Video Compression Commander: Plug-and-Play Inference Acceleration for Video Large Language Models

Xuyang Liu[1,2*] **Yiyu Wang**[1*] **Junpeng Ma**[3] **Linfeng Zhang**[1✉]
[1]Shanghai Jiao Tong University [2]Sichuan University [3]Fudan University

Abstract

Video large language models (VideoLLM) excel at video understanding, but face efficiency challenges due to the quadratic complexity of abundant visual tokens. Our systematic analysis of token compression methods for VideoLLMs reveals two critical issues: **(i)** overlooking distinctive visual signals across frames, leading to information loss; **(ii)** suffering from implementation constraints, causing incompatibility with modern architectures or efficient operators. To address these challenges, we distill three design principles for VideoLLM token compression and propose a plug-and-play inference acceleration framework "**Vid**eo **Com**pression **Com**mander" (**VidCom**^{2}). By quantifying each frame's uniqueness, VidCom^2 adaptively adjusts compression intensity across frames, effectively preserving essential information while reducing redundancy in video sequences. Extensive experiments across various VideoLLMs and benchmarks demonstrate the superior performance and efficiency of our VidCom^2. With only **25%** visual tokens, VidCom^2 achieves **99.6%** of the original performance on LLaVA-OV while reducing **70.8%** of the LLM generation latency. Notably, our Frame Compression Adjustment strategy is compatible with other token compression methods to further improve their performance. Our code is available at `https://github.com/xuyang-liu16/VidCom2`.

1 Introduction

Recently, Video Large Language Models (VideoLLMs) have demonstrated remarkable performance in video understanding and reasoning tasks (Zhang et al., 2023; Wang et al., 2025). However, videos inherently contain multiple consecutive frames, resulting in a significantly higher number of visual tokens compared to images. For instance, LLaVA-OneVision (Li et al., 2024a) processes 32 × 196 visual tokens per video, while LLaVA-Video (Zhang et al., 2024c) handles even more at 64 × 182 visual tokens. This high token count inevitably leads to expensive computation (Liu et al., 2025b), especially for long video understanding (Chen et al., 2024b).

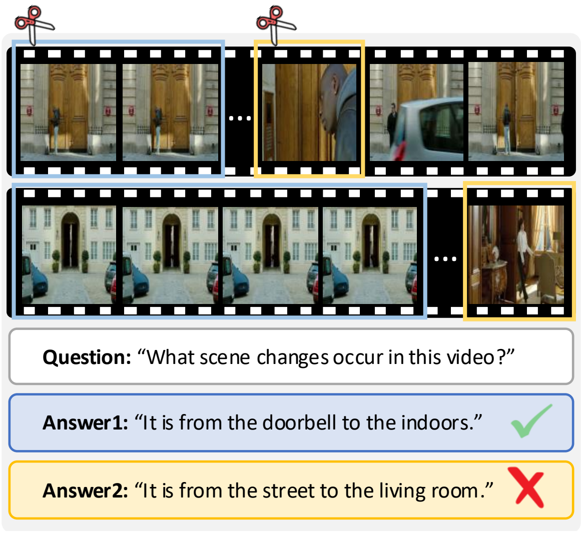
Question: "What scene changes occur in this video?"
Answer1: "It is from the doorbell to the indoors."
Answer2: "It is from the street to the living room."

Figure 1: **Power of frame uniqueness.** Removing **24** redundant frames results in **accurate** video understanding by VideoLLMs, while dropping just **8** unique frames leads to **inaccurate** video comprehension, highlighting the critical role of unique frames for VideoLLMs.

To mitigate this computational burden, researchers have turned to token compression methods (Chen et al., 2024a; Yang et al., 2025), considering the inherent visual redundancy and aiming to minimize redundant visual information. These approaches can be categorized as pre-LLM (Zhang et al., 2024b) or intra-LLM (Chen et al., 2024a) methods, based on whether compression occurs before or within the LLM. Most of these methods are training-free, enabling plug-and-play inference acceleration for existing VideoLLMs. However, despite these efforts, existing token compression methods suffer from ***two critical issues***:

(I) Design Myopia: In human video perception,

[*]Equal contribution. Work done during a visit to the EPIC Lab at Shanghai Jiao Tong University.
✉Corresponding author: zhanglinfeng@sjtu.edu.cn

Methods	Pre-LLM	Intra-LLM	[CLS] Dependency	Video-Specific	Frame Uniqueness	Efficient Attention
FastV		✓				
PDrop		✓				
SparseVLM		✓				
MUSTDrop	✓	✓	✓			
FiCoCo	✓	✓	✓			
FasterVLM	✓		✓			✓
DyCoke	✓			✓		✓
VidCom^2	✓			✓	✓	✓

Table 1: **Feature comparison with existing training-free token compression methods.** Most suffer from design myopia and implementation constraints.

we naturally focus on distinctive frames (*e.g.*, those with significant spatio-temporal changes) while ignoring repetitive and redundant visual information (Ma et al., 2025). By contrast, most existing token compression methods apply a *uniform* compression strategy across all frames, treating each one as equally informative. Even recent VideoLLM-specific method DyCoke (Tao et al., 2025) exhibits this limitation by grouping every four consecutive frames into a fixed window and compressing them identically, without regard for the varying distinctiveness of individual frames. Figure 1 further illustrates the critical nature of this issue: removing 24 redundant frames does not affect the accurate response of the LLaVA-OneVision, whereas dropping just 8 unique frames causes it to fail, despite being only a **third** of the number. This contrast shows that uniform compression risks discarding critical information in unique frames that VideoLLMs may rely on, thereby significantly impacting overall performance. Notably, Table 2 indicates that some methods even **underperform** random token dropping, further indicating their sub-optimal performance.

(II) Implementation Constraints: Beyond design limitations, existing methods face practical constraints. Some token compression works (Zhang et al., 2024b; Liu et al., 2024) rely on [CLS] attention weights in ViT for informative token preservation, yet modern VideoLLMs adopt SigLIP (Zhai et al., 2023) as visual encoder without [CLS] token. Meanwhile, certain methods (Zhang et al., 2025; Xing et al., 2025) aim to leverage textual information but require explicit attention weights in specific LLM layers, making them incompatible with efficient attention operators (Dao et al., 2022). This incompatibility leads to higher peak memory usage, even **surpassing** that of uncompressed processing (see Table 4), which is especially problematic for long video understanding (Wen et al., 2025a,b).

We summarize existing works in Table 1 and identify ***three key principles*** for designing effective and efficient token compression methods for VideoLLM: **(i) Model Adaptability:** The method should be easily compatible with and adaptable to the majority of existing VideoLLMs (Zhang et al., 2024c; Wang et al., 2024); **(ii) Frame Uniqueness:** The method should consider varying distinctiveness across video frames; **(iii) Operator Compatibility:** The method should maintain compatibility with efficient operators (Dao, 2024).

Based on above analysis, we propose "**Vid**eo **Com**pression **Com**mander" (*i.e.*, **VidCom^2**), an efficient plug-and-play token compression method for VideoLLMs from the perspective of frame uniqueness. Our VidCom^2 follows a principled two-stage approach: first adjusting frame-wise compression intensity based on each frame's uniqueness in the video sequence, then performing token compression by evaluating token distinctiveness both within individual frames and across the entire video. Through this careful design, VidCom^2 mimics human video perception by adaptively adjusting attention to different frames (see Figure 3), preserving information from key frames while minimizing redundant visual content.

In summary, our contributions are three-fold:

- **Empirical Method Analysis:** We critically analyze existing token compression methods, unveiling their inherent limitations and delineating three key design principles for effective and efficient VideoLLM token compression.

- **Video Compression Commander:** We are the first to propose a VideoLLM token compression framework based on frame uniqueness, offering a plug-and-play method with frame-wise dynamic compression.

- **Outstanding Performance & Efficiency:** Extensive experiments on diverse benchmarks demonstrate superior efficiency-performance trade-offs. With 15% tokens, VidCom^2 outperforms the second-best method by **3.9%** and **2.2%** on LLaVA-OV and LLaVA-Video.

