COMPVID: WHICH VIDEO REGIONS DESERVE MORE TOKENS? COMPLEXITY-AWARE TOKEN COMPRESSION FOR VIDEO LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing Video Large Language Model (VLLM) token pruning methods typically perform importance- or diversity-based visual token compression at the video, segment, or frame level. However, they overlook an inherent property of video data: video complexity is distributed unevenly across temporal and spatial dimensions, leading to suboptimal token allocation among different spatiotemporal regions. Through empirical analysis, we find that allocating tokens to higher-complexity regions yields larger logit margin gains and contributes to improving video understanding capabilities. Inspired by this, we propose a new perspective: regional complexity determines how much representation budget a region requires, whereas dimension-wise complexity determines how each region should be partitioned. Building on this insight, we develop **CompVID**, a training-free framework for complexity-aware token allocation. Specifically, **CompVID** utilizes dimension-wise singular value decomposition (SVD) to expose the rank structure along each dimension and quantify its intrinsic complexity through singular-value decay. Based on these complexity estimates, we introduce **SVDtree**, an SVD-guided adaptive partition tree that recursively refines the video feature volume along the complexity-dominant dimension and allocates token budgets to the resulting regions according to their regional complexity. Evaluations on **4** VLLMs across **7** video understanding benchmarks show that **CompVID** achieves a state-of-the-art efficiency-accuracy trade-off. By pruning **90%** of tokens on LLaVA-OneVision-7B, **CompVID** achieves an **11.8**× reduction in TFLOPs and accelerates the LLM prefill stage by **3.7**× while preserving **99.7%** of the full-token performance.

1 INTRODUCTION

Video Large Language Models (VLLMs) (Li et al., 2025a; Zhang et al., 2024; Bai et al., 2025b; Bai et al., 2025a; Comanici et al., 2025) have significantly improved the capability of video understanding by integrating vision encoders with large language models. Despite their strong performance, the practical deployment of VLLMs remains challenging due to the massive computational cost (Chandrasegaran et al., 2024; Fu et al., 2024; Wu et al., 2024) induced by a large number of visual tokens. In particular, videos contain dense spatial and temporal information, leading to tens of thousands of tokens.

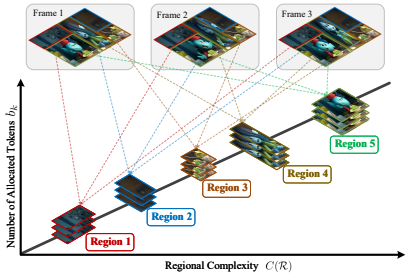


Figure 1: **Our perspective.** Regional token allocation should match local video complexity to balance token usage across video regions.

To improve inference efficiency, recent studies have explored VLLM token pruning methods, which can be broadly categorized into three groups. First, video-level methods (Fig. 4a) jointly reduce inter-frame and intra-frame redundancy over the entire video, typically selecting representative tokens based on importance and diversity (Fan et al., 2026; Cho et al., 2026; Li et al., 2026). Second, segment-level methods (Fig. 4b) partition a video into temporally coherent segments and perform token compression independently within each segment (Shen et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

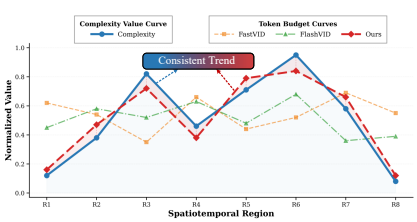


Figure 2: **Comparison of token allocation** across regions with different complexity levels, showing that our **CompVID** achieves better complexity–budget matching.

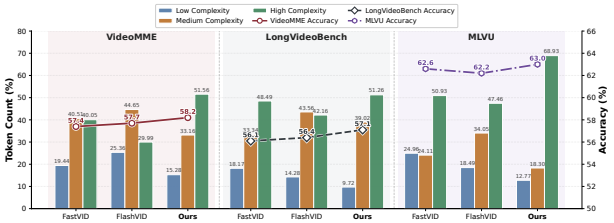
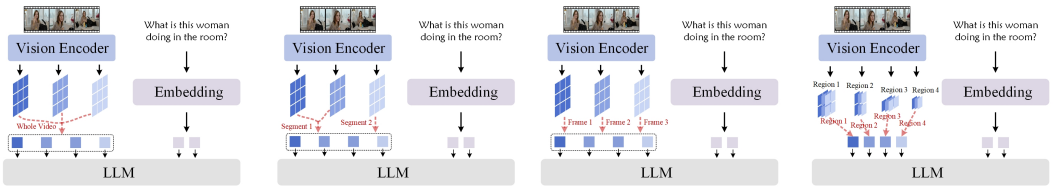


Figure 3: **Post-pruning token-level complexity distributions and accuracy** across VideoMME, LongVideoBench, and MLVU, showing that our **CompVID** allocates more tokens to complex regions and fewer to simple regions while achieving superior benchmark performance.



