HoliTom : Holistic Token Merging for Fast Video Large Language Models

Kele Shao[1,2,3], **Keda Tao**[1,3], **Can Qin**[4], **Haoxuan You**[5], **Yang Sui**[6], **Huan Wang**[3,*]

[1]Zhejiang University [2]Shanghai Innovation Institute [3]Westlake University
[4]Salesforce AI Research [5]Columbia University [6]Rice University

https://github.com/cokeshao/HoliTom

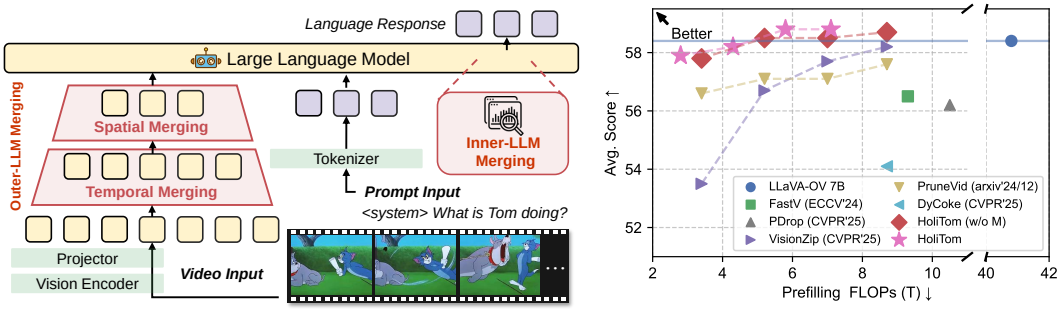


Figure 1: **Left:** We introduce *HoliTom*, a training-free holistic token merge method for fast video LLMs. Its key innovation lies in its global, redundancy-aware outer-LLM spatio-temporal compression and robust, token similarity-based inner-LLM compression. **Right:** The Efficiency/Performance trade-off curve of multiple training-free methods on four widely used video understanding benchmarks: MVBench, EgoSchema, LongVideoBench, and VideoMME. Our method, *HoliTom*, surpasses the SoTA approaches by maintaining 99.1% average performance while reducing FLOPs to 6.9%.

Abstract

Video large language models (video LLMs) excel at video comprehension but face significant computational inefficiency due to redundant video tokens. Existing token pruning methods offer solutions. However, approaches operating within the LLM (inner-LLM pruning), such as FastV, incur intrinsic computational overhead in shallow layers. In contrast, methods performing token pruning before the LLM (outer-LLM pruning) primarily address spatial redundancy within individual frames or limited temporal windows, neglecting the crucial global temporal dynamics and correlations across longer video sequences. This leads to sub-optimal spatio-temporal reduction and does not leverage video compressibility fully. Crucially, the synergistic potential and mutual influence of integrating these strategies remain unexplored. To further reduce redundancy, we introduce *HoliTom*, a novel training-free holistic token merging framework. *HoliTom* employs outer-LLM pruning through global redundancy-aware temporal segmentation, followed by spatial-temporal merging to reduce visual tokens by over 90%, significantly alleviating the LLM's computational burden. Complementing this, we introduce a robust inner-LLM token similarity-based merging approach, designed for superior performance and compatibility with outer-LLM pruning. Evaluations demonstrate our method's promising efficiency-performance trade-off on LLaVA-OneVision-7B, reducing computational costs to 6.9% of FLOPs while maintaining 99.1% of the original performance. Furthermore, we achieve a 2.28× reduction in Time-To-First-Token (TTFT) and a 1.32× acceleration in decoding throughput, highlighting the practical benefits of our integrated pruning approach for efficient video LLMs inference.

*Corresponding authors: wanghuan@westlake.edu.cn

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

1 Introduction

Video large language models (video LLMs) [21, 62, 48, 7, 8, 23, 26, 55, 59, 15, 43] have shown remarkable potential in understanding complex video content. However, their practical deployment is hindered by significant computational inefficiency. This inefficiency stems from processing high volumes of video tokens generated by encoding sampled frames, leading to substantial overhead, particularly due to the quadratic complexity of the attention mechanism in the LLMs. For videos with numerous frames, the input token count can easily reach tens of thousands, making inference computationally expensive. While prior works [6, 57, 53, 41, 30] have explored model compression and token pruning, achieving a desirable balance between efficiency and performance in video tasks remains an open challenge. Thus, developing effective methods to reduce video token redundancy while preserving critical semantic information is crucial for the widespread adoption of video LLMs.

Token pruning is a promising direction. These approaches generally fall into two categories depending on where pruning occurs. *Inner-LLM pruning* methods, such as FastV [6], TopV [56], and PDrop [53], operate within the LLM layers. However, they incur intrinsic computational and memory costs in the initial layers before pruning takes effect, limiting overall FLOPs reduction. *Outer-LLM pruning* methods process tokens before the main LLM computation. Some methods address spatial redundancy (VisionZip [57], PruMerge [36]), others tackle temporal aspects within limited temporal windows (DyCoke [41], PruneVid [17]), thus preventing a global understanding of video dynamics and comprehensive spatio-temporal optimization. Furthermore, despite the potential for synergy, no prior work has systematically explored integrating *inner-LLM* and *outer-LLM pruning* strategies or analyzed their mutual benefits. The current methods, while offering some benefits, still leave room for improvement.

Table 1: **Compression scope of vision-language model acceleration methods.** This table outlines where different methods apply compression. *Spatial* and *Temporal* refer to compression of the input visual data, while *Inner-LLM* indicates compression mechanisms applied within the model's processing.

Methods	Spatial	Temporal	Inner-LLM
FastV [6]	✗	✗	✓
PDrop [53]	✗	✗	✓
LLaVA-PruMerge [36]	✓	✗	✗
VisionZip [57]	✓	✗	✗
DyCoke [41]	✗	✓	✓
FastVID [37]	✓	✓	✗
Ours	✓	✓	✓

To address these limitations, we propose a holistic token pruning for video LLMs that leverages external and internal strategies. Our method first tackles temporal redundancy through a global redundancy-aware video segmentation process, followed by spatio-temporal merging. This external step reduces visual tokens to less than 10%, significantly alleviating the computational burden on the subsequent LLM. Complementing this, we introduce a new and robust inner-LLM token similarity-based merging method, specifically designed for integration with our outer-LLM pruning method, enabling mutual benefits. This integrated strategy offers a more holistic and efficient solution to handle long videos with LLMs, as summarized in Tab. 1, which contrasts the compression achieved by our approach in both the spatio-temporal domain and within the inner-LLM against other methods.

Empirical evaluations validate the effectiveness of our proposed method in achieving a compelling efficiency-performance trade-off. Specifically, as shown in Fig. 1 (right), on the LLaVA-OneVision-7B model [21], our approach reduces computational costs to just 6.9% of the original FLOPs while remarkably preserving 99.1% of the original model's performance. Moreover, we observe significant gains in inference efficiency, achieving a 2.28× reduction in Time-To-First-Token (TTFT) and a 1.32× acceleration in decoding throughput. These results clearly demonstrate the substantial practical advantages of our holistic token merging framework for efficient video LLM inference.

Our key contributions are summarized as follows:

1. We analyze the phenomenon of temporal redundancy in the context of video LLMs and propose a global redundancy-aware temporal merging method to effectively address the inefficiency in video LLMs before LLM processing in a plug-and-play fashion.
2. We introduce a robust inner-LLM similarity-based merging technique specifically designed for integration with the outer-LLM pruning method, facilitating synergistic optimization.
3. Extensive evaluations on LLaVA-OneVision and LLaVA-Video demonstrate that our integrated pruning framework achieves a state-of-the-art efficiency-performance trade-off, significantly reducing computational costs and accelerating inference while preserving model performance.

2 Related Work

2.1 Video Large Language Models

The rapid progress of multimodal large language models has led to the integration of video encoders, creating video LLMs that excel in video understanding and question answering tasks [55, 14, 21, 2, 48, 3, 23, 24, 26, 59, 40, 42, 19, 1, 39, 25]. However, the substantial number of tokens generated by processing numerous video frames hinders inference efficiency, thereby impeding the widespread adoption of video LLMs. Existing approaches have attempted to mitigate this issue. For instance, VideoLLaMA [59] employs a Q-Former module [22] to aggregate video tokens, while MovieChat [40] introduces a memory module to merge and store token representations. Although pooling mechanisms in LLaVA-OneVision [21] reduce token counts, each video frame still produces hundreds of tokens for downstream processing. Consequently, handling tens of thousands of visual tokens for long video inputs substantially increases inference time and memory consumption. While works such as VILA [28] and NVILA [29] aim to optimize token usage, these methods often require model fine-tuning, demanding considerable hardware resources [18, 26, 28, 29, 47]. This underscores a critical need for developing more efficient, training-free token compression methods specifically for video LLMs, bypassing the need for costly model adaptations and significant hardware investment.