2 Related Work

2.1 Video Large Language Models

Large vision-language models (LVLMs) combine vision encoders with LLMs for exceptional visual understanding (Li et al., 2024a; Wang et al., 2024). While LVLMs can handle basic video tasks, the growing demand has led to specialized video large language models (VideoLLMs) (Zhang et al., 2024c, 2023). These VideoLLMs enhance video understanding through extensive datasets and targeted training strategies, as demonstrated by LLaVA-OneVision (Li et al., 2024a) for multi-modal tasks and LLaVA-Video (Zhang et al., 2024c) for video instruction-following. However, the long sequences of visual tokens from continuous video frames limit their practical applications.

2.2 Token Compression for LVLMs

Recently, with the increase in visual tokens in LVLMs, research has shifted from *training-aware* (Li et al., 2024c) to *training-free* token compression methods (Yang et al., 2025). Training-free approaches are generally categorized as: **(a)** Pre-LLM token compression at the ViT or projector level (Zhang et al., 2024b; Liu et al., 2025a); **(b)** Intra-LLM token compression within the LLM decoder (Chen et al., 2024a; Zhang et al., 2025; Chen et al., 2025); and **(c)** Hybrid token compression that compresses tokens at both ViT and LLM (Han et al., 2024). However, these methods treat video frames as separate images, overlooking temporal relationships. While recent work DyCoke (Tao et al., 2025) introduces temporal token merging across consecutive frame windows, it cannot achieve retention ratios below 25%. More importantly, existing methods, including DyCoke, adopt uniform compression across frames without considering frame uniqueness, and many face compatibility issues with efficient operators (Dao et al., 2022).

In this work, we propose a plug-and-play efficient token compression strategy that leverages frame-specific features to tackle current challenges in efficient VideoLLM inference.

3 Methodology

3.1 Preliminary

VideoLLM Architecture. Most current VideoLLMs follow the "ViT-MLP-LLM" paradigm (Li et al., 2024a; Zhang et al., 2024c). For example, in LLaVA-Video, a video sequence $\mathbf{V} = \{\mathbf{v}_t\}_{t=1}^{T} \in \mathbb{R}^{T\times H\times W\times 3}$ is first encoded by ViT into embeddings $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^{T} \in \mathbb{R}^{T\times N\times D}$. These embeddings are projected by a 2-layer MLP and pooled to produce visual tokens $\mathbf{X}^v = \{\mathbf{x}_t^v\}_{t=1}^{T} \in \mathbb{R}^{T\times M\times D'}$, with $M < N$, which are then fed into the LLM for autoregressive instruction-following:

$$p\left(\mathbf{Y} \mid \mathbf{X}^v, \mathbf{X}^t\right) = \prod_{i=1}^{L} p\left(\mathbf{y}_i \mid \mathbf{X}^v, \mathbf{X}^t, \mathbf{Y}_{1:i-1}\right), \tag{1}$$

where $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^{L}$ are the generated response tokens, and $\mathbf{X}^t$ are the textual tokens.

Token Compression for VideoLLMs. Token compression aims to reduce data redundancy by directly compressing token representations for inference acceleration. For VideoLLMs, this typically involves compressing visual token sequences $\mathbf{X}_t^v$ into a reduced representation $\hat{\mathbf{X}}^v$:

$$\hat{\mathbf{X}}^v = \mathbf{\Phi}(\mathbf{X}^v), \quad \text{where} \quad |\hat{\mathbf{X}}^v| < |\mathbf{X}^v| \tag{2}$$

where $\mathbf{\Phi}$ represents the token compression operator and $|\cdot|$ denotes the token length.

Token compression is particularly crucial for VideoLLMs due to their processing of substantially more visual tokens compared to standard LVLMs, a result of the multi-frame nature of videos. Consecutive frames often share high similarity, leading to significant visual redundancy. While recent method DyCoke (Tao et al., 2025) address some aspects of multi-frame redundancy, it struggles with uneven frame distinctiveness and achieving aggressive compression rates. Our work focuses on designing an effective token compression operator $\mathbf{\Phi}$ that adaptively handles frame-wise distinctiveness while enabling flexible compression rates, addressing these key challenges for VideoLLMs.

3.2 Video Compression Commander

To improve the computational efficiency of VideoLLMs, we propose "**Vid**eo **Com**pression **Com**mander" (**VidCom^2**), a novel token compression framework that adaptively minimizes visual redundancy within a predefined token budget while preserving distinctive visual information. VidCom^2 maintains compatibility with efficient attention operators (Dao et al., 2022; Dao, 2024) and supports flexible compression rates, enabling plug-and-play inference acceleration.

Figure 2 illustrates the overall framework of VidCom^2, which achieves efficient token compression for VideoLLMs through a methodical ***two-stage*** framework: **(i) Frame Compression Adjustment**, which evaluates frame uniqueness within the video sequence and dynamically allocates optimal token budgets through compression intensity adjustment; and **(ii) Adaptive Token Compression**,

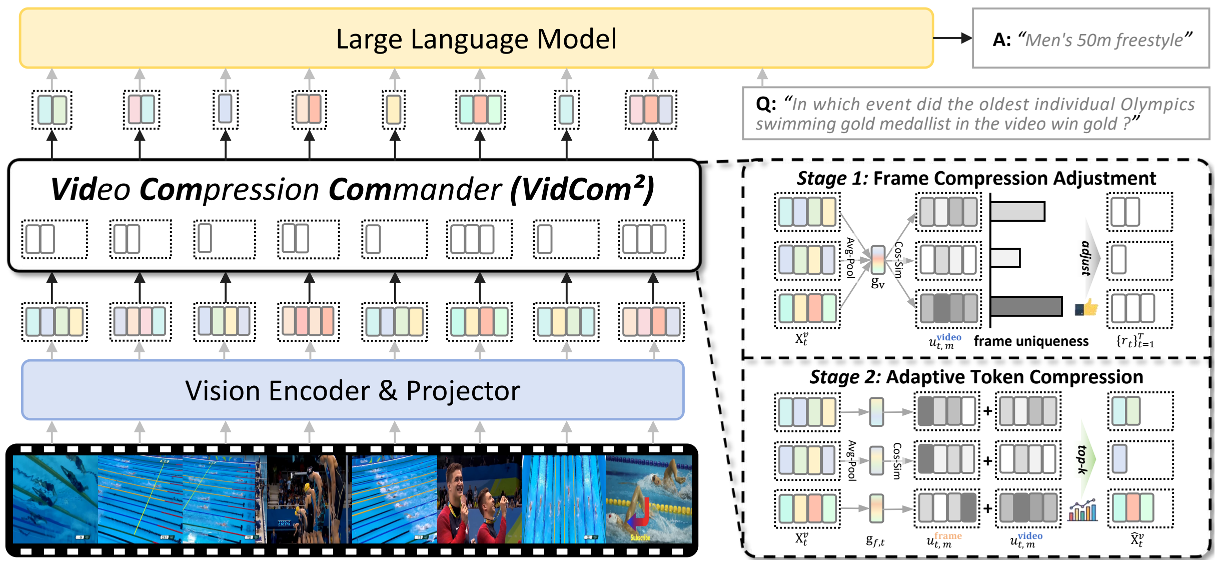
Large Language Model
A: "Men's 50m freestyle"
Q: "In which event did the oldest individual Olympics swimming gold medallist in the video win gold ?"
Video Compression Commander (VidCom²)
Vision Encoder & Projector
Stage 1: Frame Compression Adjustment
Avg-Pool
Cos-Sim
g_v
adjust
X_t^v
$u_{t,m}^{\text{video}}$
frame uniqueness
$\{r_t\}_{t=1}^T$
Stage 2: Adaptive Token Compression
Avg-Pool
Cos-Sim
top-k
X_t^v
$g_{f,t}$
$u_{t,m}^{\text{frame}}$
$u_{t,m}^{\text{video}}$
$\hat{X}_t^v$

Figure 2: **Overall framework of VidCom^2.** Our VidCom^2 performs plug-and-play token compression in two stages: **(i) Frame Compression Adjustment**: adjusts compression intensity based on frame uniqueness (see Figure 3), **(ii) Adaptive Token Compression**: preserves tokens based on their within-frame and cross-video uniqueness.

which assesses token distinctiveness both within-frame and across-video, strategically performing compression based on the frame-specific budgets from the previous stage. Below, we elaborate on the detailed operations of these two stages.