(a) Video-Level Pruning method (b) Segment-Level Pruning method (c) Frame-Level Pruning method (d) Our Region-Level Pruning method

Figure 4: **Comparison of before-LLM pruning methods.** (a) Video-level methods select representative tokens from the entire video; (b) Segment-level methods compress temporally coherent segments independently; (c) Frame-level methods assign different compression ratios across frames; (d) our region-level method partitions video features into local spatiotemporal regions for token allocation.

2025; Yuan et al., 2026). Third, frame-level methods (Fig. 4c) identify keyframes to assign different compression ratios to individual frames. In addition to these before-LLM compression strategies, several approaches further perform token reduction inside the LLM, progressively pruning redundant visual tokens during inference (Huang et al., 2024; Shao et al., 2025; Tao et al., 2025). Despite their different pruning strategies, these methods generally operate at coarse compression levels, overlooking the varying representation demands of local spatiotemporal regions.

This limitation is particularly important for video data, where complexity is unevenly distributed across spatiotemporal regions (Duanmu et al., 2017). As illustrated in Fig. 2, we uniformly partition the video feature volume into eight spatiotemporal regions with different complexity levels. Existing pruning methods yield token allocations that deviate from the complexity distribution, assigning fewer tokens to complex regions while preserving excessive tokens in relatively simple ones. Such complexity-budget mismatch motivates a question: *how much token budget should each spatiotemporal region receive according to its representation demand?* We argue that regional token allocation should match local video complexity to better balance token usage across video regions (Fig. 1).

To address this challenge, we introduce **CompVID**, a training-free framework for complexity-aware video token allocation (Fig. 4d). The key principle of **CompVID** is to combine two decisions: *how much representation a region requires* and *how to partition different regions*. To this end, we perform dimension-wise singular value decomposition (SVD) on each spatiotemporal region to derive regional and dimension complexity from the rank structure revealed by the singular-value distributions. Based on these complexity indicators, we introduce **SVDtree**, an SVD-guided hierarchical partitioning scheme that recursively refines video regions along their complexity-dominant dimensions and allocates token budgets according to regional complexity. Unlike prior pruning methods, **CompVID** performs complexity estimation and representation refinement jointly within the recursive partitioning process, allowing the token budget to adapt to local representation demand. As shown in Fig. 3, under a 10% retention ratio on LLaVA-OneVision-7B, **CompVID** preserves more tokens in high-complexity regions while reducing token expenditure on low-complexity regions. The complexity-level categorization is detailed in Appendix B. Compared with FastVID and FlashVID, **CompVID** shows stronger correspondence between complexity and token distribution while achieving higher accuracy, demonstrating more effective use of visual tokens.

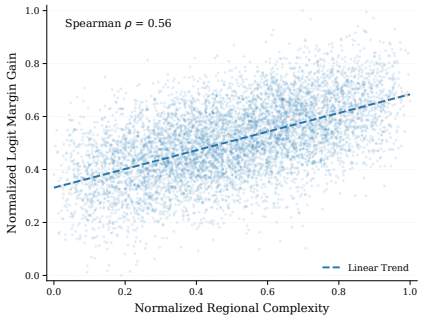


Figure 5: Visualization of the correlation between regional complexity and logit margin gain.

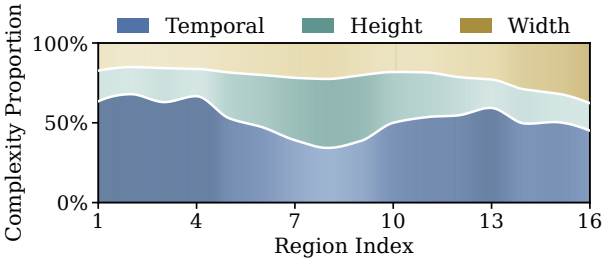


Figure 6: Visualization of complexity distributions across temporal, height, and width dimensions in different video regions.

- We identify a previously overlooked limitation of existing VLLM token pruning methods: they overlook the uneven complexity distribution across spatiotemporal regions, resulting in mismatched token allocation between complex and simple video content.
- We present **`CompVID`**, a training-free acceleration framework built on **SVDtree**, an SVD-guided hierarchical partitioning scheme that derives regional and dimension-wise complexity from singular-value distributions to jointly guide adaptive region partitioning and token allocation, concentrating more visual tokens on complex regions while reducing unnecessary token expenditure on simple regions.
- Extensive experiments on **4** VLLMs across **7** benchmarks demonstrate that **`CompVID`** achieves state-of-the-art efficiency-performance trade-off under varying token budgets. Further analyses validate the effectiveness of complexity-aware token allocation.

