a natural question arises: *How can we eliminate the temporal redundancy of consecutive frames within each segment?* DyCoke (Tao et al. 2025) preserves the visual tokens in the first frame while pruning those in subsequent frames (Fig. 2 (b)), whereas PruneVid (Huang, Zhou, and Han 2024) merges similar tokens to reduce redundancy (Fig. 2 (a)). To compare these two schemes, we conduct both temporal merge and prune experiments for DyCoke and PruneVid.

Interestingly, we observe similar results for these two algorithms (Tab. 7). Temporal merge outperforms temporal prune (61.22 *vs.* 60.77 for DyCoke and 60.86 *vs.* 60.65 for PruneVid). *These findings indicate that merge may be more suitable than prune for overcoming temporal redundancy.* The main reason is the high similarity between tokens at corresponding positions across consecutive frames within the same segment. See the Appendix for details.

Finding 1. **Spatial Redundancy:** ① Vision Tower: Similarity-based and Attention-based metrics perform comparably, while Prune generally outperforms Merge. ② LLM: Sophisticated metrics and solutions should be chosen to suit different scenarios. **Temporal Redundancy:** Dynamic segment outperforms Fixed ones. Merge is slightly more effective than prune.

4 Compression Struggles in Practical Tasks

Although the aforementioned methods (Yang et al. 2025; Zhang et al. 2024c; Wen et al. 2025; Alvar et al. 2025) have demonstrated strong performance on general single-turn VQA tasks, they neglect systematic evaluation on practical tasks. First, real-world applications often involve fine-grained tasks that require a far more precise understanding of visual details, such as DocVQA (Tito, Karatzas, and Valveny 2023) and OCRBench (Liu et al. 2024e). Second,

Method	∼640 Tokens		∼320 Tokens		∼160 Tokens	
	DOC	OCR	DOC	OCR	DOC	OCR
Vanilla	68.5	52.1	68.5	52.1	68.5	52.1
FastV	46.6	38.9	29.7	24.8	17.8	15.4
DART	54.6	44.2	42.3	35.4	29.4	26.8
SparseVLM	49.1	40.6	31.1	29.0	17.3	18.6
VisionZip	56.9	48.5	42.5	39.5	28.8	29.5
VisPruner	53.9	44.8	39.6	37.3	27.4	31.7
DivPrune	44.8	37.6	32.8	33.0	26.1	26.6

Table 8: DocVQA (Tito, Karatzas, and Valveny 2023) and OCRBench (Liu et al. 2024e) results of token reduction.

Method	Text Info	Score
VisionZip (Yang et al. 2025)	None	0.938
DivPrune (Alvar et al. 2025)	None	0.929
DART (Wen et al. 2025)	Low	0.903
FastV (Chen et al. 2024a)	High	0.896
SparseVLM (Zhang et al. 2024c)	High	0.882

Table 9: Performance comparison of various methods on multi-turn dialogue tasks.

modern inference engines (Zheng et al. 2024; Gong et al. 2025a) for VLMs widely use prefix caching in multi-turn dialogue, reusing encoded visual-text prefixes to reduce redundant computation. Hence, evaluating the effectiveness of compression techniques on both task types is essential.

Fine-Grained Tasks. We select two detail-sensitive tasks, DocVQA (Tito, Karatzas, and Valveny 2023) and OCRBench (Liu et al. 2024e), to evaluate the accuracy degradation introduced by token reduction under different compression ratios on LLaVA-NeXT-7B (Li et al. 2024a). *In Tab. 8, we observe that the accuracy of token reduction on these tasks is still far from satisfactory.* For example, retaining approximately 160 tokens leads to non-trivial performance degradation on these tasks, with accuracy dropping to around 50%, in stark contrast to the around 80% accuracy achieved on general VQA benchmarks (Tab. 3 and Tab. 5). Therefore, subsequent works should place greater emphasis on these more challenging fine-grained tasks.

Multi-Turn Tasks. We further build a multi-turn dialogue dataset tailored for token reduction in a simple-yet-effective manner. Instead of constructing a dataset from scratch, we build upon the existing visual question answering benchmark, GQA (Hudson and Manning 2019), by selecting two semantically distinct questions for each image, serving as the first-turn and second-turn questions. Since the first-turn and second-turn questions are different, and their difficulty levels may vary randomly, we further swap the order of the two questions to create a new question pair. As a result, each question appears once in the first turn and once in the second turn, allowing us to eliminate randomness.