3.3 Stage 1: Frame Compression Adjustment

The core of this stage is to adaptively adjust compression intensity based on frame uniqueness across the video. A natural question arises: ***How can a frame's uniqueness be quantified within the video context?***. Since each frame $\mathbf{x}_t^v \in \mathbb{R}^{M\times D'}$ consists of M visual tokens, we define frame uniqueness through the collective distinctiveness of its constituent tokens.

Specifically, we first obtain a global video representation $\mathbf{g_v}$ by average pooling all tokens across T frames, each with M tokens:

$$\mathbf{g_v} = \frac{1}{T\cdot M}\sum_{t=1}^{T}\sum_{m=1}^{M}\mathbf{x}_{t,m}^v, \quad \mathbf{g_v}\in\mathbb{R}^{D'}, \tag{3}$$

where $\mathbf{g_v}$ serves as a coarse-grained summary of the entire video. Then, inspired by existing efforts (Sun et al., 2025), we compute the similarity between each token $\mathbf{x}_{t,m}^v$ and global video representation $\mathbf{g_v}$ in high-dimensional space:

$$s_{t,m}^{\text{video}} = \frac{\mathbf{x}_{t,m}^v\cdot\mathbf{g_v}}{\|\mathbf{x}_{t,m}^v\|\,\|\mathbf{g_v}\|}, \quad s_{t,m}^{\text{video}}\in[-1,1], \tag{4}$$

where a lower $s_{t,m}^{\text{video}}$ implies that token $\mathbf{x}_{t,m}^v$ is less redundant (more unique) relative to the full video. We define the *video-level uniqueness score* of token $\mathbf{x}_{t,m}^v$ as $u_{t,m}^{\text{video}} = -s_{t,m}^{\text{video}}$ and compute the *frame uniqueness score* $u_t = \frac{1}{M}\sum_{m=1}^{M}u_{t,m}^{\text{video}}$, where a larger u_t indicates higher density of distinctive tokens in frame t compared to the rest of the video. Figure 3 demonstrates how u_t effectively quantifies frame-wise uniqueness density within video sequences. More cases are in Appendix F.

These frame-wise scores $\{u_t\}_{t=1}^T$ are used to modulate per-frame compression intensity. To stabilize the scores, we compute $\tilde{u}_t = (u_t - \max(u_t))/\tau$ ($\tau = 0.01$), and obtain the relative importance weight σ_t of each frame via softmax:

$$\sigma_t = \frac{\exp(\tilde{u}_t)}{\sum_{l=1}^{T}\exp(\tilde{u}_l)+\epsilon}, \tag{5}$$

where $\epsilon = 10^{-8}$ prevents division by zero. Based on these weights, we adjust the preset retention ratio $R(\%)$ for each frame:

$$r_t = R\times\left(1+\sigma_t-\frac{1}{T}\right), \tag{6}$$

where $\sigma_t - \frac{1}{T}$ represents the relative deviation from average importance. Consequently, VidCom^2 adaptively adjusts compression intensity (*i.e.*, $\{r_t\}_{t=1}^T$) based on frame uniqueness, enabling differentiated token compression degrees across frames while maintaining the average retention ratio R.

3.4 Stage 2: Adaptive Token Compression

The core of this stage lies in how to select and retain more unique visual information based on the compression degrees $\{r_t\}_{t=1}^T$ determined in the previous stage. Since visual information is composed of tokens, this problem naturally transforms

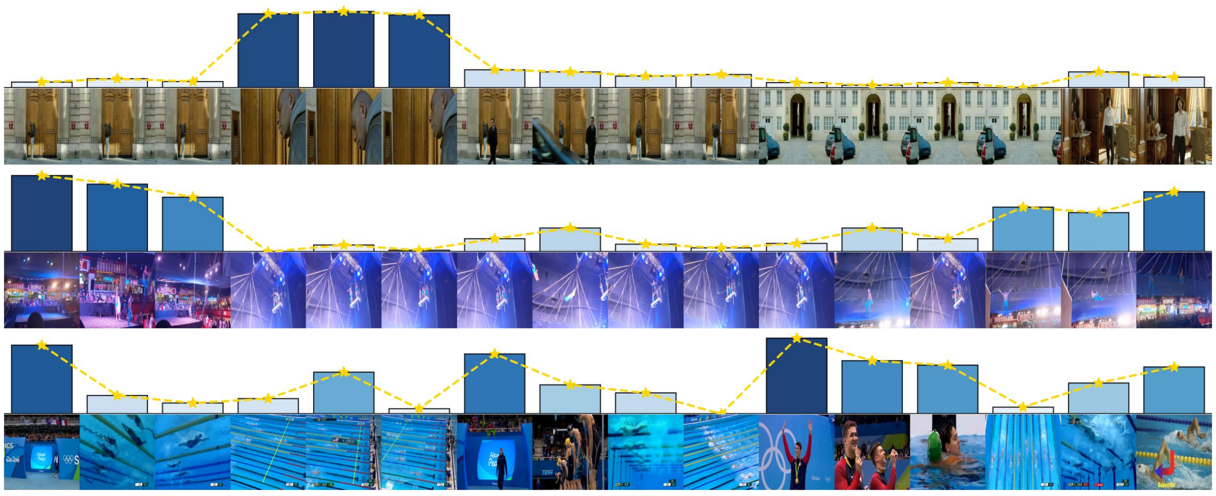
Figure 3: **Visualization of frame uniqueness quantified by our VidCom^2.** Taller and darker bars indicate frame uniqueness, where VidCom^2 allocates more tokens to unique frames to preserve critical visual information.

into: ***How can a token's uniqueness be quantified within the video context?***. Given the multi-frame nature of videos, a token's uniqueness could be evaluated both locally and globally, *i.e.*, within its frame and across the entire video sequence.

As for token uniqueness within its frame, we can quantify it by measuring its relationship with the frame's global representation. Specifically, for the t-th frame, we obtain its global representation through average pooling:

$$\mathbf{g}_{f,t} = \frac{1}{M}\sum_{m=1}^{M} \mathbf{x}_{t,m}^{v}, \quad \mathbf{g}_{f,t} \in \mathbb{R}^{D'}, \tag{7}$$

By computing the cosine similarity of the m-th token within the t-th frame with its frame-level global representation $\mathbf{g}_{f,t}$, we first define:

$$s_{t,m}^{\text{frame}} = \frac{\mathbf{x}_{t,m}^{v} \cdot \mathbf{g}_{f,t}}{\|\mathbf{x}_{t,m}^{v}\| \, \|\mathbf{g}_{f,t}\|}, \quad s_{t,m} \in [-1, 1], \tag{8}$$

We then define the *frame-level uniqueness score* as $u_{t,m}^{\text{frame}} = -s_{t,m}^{\text{frame}}$, where higher values indicate greater token uniqueness within the frame.

Moreover, since we have already obtained the *video-level uniqueness score* $u_{t,m}^{\text{video}} = -s_{t,m}^{\text{video}}$ of token $\mathbf{x}_{t,m}^{v}$ in the previous stage, we combine these two uniqueness scores to derive *comprehensive uniqueness score* of token $\mathbf{x}_{t,m}^{v}$ by:

$$u_{t,m} = u_{t,m}^{\text{frame}} + u_{t,m}^{\text{video}}, \tag{9}$$

which provides a balanced assessment of the token's distinctiveness both within its frame and across the entire video.

Given the adjusted compression intensity (*i.e.*, $\{r_t\}_{t=1}^{T}$) based on frame uniqueness in the previous stage, the token compression process for the t-th frame can be formulated as:

$$\mathbf{X}_t^v \to \hat{\mathbf{X}}_t^v = \text{TopK}(\mathbf{X}_t^v, \{u_{t,m}\}_{m=1}^{M}, r_t \times M) \tag{10}$$

where $\hat{\mathbf{X}}_t^v$ represents the compressed token sequence for the t-th frame, $\{u_{t,m}\}_{m=1}^{M}$ are the *comprehensive uniqueness scores* of each token in $\mathbf{X}_t^v$, and r_t is the frame-specific retention ratio.

To this end, our VidCom^2 adaptively adjusts the compression intensity based on frame uniqueness, selectively retaining tokens that are distinctive both within their frames and across the entire video, thereby minimizing information redundancy. The complete algorithm is detailed in Appendix E.