2 PRELIMINARY

Notation. Given a video $\mathcal{V}$ with T sampled frames, a frozen vision encoder produces a spatiotemporal feature volume $\mathbf{X} \in \mathbb{R}^{T\times H\times W\times D}$, where H and W denote spatial dimensions and D the feature dimension. The volume contains $N = THW$ visual tokens. We denote a local spatiotemporal region by $\mathcal{R}$ and the partition by $\mathcal{P} = \{\mathcal{R}_1, \ldots, \mathcal{R}_K\}$. Each region $\mathcal{R}_k$ is assigned a token budget b_k under a global budget $B = \sum_{k=1}^{K} b_k$. The compressed visual tokens are fed into the frozen VLLM.

Problem Formalization. Existing token pruning generally focuses on overall redundancy reduction, while overlooking representation demands across spatiotemporal regions. We therefore formulate video token compression as a joint *region partitioning and token allocation* problem:

$$(\mathcal{P}^*, \mathbf{b}^*) = \arg\max_{\mathcal{P},\mathbf{b}} \mathcal{U}\big(\widetilde{\mathbf{X}}(\mathcal{P}, \mathbf{b})\big), \qquad \text{s.t.} \quad \sum_{\mathcal{R}_k\in\mathcal{P}} b_k \le B. \tag{1}$$

where $\mathcal{P}^*$ denotes the optimal partition, $\mathbf{b}^* = (b_1^*, \ldots, b_K^*)$ denotes the corresponding token allocation, $\widetilde{\mathbf{X}}(\mathcal{P}, \mathbf{b})$ denotes the compressed visual tokens obtained under partition $\mathcal{P}$ and token allocation $\mathbf{b}$, and $\mathcal{U}(\cdot)$ measures the information preserved by the compressed visual tokens. To characterize regional representation demand, we define *regional complexity* $C(\mathcal{R})$ and *dimension-wise complexity* $\mathbf{c}(\mathcal{R}) = [c_T(\mathcal{R}), c_H(\mathcal{R}), c_W(\mathcal{R})]$. The former determines the token budget, whereas the latter determines the partition dimension, given by $d^*(\mathcal{R}) = \arg\max_{d\in\{T,H,W\}} c_d(\mathcal{R})$.

3 MOTIVATION

Existing before-LLM video token compression methods typically operate at the video, segment, or frame level. Such coarse-grained compression fails to reflect the varying representation demands of local regions. In this study, we identify two critical factors overlooked by existing methods: region-wise complexity differences and dimension-wise complexity differences. Modeling these factors is crucial, as they jointly determine regional token allocation and the direction of local representation refinement.

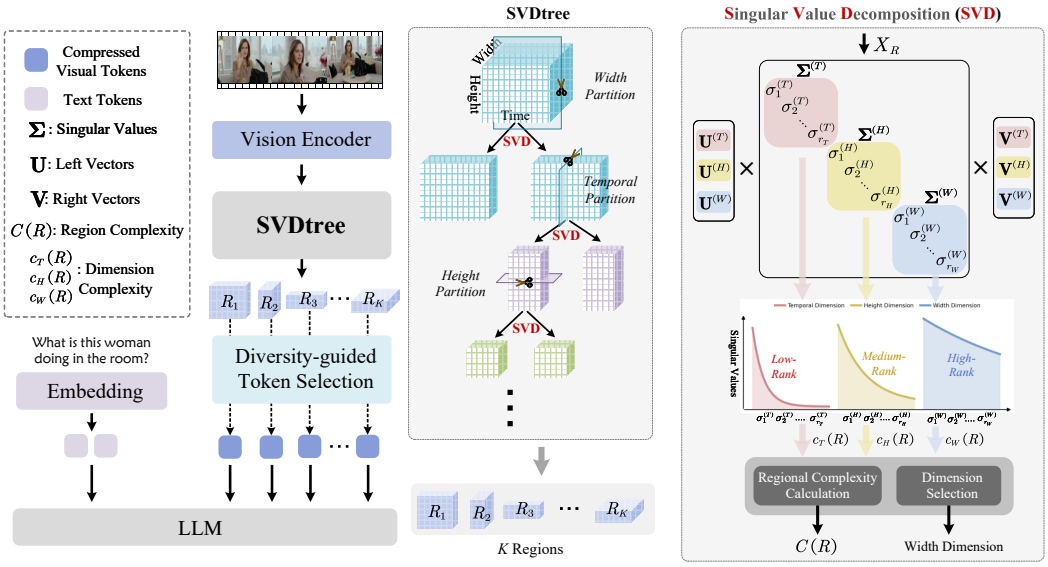


Figure 7: **Overview of CompVID.** (1) Dimension-wise SVD quantifies regional and dimension-wise complexity. (2) SVDtree recursively partitions the feature volume along complexity-dominant dimensions. (3) Diversity-guided token selection retains representative tokens for each region.