2.2 Visual Token Compression

Token compression has emerged as an effective strategy for reducing token redundancy in vision transformers and large language models. ToMe [4] merges similar tokens in ViTs to alleviate spatial redundancy, while TempMe [38] focuses on minimizing temporal redundancy by merging adjacent video clips. TESTA [34] achieves up to a 75% reduction in processed tokens by employing temporal and spatial aggregation modules. For MLLMs, FastV [6] prunes non-essential visual tokens in early layers of LLM. TopV [56] proposes an optimization framework to prune unnecessary visual tokens. DyMU [49] introduces token merging in the visual encoder and virtual unmerging in the LLM decoder. PDrop [53] performs progressive pruning of tokens at different stages within the LLM. LLaVA-PruMerge [36] and VisionZip [57] leverage attention weight analysis in visual encoders to eliminate spatial redundancy. However, the inherent temporal dependencies between video frames necessitate specialized compression designs. Consequently, recent methods specifically for video token compression have gained increasing attention. DyCoke [41] consolidates tokens across frames and implements dynamic key-value cache reduction. PruneVID [17] clusters video tokens, whereas FastVID [37] enhances compression by combining temporal segmentation with spatio-temporal token merging. In this paper, we propose a new token merging strategy specifically designed for video LLMs, which fully considers spatio-temporal characteristics to maximize performance retention.

3 Method

3.1 Background on Video LLMs Inference

The inference process of video LLMs involves three key stages: before LLM, prefilling, and decoding.

(1) Before LLM. Given an input video with B frames, a vision encoder processes each frame to produce N_v embedding vectors. These are projected into the text embedding space, yielding visual tokens $H_v \in \mathbb{R}^{BN_v \times d}$, where d represents the dimension of the hidden state space. A text prompt $T = \{t_i\}_{i=1}^{N_q}$ is tokenized and embedded into text tokens $H_q \in \mathbb{R}^{N_q \times d}$ similarly. Finally, the visual and text tokens are concatenated to form $H = \text{concat}[H_v, H_q]$, which serves as the LLM input.

(2) Prefilling Stage. During prefilling, each transformer layer l of the LLM performs self-attention operations on the concatenated input H. It begins with linear transformations to compute query $\mathbf{Q}^l = H\mathbf{W}_Q^l$, key $\mathbf{K}^l = H\mathbf{W}_K^l$, and value $\mathbf{V}^l = H\mathbf{W}_V^l$, where $\mathbf{W}_Q^l$, $\mathbf{W}_K^l$, $\mathbf{W}_V^l \in \mathbb{R}^{d \times d}$ are learnable projection matrices. The resulting key-value pairs ($\mathbf{K}^l$ and $\mathbf{V}^l$) are then cached (KV cache) to enhance the efficiency of token generation during the subsequent decoding phase.

(3) Decoding Stage. The decoding stage generates tokens autoregressively, leveraging the KV cache. At each time step t, only the new token h_t is processed to compute its key and value representations, avoiding recalculating attention weights over the entire history. The KV cache is updated by appending the new key-value pairs: $\mathbf{K} \leftarrow [\mathbf{K}, h_t\mathbf{W}_K]$, $\mathbf{V} \leftarrow [\mathbf{V}, h_t\mathbf{W}_V]$. This caching mechanism substantially reduces the computational complexity of the generation process.

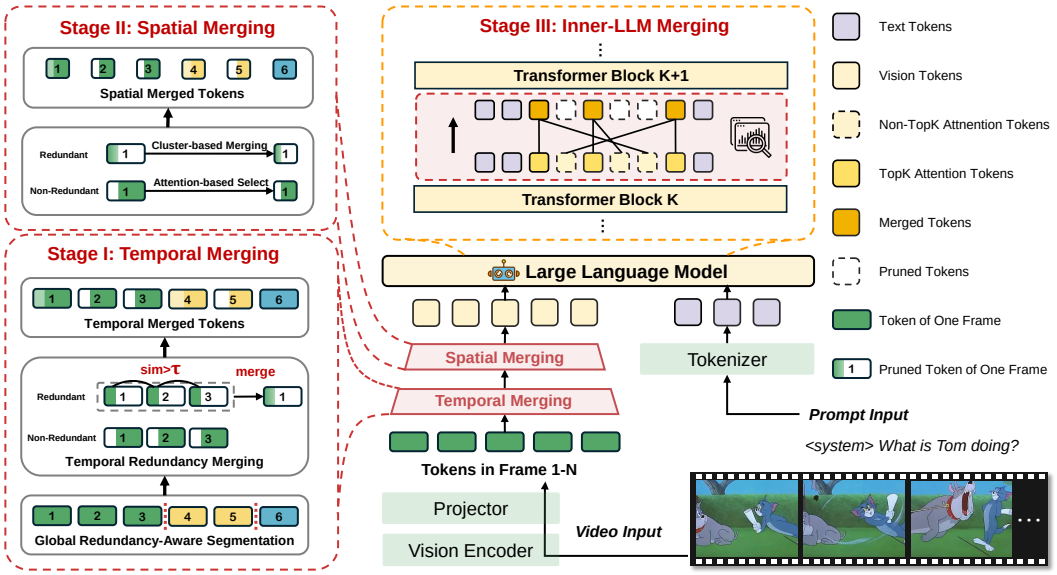


Figure 2: **Overview of our *HoliTom* method.** *HoliTom* compresses video LLMs across three scopes; the first two are outer-LLM pruning. **Temporal Merging** maximizes temporal compression via global redundancy-aware segmentation, merging similar tokens into their first occurrence. **Spatial Merging** further reduces redundancy by applying tailored spatial compression based on the characteristics of remaining temporal variations. **Inner-LLM Merging** leverages attention within the LLM to identify key tokens and merges less important, similar tokens, streamlining information within the LLM.

3.2 Global Redundancy-Aware Temporal Merging

Temporal redundancy describes feature persistence at fixed spatial locations across consecutive frames. We identify this redundancy for the k-th feature between frames m and $m+1$ using their respective feature vectors $h_{m,k}$ and $h_{m+1,k}$. A feature is considered temporally redundant if its normalized similarity, $sim(h_{m,k}, h_{m+1,k})$, exceeds a defined threshold $\tau \in [0, 1]$.

For a temporal segment defined by a start frame t_s and an end frame t_e (covering $[t_s, t_e)$), the total number of prunable tokens, $g(t_s, t_e)$, is calculated. This involves counting tokens $N(t_s, t_e)$ that are consecutively redundant across *all* frames from t_s to $t_e - 1$, and then multiplying by the number of subsequent frames $(t_e - t_s - 1)$ within the segment where these tokens can be pruned. Our method prunes the redundant by merging these subsequent occurrence tokens into their first appearance at start frame t_s, treating them as *temporal redundant tokens*, as shown in Fig. 2. The formulation is:

$$g(t_s, t_e) = \underbrace{\left(\sum_{k=1}^{N_v} \prod_{m=t_s}^{t_e-2} \mathbb{I}(sim(h_{m,k}, h_{m+1,k}) > \tau) \right)}_{N(t_s,t_e)} \times (t_e - t_s - 1), \tag{1}$$

where N_v is the total number of features per frame, and $\mathbb{I}(\cdot)$ the indicator function.