4 Experiments

4.1 Experimental Setting

Benchmark. We conduct comprehensive comparative experiments across multiple benchmarks, including: MVBench (Li et al., 2024b), LongVideoBench (Wu et al., 2024), MLVU (Zhou et al., 2024), VideoMME (Fu et al., 2024), EgoSchema (Mangalam et al., 2023), and PerceptionTest (Patraucean et al., 2023), employing LMMs-Eval (Zhang et al., 2024a) evaluation framework. More details are in Appendix A.

Implementations. We evaluate our method on popular VideoLLMs: LLaVA-OneVision (LLaVA-OV) (Li et al., 2024a), LLaVA-Video (Zhang et al., 2024c), and Qwen2-VL (Wang et al., 2024). Detailed model information is in Appendix B. All experiments use NVIDIA A100-SXM4-80GB GPUs.

Baselines. We evaluate our method against various training-free token compression strategies, including: FastV (Chen et al., 2024a), PDrop (Xing et al., 2025), SparseVLM (Zhang et al., 2025), and Dy-

Methods	MVBench	LongVideoBench	MLVU	VideoMME Overall	VideoMME Short	VideoMME Medium	VideoMME Long	Average (%)
Upper Bound								
LLaVA-OV-7B	56.9	56.4	63.0	58.6	70.3	56.6	48.8	100.0
Retention Ratio=30%								
DyCoke [CVPR'25]	56.6	54.7	60.3	56.1	67.1	54.6	46.6	96.5
Retention Ratio=25%								
Random	54.2	52.7	59.7	55.6	65.4	53.0	48.3	94.8
FastV [ECCV'24]	55.5	53.3	59.6	55.3	65.0	53.8	47.0	94.9
PDrop [CVPR'25]	55.3	51.3	57.1	55.5	64.7	53.1	48.7	94.1
SparseVLM [ICML'25]	56.4	53.9	60.7	57.3	68.4	55.2	48.1	97.5
DyCoke [CVPR'25]	49.5	48.1	55.8	51.0	61.1	48.6	43.2	87.0
VidCom^2	**57.2**	**54.9**	**62.5**	**58.6**	**69.8**	**56.4**	**49.4**	**99.6**
Retention Ratio=15%								
FastV [ECCV'24]	51.6	48.3	55.0	48.1	51.4	49.4	43.3	85.0
PDrop [CVPR'25]	53.2	47.6	54.7	50.1	58.7	48.7	45.0	87.4
SparseVLM [ICML'25]	52.9	49.7	57.4	53.4	61.0	52.1	47.0	91.2
VidCom^2	**54.3**	**52.0**	**58.9**	**56.2**	**65.8**	**54.8**	**48.1**	**95.1**
Upper Bound								
LLaVA-Video-7B	60.4	59.6	70.3	64.3	77.2	62.1	53.4	100.0
Retention Ratio=30%								
DyCoke [CVPR'25]	57.5	55.5	60.6	61.3	73.4	59.3	51.2	93.8
Retention Ratio=25%								
FastV [ECCV'24]	53.8	51.2	57.8	59.3	67.1	60.0	**50.8**	89.7
SparseVLM [ICML'25]	55.4	54.2	58.9	60.1	71.1	59.1	50.1	91.6
DyCoke [CVPR'25]	50.8	53.0	56.9	56.1	65.8	53.6	48.9	86.3
VidCom^2	**57.0**	**55.5**	**59.0**	**61.7**	**73.0**	**61.7**	50.0	**93.6**
Retention Ratio=15%								
FastV [ECCV'24]	44.0	44.6	53.8	51.3	56.4	51.1	46.2	78.0
SparseVLM [ICML'25]	53.1	**52.7**	56.2	55.7	65.0	53.9	48.3	86.3
VidCom^2	**53.3**	51.5	**56.8**	**58.3**	**68.0**	**57.3**	**49.7**	**88.5**

Table 2: **Performance comparison with other baselines with LLaVA-OV-7B and LLaVA-Video-7B across different benchmarks.** "Average" shows the mean performance across different benchmarks. DyCoke requires pruning similar tokens from consecutive 4 frames, making it not possible for the retention ratio of $R < 25\%$.

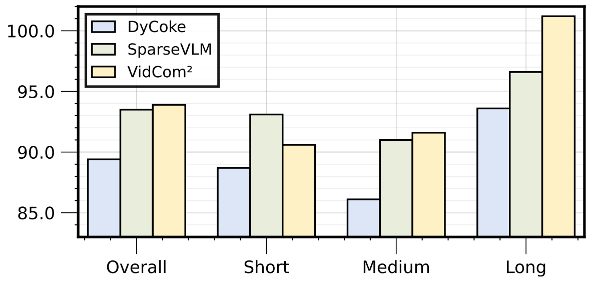

Figure 4: **Performance with Qwen2-VL.** At $R = 25\%$, VidCom^2 surpasses DyCoke and SparseVLM by **7.6%** and **4.6%** of original performance in long video tasks.

Methods	EgoSchema	PerceptionTest
Upper Bound		
LLaVA-OV-7B	60.4 (100%)	57.1 (100%)
Retention Ratio=25%		
FastV [ECCV'24]	57.5 (95.2%)	55.4 (97.0%)
PDrop [CVPR'25]	58.0 (96.0%)	55.6 (97.4%)
DyCoke [CVPR'25]	59.5 (98.5%)	56.4 (98.8%)
VidCom^2	**59.7 (98.8%)**	**56.7 (99.3%)**

Table 3: **Performance comparison on EgoSchema and PerceptionTest.** Percentages represent ratios to the original performance of LLaVA-OV-7B.

Coke (Tao et al., 2025), more introduction can be seen in Appendix C. Following SparseVLM, we use the "equivalent retention ratio"[1] for fair comparisons. Unlike others, DyCoke compresses both visual tokens and KV cache. For fair comparison, we evaluate only on its token compression strategy.

4.2 Main Comparisons

Performance Comparisons. Table 2 presents a comparative analysis of our VidCom^2 against multiple token compression methods across various benchmarks. The experimental results reveal ***two key performance advantages*** of VidCom^2:

(i) State-of-the-art Performance: VidCom^2 demonstrates exceptional performance across diverse video understanding benchmarks. On LLaVA-OV and LLaVA-Video with compression ratio $R = 25\%$, VidCom^2 substantially outperforms DyCoke by margins of **12.6%** and **7.3%**, respectively. Remarkably, VidCom^2 at $R = 25\%$ (achieving 99.6% performance retention) even surpasses DyCoke operating at a higher compression ratio of $R = 30\%$ (96.5% performance retention). This superiority extends to long-form video understanding tasks with Qwen2-VL (Figure 4), where VidCom^2 achieves **101.2%** performance on VideoMME (Long), surpassing both Dy-

[1]"Equivalent retention ratio" represents the average percentage of visual tokens retained across all LLM layers.

Methods	LLM Generation↓ Latency (s)	Model Generation↓ Latency (s)	Total↓ Latency (min:sec)	GPU Peak↓ Memory (GB)	Throughput↑ (samples/s)	Performance↑
LLaVA-OV-7B	618.0	1008.4	26:03	17.7	0.64	56.9
Retention Ratio=25%						
Random	178.2 (↓71.2%)	566.0 (↓43.9%)	18:44 (↓28.1%)	16.0 (↓9.6%)	0.89 (1.39×)	54.6 (↓2.3)
FastV [ECCV'24]	260.9 (↓57.8%)	648.6 (↓35.7%)	20:07 (↓22.8%)	24.7 (↑39.5%)	0.83 (1.30×)	55.5 (↓1.4)
PDrop [CVPR'25]	205.6 (↓66.7%)	592.6 (↓41.2%)	18:50 (↓27.7%)	24.5 (↑38.4%)	0.88 (1.38×)	55.3 (↓1.6)
SparseVLM [ICML'25]	410.6 (↓33.6%)	807.7 (↓19.9%)	25:03 (↓3.8%)	27.1 (↑53.1%)	0.67 (1.05×)	56.4 (↓0.5)
DyCoke [CVPR'25]	205.2 (↓66.8%)	598.0 (↓40.7%)	18:56 (↓27.4%)	16.1 (↓9.0%)	0.88 (1.38×)	49.5 (↓7.4)
VidCom^2	**180.7 (↓70.8%)**	**574.7 (↓43.0%)**	**18:46 (↓28.0%)**	**16.0 (↓9.6%)**	**0.88 (1.38×)**	**57.2 (↑0.3)**
Retention Ratio=15%						
Random	130.3 (↓78.9%)	532.5 (↓47.2%)	18:02 (↓30.8%)	15.8 (↓10.7%)	0.92 (1.44×)	53.1 (↓3.8)
FastV [ECCV'24]	172.4 (↓72.1%)	599.3 (↓40.6%)	18:19 (↓29.7%)	24.6 (↑39.0%)	0.91 (1.42×)	51.6 (↓5.3)
PDrop [CVPR'25]	165.3 (↓73.3%)	552.6 (↓45.2%)	18:32 (↓28.9%)	24.5 (↑38.4%)	0.90 (1.41×)	53.2 (↓3.7)
SparseVLM [ICML'25]	370.4 (↓40.1%)	764.8 (↓24.2%)	24:09 (↓7.3%)	27.1 (↑53.1%)	0.69 (1.08×)	52.9 (↓4.0)
VidCom^2	**129.2 (↓79.1%)**	**533.0 (↓47.1%)**	**18:11 (↓30.2%)**	**15.8 (↓10.7%)**	**0.92 (1.44×)**	**54.3 (↓2.6)**