Observation (i): Regional complexity reflects different representation demands. To examine whether different regions benefit from additional representation, we compare normalized regional complexity with normalized logit margin gain. As shown in Fig. 5, the two exhibit a clear positive correlation, with a Spearman coefficient of $\rho = 0.56$. This suggests that complex regions have greater representation demands, whereas simpler regions can be represented with fewer tokens. Details of the calculation for Fig. 5 are provided in Appendix E.2.

Insight (i): Regional token budgets should adapt to regional complexity, assigning more tokens to complex regions and fewer to simple ones.

Observation (ii): Dimension-wise complexity varies across local regions. We further decompose regional complexity along the temporal, height, and width dimensions. As shown in Fig. 6, their relative complexity proportions vary considerably across regions, and the complexity-dominant dimension changes with local video content. A fixed partition direction therefore cannot adapt to these region-specific variations and may result in suboptimal refinement of local video features.

Insight (ii): Each region should be partitioned along its complexity-dominant dimension, e.g., along the temporal dimension for motion-dominant regions and spatially for structurally richer regions.

4 Methodology

We aim to address the complexity–budget mismatch in existing video token compression. We present **CompVID** (Fig. 7), a training-free framework that adaptively allocates visual tokens according to the representation demand of different video regions. **CompVID** consists of two key components. First, dimension-wise singular value decomposition (SVD) characterizes the rank structure of each region and derives its regional and dimension-wise complexity (§4.2). Second, we propose SVDtree to recursively partition the video feature volume, where regional complexity controls regional token budget and dimension-wise complexity determines partition directions (§4.3).

4.1 CompVID: Complexity-Aware Video Token Allocation

Given the video feature volume $\mathbf{X} \in \mathbb{R}^{T\times H\times W\times D}$ and a global token budget B, **CompVID** progressively partitions $\mathbf{X}$ into local video regions. Each region $\mathcal{R}$ is characterized by regional complexity $C(\mathcal{R})$ and dimension-wise complexity $\mathbf{c}(\mathcal{R}) = [c_T(\mathcal{R}), c_H(\mathcal{R}), c_W(\mathcal{R})]$. The former reflects its representation demand, while the latter determines the partition direction.

The partitioning process is recursive. At each iteration, the region with the highest regional complexity is selected for refinement along its most complex dimension. Thus, complex regions are refined into finer subregions, whereas simple regions remain coarse. The recursion terminates when the number of regions reaches K. Given $\mathcal{P} = \mathcal{R}_1, \ldots, \mathcal{R}_K$, the global budget B is distributed according to regional complexity. Thus, dimension-wise complexity guides partitioning, while regional complexity determines token allocation.

4.2 Dimension-Wise SVD Complexity Estimation

Dimension-wise SVD. The key to complexity-aware allocation is to quantify both the complexity of a region and how this complexity varies across different dimensions. For a region $\mathcal{R}$ with feature volume $\mathbf{X}_\mathcal{R} \in \mathbb{R}^{T_\mathcal{R} \times H_\mathcal{R} \times W_\mathcal{R} \times D}$, we unfold it separately along the temporal, height, and width dimensions. For each dimension $d \in \{T, H, W\}$, $\mathbf{X}_\mathcal{R}$ is unfolded along d into a matrix $\mathbf{X}_\mathcal{R}^{(d)} \in \mathbb{R}^{n_d \times (T_\mathcal{R} H_\mathcal{R} W_\mathcal{R} D / n_d)}$, where $n_T = T_\mathcal{R}$, $n_H = H_\mathcal{R}$, and $n_W = W_\mathcal{R}$. The three dimension-wise SVDs can be jointly represented as:

$$\begin{bmatrix} \mathbf{X}_\mathcal{R}^{(T)} & 0 & 0 \\ 0 & \mathbf{X}_\mathcal{R}^{(H)} & 0 \\ 0 & 0 & \mathbf{X}_\mathcal{R}^{(W)} \end{bmatrix} = \begin{bmatrix} \mathbf{U}^{(T)} & 0 & 0 \\ 0 & \mathbf{U}^{(H)} & 0 \\ 0 & 0 & \mathbf{U}^{(W)} \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}^{(T)} & 0 & 0 \\ 0 & \mathbf{\Sigma}^{(H)} & 0 \\ 0 & 0 & \mathbf{\Sigma}^{(W)} \end{bmatrix} \begin{bmatrix} \mathbf{V}^{(T)\top} & 0 & 0 \\ 0 & \mathbf{V}^{(H)\top} & 0 \\ 0 & 0 & \mathbf{V}^{(W)\top} \end{bmatrix}. \tag{2}$$