Given a video of B frames, our objective is to find a segmentation into K consecutive segments $[t_i, t_{i+1})$ (with $t_1 = 1$, $t_{K+1} = B + 1$, and $t_i < t_{i+1}$) that maximizes the total prunable features:

$$\max_{K, \{t_i\}_{i=1}^{K+1}} \sum_{i=1}^{K} g(t_i, t_{i+1}). \tag{2}$$

This optimization is solved using dynamic programming to achieve *global* optimization. Let $dp[i]$ be the maximum prunable features for a video ending at frame i (exclusive, i.e., considering frames $1, ..., i-1$), where frame i marks the exclusive end of the last segment. The value $prev[i]$ stores the optimal starting frame j^* of this final segment $[j^*, i)$. The state transition is given by:

$$dp[i] = \max_{1 \le j < i} \{dp[j] + g(j, i)\}, \quad \text{with} \quad prev[i] = \arg\max_{1 \le j < i} \{dp[j] + g(j, i)\}. \tag{3}$$

The base case is $dp[1] = 0$. The maximum prunable features for the entire video are $dp[B+1]$. The optimal segmentation is reconstructed by backtracking from $B + 1$ using the $prev$ array.

3.3 Spatial Merging

After temporal merging, tokens are classified as *non-redundant* or *redundant* temporal tokens. We first process the former. Inspired by works [36, 57, 61, 44], we utilize the CLS tokens for spatial feature selection. For vision encoders like Siglip [58] that do not have an explicit CLS token, a method is detailed to derive CLS-equivalent attention. Specifically, we compute the attention matrix:

$$A = \text{Softmax}(QK^T/\sqrt{d}) \in \mathbb{R}^{B\times N_v\times N_v}, \tag{4}$$

where d is the state dimension. Token importance is quantified by averaging the attention weights each token receives from all other tokens within the same frame in the vision tower, yielding a score vector $A_{\text{avg}} \in \mathbb{R}^{B\times N_v}$. Tokens receiving higher average attention are considered more salient.

Consistent with video LLMs of applying spatial pooling (e.g., after the projector to reduce tokens), we reshape A_{avg} to its original spatial grid dimensions ($H \times W = N_v$) and apply an analogous pooling operation. This results in a spatially downsampled importance map $\overline{A}_{\text{avg}} \in \mathbb{R}^{B\times\overline{H}\times\overline{W}}$. Ultimately, we select the visual features corresponding to the highest scores in $\overline{A}_{\text{avg}}$ as the representative and most informative spatial tokens, known as *attention-based select*, discarding all others.

The computation of vision tower attention is intra-frame. Averaging these attention weights across frames lacks theoretical justification, invalidating the attention-based method for *redundant temporal tokens*. To process these features, we employ a *cluster-based merging* method utilizing density peak clustering based on k-nearest neighbors (DPC-KNN) [12, 35]. Given a set of N redundant temporal tokens $[v_1, v_2, ..., v_N]$ within the first frame of the segmentation. For each token v_i, we calculate its local density ρ_i, distance to the closest higher-density token δ_i and the final density score $\gamma_i = \rho_i \times \delta_i$:

$$\rho_i = \exp\left(-\frac{1}{k}\sum_{v_j\in\text{kNN}(v_i)} d(v_i,v_j)^2\right), \quad \delta_i = \begin{cases} \max\limits_{j\neq i} d(v_i,v_j) & \text{if } \rho_i = \max\limits_k \rho_k \\ \min\limits_{j:\rho_j>\rho_i} d(v_i,v_j) & \text{otherwise} \end{cases}. \tag{5}$$

Tokens with high γ_i are selected as cluster centers. After selecting the cluster centers, each remaining feature is assigned to the cluster whose center is closest in feature space. Finally, the representative feature for each cluster is then computed by averaging the features assigned to it. Ultimately, the compressed features, derived from this clustering process, along with the non-redundant features, are concatenated according to their original spatial order, thereby preserving positional characteristics.

3.4 Inner-LLM Merging

Inefficient visual attention in large vision language models has been widely discussed [6, 53]. Existing methods, such as FastV [6] and PDrop [53], directly *discard* redundant visual tokens, which may lead to performance degradation due to information loss. Unlike these approaches, our proposed method addresses it by *merging* the information from potentially redundant tokens instead of simply discarding them. Specifically, at the K-th layer of the LLM, to reduce the number of visual tokens by R%, we employ a token selection strategy based on attention scores. We use the attention weights of the last token to rank all vision tokens at layer K. The R% of visual tokens exhibiting the lowest attention scores are identified as candidates for merging. We find its most similar visual token within the set of tokens designated for retention. For a retained token v_r and its associated set of low attention tokens $V_m = \{v_{m_1}, v_{m_2}, ..., v_{m_n}\}$. The updated retained token v'_r is:

$$v'_r = \text{average}(v_r, v_{m_1}, ..., v_{m_n}). \tag{6}$$

This selective merging preserves relevant features from tokens that would otherwise be removed, mitigating information loss while achieving the desired token reduction.

4 Experimental Results

4.1 Experimental Settings

Benchmarks. We evaluate our method on four widely-used video understanding benchmarks: MVBench [24], EgoSchema [31], LongVideoBench [51], and VideoMME [14]. Comprising videos of varying lengths and complex scenarios, these benchmarks provide a comprehensive testbed for assessing the effectiveness and generalization of our method.

Table 2: **Comparison of state-of-the-art methods across benchmarks. Best** and **most efficient** results are in bold, second best underlined. Here, "HoliTom" means the full version of our method; "HoliTom (w/o M)" means our method *without* inner-LLM merging, for reference.

Method	Prefilling FLOPs (T) ↓	FLOPs Ratio ↓	Before LLM Retained Ratio	MVBench ↑	EgoSchema ↑	LongVideo Bench ↑	VideoMME ↑	Avg. ↑ Score	Avg. ↑ %
LLaVA-OV-7B	40.8	100%	100%	58.3	60.4	56.4	58.6	58.4	100
FastV [6]	9.3	22.8%	100%	55.9	57.5	56.7	56.1	56.5	96.7
PDrop [53]	10.5	25.7%	100%	56.1	58.0	54.1	56.4	56.2	96.2
DyCoke [41]	8.7	21.3%	25%	53.1	59.5	49.5	54.3	54.1	92.6
VisionZip [57]	8.7	21.3%	25%	57.9	60.3	56.5	58.2	58.2	99.7
PruneVid [17]	8.7	21.3%	25%	57.4	59.9	55.7	57.4	57.6	98.6
FastVID [37]	8.7	21.3%	25%	56.5	-	56.3	58.0	-	-
HoliTom (w/o M)	8.7	21.3%	25%	**58.5**	60.8	56.5	**59.1**	58.7	100.5
HoliTom	**7.1**	**17.4%**	25%	58.4	**61.2**	**56.7**	58.9	**58.8**	**100.7**
VisionZip [57]	7.0	17.2%	20%	57.7	59.8	55.2	57.9	57.7	98.8
PruneVid [17]	7.0	17.2%	20%	57.2	59.7	54.7	56.9	57.1	97.8
FastVID [37]	7.0	17.2%	20%	56.3	-	**57.1**	57.9	-	-
HoliTom (w/o M)	7.0	17.2%	20%	58.5	60.7	56.3	**58.6**	58.5	100.2
HoliTom	**5.8**	**14.2%**	20%	**58.7**	**61.0**	**57.1**	**58.6**	**58.8**	**100.7**
VisionZip [57]	5.2	12.7%	15%	56.5	59.8	54.4	56.1	56.7	97.1
PruneVid [17]	5.2	12.7%	15%	56.8	59.7	55.4	56.6	57.1	97.8
FastVID [37]	5.2	12.7%	15%	56.0	-	56.2	57.7	-	-
HoliTom (w/o M)	5.2	12.7%	15%	**58.1**	61.0	**57.0**	**58.1**	**58.5**	**100.2**
HoliTom	**4.3**	**10.5%**	15%	**58.1**	**61.2**	56.4	57.3	58.2	99.7
VisionZip [57]	3.4	8.3%	10%	53.5	58.0	49.3	53.4	53.5	91.6
PruneVid [17]	3.4	8.3%	10%	56.2	59.8	54.5	56.0	56.6	96.9
FastVID [37]	3.4	8.3%	10%	55.9	-	56.3	**57.3**	-	-
HoliTom (w/o M)	3.4	8.3%	10%	**56.9**	61.1	**56.5**	56.9	57.8	99.0
HoliTom	**2.8**	**6.9%**	10%	**57.3**	**61.2**	56.3	56.8	**57.9**	**99.1**

Compared Methods. We compare our proposed *HoliTom* against **6** strong training-free baselines: 1) FastV [6], identifies key tokens during prefilling using attention scores between predicted and vision tokens; 2) PDrop [53], prunes visual tokens within partitioned LLM stages, guided by image and instruction tokens; 3) Visionzip [57], prunes tokens before LLM via spatial token merging; 4) DyCoke [41], employs temporal merging before LLM and dynamic KV cache pruning in decoding; 5) PruneVid [17], minimizes video redundancy via spatio-temporal token clustering; and 6) FastVID [37], a concurrent work, partitions videos and applies density-based token pruning. Due to the lack of public code, we compare FastVID to its reported results. For all other baselines and our method, experiments use their open-source code under identical hardware condition.