Table 4: **Efficiency comparisons on LLaVA-OV-7B.** "LLM Generation Latency": time for LLM-only response generation; "Model Generation Latency": time for model to generate response; "Total Latency": total time to complete MVBench; and "Throughput": number of MVBench samples processed per second.

Coke (93.6%) and SparseVLM (96.6%) by substantial margins of **7.6%** and **4.6%**, respectively. Additional comparisons in Table 3 further validate the superior performance advantages of VidCom^2 across various video understanding scenarios.

(ii) Robustness in Extreme Compression: Under aggressive compression with $R = 15\%$, most baselines such as FastV and PDrop exhibit significant performance degradation. Even the VideoLLM-specific method DyCoke **fails to achieve** such aggressive compression due to inherent design limitations. However, VidCom^2 maintains robust performance, outperforming the second-best method SparseVLM by an average of **3.9%** and **2.1%** on LLaVA-OV and LLaVA-Video. This demonstrates VidCom^2's superiority in frame-adaptive compression, dynamically adjusting intensity to preserve distinctive visual information.

Besides, we observe an interesting phenomenon that Intra-LLM methods (*e.g.*, SparseVLM), which incorporate textual information, perform relatively better on long video tasks (*e.g.*, LongVideoBench and VideoMME (long)) compared to shorter video benchmarks like MVBench and VideoMME (Short). For instance, SparseVLM slightly outperforms VidCom^2 on LongVideoBench with LLaVA-Video at $R = 15\%$. This suggests that for longer videos with fixed frame counts, leveraging textual information for visual token compression helps VideoLLMs focus on text-relevant visual areas, potentially leading to improved performance.

Efficiency Comparisons. Beyond performance, Table 4 presents comprehensive real-world inference efficiency comparisons among different token compression methods on MVBench, with all experiments conducted on four NVIDIA A100 GPUs. We follow the original implementation of each baseline method, and unless otherwise specified, Flash Attention 2 (Dao, 2024) is used as the efficient attention operator throughout comparisons. The comparison results in Table 4 reveal ***two key efficiency advantages*** of our VidCom^2:

(i) State-of-the-art Efficiency: VidCom^2 achieves remarkable inference efficiency, **comparable to** simple random token dropping. With 25% visual tokens retained, the additional computation of VidCom^2 is **negligible** – only 2.5s extra (**1.3%** of LLM generation time) for the entire MVBench inference. Despite this minimal overhead, VidCom^2 significantly reduces both the LLM generation latency and overall model latency (primarily from ViT and LLM) by **70.8%** and **43.0%** respectively, achieving **1.38**× throughput while maintaining **99.6%** average performance across benchmarks. These results highlight the efficiency of VidCom^2 in accelerating inference for VideoLLMs.

(ii) Efficient Operator Compatibility: Pre-LLM methods like DyCoke and our VidCom^2 maintain Flash Attention compatibility while continuously reducing peak memory usage, showcasing their efficiency. When equipped with Flash Attention, both VidCom^2 and random dropping further reduce peak memory usage by approximately 2 GB compared to standard Flash Attention, demonstrating that VidCom^2's computation introduces no additional memory overhead. In contrast, Intra-LLM methods (*e.g.*, PDrop and FastV) even substantially **increase** memory consumption. For instance, FastV increases the original peak memory by significantly **39.5%**. This dramatic increase stems from their reliance on **explicit** attention weights, rendering them incompatible with Flash Attention in certain layers.

Metrics	MLVU	VideoMME				Avg.
		Overall	Short	Medium	Long	
Vanilla	63.0	58.6	70.3	56.6	48.8	100.0
$s_{t,m}^{\text{frame}}$	59.5	54.0	62.2	54.2	45.3	94.1
$-s_{t,m}^{\text{frame}}$	61.9	57.9	68.8	56.9	48.1	98.8
$s_{t,m}^{\text{video}}$	58.9	53.3	61.7	52.1	46.1	93.2
$-s_{t,m}^{\text{video}}$	61.4	58.3	69.3	56.1	49.3	99.3
$u_{t,m}^{\text{frame}} + u_{t,m}^{\text{video}}$	**62.1**	**58.5**	**69.6**	**56.3**	**49.3**	**99.7**

Table 5: **Effects of different token evaluation metrics.** The first two parts explores the optimal $u_{t,m}^{\text{frame}}$ and $u_{t,m}^{\text{video}}$, while the last part examines the optimal $u_{t,m}$.

Given the large number of frames and tokens in video sequences, such memory-intensive methods show limited practical value for VideoLLMs.

4.3 Ablation Study and Analysis

We conduct multiple ablation studies and analyses with $R = 25\%$ on LLaVA-OV-7B, exploring optimal token evaluation strategies and validating the effectiveness of Frame Compression Adjustment for both VidCom^2 and other methods.

Effects of Different Token Evaluation Metrics. Table 5 presents various metrics for token evaluation, consisting of three parts: **(a)** frame-level uniqueness score $u_{t,m}^{\text{frame}}$, **(b)** video-level uniqueness score $u_{t,m}^{\text{video}}$, and **(c)** the final score $u_{t,m}$ that combines $u_{t,m}^{\text{frame}}$ and $u_{t,m}^{\text{video}}$ to guide our token preservation strategy.

For frame-level uniqueness, defining $u_{t,m}^{\text{frame}}$ as the negative similarity to frame-level global representation ($-s_{t,m}^{\text{frame}}$) outperforms positive similarity. Similarly, for video-level uniqueness, tokens less similar to the video-level global representation prove more informative. These results indicate that unique tokens, both within frames and across the video, should be prioritized during token compression to preserve richer visual information.

Token compression guided by either frame-level or video-level uniqueness scores outperforms the baselines in Table 2, showcasing the effectiveness of uniqueness-based selection. Their combination further achieves optimal performance, suggesting that token uniqueness should be evaluated both **within-frame** and **across-video** to maximize visual content preservation during token compression.

Effects of Frame Compression Adjustment. Table 6 compares different compression adjustment strategies: **(a)** "Uniform" with fixed $R = 25\%$ (no adjustment); **(b)** "$\max_m u_{t,m}^{\text{video}}$" and **(c)** "$\overline{u_{t,m}^{\text{video}}}$", which compute frame uniqueness score u_t for token budget allocation using maximum and average operations of $u_{t,m}^{\text{video}}$ in frame t, where larger u_t leads to more tokens preserved in frame t.

Metrics	MLVU	VideoMME				Avg.
		Overall	Short	Medium	Long	
Vanilla	63.0	58.6	70.3	56.6	48.8	100.0
Uniform	61.9	57.9	68.8	**56.9**	48.1	98.8
Frame Compression Adjustment						
$\max_m u_{t,m}^{\text{video}}$	62.1	58.1	68.4	56.7	49.3	99.4
$\overline{u_{t,m}^{\text{video}}}$	**62.3**	**58.2**	**69.1**	55.9	**49.6**	**99.6**

Table 6: **Effects of different compression adjustment.** "Uniform": fixed $R = 25\%$. "$\max u_{t,m}^{\text{video}}$" and "$\overline{u_{t,m}^{\text{video}}}$" denote frame uniqueness score u_t of frame t computed by maximum and average operations of $u_{t,m}^{\text{video}}$.