where $\mathbf{U}^{(d)}$ and $\mathbf{V}^{(d)}$ contain the left and right singular vectors of $\mathbf{X}_\mathcal{R}^{(d)}$, respectively, and $\mathbf{\Sigma}^{(d)} = \mathrm{diag}(\sigma_1^{(d)}, \ldots, \sigma_{r_d}^{(d)})$ contains the corresponding singular values. In the unified representation, the temporal, height, and width components are organized as column blocks in $\mathbf{U}$, diagonal blocks in $\mathbf{\Sigma}$, and row blocks in $\mathbf{V}^\top$, respectively. For each dimension d, the singular values are ordered as $\sigma_1^{(d)} \geq \cdots \geq \sigma_{r_d}^{(d)}$. Rapid singular-value decay indicates an approximately low-rank and relatively simple structure along dimension d, whereas slow decay reflects a higher-rank structure and correspondingly greater complexity.

Dimension-wise complexity. To quantify the complexity along each dimension, we normalize the singular values within each block as $p_i^{(d)} = \sigma_i^{(d)} / \sum_{j=1}^{r_d} \sigma_j^{(d)}$ and define the dimension-wise complexity as

$$c_d(\mathcal{R}) = -\frac{1}{\log r_d} \sum_{i=1}^{r_d} p_i^{(d)} \log p_i^{(d)}, \qquad d \in \{T, H, W\}. \tag{3}$$

The resulting $\mathbf{c}(\mathcal{R}) = [c_T(\mathcal{R}), c_H(\mathcal{R}), c_W(\mathcal{R})]$ represents the complexity distribution across the temporal, height, and width dimensions. A larger $c_d(\mathcal{R})$ corresponds to slower singular-value decay and therefore indicates that dimension d requires finer partitioning.

Regional complexity. We further aggregate the dimension-wise complexity to obtain the regional complexity:

$$C(\mathcal{R}) = |\mathcal{R}| \cdot \frac{c_T(\mathcal{R}) + c_H(\mathcal{R}) + c_W(\mathcal{R})}{3}, \tag{4}$$

where $|\mathcal{R}| = T_\mathcal{R} H_\mathcal{R} W_\mathcal{R}$ denotes the number of visual tokens contained in $\mathcal{R}$. This formulation considers both regional complexity and region size, allowing complex regions to receive more tokens.

4.3 SVDTree

Building on *Insight (i)* and *Insight (ii)*, we propose SVDtree, which incorporates the SVD-derived complexity into recursive partitioning and token allocation. It dynamically decides both *which region to refine* and *along which dimension to partition it*. Upon reaching the predefined number of regions, the global token budget is further distributed according to their regional complexity.

Dimension-complexity-guided partitioning. We first use dimension-wise complexity to partition the video feature volume into K spatiotemporal regions. Let $\mathcal{L}$ denote the current set of leaf regions, initialized as $\mathcal{L} = \{\mathcal{R}_0\}$, where $\mathcal{R}_0$ covers the entire feature volume. For each region $\mathcal{R} \in \mathcal{L}$, its dominant dimension complexity is defined as $\max_{d \in \{T,H,W\}} c_d(\mathcal{R})$. At each iteration, we select

the region with the highest regional complexity and determine its partition direction according to its dimension-wise complexity:

$$\mathcal{R}^* = \arg\max_{\mathcal{R}\in\mathcal{L}} C(\mathcal{R}), \qquad d^* = \arg\max_{d\in\{T,H,W\}} c_d(\mathcal{R}^*). \tag{5}$$

To further determine where to partition along d^*, we construct the SVD-derived coordinates $\mathbf{Z}^{(d^*)} = \mathbf{U}^{(d^*)}\mathbf{\Sigma}^{(d^*)}$, where each row $\mathbf{z}_i^{(d^*)}$ represents the i-th position along the selected dimension in the SVD-derived feature space, with each component scaled by its corresponding singular value. We select the partition point with the largest difference between adjacent positions:

$$s^* = \arg\max_{1\le i<n_{d^*}} \left\| \mathbf{z}_{i+1}^{(d^*)} - \mathbf{z}_i^{(d^*)} \right\|_2. \tag{6}$$

where n_{d^*} denotes the number of positions of $\mathcal{R}^*$ along the selected dimension d^*. The selected region $\mathcal{R}^*$ is then split at s^* along d^* into two child regions, whose dimension-wise complexities are subsequently recomputed. The parent region is replaced by its children, and this recursive procedure continues until $|\mathcal{L}| = K$, yielding $\mathcal{P} = \{\mathcal{R}_1, \ldots, \mathcal{R}_K\}$. In this way, SVD determines both the partition direction and the partition point, progressively refining complex regions while preserving coarser partitions for simpler regions.