Moreover, rather than relying solely on traditional accuracy-based metrics, we focus on evaluating an algorithm's consistency in multi-turn dialogue. Specifically, we are interested in the probability that a question is answered

Method	GQA	MMB_EN	MME	POPE	TextVQA	Avg.
LLaVA-1.5-7B	61.2	62.9	1805	85.5	48.6	100%
FastV	56.1	61.8	1744	79.5	42.8	93.4%
GPTQ	60.9	62.8	1790	85.1	48.3	99.5%
GPTQ + FastV	55.3	60.5	1732	79.2	42.6	92.5%
SQ	61.2	63.7	1781	85.4	48.5	99.9%
SQ + FastV	56.0	61.2	1702	79.6	42.9	92.9%
LLaVA-1.5-13B	62.6	68.3	1854	85.7	52.8	100%
FastV	58.6	66.8	1747	80.7	45.1	93.0%
GPTQ	62.2	67.4	1840	85.8	52.6	99.4%
GPTQ + FastV	58.5	65.6	1749	80.8	44.9	92.6%
SQ	62.6	68.5	1840	85.6	53.0	100.0%
SQ + FastV	58.3	66.1	1757	80.6	44.9	92.8%

Table 10: Results of joint token reduction and quantization on LLaVA-1.5, where SQ denotes SmoothQuant and 192 visual tokens are preserved for FastV.

correctly in the second turn, if it can be answered correctly in the first turn. To compute this, we propose a novel metric that measures this conditional accuracy:

$$P(Q_2^T|Q_1^T) = \frac{P(Q_2^T, Q_1^T)}{P(Q_1^T)} \approx \frac{N(Q_2^T, Q_1^T)}{N(Q_1^T)} \quad (1)$$

where Q_i^T indicates that the question at turn i is answered correctly. P and N denote the probability and corresponding number. We select several representative algorithms and categorize them based on the extent to which they utilize textual information. As we can see in Tab. 9, text-agnostic schemes, such as VisionZip and DivPrune, surpass text-relevant ones (*i.e.*, FastV and SparseVLM). *This suggests that question-dependent approaches may prune visual tokens that are required in subsequent turns, leading to performance degradation in multi-turn dialogue scenarios.* Visualization results can be found in the Appendix.

Finding 2. Token reduction suffers from significant accuracy drops on fine-grained tasks, and prompt-dependent methods yield unsatisfactory results in multi-turn dialogue. Future work should give more attention to the performance in these tasks.

5 Achieving Extreme Compression

Although token reduction can significantly reduce inference latency, it does not lower the peak memory usage during inference. The rationale lies in the fact that model weights dominate memory consumption, which often accounts for over 90%, while token reduction primarily reduces the storage cost of the KV Cache (shown in Fig. 3). To further address the memory bottleneck in VLM inference, we adopt post-training quantization (Xiao et al. 2023; Frantar et al. 2022; Lin et al. 2024), which is more practical during deployment and application.

Following the categorization of quantization methods, we choose GPTQ (Frantar et al. 2022) as a representative algorithm for weight-only quantization (W4A16), and

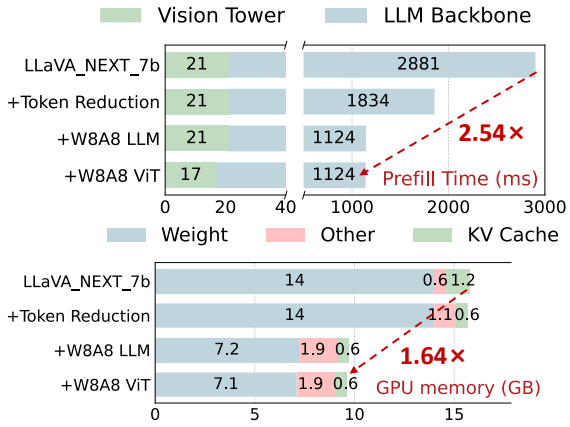

Figure 3: Real inference efficiency on LLaVA-NeXT.

SmoothQuant (Xiao et al. 2023) for weight-activation quantization (W8A8). In Tab. 10, we present detailed results for the joint application of token reduction and quantization. We summarize the following key observations: 1) Using quantization alone (*e.g.*, W8A8 and W4A16) can retain near-lossless accuracy on VLMs. When combined with token reduction, overall performance mainly depends on the effectiveness of the token reduction method. 2) W8A8 consistently achieves better accuracy than W4A16, regardless of whether token reduction is applied.

To further validate the actual speedup and memory savings of quantized VLM, we deploy LLaVA-NeXT-7B on an NVIDIA RTX 4090 GPU. We utilize the int8 kernel from vLLM (Kwon et al. 2023) for "LLM Backbone" and the int8 kernel for "Vision Tower". When the number of visual tokens decreases from 2487 to 1243 (50% is considered a safe pruning ratio (Chen et al. 2024a)), the model achieves a 1.56× speedup, but with negligible change in memory usage. However, when combined with quantization, the model achieves a 2.54× speedup along with a 1.64× reduction in memory consumption. Therefore, our recommended best practice is to combine token reduction with efficient *model-level* compression techniques, such as quantization.