Size	MVBench	VideoMME				Avg.
		Overall	Short	Medium	Long	
Vanilla	56.9	58.6	70.3	56.6	48.8	100.0
4	56.8	57.9	69.6	55.6	48.7	99.1
8	56.8	58.3	69.8	56.4	48.6	99.6
16	57.2	58.5	**70.0**	**56.7**	48.9	100.1
32	**57.2**	**58.6**	69.8	56.4	**49.4**	**100.1**

Table 7: **Effects of different window sizes for local g_v computation.** Window sizes up to 32 (global perspective) are evaluated on LLaVA-OV-7B.

Generally, Frame Compression Adjustment strategies demonstrate performance improvements over uniform compression, validating the effectiveness of dynamically adjusting compression intensity based on frame uniqueness. This confirms our intuition that allocating more token budget to distinctive frames helps preserve important visual information along the temporal dimension. Moreover, averaging token uniqueness ($\overline{u_{t,m}^{\text{video}}}$) outperforms maximum operation ($\max_m u_{t,m}^{\text{video}}$), as it better captures the overall **uniqueness density** of a frame rather than focusing on isolated distinctive features, providing a more comprehensive measure of frame-level temporal uniqueness.

Effects of Different Window Sizes for Local g_v Computation We explore sliding window strategies for computing local g_v representations to investigate the effectiveness of adjusting frame compression intensity from local perspectives. We evaluate different window sizes (4, 8, 16, 32) on LLaVA-OV-7B with fixed 32 frames.

As shown in Table 7, performance consistently improves as window size increases across both MVBench and VideoMME. Notably, when window sizes reach 16 and 32, the performance gap becomes marginal. Window size 16 achieves better results on VideoMME short and medium videos, while window size 32 (global perspective) demonstrates superior performance on VideoMME long

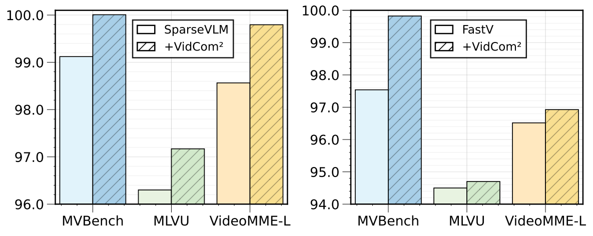

Figure 5: **Effects of Frame Compression Adjustment on other methods.**"+VidCom^2" indicates the application of our Frame Compression Adjustment strategy.

videos. Therefore, we adopt the global perspective for adjusting compression intensity by default to achieve better long video understanding.

Broad Applicability of Frame Compression Adjustment. Figure 5 further demonstrates the effectiveness of integrating our Frame Compression Adjustment strategy with other methods.

Results show consistent performance improvements compared to their original performance on LLaVA-OV-7B across short (MVBench) and long (MVLU and VideoMME-L) video understanding tasks. Notably, SparseVLM and FastV show significant gains on MVBench, where spatiotemporal changes are more pronounced. This improvement stems from the complementary nature of our approach: while Intra-LLM methods focus on **textual relevance**, our strategy considers **visual uniqueness**. This combination enables more comprehensive token preservation, capturing both distinctive visual content and instruction-relevant information, thus mitigating unique visual information loss that often occurs in text-centric approaches during token compression in LLM.

5 Conclusion

In this work, we first analyze existing token compression methods for VideoLLMs, identifying two key limitations: design myopia and implementation constraints. We then derive three principles for effective token compression: model adaptability, frame uniqueness, and operator compatibility. Guided by the three principles, we propose VidCom^2, a novel plug-and-play acceleration framework. VidCom^2 dynamically adjusts compression intensity based on frame uniqueness, effectively preserving the most distinctive tokens both within each frame and across the entire video. Extensive experiments demonstrate VidCom^2 achieves state-of-the-art performance and efficiency across diverse benchmarks.

6 Limitations

In our work, we propose a plug-and-play efficient token compression framework for VideoLLM acceleration. Due to computational constraints, we couldn't evaluate our method on larger models like LLaVA-Video-72B and Qwen2-VL-72B. However, given VidCom^2's simplicity and the significant advantages demonstrated in Table 2 and Table 4, we anticipate its benefits may extend to or even amplify in larger architectures. This expectation is based on the increased importance of efficient token management in more complex models. Future work will focus on comprehensive evaluation across various model sizes to further validate and explore VidCom^2's potential in larger-scale scenarios. Additionally, we aim to adapt VidCom^2 for real-time streaming video understanding scenarios, further expanding its practical applications.

Acknowledgement

This research was supported by the Shanghai Science and Technology Program (Grant No. 25ZR1402278). Besides, we thank Huawei Ascend Cloud Ecological Development Project for the support of Ascend 910 processors.

References

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

Junjie Chen, Xuyang Liu, Zichen Wen, Yiyu Wang, Siteng Huang, and Honggang Chen. 2025. Variation-aware vision token dropping for faster large vision-language models. *arXiv preprint arXiv:2509.01552*.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2024a. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *Proceedings of the European Conference on Computer Vision*.

Yukang Chen, Fuzhao Xue, Dacheng Li, Qinghao Hu, Ligeng Zhu, Xiuyu Li, Yunhao Fang, Haotian Tang, Shang Yang, Zhijian Liu, and 1 others. 2024b. Longvila: Scaling long-context visual language models for long videos. *arXiv preprint arXiv:2408.10188*.

Tri Dao. 2024. Flashattention-2: Faster attention with better parallelism and work partitioning. In *Proceedings of the International Conference on Learning Representations*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Proceedings of the Advances in Neural Information Processing Systems*.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, and 1 others. 2024. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*.

Yuhang Han, Xuyang Liu, Pengxiang Ding, Donglin Wang, Honggang Chen, Qingsen Yan, and Siteng Huang. 2024. Rethinking token reduction in mllms: Towards a unified paradigm for training-free acceleration. *arXiv preprint arXiv:2411.17686*.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. 2024a. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, and 1 others. 2024b. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206.

Wentong Li, Yuqian Yuan, Jian Liu, Dongqi Tang, Song Wang, Jianke Zhu, and Lei Zhang. 2024c. Token-Packer: Efficient visual projector for multimodal LLM. *arXiv preprint arXiv:2407.02392*.

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

Xuyang Liu, Ziming Wang, Yuhang Han, Yingyao Wang, Jiale Yuan, Jun Song, Bo Zheng, Linfeng Zhang, Siteng Huang, and Honggang Chen. 2025a. Global compression commander: Plug-and-play inference acceleration for high-resolution large vision-language models. *arXiv preprint arXiv:2501.05179*.

Xuyang Liu, Zichen Wen, Shaobo Wang, Junjie Chen, Zhishan Tao, Yubo Wang, Xiangqi Jin, Chang Zou, Yiyu Wang, Chenfei Liao, and 1 others. 2025b. Shifting ai efficiency from model-centric to data-centric compression. *arXiv preprint arXiv:2505.19147*.

Junpeng Ma, Qizhe Zhang, Ming Lu, Zhibin Wang, Qiang Zhou, Jun Song, and Shanghang Zhang. 2025. Mmg-vid: Maximizing marginal gains at segment-level and token-level for efficient video llms. *arXiv preprint arXiv:2508.21044*.

Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. 2023. Egoschema: A diagnostic benchmark for very long-form video language understanding. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 36, pages 46212–46244.

Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Mateusz Malinowski, Yi Yang, Carl Doersch, and 1 others. 2023. Perception test: A diagnostic benchmark for multimodal video models. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 36, pages 42748–42761.

Hui Sun, Shiyin Lu, Huanyu Wang, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Ming Li. 2025. Mdp3: A training-free approach for list-wise frame selection in video-llms. *arXiv preprint arXiv:2501.02885*.

Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. 2025. Dycoke: Dynamic compression of tokens for fast video large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Yi Wang, Xinhao Li, Ziang Yan, Yinan He, Jiashuo Yu, Xiangyu Zeng, Chenting Wang, Changlian Ma, Haian Huang, Jianfei Gao, and 1 others. 2025. Internvideo2. 5: Empowering video mllms with long and rich context modeling. *arXiv preprint arXiv:2501.12386*.