Regional-complexity-guided token allocation. Given K regions, we distribute the global token budget B according to their regional complexity. Specifically, the token budget assigned to region $\mathcal{R}_k$ is proportional to $C(\mathcal{R}_k)$:

$$b_k = B\frac{C(\mathcal{R}_k)}{\sum_{j=1}^{K} C(\mathcal{R}_j)}, \qquad k = 1, \ldots, K, \tag{7}$$

where the token budgets are discretized to integers with a residual adjustment such that $\sum_{k=1}^{K} b_k = B$. Consequently, regions with higher regional complexity receive more visual tokens, whereas simpler regions are assigned fewer tokens.

Diversity-guided token selection. Given the allocated budget b_k, we employ Max-Min Diversity (Alvar et al., 2025) to select representative tokens within each region $\mathcal{R}_k$. Specifically, we initialize the selected set $\mathcal{S}_k$ with the token closest to the regional feature centroid. We then iteratively select the token that maximizes its minimum distance to the already selected tokens:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}_i\in\mathcal{R}_k\setminus\mathcal{S}_k} \min_{\mathbf{x}_j\in\mathcal{S}_k} D(\mathbf{x}_i, \mathbf{x}_j), \tag{8}$$

where $D(\mathbf{x}_i, \mathbf{x}_j) = 1 - \mathrm{sim}(\mathbf{x}_i, \mathbf{x}_j)$ denotes the cosine distance between two visual tokens. The selected token $\mathbf{x}^*$ is added to $\mathcal{S}_k$, and the process is repeated until $|\mathcal{S}_k| = b_k$. By maximizing the minimum pairwise distance at each step, Max-Min Diversity encourages the retained tokens to cover distinct visual content while reducing redundancy within each region. Finally, the selected tokens from all regions are concatenated to form the compressed visual sequence $\widetilde{\mathbf{X}} \in \mathbb{R}^{B\times D}$.

5 Experiments

5.1 Experimental Settings

Benchmarks. We evaluate **`CompVID`** on **7** video understanding benchmarks: **VideoMME** [Fu et al., 2024] for diverse durations and domains, **LongVideoBench** [Wu et al., 2024] for long-form understanding, **MLVU** [Zhou et al., 2025] for multi-task understanding, **EgoSchema** [Mangalam et al., 2023] for ego-centric understanding, **MVBench** [Li et al., 2024] for temporal perception and reasoning, **ActivityNet-Grounding** [Caba Heilbron et al., 2015] for long-video grounding, and **VNBench** [Zhao et al., 2024] for fine-grained understanding. Details are provided in Appendix C.3.

Evaluation Setup. We evaluate **`CompVID`** on **4** VLLMs: **LLaVA-OneVision-7B** [Li et al., 2025a], **Qwen3-VL-8B** [Bai et al., 2025a], **Qwen3-VL-32B** [Bai et al., 2025a], and **InternVL3-8B** [Zhu et al., 2025], covering diverse architectures and scales. For LLaVA-OneVision, we sample 32 frames per video, yielding 32×196 visual tokens. For Qwen3-VL-8B/32B and InternVL3-8B, we use up

Table 1: Performance comparison under different retention ratios on **LLaVA-OneVision-7B**. Score is the average performance across all benchmarks, and Acc (%) represents the relative performance with respect to the full-token Vanilla baseline. The best and second-best results under each retention ratio are highlighted in **bold** and underline, respectively.