Inference Cost Evaluation. We evaluate the inference cost of transformer layers, each composed of multi-head attention (MHA) and feed-forward network (FFN) modules. Following previous work [6, 53, 41], the FLOPs for processing n_i *vision* tokens in layer i, with hidden state size d and FFN intermediate size m, are defined as $4n_id^2 + 2n_i^2d + 2n_idm$. For an LLM with T transformer layers, the total FLOPs span the prefilling and decoding phases, calculated as:

$$\sum_{i=1}^{T} \underbrace{(4n_id^2 + 2n_i^2d + 2n_idm)}_{\text{Prefilling FLOPs per layer}} + \underbrace{R((4d^2 + 2dm) + 2(dn_i + \frac{1}{2}d(R+1)))}_{\text{Decoding FLOPs per layer}}. \tag{7}$$

For consistency, the decoding calculation is fixed for predicting $R = 100$ tokens, accounting for the the KV cache. In video LLMs, the decoding phase FLOPs contribute only approximately **2%** of the total. Consequently, our primary optimization focus is on the *prefilling* stage. When considering prefilling optimization, inner-LLM pruning methods like FastV [6], the FLOPs incurred in the first 2 shallow layers can amount to 2.9 TFLOPs in LLaVA-OneVision-7B. Even pruning 100% token in the layer, these methods cannot match the potential efficiency compared to outer-LLM pruning methods. Thus, outer-LLM pruning offers a more impactful optimization approach for this domain.

Implementation Details. Our method is implemented on LLaVA-OneVision-7B/72B [21] and LLaVA-Video-7B [62] models. Evaluation uses NVIDIA A100 GPUs, while inference is on an RTX A6000. Inference cost is measured by prefilling FLOPs, with baselines configured for comparable FLOPs (details in the appendix A). The default τ is 0.8; for 10% compression, τ is 0.65. Following official practice, LLaVA-OneVision models utilize 32 input video frames (N_v = 196), while LLaVA-Video uses 64 frames (N_v = 169). All benchmarks are conducted using LMMs-Eval [60, 20].

Table 3: **Cross-backbone method comparison.** Performance comparison of our method against state-of-the-art methods across different backbones, demonstrating consistent effectiveness.

Model	Method	Prefilling FLOPs (T) ↓	FLOPs Ratio ↓	Before LLM Retained Ratio	MVBench ↑	EgoSchema ↑	LongVideo Bench ↑	VideoMME ↑	Avg. ↑ Score	Avg. ↑ %
LLaVA-OneVision-72B	Vanilla	429.3	100%	100%	60.9	61.1	62.7	65.7	62.6	100
	FastV [6]	86.4	20.1%	100%	56.1	57.1	57.0	61.2	57.9	92.5
	Visionzip [57]	59.0	13.7%	15%	58.4	59.3	57.4	63.8	59.7	95.4
	PruneVid [17]	59.0	13.7%	15%	56.8	57.7	57.4	62.7	58.6	93.6
	HoliTom (w/o M)	59.0	13.7%	15%	58.5	60.0	**57.8**	64.1	**60.1**	**96.0**
	HoliTom	**51.6**	**12.0%**	15%	**58.7**	**60.1**	57.2	**64.3**	**60.1**	**96.0**
LLaVA-Video-7B	Vanilla	80.2	100%	100%	60.4	57.2	58.9	64.3	60.2	100
	FastV [6]	17.1	21.3%	100%	54.3	54.1	55.0	58.8	55.6	92.4
	PDrop [53]	19.5	24.3%	100%	55.9	54.3	54.7	61.9	56.7	94.2
	VisionZip [57]	9.3	11.6%	15%	56.7	54.7	54.7	60.7	56.7	94.2
	HoliTom (w/o M)	9.3	11.6%	15%	**57.8**	**54.8**	55.6	61.9	57.5	95.5
	HoliTom	**7.6**	**9.5%**	15%	57.7	**54.8**	**56.2**	**62.1**	**57.7**	**95.8**

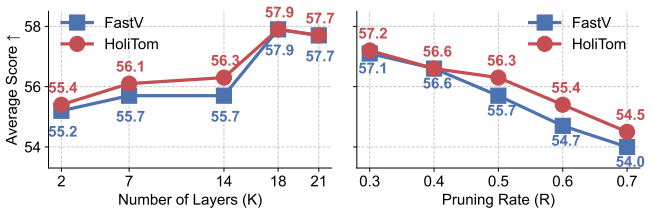


Figure 3: **Left:** Performance of our method *vs.* FastV when pruning various layers at rate R=50%. **Right:** Performance comparison with varying pruning rates at a fixed layer (K=14).

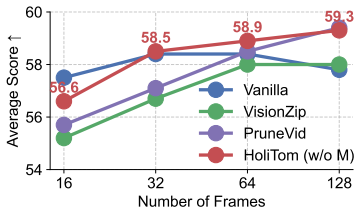


Figure 4: Performance *vs.* number of frames for our method and other token compression methods.

4.2 Main Results

Comparison with State-of-the-Art Methods. Tab. 2 benchmarks *Holitom* against state-of-the-art approaches on the LLaVA-OneVision-7B model, analyzing performance and inference cost (FLOPs) at various token retention ratios (25%, 20%, 15%, and 10%) prior to LLM processing. Inner-LLM pruning methods, such as FastV [6] and PDrop [53], often struggle to balance performance and efficiency, especially at lower token retention ratios (25%). DyCoke [41], which segments video frames into groups of 4 and prunes all but the first frame, is limited by its design, capping its lowest retention ratio at 25%. Spatial pruning methods like VisionZip [57] show a significant performance drop (up to 8.4%) at 10% retention. This decline stems from relying solely on spatial compression, less effective at preserving crucial temporal information needed for performance under aggressive pruning. Crucially, even **without** our inner-LLM merging technique, our method achieves state-of-the-art performance and efficiency consistently across the evaluated retention ratios. This highlights the superior robustness and adaptability of our approach compared to prior methods. Our inner-LLM merging method further enhances efficiency, driving optimization further. For instance, we retain only 6.9% of the original FLOPs, while preserving 99.1% of the baseline performance.

Performance Comparison Across Different Backbones. Tab. 3 assesses our method's performance across various backbones. For the powerful LLaVA-OneVision-72B model, sensitive to aggressive compression, our approach reduces computational cost to 11.3%, keeping 96% of its original performance. LLaVA-Video-7B presents a greater compression challenge due to its higher initial pooling rate (169 *vs.* 196 tokens/frame in LLaVA-OneVision). Despite this, our method achieves a reduction to just 9.5% of the original FLOPs, retaining 95.8% performance and outperforming existing methods. Overall, achieving significant token compression with minimal performance drop is indeed tougher for LLaVA-OneVision-72B and LLaVA-Video-7B than for LLaVA-OV-7B.

HoliTom *vs.* FastV under Outer-LLM Compression. Building on the challenges faced by inner-LLM pruning methods discussed in Section 4.1, we compare our inner-LLM merging method with FastV, specifically in scenarios where outer-LLM compression is already applied. In this compressed context, the property "*an image is worth 1/2 tokens after layer 2*" [6] is not consistently observed. This is because outer-LLM compression concentrates information, making trivial token discarding more difficult using attention mechanisms. As illustrated in Fig. 3, our method demonstrates superior performance compared to FastV when pruning 50% at shallower layers. Furthermore, at equivalent layers, our approach consistently surpasses FastV across a wide range of pruning rates, underscoring its effectiveness. This effectiveness stems from our inner-LLM merging method, which better preserves information rather than directly discarding it.