Zichen Wen, Yifeng Gao, Weijia Li, Conghui He, and Linfeng Zhang. 2025a. Token pruning in multimodal large language models: Are we solving the right problem? In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 15537–15549.

Zichen Wen, Yifeng Gao, Shaobo Wang, Junyuan Zhang, Qintong Zhang, Weijia Li, Conghui He, and Linfeng Zhang. 2025b. Stop looking for important tokens in multimodal language models: Duplication matters more. *arXiv preprint arXiv:2502.11494*.

Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. 2024. Longvideobench: A benchmark for long-context interleaved video-language understanding. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 37, pages 28828–28857.

Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, and 1 others. 2025. Pyramiddrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. 2025. Visionzip: Longer is better but not necessary in vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11941–11952.

Hang Zhang, Xin Li, and Lidong Bing. 2023. Video-LLaMA: An instruction-tuned audio-visual language model for video understanding. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 543–553.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and 1 others. 2024a. Lmms-eval: Reality check on the evaluation of large multimodal models. *arXiv preprint arXiv:2407.12772*.

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. *arXiv 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, and Shanghang Zhang. 2025. SparseVLM: Visual token sparsification for efficient vision-language model inference. In *Proceedings of the International Conference on Machine Learning*.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. 2024c. Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*.

Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Shitao Xiao, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. 2024. Mlvu: A comprehensive benchmark for multi-task long video understanding. *arXiv preprint arXiv:2406.04264*.

In the appendix, we provide more benchmark details in Section A, model details in Section B, more baseline details in Section C, sensitivity analysis in Section D, algorithm details in Section E, and more visualization of frame uniqueness quantified by VidCom^2 in Section F.

A Benchmark Details

We present a detailed overview of video understanding benchmarks, as described below:

We present a detailed overview of video understanding benchmarks, as described below:

- **MVBench** (Li et al., 2024b) defines 20 video understanding tasks that require deep comprehension of temporal dimensions, beyond single-frame analysis.
- **LongVideoBench** (Wu et al., 2024) focuses on long-context video understanding with 3,763 videos up to one hour long. It includes 6,678 multiple-choice questions across 17 categories, emphasizing temporal information retrieval and analysis.
- **MLVU** (Zhou et al., 2024) features videos ranging from 3 minutes to 2 hours, encompassing 9 evaluation tasks including topic reasoning, anomaly recognition, video summarization, and plot question-answering.
- **VideoMME** (Fu et al., 2024) comprises 900 videos and 2,700 multiple-choice questions across six domains, with durations from 11 seconds to 1 hour, categorized into short, medium, and long subsets.
- **EgoSchema** (Mangalam et al., 2023) consists of 5,000 egocentric videos with multiple-choice questions requiring comprehensive understanding of procedural activities and temporal reasoning over extended sequences, challenging models with first-person perspective video analysis.
- **PerceptionTest** (Patraucean et al., 2023) presents 11,609 real-world videos with 38,565 multiple-choice questions evaluating diverse perceptual skills including object tracking, action recognition, and temporal localization across varied scenarios and contexts.

B Model Details

We introduce the VideoLLMs used for evaluation in main text, as follows:

- **LLaVA-OneVision** (Li et al., 2024a) unifies single-image, multi-image, and video tasks in a single LLaVA-OneVision model. It represents videos as long visual token sequences in the same "interleaved" format used for images, enabling smooth task transfer from images to videos and facilitating strong zero-shot video understanding capabilities.
- **LLaVA-Video** (Zhang et al., 2024c) builds upon the single-image stage checkpoint of LLaVA-OneVision. It is fine-tuned on a large synthetic video-instruction dataset (LLaVA-Video-178K), covering detailed captioning, open-ended QA, and multiple-choice QA. By employing the SigLIP visual encoder and Qwen2 as the LLM, LLaVA-Video achieves robust video comprehension across various benchmarks.
- **Qwen2-VL** (Wang et al., 2024) introduces Naive Dynamic Resolution to adaptively convert frames of any resolution into visual tokens. It utilizes Multimodal Rotary Position Embedding within a unified image-and-video processing paradigm, enabling the handling of long videos (20+ minutes) for high-quality QA, dialogue, and content creation.

C Baseline Details

We provide detailed introductions and comparisons of existing token compression methods mentioned in the main text, as follows:

- **FastV** (Chen et al., 2024a) performs one-time token pruning as an intra-LLM compression method, utilizing attention weights associated with the output token after a selected LLM layer. However, its explicit dependence on attention weights makes it incompatible with Flash Attention (Dao et al., 2022) in LLM.
- **PDrop** (Xing et al., 2025) extends intra-LLM compression by implementing progressive token pruning across multiple LLM layers, based on attention weights of output tokens. Similarly, this explicit attention mechanism prevents compatibility with Flash Attention (Dao et al., 2022) in LLM.

- **SparseVLM** (Zhang et al., 2025) functions as an intra-LLM compression method, ranking token importance using text-visual attention maps and pruning via pre-selected text prompts to mitigate attention noise. Similar to FastV, SparseVLM is also incompatible with Flash Attention (Dao et al., 2022) in LLM.

- **MUSTDrop** (Liu et al., 2024) is a three-stage compression method operating in ViT and LLM stages. It relies on `[CLS]` token attention and text-visual attention for token selection and pruning. This approach faces compatibility issues with `[CLS]`-free VideoLLMs and prevents Flash Attention support in LLM due to its explicit use of attention weights.

- **FiCoCo** (Han et al., 2024) is a two-stage compression method that merges tokens in ViT using `[CLS]` and patch-patch attention, then further compresses in LLM using text-visual attention. It suffers from `[CLS]` dependency and lacks Flash Attention compatibility.

- **FasterVLM** (Zhang et al., 2024b) is another pre-LLM compression method that relies on `[CLS]` token attention weights to retain informative visual tokens. It also faces compatibility issues with `[CLS]`-free VideoLLMs and Flash Attention integration in ViT.

- **DyCoke** (Tao et al., 2025) is a two-stage VideoLLM-specific method that first prunes similar tokens along the temporal dimension and then uses attention weights in the LLM to compress the less attended visual tokens in the KV cache. Due to its reliance on dividing frame sets into parts and compressing them through similarity calculations, similar to ToMe (Bolya et al., 2023), it cannot achieve aggressive token compression in one go. While its token compression stage is compatible with Flash Attention (Dao et al., 2022), its KV cache compression requires explicit attention weights and thus remains incompatible with efficient attention operators.

D Sensitivity Analysis

Table 8 further explores the hyper-parameter that balances the influence of $u_{t,m}^{\text{frame}}$ and $u_{t,m}^{\text{video}}$ on $u_{t,m}$ in our VidCom^2 method. We observe that our method is not particularly sensitive to the balancing coefficient, as different degrees of balancing result in minimal performance differences. However, all balanced configurations outperform using either $u_{t,m}^{\text{frame}}$ or $u_{t,m}^{\text{video}}$ alone. This suggests that when performing token compression in VideoLLMs, it is crucial to consider the uniqueness of each token both within its frame and across the entire video to preserve more distinctive visual information. Notably, we find that $u_{t,m} = u_{t,m}^{\text{frame}} + u_{t,m}^{\text{video}}$ yields the best performance, indicating that $u_{t,m}^{\text{frame}}$ and $u_{t,m}^{\text{video}}$ are equally important. Therefore, we adopt $u_{t,m} = u_{t,m}^{\text{frame}} + u_{t,m}^{\text{video}}$ as our default configuration.

Metrics	**MVBench**	**VideoMME**				**Avg.**
		Overall	**Short**	**Medium**	**Long**	
Vanilla	56.9	58.6	70.3	56.6	48.8	100.0
$u_{t,m}^{\text{frame}}$	56.8	57.9	68.8	56.9	48.1	98.8
$u_{t,m}^{\text{video}}$	56.8	58.3	69.3	56.1	49.3	99.3
Combination						
$u_{t,m}^{\text{frame}} + u_{t,m}^{\text{video}}$	57.2	58.6	69.8	56.4	49.4	100.3
$u_{t,m}^{\text{frame}} + 2u_{t,m}^{\text{video}}$	56.1	58.4	69.7	56.4	49.0	99.5
$2u_{t,m}^{\text{frame}} + u_{t,m}^{\text{video}}$	56.9	58.6	69.7	56.8	49.3	100.0

Table 8: **Effects of balancing hyper-parameters between $u_{t,m}^{\text{frame}}$ and $u_{t,m}^{\text{video}}$ on VidCom^2 performance.**

E Algorithm Details of VidCom^2

Algorithm 1 present the algorithm workflow of our VidCom^2. This algorithm details the step-by-step process of our token compression framework, illustrating how VidCom^2 dynamically adjusts compression intensity based on frame uniqueness and preserves the most distinctive tokens both within each frame and across the entire video sequence.