Method	Retention Ratio	VideoMME				EgoSchema		LongVideo Bench	MLVU	Avg.	
		Short	Medium	Long	Overall	Subset	Total			Score	Acc (%)
LLaVA-OneVision-7B											
Vanilla	100%	69.7	56.7	48.8	58.5	62.1	60.3	56.4	64.7	59.7	100.0
FastV (ECCV24)	25%	68.9	53.4	46.0	56.1	60.2	57.4	55.1	61.7	57.4	96.0
VisionZip (CVPR25)		67.9	56.4	47.9	57.4	62.3	60.2	56.5	**64.9**	59.2	99.2
PruneVID (ACL25)		67.9	55.4	47.7	57.0	61.3	58.0	55.8	62.2	58.2	97.5
FastVID (NeurIPS25)		70.2	56.5	47.6	58.1	61.4	59.5	55.7	64.2	59.2	99.1
HoliTom (NeurIPS25)		71.0	57.0	48.1	58.7	63.4	61.3	56.2	64.1	60.0	100.5
AOT (CVPR26)		68.3	55.2	46.6	56.7	62.9	61.1	56.3	63.8	58.9	98.6
FlashVID (ICLR26)		**71.5**	56.4	**49.1**	**59.0**	63.3	60.6	56.8	63.5	60.0	100.6
Ours		70.5	**57.7**	48.8	**59.0**	**64.3**	**61.6**	**57.4**	64.6	**60.5**	**101.4**
FastV (ECCV24)	20%	66.7	53.6	47.4	55.9	60.2	57.4	56.3	61.0	57.3	96.2
VisionZip (CVPR25)		68.1	**57.4**	47.9	57.8	62.5	60.3	55.5	64.2	59.2	99.4
PruneVID (ACL25)		67.7	53.6	48.8	56.7	63.0	59.7	55.1	62.5	58.4	97.9
FastVID (NeurIPS25)		69.6	56.6	47.2	57.8	61.4	59.6	55.7	63.8	59.0	98.8
HoliTom (NeurIPS25)		70.5	56.1	48.3	58.3	63.0	61.1	57.4	63.1	59.7	100.1
AOT (CVPR26)		68.1	54.9	46.2	56.4	63.2	**61.3**	55.7	62.9	58.6	98.1
FlashVID (ICLR26)		70.4	55.1	**49.7**	58.5	63.1	60.0	**58.3**	62.9	59.7	100.1
Ours		**71.1**	57.1	48.4	**58.9**	**63.6**	61.2	56.8	**64.7**	**60.2**	**100.9**
FastV (ECCV24)	15%	64.1	54.3	44.8	54.4	59.5	56.5	54.6	59.9	56.0	93.9
VisionZip (CVPR25)		63.1	54.9	47.3	55.1	62.5	59.8	54.2	63.3	57.5	96.5
PruneVID (ACL25)		67.6	52.5	48.8	56.3	61.7	57.5	54.3	62.4	57.6	96.7
FastVID (NeurIPS25)		69.4	56.2	47.5	57.7	58.6	58.8	56.1	63.4	58.5	98.0
HoliTom (NeurIPS25)		68.9	55.6	47.1	57.2	62.8	61.1	56.5	62.6	59.0	98.8
AOT (CVPR26)		67.8	54.9	46.2	56.3	63.5	**61.3**	55.5	62.2	58.5	97.9
FlashVID (ICLR26)		69.2	56.4	48.4	57.8	62.7	60.4	57.6	62.1	59.4	99.5
Ours		**69.6**	**56.7**	**49.2**	**58.5**	**63.6**	61.0	**57.7**	**63.7**	**60.0**	**100.5**
FastV (ECCV24)	10%	60.5	52.5	44.5	52.5	58.4	55.8	52.2	57.8	54.3	91.1
VisionZip (CVPR25)		59.8	53.2	46.0	53.0	61.2	58.3	49.6	62.7	55.5	93.1
PruneVID (ACL25)		66.4	52.5	46.1	55.0	60.1	57.2	54.2	60.4	56.5	94.7
FastVID (NeurIPS25)		**68.5**	55.4	48.3	57.4	58.5	58.8	56.1	62.6	58.2	97.6
HoliTom (NeurIPS25)		67.8	55.7	47.2	56.9	62.7	**61.0**	56.5	61.0	58.6	98.3
AOT (CVPR26)		67.1	56.0	46.1	56.4	61.5	60.1	53.6	61.3	57.8	96.8
FlashVID (ICLR26)		67.1	57.4	48.6	57.7	62.5	60.2	56.4	62.2	59.0	99.0
Ours		68.1	**57.5**	**49.1**	**58.2**	**62.9**	60.4	**57.1**	**63.0**	**59.5**	**99.7**

to 32 frames. We set $K = 16$. Experiments are conducted on NVIDIA A100 80G GPUs using LMMs-Eval [Zhang et al., 2025a]. Details are provided in Appendix C.2.

Compared Baselines. We compare **CompVID** with **7** representative pruning baselines, including FastV [Chen et al., 2024], VisionZip [Yang et al., 2025], PruneVID [Huang et al., 2024], HoliTom [Shao et al., 2025], FastVID [Shen et al., 2025], FlashVID [Fan et al., 2026], AOT [Li et al., 2026]. Reproduction details of these baselines are provided in Appendix C.1.

5.2 Main Results

Comparisons on LLaVA-OneVision-7B. Table 1 compares **CompVID** with existing methods on LLaVA-OneVision-7B and reveals two findings. **(i) Consistent superiority across budgets:** **CompVID** achieves the highest Acc (%) at all retention ratios, reaching **101.4%**, **100.9%**, **100.5%**, and **99.7%** under 25%, 20%, 15%, and 10% retention ratios, respectively, outperforming the strongest baselines by **0.8%**, **0.8%**, **1.0%**, and **0.7%**. Image-based methods such as FastV and VisionZip compress only spatial tokens, leading to clear degradation under low retention ratios. Although video-based methods such as HoliTom and FlashVID outperform Vanilla under 25% and 20% retention ratios via spatiotemporal merging, they still ignore non-uniform local video complexity and may cause suboptimal token allocation. While FastVID relies on temporal segmentation, it overlooks the 3D structure and leads to coarse token allocation. In contrast, our **CompVID** adaptively partitions the 3D spatiotemporal volume according to the representation demands of specific regions, achieving a stronger efficiency-accuracy trade-off. **(ii) Robustness under aggressive pruning:** even with only **10%** of the visual tokens retained, **CompVID** preserves **99.7%** of the full-token performance, substantially outperforming all competing methods under this aggressive pruning setting.