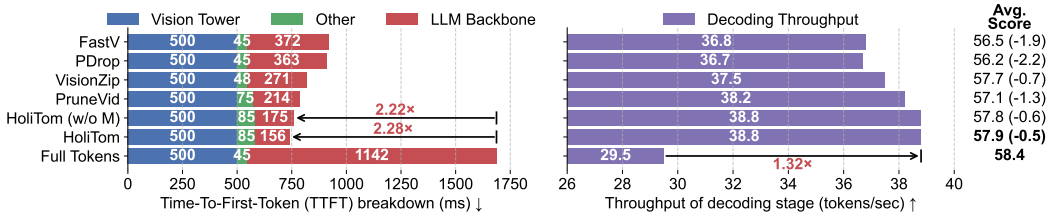


Figure 5: **Achieving superior inference.** "Other" indicates token pre-processing time (e.g., pooling). Our proposed method reduces Time-To-First-Token (TTFT) by 2.28× and achieves 1.32× higher decoding throughput, outperforming all other token compression methods and the vanilla model.

Table 4: **Ablation study on merging modules.** Our temporal merging module reduces FLOPs to 75.7% without performance loss, alleviates the performance degradation caused by aggressive spatial pruning. The integration of all 3 modules achieves the best performance-efficiency trade-off.

Method	Prefilling FLOPs (T) ↓	FLOPs Ratio ↓	Before LLM Retained Ratio	MVBench ↑	EgoSchema ↑	LongVideo Bench ↑	VideoMME ↑	Avg. ↑ Score	Avg. ↑ %
Vanilla	40.8	100%	100%	58.3	60.4	56.4	58.6	58.4	100
Only Temporal	30.9	75.7%	79%	**58.9**	60.5	56.5	**59.1**	**58.8**	**100.7**
Only Spatial	5.2	12.7%	15%	57.9	60.8	54.2	56.8	57.4	98.3
HoliTom (w/o M)	5.2	12.7%	15%	58.1	61.0	**57.0**	58.1	58.5	100.2
HoliTom	**4.3**	**10.5%**	15%	58.1	**61.2**	56.4	57.3	58.2	99.7

Performance Scaling with more frames. Our method scales performance robustly with increasing input frames (Fig. 4). A challenge for video LLMs is that uniformly sampled frames may miss crucial information required for accurate answers. Therefore, an effective token pruning method is essential to process more frames and capture sufficient context. Fig. 4 shows our approach consistently outperforms other compression methods across frame rates. At 16 frames, where less temporal redundancy exists, our method, while slightly below the vanilla, still outperforms all other compression techniques. With 64 frames, our method is more efficient and achieves superior performance over the vanilla model. Furthermore, when processing 128 frames, our token compression approach avoids the maximum context length limitations that bottleneck vanilla models. This capability is particularly beneficial for tasks that require an extensive temporal context or to answer complex questions with long text, resulting in improved performance.

Discussion: Improved Performance after Token Compression Tabs. 2, 4, and Fig. 4 present a key finding: models employing our token compression technique **outperform the original models** on various benchmarks. This surprising result underscores a fundamental principle for achieving superior performance at the input stage: *the value of key information over exhaustive information*. Excessive, irrelevant, or redundant data acts as noise, obscuring essential signals critical for effective processing. This information overload impedes the capacity of the model to accurately identify and process critical details, thereby degrading understanding and response generation. By providing a refined input that retains pertinent information while shedding redundant information, our compression method facilitates deeper comprehension and yields more accurate, relevant outputs. Collectively, these results underscore the efficacy of our technique in distilling key information and demonstrate that intelligent input refinement is crucial for superior model performance.

4.3 Efficiency Results

Fig. 5 summarizes the impact of various token compression methods on the inference efficiency of video LLMs. As shown, all the evaluated methods demonstrably reduce LLM prefilling time. Our method, in particular, reduces it to just 13.7% of the original. For VisionZip [57], PruneVid [17], and our *HoliTom* require token pre-processing, which introduces additional "other" time. Furthermore, both PruneVid and our method produce a variable number of tokens per frame, complicating batch processing, which contributes to extra overhead. Our method, designed to maximize temporal redundancy pruning, leads to finer-grained segmentation, further influencing this observation. Nevertheless, our method achieves the maximum reduction in Time-To-First-Token latency, reducing it by 2.28×, while maintaining optimal performance. Although we did not specifically optimize for decoding, our model's decoding speed still benefits from the reduced number of vision tokens. Our throughput increased by 1.32× compared to the original model, the highest among all methods evaluated.

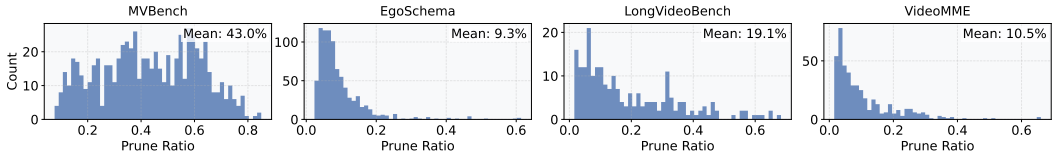


Figure 6: **Histogram of temporal pruning rates across four benchmarks** ($\tau = 0.80$). The average pruning ratio for each benchmark is annotated in the top right. MVBench (16s duration) exhibits the highest ratio, reflecting greater temporal redundancy, while EgoSchema is the least.

Table 5: **Ablation study on video segmentation methods.** This table compares different video segmentation strategies: Fixed-interval segmentation partitions the video at equal intervals; DySeg adaptively segments based on transition similarity; and our proposed global redundancy-aware segmentation.

Methods	MVBench	EgoSchema	LongVideo Bench	VideoMME	Avg.
Fixed-interval	**57.0**	60.9	53.8	56.4	57.0
DySeg [37]	56.8	60.8	54.1	56.6	57.1
HoliTom (w/o M)	56.9	**61.1**	**56.5**	**56.9**	**57.8**

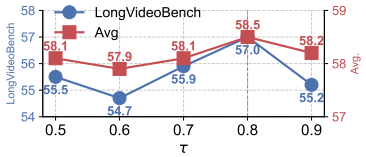


Figure 7: **Ablation study on τ.** Performance of our method is analyzed with varying τ at a target before LLM retained ratio of 15%.

4.4 Ablation Study

Ablation study on merging modules. Tab. 4 provides a detailed ablation study on the contribution of our proposed merging modules. We first evaluated the temporal merging module ($\tau = 0.8$), designed to eliminate temporal redundancy, which demonstrated efficiency gains while preserving performance. Across the four benchmarks, our method achieved 100.7% of the baseline performance while reducing FLOPs to 75.7%. Note that the reported average pruning rate is calculated over four datasets. The average pruning rate varied across datasets is illustrated in Fig. 6. For instance, MVBench(16s), with its shortest duration, exhibits the highest temporal redundancy, allowing approximately 43% pruning, whereas EgoSchema contains the least, permitting only about 9.3%. We then investigate combining temporal with spatial pruning. Applying our temporal pruning method significantly mitigates the performance degradation typically associated with aggressive spatial pruning alone. Furthermore, incorporating the inner merging module allowed us to push the efficiency boundaries even further, ultimately retaining 99.7% performance with a mere 10.5% of the original FLOPs.

Ablation study on temporal segmentation method. Tab. 5 compares different temporal segmentation methods. Fixed-interval Segmentation generates 8 segments with an interval of 4. DySeg [37] selects segment start points using the 8 largest inter-frame differences and includes frames below a 0.90 similarity threshold. Our proposed global redundancy-aware segmentation maximally leverages spatial redundancy and achieves a better performance.

Ablation study on τ. The hyperparameter τ controls the sensitivity of the temporal pruning mechanism, with lower values leading to more aggressive pruning. For a fixed retained ratio, τ also governs the balance between the amount of spatial and temporal pruning applied. Fig. 7, demonstrates the easy tunability of τ, with peak performance observed around $\tau = 0.8$. This value is adopted uniformly without performance degradation, as shown in Tab. 4. For a 10% pruning target, we set $\tau = 0.65$ to mitigate performance degradation from aggressive spatial pruning.

5 Conclusion

This paper presents *HoliTom*, a new training-free holistic token merging framework for boosting the efficiency of video LLMs by effectively handling redundant visual tokens. *HoliTom* achieves this through a synergistic integration of outer-LLM spatio-temporal reduction, drastically reducing initial token counts, and a robust inner-LLM token merging mechanism tailored for compatibility and further optimization. Evaluated on prominent video LLMs, *HoliTom* achieves a state-of-the-art efficiency-performance trade-off, substantially reducing computational costs (e.g., to 6.9% FLOPs) while preserving high performance (e.g., 99.1% accuracy), and accelerating inference (2.28× TTFT, 1.32× throughput). These results underscore the effectiveness of *HoliTom* in enabling practical and efficient video LLMs inference for complex, long-form video understanding.