Finding 3. Introducing relatively stable W8A8 or W4A16 quantization into token reduction yields greater compression rates without a significant performance drop.

6 Conclusion

In this study, we introduce LLMC+, a VLM compression benchmark with a versatile toolkit. We address key limitations in prior research and systematically evaluate various compression methods. Our contributions are threefold: 1) analyzing techniques for spatial and temporal redundancy with modular comparisons. 2) assessing compression performance on fine-grained tasks and multi-turn dialogue, highlighting practical limitations. 3) Examining the effectiveness and trade-offs of combining methods such as token reduction and quantization. LLMC+ offers insights and guidance for advancing efficient VLM compression.

Acknowledgements

This work was supported by the Beijing Natural Science Foundation (QY24138), the Fundamental Research Funds for the Central Universities, the Postdoctoral Fellowship Program and China Postdoctoral Science Foundation (No. BX20250487).

References

Alvar, S. R.; Singh, G.; Akbari, M.; and Zhang, Y. 2025. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 9392–9401.

Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Zitnick, C. L.; and Parikh, D. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, 2425–2433.

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; et al. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Bigham, J. P.; Jayant, C.; Ji, H.; Little, G.; Miller, A.; Miller, R. C.; Miller, R.; Tatarowicz, A.; White, B.; White, S.; et al. 2010. Vizwiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, 333–342.

Bolya, D.; Fu, C.-Y.; Dai, X.; Zhang, P.; Feichtenhofer, C.; and Hoffman, J. 2022. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chen, L.; Zhao, H.; Liu, T.; Bai, S.; Lin, J.; Zhou, C.; and Chang, B. 2024a. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, 19–35. Springer.

Chen, Z.; Wu, J.; Wang, W.; Su, W.; Chen, G.; Xing, S.; Zhong, M.; Zhang, Q.; Zhu, X.; Lu, L.; et al. 2024b. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 24185–24198.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Fu, C.; Chen, P.; Shen, Y.; Qin, Y.; Zhang, M.; Lin, X.; Yang, J.; Zheng, X.; Li, K.; Sun, X.; Wu, Y.; and Ji, R. 2023. MME: A Comprehensive Evaluation Benchmark for Multimodal Large Language Models. *arXiv preprint arXiv:2306.13394*.

Fu, C.; Dai, Y.; Luo, Y.; Li, L.; Ren, S.; Zhang, R.; Wang, Z.; Zhou, C.; Shen, Y.; Zhang, M.; et al. 2025. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 24108–24118.

Fu, T.; Liu, T.; Han, Q.; Dai, G.; Yan, S.; Yang, H.; Ning, X.; and Wang, Y. 2024. Framefusion: Combining similarity and importance for video token reduction on large visual language models. *arXiv preprint arXiv:2501.01986*.

Gong, R.; Bai, S.; Wu, S.; Fan, Y.; Wang, Z.; Li, X.; Yang, H.; and Liu, X. 2025a. Past-future scheduler for llm serving under sla guarantees. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, 798–813.

Gong, R.; Ding, Y.; Wang, Z.; Lv, C.; Zheng, X.; Du, J.; Yong, Y.; Gu, S.; Qin, H.; Guo, J.; et al. 2025b. A survey of low-bit large language models: Basics, systems, and algorithms. *Neural Networks*, 107856.

Huang, X.; Zhou, H.; and Han, K. 2024. Prunevid: Visual token pruning for efficient video large language models. *arXiv preprint arXiv:2412.16117*.

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

Kwon, W.; Li, Z.; Zhuang, S.; Sheng, Y.; Zheng, L.; Yu, C. H.; Gonzalez, J. E.; Zhang, H.; and Stoica, I. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Li, F.; Zhang, R.; Zhang, H.; Zhang, Y.; Li, B.; Li, W.; Ma, Z.; and Li, C. 2024a. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. *arXiv preprint arXiv:2407.07895*.

Li, K.; Wang, Y.; He, Y.; Li, Y.; Wang, Y.; Liu, Y.; Wang, Z.; Xu, J.; Chen, G.; Luo, P.; et al. 2024b. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 22195–22206.

Li, Y.; Du, Y.; Zhou, K.; Wang, J.; Zhao, W. X.; and Wen, J.-R. 2023. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*.

Lin, B.; Ye, Y.; Zhu, B.; Cui, J.; Ning, M.; Jin, P.; and Yuan, L. 2023. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Chen, W.-M.; Wang, W.-C.; Xiao, G.; Dang, X.; Gan, C.; and Han, S. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6: 87–100.

Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; et al. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Liu, H.; Li, C.; Li, Y.; and Lee, Y. J. 2024b. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 26296–26306.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual instruction tuning. *Advances in neural information processing systems*, 36: 34892–34916.

Liu, T.; Shi, L.; Hong, R.; Hu, Y.; Yin, Q.; and Zhang, L. 2024c. Multi-stage vision token dropping: Towards efficient multimodal large language model. *arXiv preprint arXiv:2411.10803*.

Liu, Y.; Duan, H.; Zhang, Y.; Li, B.; Zhang, S.; Zhao, W.; Yuan, Y.; Wang, J.; He, C.; Liu, Z.; et al. 2024d. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, 216–233. Springer.

Liu, Y.; Li, Z.; Huang, M.; Yang, B.; Yu, W.; Li, C.; Yin, X.-C.; Liu, C.-L.; Jin, L.; and Bai, X. 2024e. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12): 220102.

Saikh, T.; Ghosal, T.; Mittal, A.; Ekbal, A.; and Bhattacharyya, P. 2022. Scienceqa: A novel resource for question answering on scholarly articles. *International Journal on Digital Libraries*, 23(3): 289–301.

Shao, K.; Tao, K.; Qin, C.; You, H.; Sui, Y.; and Wang, H. 2025. HoliTom: Holistic Token Merging for Fast Video Large Language Models. *arXiv preprint arXiv:2505.21334*.

Shen, L.; Gong, G.; He, T.; Zhang, Y.; Liu, P.; Zhao, S.; and Ding, G. 2025. Fastvid: Dynamic density pruning for fast video large language models. *arXiv preprint arXiv:2503.11187*.

Singh, A.; Natarajan, V.; Shah, M.; Jiang, Y.; Chen, X.; Batra, D.; Parikh, D.; and Rohrbach, M. 2019. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8317–8326.

Sun, M.; Liu, Z.; Bair, A.; and Kolter, J. Z. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Tao, K.; Qin, C.; You, H.; Sui, Y.; and Wang, H. 2025. DyCoke: Dynamic Compression of Tokens for Fast Video Large Language Models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 18992–19001.

Tito, R.; Karatzas, D.; and Valveny, E. 2023. Hierarchical multimodal transformers for multipage docvqa. *Pattern Recognition*, 144: 109834.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Wang, P.; Bai, S.; Tan, S.; Wang, S.; Fan, Z.; Bai, J.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; et al. 2024a. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Wang, X.; Zheng, Y.; Wan, Z.; and Zhang, M. 2024b. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Wen, Z.; Gao, Y.; Wang, S.; Zhang, J.; Zhang, Q.; Li, W.; He, C.; and Zhang, L. 2025. Stop looking for important tokens in multimodal language models: Duplication matters more. *arXiv preprint arXiv:2502.11494*.

Wu, H.; Li, D.; Chen, B.; and Li, J. 2024. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 37: 28828–28857.

Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; and Han, S. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, 38087–38099. PMLR.

Yang, S.; Chen, Y.; Tian, Z.; Wang, C.; Li, J.; Yu, B.; and Jia, J. 2025. Visionzip: Longer is better but not necessary in vision language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 19792–19802.

Zhai, X.; Mustafa, B.; Kolesnikov, A.; and Beyer, L. 2023. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, 11975–11986.

Zhang, K.; Li, B.; Zhang, P.; Pu, F.; Cahyono, J. A.; Hu, K.; Liu, S.; Zhang, Y.; Yang, J.; Li, C.; et al. 2024a. Lmms-eval: Reality check on the evaluation of large multimodal models. *arXiv preprint arXiv:2407.12772*.

Zhang, Q.; Cheng, A.; Lu, M.; Zhang, R.; Zhuo, Z.; Cao, J.; Guo, S.; She, Q.; and Zhang, S. 2024b. Beyond text-visual attention: Exploiting visual cues for effective token pruning in vlms. *arXiv preprint arXiv:2412.01818*.

Zhang, Y.; Fan, C.-K.; Ma, J.; Zheng, W.; Huang, T.; Cheng, K.; Gudovskiy, D.; Okuno, T.; Nakata, Y.; Keutzer, K.; et al. 2024c. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*.

Zheng, L.; Yin, L.; Xie, Z.; Sun, C. L.; Huang, J.; Yu, C. H.; Cao, S.; Kozyrakis, C.; Stoica, I.; Gonzalez, J. E.; et al. 2024. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37: 62557–62583.

Zhou, J.; Shu, Y.; Zhao, B.; Wu, B.; Xiao, S.; Yang, X.; Xiong, Y.; Zhang, B.; Huang, T.; and Liu, Z. 2024. Mlvu: A comprehensive benchmark for multi-task long video understanding. *arXiv e-prints*, arXiv–2406.