F More Visualization of Frame Uniqueness

Figure 6 presents additional visualizations of frame distinctiveness as quantified by our VidCom^2. These cases cover a diverse range of scenarios, including everyday life situations, sports activities, dynamic scenes, and scientific domains. The visualizations demonstrate that VidCom^2 effectively quantifies frame uniqueness across these varied contexts, consistently allocating more token budget to distinctive frames. This approach ensures the preservation of more visually unique information across diverse scenarios, which is crucial for accurate video understanding by VideoLLMs.

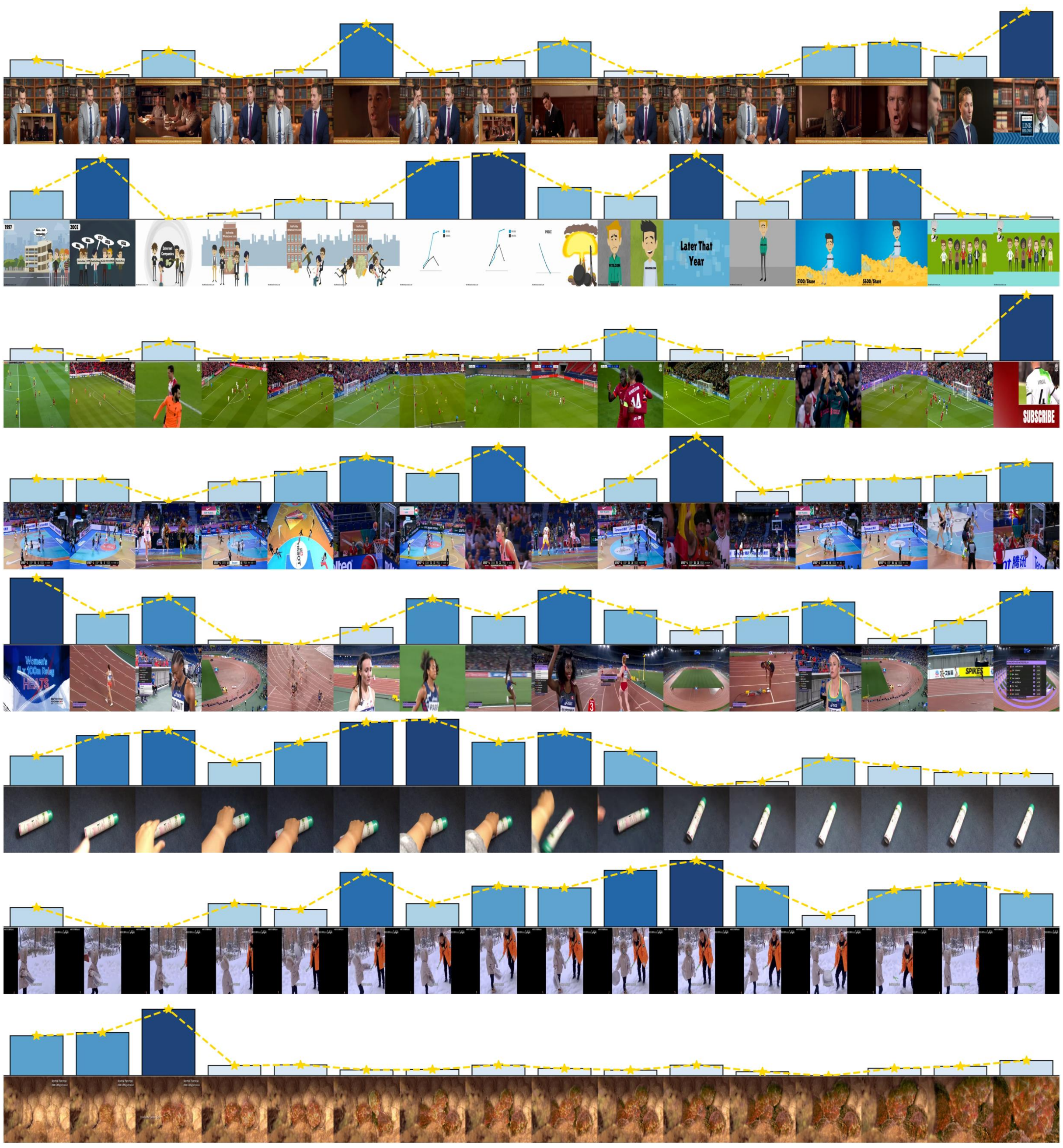

Figure 6: **More visualization of frame uniqueness quantified by our VidCom^2.** In most cases, the frame uniqueness determined by VidCom^2 aligns well with human video perception.

Algorithm 1 VidCom^2: Plug-and-Play Token Compression for VideoLLMs

```
Require: Video tokens $\mathbf{X}^v = \{\mathbf{x}^v_{t,m}\}_{t=1,m=1}^{T,M}$,
    Preset retention ratio $R \in (0,1]$, Temperature
    $\tau > 0$, Stability epsilon $\epsilon > 0$
Ensure: Compressed tokens $\{\hat{\mathbf{X}}^v_t\}_{t=1}^{T}$
 1: Stage 1: Frame Compression Adjustment
 2: // 1. Compute global summary
 3: $\mathbf{g}_v \leftarrow \frac{1}{T \cdot M} \sum_{t=1}^{T} \sum_{m=1}^{M} \mathbf{x}^v_{t,m}$
 4: // 2. Token–video similarity
 5: for $t = 1 \to T,\ m = 1 \to M$ do
 6:     $s^{\text{video}}_{t,m} \leftarrow \frac{\mathbf{x}^v_{t,m} \cdot \mathbf{g}_v}{\|\mathbf{x}^v_{t,m}\| \|\mathbf{g}_v\|}$
 7:     $u^{\text{video}}_{t,m} \leftarrow -\, s^{\text{video}}_{t,m}$
 8: end for
 9: // 3. Frame uniqueness
10: for $t = 1 \to T$ do
11:     $u_t \leftarrow \frac{1}{M} \sum_{m=1}^{M} u^{\text{video}}_{t,m}$
12: end for
13: // 4. Normalize & weigh
14: for $t = 1 \to T$ do
15:     $\tilde{u}_t \leftarrow (u_t - \max_k u_k)/\tau$
16:     $\sigma_t \leftarrow \exp(\tilde{u}_t) \,/\, (\sum_{k=1}^{T} \exp(\tilde{u}_k) + \epsilon)$
17:     $r_t \leftarrow R\big(1 + \sigma_t - \frac{1}{T}\big)$
18: end for
19: Stage 2: Adaptive Token Compression
20: for $t = 1 \to T$ do
21:     // 1. Frame-level token uniqueness
22:     $\mathbf{g}_{f,t} \leftarrow \frac{1}{M} \sum_{m=1}^{M} \mathbf{x}^v_{t,m}$
23:     for $m = 1 \to M$ do
24:         $s^{\text{frame}}_{t,m} \leftarrow \frac{\mathbf{x}^v_{t,m} \cdot \mathbf{g}_{f,t}}{\|\mathbf{x}^v_{t,m}\| \|\mathbf{g}_{f,t}\|}$
25:         $u^{\text{frame}}_{t,m} \leftarrow -s^{\text{frame}}_{t,m}$
26:     end for
27:     // 2. Combine video & frame uniqueness
28:     for $m = 1 \to M$ do
29:         $u_{t,m} \leftarrow u^{\text{video}}_{t,m} + u^{\text{frame}}_{t,m}$
30:     end for
31:     // 3. Top-$k$ selection
32:     $k_t \leftarrow \lceil r_t \times M \rceil$
33:     $\hat{\mathbf{X}}^v_t \leftarrow \text{TopK}(\{\mathbf{x}^v_{t,m}\}, \{u_{t,m}\}, k_t)$
34: end for
35: return $\{\hat{\mathbf{X}}^v_t\}_{t=1}^{T}$
```