Acknowledgment

This paper is supported by the Young Scientists Fund of the National Natural Science Foundation of China (No. 62506305), Zhejiang Leading Innovative and Entrepreneur Team Introduction Program (No. 2024R01007), Key Research and Development Program of Zhejiang Province (No. 2025C01026), and Scientific Research Project of Westlake University (No. WU2025WF003).

References

[1] Kirolos Ataallah, Xiaoqian Shen, Eslam Abdelrahman, Essam Sleiman, Deyao Zhu, Jian Ding, and Mohamed Elhoseiny. Minigpt4-video: Advancing multimodal llms for video understanding with interleaved visual-textual tokens. *arXiv preprint arXiv:2404.03413*, 2024.

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

[3] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[4] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *ICLR*, 2023.

[5] Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. Videollm-online: Online video large language model for streaming video. In *CVPR*, 2024.

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

[7] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024.

[8] Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, et al. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv preprint arXiv:2406.07476*, 2024.

[9] Rohan Choudhury, Guanglei Zhu, Sihan Liu, Koichiro Niinuma, Kris Kitani, and László Jeni. Don't look twice: Faster video transformers with run-length tokenization. In *NeurIPS*, 2024.

[10] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *ICLR*, 2024.

[11] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, 2022.

[12] Mingjing Du, Shifei Ding, and Hongjie Jia. Study on density peaks clustering based on k-nearest neighbors and principal component analysis. *Knowledge-Based Systems*, 99:135–145, 2016.

[13] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training compression for generative pretrained transformers. In *ICLR*, 2023.

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

[15] Zhangwei Gao, Zhe Chen, Erfei Cui, Yiming Ren, Weiyun Wang, Jinguo Zhu, Hao Tian, Shenglong Ye, Junjun He, Xizhou Zhu, et al. Mini-internvl: a flexible-transfer pocket multimodal model with 5% parameters and 90% performance. *Visual Intelligence*, 2(1):32, 2024.

[16] Wei Huang, Xingyu Zheng, Xudong Ma, Haotong Qin, Chengtao Lv, Hong Chen, Jie Luo, Xiaojuan Qi, Xianglong Liu, and Michele Magno. An empirical study of llama3 quantization: From llms to mllms. *Visual Intelligence*, 2(1):36, 2024.

[17] Xiaohu Huang, Hao Zhou, and Kai Han. Prunevid: Visual token pruning for efficient video large language models. *arXiv preprint arXiv:2412.16117*, 2024.

[18] Peng Jin, Ryuichi Takanobu, Wancai Zhang, Xiaochun Cao, and Li Yuan. Chat-univi: Unified visual representation empowers large language models with image and video understanding. In *CVPR*, 2024.

[19] Yang Jin, Zhicheng Sun, Kun Xu, Liwei Chen, Hao Jiang, Quzhe Huang, Chengru Song, Yuliang Liu, Di Zhang, Yang Song, et al. Video-lavit: Unified video-language pre-training with decoupled visual-motional tokenization. In *ICML*, 2024.

[20] Bo Li, Peiyuan Zhang, Kaichen Zhang, Fanyi Pu, Xinrun Du, Yuhao Dong, Haotian Liu, Yuanhan Zhang, Ge Zhang, Chunyuan Li, and Ziwei Liu. Lmms-eval: Accelerating the development of large multimoal models, 2024.

[21] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

[22] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023.

[23] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding. *arXiv preprint arXiv:2305.06355*, 2023.

[24] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *CVPR*, 2024.

[25] Xinhao Li, Yi Wang, Jiashuo Yu, Xiangyu Zeng, Yuhan Zhu, Haian Huang, Jianfei Gao, Kunchang Li, Yinan He, Chenting Wang, et al. Videochat-flash: Hierarchical compression for long-context video modeling. *arXiv preprint arXiv:2501.00574*, 2024.

[26] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *ECCV*, 2024.

[27] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. In *MLSys*, 2024.

[28] Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. Vila: On pre-training for visual language models. In *CVPR*, 2024.

[29] Zhijian Liu, Ligeng Zhu, Baifeng Shi, Zhuoyang Zhang, Yuming Lou, Shang Yang, Haocheng Xi, Shiyi Cao, Yuxian Gu, Dacheng Li, et al. Nvila: Efficient frontier visual language models. *arXiv preprint arXiv:2412.04468*, 2024.

[30] Yongdong Luo, Wang Chen, Xiawu Zheng, Weizhong Huang, Shukang Yin, Haojia Lin, Chaoyou Fu, Jinfa Huang, Jiayi Ji, Jiebo Luo, et al. Quota: Query-oriented token assignment via cot query decouple for long video comprehension. *arXiv preprint arXiv:2503.08689*, 2025.

[31] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. In *NeurIPS*, 2023.

[32] Rui Qian, Shuangrui Ding, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Dahua Lin, and Jiaqi Wang. Dispider: Enabling video llms with active real-time interaction via disentangled perception, decision, and reaction. In *CVPR*, 2025.

[33] Rui Qian, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Shuangrui Ding, Dahua Lin, and Jiaqi Wang. Streaming long video understanding with large language models. In *NeurIPS*, 2024.

[34] Shuhuai Ren, Sishuo Chen, Shicheng Li, Xu Sun, and Lu Hou. Testa: Temporal-spatial token aggregation for long-form video-language understanding. In *EMNLP*, 2023.

[35] Alex Rodriguez and Alessandro Laio. Clustering by fast search and find of density peaks. *science*, 344(6191):1492–1496, 2014.

[36] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *arXiv preprint arXiv:2403.15388*, 2024.

[37] Leqi Shen, Guoqiang Gong, Tao He, Yifeng Zhang, Pengzhang Liu, Sicheng Zhao, and Guiguang Ding. Fastvid: Dynamic density pruning for fast video large language models. *arXiv preprint arXiv:2503.11187*, 2025.

[38] Leqi Shen, Tianxiang Hao, Sicheng Zhao, Yifeng Zhang, Pengzhang Liu, Yongjun Bao, and Guiguang Ding. Tempme: Video temporal token merging for efficient text-video retrieval. In *ICLR*, 2025.

[39] Yan Shu, Zheng Liu, Peitian Zhang, Minghao Qin, Junjie Zhou, Zhengyang Liang, Tiejun Huang, and Bo Zhao. Video-xl: Extra-long vision language model for hour-scale video understanding. In *CVPR*, 2025.

[40] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, et al. Moviechat: From dense token to sparse memory for long video understanding. In *CVPR*, 2024.

[41] Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. Dycoke: Dynamic compression of tokens for fast video large language models. In *CVPR*, 2025.

[42] Keda Tao, Haoxuan You, Yang Sui, Can Qin, and Huan Wang. Plug-and-play 1. x-bit kv cache quantization for video large language models. *arXiv preprint arXiv:2503.16257*, 2025.

[43] Xiaoguang Tu, Zhi He, Yi Huang, Zhi-Hao Zhang, Ming Yang, and Jian Zhao. An overview of large ai models and their applications. *Visual Intelligence*, 2(1):34, 2024.

[44] Ao Wang, Fengyuan Sun, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. [cls] token tells everything needed for training-free efficient mllms. *arXiv preprint arXiv:2412.05819*, 2024.

[45] Haicheng Wang, Zhemeng Yu, Gabriele Spadaro, Chen Ju, Victor Quétu, Shuai Xiao, and Enzo Tartaglione. Folder: Accelerating multi-modal large language models with enhanced performance. *arXiv preprint arXiv:2501.02430*, 2025.

[46] Hanzhen Wang, Jiaming Xu, Jiayi Pan, Yongkang Zhou, and Guohao Dai. Specprune-vla: Accelerating vision-language-action models via action-aware self-speculative pruning. *arXiv preprint arXiv:2509.05614*, 2025.

[47] Jiawei Wang, Liping Yuan, Yuchen Zhang, and Haomiao Sun. Tarsier: Recipes for training and evaluating large video description models. *arXiv preprint arXiv:2407.00634*, 2024.

[48] 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. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[49] Zhenhailong Wang, Senthil Purushwalkam, Caiming Xiong, Silvio Savarese, Heng Ji, and Ran Xu. Dymu: Dynamic merging and virtual unmerging for efficient vlms. *arXiv preprint arXiv:2504.17040*, 2025.

[50] Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, et al. Streamvln: Streaming vision-and-language navigation via slowfast context modeling. *arXiv preprint arXiv:2507.05240*, 2025.

[51] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. In *NeurIPS*, 2024.

[52] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *ICML*, 2023.

[53] Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, et al. Pyramiddrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. In *CVPR*, 2025.

[54] Jiaming Xu, Jiayi Pan, Yongkang Zhou, Siming Chen, Jinhao Li, Yaoxiu Lian, Junyi Wu, and Guohao Dai. Specee: Accelerating large language model inference with speculative early exiting. In *ISCA*, 2025.

[55] Lin Xu, Yilin Zhao, Daquan Zhou, Zhijie Lin, See Kiong Ng, and Jiashi Feng. Pllava: Parameter-free llava extension from images to videos for video dense captioning. *arXiv preprint arXiv:2404.16994*, 2024.

[56] Cheng Yang, Yang Sui, Jinqi Xiao, Lingyi Huang, Yu Gong, Chendi Li, Jinghua Yan, Yu Bai, Ponnuswamy Sadayappan, Xia Hu, et al. Topv: Compatible token pruning with inference time optimization for fast and low-memory multimodal vision language model. In *CVPR*, 2025.

[57] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. In *CVPR*, 2025.

[58] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

[59] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. In *EMNLP*, 2023.

[60] Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. Lmms-eval: Reality check on the evaluation of large multimodal models, 2024.

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

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

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly outline the problem, propose a new solution combining external and internal pruning, and list key contributions that align with the methods and goals described in the text.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Appendix D details the limitations of our current method and outlines future research directions.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [NA]

Justification: No formal theorem or proposition is provided in the main manuscript.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We have made our best efforts to ensure the reproducibility of our method. The algorithmic details are provided in Section 3, and the experimental setup and procedures are detailed in Section 4.1 and Appendix A.

   Guidelines:

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

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

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

Answer: [No]

Justification: The datasets are from publicly available online sources. We guarantee the code will be open-sourced after the review period.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: The experimental details, including the model structure, hyperparameters, and comparison methods, have been elaborated in detail in Section 4 and Appendix A.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: For evaluation, we did not report error bars, consistent with previous work in this area. Due to the computational cost, we cannot replicate the experiments multiple times. This said, we have evaluated our method on multiple datasets and models (see Tab. 2, 3) to confirm the performance robustness of our method.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: In Section 4.1, we have detailed the computational resources required for our evaluation and inference tests.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We believe we have followed the Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: As shown in appendix E, the societal impact of our method has been fully discussed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: Our method is training-free and plug-and-play, which means this is no risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We have cited all datasets and baseline models used in our experiment.

    Guidelines:

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

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

13. **New assets**

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

    Answer: [NA]

    Justification: Our work does not provide new assets.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: Our work does not involve any crowd sourcing or experiments with human subjects.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: Our work does not involve experiments with human subjects.

    Guidelines:

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

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

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: The method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

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

Table 6: **Comparison of state-of-the-art methods on Qwen2.5-VL 7B.** Qwen2.5-VL accepts video input at 2 fps, capped at a maximum of 768 frames, maximum video token limit to 24,576 (as the technical report recommends). A/B in the # Token column indicates that A tokens are first provided to the LLM, and then compressed to B tokens during the LLM forward pass. The token number is derived from the average video token count per task within VideoMME. FastV performs full attention matrix calculation in memory, causing OOM errors due to the large number of video tokens.

Method	# Token ↓		TFLOPs ↓		VideoMME ↑				
					Short	Medium	Long	Overall	
Qwen2.5-VL-7B	18442	100%	377.2	100%	77.4	68.1	55.6	67.0	100%
FastV [6]	18442 / 9221	100% / 50.0%	170.4	45.2%			OOM		
Visionzip [57]	9221	50.0%	154.5	41.0%	**74.9**	**66.6**	55.7	65.7	98.1%
PruneVid [17]	9173	49.7%	153.5	40.7%	72.3	64.8	54.7	63.9	95.4%
HoliTom (w/o M)	**6513**	**34.9%**	**102.0**	**27.0%**	74.4	66.4	**56.4**	**65.8**	**98.2%**
FastV [6]	18442 / 4610	100% / 25.0%	90.7	24.0%			OOM		
Visionzip [57]	4610	25.0%	68.7	18.2%	**73.1**	63.3	55.9	64.1	95.7%
PruneVid [17]	4632	25.1%	69.1	18.3%	69.3	61.1	53.2	61.2	91.3%
HoliTom (w/o M)	**4504**	**24.4%**	**66.9**	**17.7%**	72.7	**65.7**	**56.1**	**64.8**	**96.7%**

A Supplemental Implementation Details

Our method is implemented on the LLaVA-OneVision-7B/72B [21] and LLaVA-Video-7B [62] models. Evaluation utilized NVIDIA A100 (80GB) GPUs; inference was performed on an NVIDIA RTX A6000 GPU. To ensure a fair comparison of computational cost, we used total prefilling FLOPs as the primary metric. Baselines are configured for comparable FLOPs: FastV [6] prunes 80% of tokens at layer 2; PDrop [53] retains 50%, 25%, and 12.5% of vision tokens at layers 2, 7, and 14, respectively; VisionZip [57] and PruneVid [17] maintain a consistent proportion of input tokens with our method. Performance results for FastVID [37] are adopted directly from their original paper. For our proposed method, the default threshold τ is 0.8. In the specific experiments conducted on Qwen2.5-VL [3] with a maximum sampling of 768 frames, a lower threshold of $\tau = 0.2$ was used. This adjustment accounts for the higher temporal redundancy present when sampling is dense. In experiments targeting a 10% compression ratio, τ was set to 0.65. Experimental setups include pruning K=18 layers of the 7B model and K=60 layers of the 72B model, both at a ratio of R=50%. Following the official LLaVA-OneVision specifications, the default input video frames are 32 and $N_v = 196$. For LLaVA-Video, the default input consisted of 64 video frames with $N_v = 169$. All benchmark evaluations are performed using the LMMs-Eval [60, 20].

B Supplemental Experimental Results

B.1 Experiments on Qwen2.5-VL with High Frame Sampling

Existing models like LLaVA-OV [21] and LLaVA-Video [62] utilize a fixed input of 32/64 video frames, each resized to a static resolution (Tab. 2, 3). In contrast, frontier models, such as Qwen2.5-VL [3], introduce advanced features including FPS frame sampling, which extend input sequences (up to 768 frames), and dynamic resolution support. These new capabilities pose new challenges to existing token compression methods in maintaining performance for video understanding tasks.

As shown in Tab. 6, HoliTom surpasses state-of-the-art methods across both token compression rates, especially for long videos. Due to Out-of-Memory (OOM) issues arising from the full attention matrix calculation, we were unable to report results for FastV and the inner-LLM merging execution.

B.2 Impact of Token Compression on Fine-Grained Object Understanding

HoliTom applies aggressive video token compression. Does this aggressive compression impair the model's ability to comprehend fine-grained details? Tab. 7 presents the performance results on selected subtasks from the MVBench benchmark (originally detailed in Table 2).

Table 7: **Fine-grained object tasks.** Performance on MVBench object subtasks (object existence (OE), object interaction (OI), and object shuffle (OS)) improves with more aggressive token compression.

Method	FLOPs (%)	MVBench			
		OE	OI	OS	Overall
OV 7B	100	57.5	84.0	35.5	58.3
+HoliTom	17.4	61.0	83.5	36.5	58.4
+HoliTom	14.2	**63.0**	84.0	37.5	**58.7**
+HoliTom	10.5	60.5	**84.5**	**38.0**	58.1

Table 8: **Improved performance with enhanced efficiency.** Using more input frames, token compression boosts performance while controlling computational overhead.

Method	# Frame	FLOPs (%)	Avg. Score
OV 7B	32	100	58.4
+HoliTom	32	12.7	58.5
+HoliTom	64	26.5	58.9
+HoliTom	128	56.6	59.3

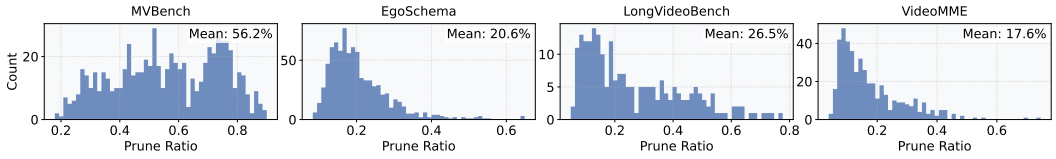


Figure 8: **Histogram of temporal pruning rates across four benchmarks** ($\tau = 0.65$). The average pruning ratio for each benchmark is annotated in the top right. MVBench (16s duration) exhibits the highest ratio, reflecting greater temporal redundancy, while VideoMME is the least ($\tau = 0.65$).

At compression rates ranging from 10% to 25%, HoliTom demonstrates increases in performance. This result would be improbable if HoliTom discards critical information about small objects. Instead, this outcome provides evidence of HoliTom's robust ability to retain fine-grained details.

B.3 Enhanced Performance with Reduced Overhead

Tab. 8 extends the findings presented in Fig. 4. By sampling more frames with HoliTom while maintaining constant or even reduced total FLOPs, we achieve better performance compared to a vanilla model operating on fewer frames. We also observe that as the number of input frames increases, the computational overhead contributed by the vision encoder becomes a non-negligible factor. This performance and efficiency trade-off is further illustrated in Fig. 5 of our paper.

B.4 Supplemental Ablation Study on τ

In section 4.4, we discussed the selection of τ ($\tau = 0.8$) and the corresponding histogram of temporal pruning rates on four benchmarks for a retain ratio of 15%. Next, we detail the selection of the hyperparameter τ for a 10% retain ratio and present the corresponding histogram. As illustrated in Fig. 9, peak performance is observed around $\tau = 0.65$. The Fig. 8 presents the histogram of temporal pruning rates on the four datasets when $\tau = 0.65$. It is evident that controlling τ regulates the aggressiveness of temporal pruning; a larger τ results in more aggressive pruning.

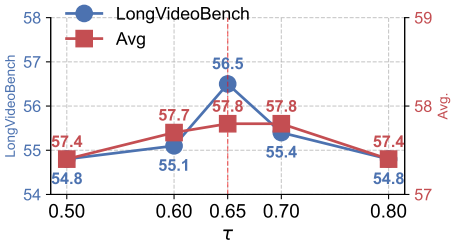


Figure 9: **Ablation study on τ.** Performance of our method is analyzed with varying τ at a target before LLM retained ratio of 10%.

B.5 Ablation Study on Merge Strategy

The core of spatio-temporal merging in HoliTom is a hybrid strategy that integrates attention-guided compression with similarity-guided clustering (DPC-KNN). As shown in Tab. 9, our mixed strategy achieves the best performance. The key to this lies in distinguishing between the two token types encountered during the merging process. For non-redundant tokens (within a single frame), the encoder's self-attention scores are *excellent priors*, making an attention-guided compression strategy highly effective. However, for redundant tokens (formed by merging tokens from adjacent frames), the original single-frame self-attention scores lack a theoretical basis as a merging metric. Therefore,

Table 9: **Ablation Study on Merge Strategy.** Our mixed strategy achieves the best performance.

Method	MVBench	EgoSchema	LongVideoBench	VideoMME	Avg. Score
Attention	**58.4**	**60.9**	55.9	57.4	58.1
DPC-KNN	57.4	59.6	53.9	56.6	56.9
HoliTom	**58.4**	**60.9**	**56.2**	**58.3**	**58.5**

DPC-KNN clustering is adopted to group these merged tokens based on feature similarity, which provides a more principled and effective approach in this specific spatio-temporal context.

C Compatible with Flash Attention

Our approach introduces two distinct merging strategies: inner-LLM and outer-LLM. The inner-LLM strategy, similar to prior work [6, 53, 41], is designed for integration with highly optimized attention implementations (e.g., Flash Attention [11, 10]). This requires obtaining attention scores from a specific layer *only once* during the prefilling stage, an operation introducing negligible computational overhead compared to total inference cost. In contrast, our outer-LLM merging strategy operates externally to the model, decoupled from the attention mechanisms of LLM.

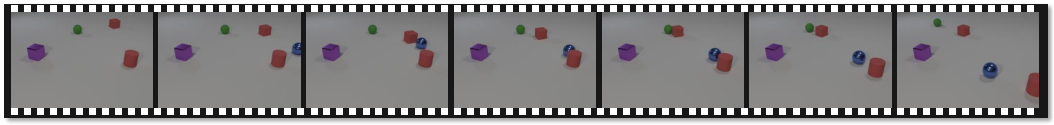
D Limitations and Future Work

While our work demonstrates an adequate acceleration of video LLMs by token merging for inference, it is important to outline its current limitations. First, the approach is primarily designed for fixed-length video clips and does not natively support online, arbitrary-length streaming video input. This poses challenges for real-time processing [5, 33, 32] and maintaining long-term context understanding. Second, as shown in Fig. 5, similar to other methods [6, 53, 57, 41] in the token pruning area, our approach does not optimize the latency of the vision tower. Further work, such as quantization [27, 52, 13, 16], methods to accelerate the vision tower [9, 45, 49] and new application areas [50, 46, 54], is worth exploring for further optimization.

E Broader impacts

This work significantly enhances video LLM efficiency, addressing a key barrier to deployment and scalability. By reducing computational needs, it broadens access to advanced video AI, enabling wider application and fostering innovation.

F More Visualizations



User:	FLOPS	What color is the object that is stationary?	What direction is the green ball moving in?
LLaVA-OV-7B	40.8T	The object that is stationary is **purple**.	The green ball moves towards the **right** side of the frame.
FastV	9.3T	The object that is stationary is **purple**.	The green ball moves towards the **right** side of the frame.
VisionZip	8.7T	The object that is stationary is **green**.	The green ball is moving towards the **right** side of the frame.
HoliTom (w/o M)	8.7T	The object that is stationary is **purple**.	The green ball moves towards the **left** side of the frame.
HoliTom	7.1T	The object that is stationary is **purple**.	The green ball moves towards the **left** side of the frame.

User:	FLOPS	Where is the hidden object at the end of the occlusion game from the camera's point of view?
LLaVA-OV-7B	40.8T	The hidden object is **under the black notebook**.
FastV	9.3T	The hidden object is **under the black notebook**.
VisionZip	8.7T	The hidden object is **under the black notebook**
HoliTom (w/o M)	8.7T	The hidden object is **under the middle cup.**
HoliTom	7.1T	The hidden object is **under the middle cup.**

***Top*: Dynamic Object Tracking**

***Bottom*: Memory; Motion Understanding**

Figure 10: **Comparison on Challenging Video Understanding.** Green: correct results, Red: incorrect results. Our method is able to produce correct answers on challenging video tasks.

User:	Describe what's happening in this video.
LLaVA-OV-7B 40.8 TFLOPS 100% Token	The video features a person standing on a stage with a dark background, wearing a black shirt and dark pants. A large hand appears from the top left corner, reaching towards the person's waist area. The text 'Anyone know what this pocket is for?' appears at the bottom of the screen, indicating curiosity about the purpose of a specific pocket on the person's clothing. The text changes to 'I've always wondered that' as the hand continues to reach out. Eventually, the hand reveals a small white device attached to the pocket, which is identified as 'this is the new iPod Nano 0'. The person holds up the iPod Nano 0, showing it to the audience. The video concludes with a close-up of the person holding the iPod Nano 0, emphasizing its design and features.
HoliTom 4.3 TFLOPS 15% Token	The video features a person standing on a stage, dressed in a dark outfit. A large hand appears from the background, reaching towards the person. The text 'I've always wondered' (I've always wondered) is displayed at the top left corner of the screen, suggesting a sense of curiosity or anticipation. As the hand moves closer, the text changes to 'well now we know' (well now we know), indicating a revelation or discovery. The hand then reveals a small device labeled 'this new iPod Nano' (this new iPod Nano), which is white with a circular button and a small screen. The text 'iPod Nano' (iPod Nano) appears below the device, confirming its identity. The video concludes with a close-up of the person holding the iPod Nano, emphasizing the significance of the new product.

Figure 11: **Qualitative generation comparison.** Green indicates correctly detailed descriptions. Our method achieves high-quality, accurate text generation even when retaining only 15% of input tokens.