Table 2: Performance comparison under different retention ratios on **Qwen3-VL-8B & 32B**. OOM represents out-of-memory error.

Method	Retention Ratio	VideoMME				MVBench	LongVideo Bench	MLVU	Avg.	
		Short	Medium	Long	Overall				Score	Acc (%)
Qwen3-VL-8B										
Vanilla	100%	76.0	60.4	57.0	64.5	68.6	60.3	63.5	64.3	100
VisionZip (CVPR25)	25%	69.6	56.7	54.2	60.1	62.2	56.7	60.8	60.0	93.5
HoliTom (NeurIPS25)		71.4	54.6	53.1	59.7	63.0	56.8	61.2	60.0	93.2
FastVID (NeurIPS25)		72.0	56.8	52.7	60.5	67.3	58.7	60.7	61.2	95.2
FlashVID (ICLR26)		**74.4**	58.7	53.9	62.3	**67.5**	58.8	60.9	62.4	96.9
Ours		74.3	**59.7**	**55.0**	**63.0**	66.8	**60.2**	**64.1**	**63.3**	**98.4**
VisionZip (CVPR25)	15%	66.9	54.9	52.9	58.2	60.1	56.2	60.0	58.5	91.0
HoliTom (NeurIPS25)		68.9	54.7	51.7	58.3	60.0	55.8	59.6	58.4	90.9
FastVID (NeurIPS25)		69.6	54.3	51.1	58.3	66.1	57.1	59.0	59.4	92.3
FlashVID (ICLR26)		71.5	**57.6**	53.2	60.8	65.8	57.0	58.5	60.6	94.2
Ours		**73.2**	57.4	**54.6**	**61.7**	**66.4**	**59.1**	**61.1**	**61.9**	**96.3**
Qwen3-VL-32B										
Vanilla	100%	78.3	67.1	62.4	69.3	73.2	62.4	66.1	68.4	100
VisionZip (CVPR25)	25%	72.8	63.1	58.6	64.8	62.9	60.0	62.2	63.5	92.9
HoliTom (NeurIPS25)		74.2	61.2	59.1	64.9	62.3	60.2	63.2	63.6	93.1
FastVID (NeurIPS25)		75.1	62.6	57.7	65.1	**71.0**	60.9	64.5	65.3	95.4
FlashVID (ICLR26)		OOM	OOM	OOM	OOM	OOM	OOM	OOM	–	–
Ours		**77.5**	**64.8**	**59.5**	**67.3**	69.8	**61.9**	**66.5**	**66.8**	**97.6**

Comparisons on Qwen3-VL-8B & 32B. Table 2 demonstrates the generalization ability of **CompVID** across advanced architectures and larger-scale models. **CompVID** consistently ranks first under different retention ratios. On Qwen3-VL-8B, **CompVID** achieves **98.4%** and **96.3%** Acc under 25% and 15% retention ratios, outperforming the second-best FlashVID by **1.5%** and **2.1%**, respectively. On the larger Qwen3-VL-32B, **CompVID** preserves **97.6%** of the original performance under a 25% retention ratio, surpassing FastVID by **2.2%**. Moreover, since **CompVID** performs pruning based on visual content itself, it avoids the additional activation and attention overhead incurred by attention-based methods such as FlashVID. **CompVID** therefore facilitates seamless deployment across diverse and large-scale VLLM architectures, demonstrating its practicality.

Table 3: Efficiency comparison on LLaVA-OneVision-7B under a 10% retention ratio on an NVIDIA A100.

Method	TFLOPs	Prefill Time (ms)			Acc (%)
		Compression	LLM Forward	Total	
Vanilla	48.82	-	466.4	466.4 (1.0x)	100
VisionZip (CVPR25)	4.26	71.3	**54.5**	**125.8 (3.7x)**	93.1
PruneVID (ACL25)	4.18	84.7	54.6	139.3 (3.3x)	94.7
HoliTom (NeurIPS25)	4.33	71.9	68.2	140.1 (3.3x)	98.3
FastVID (NeurIPS25)	4.24	75.8	58.1	133.9 (3.5x)	97.6
FlashVID (ICLR26)	4.21	**67.1**	65.5	132.6 (3.5x)	99.0
Ours	**4.14**	68.3	58.1	126.4 (3.7x)	**99.7**

Table 4: Efficiency comparison on **Qwen3-VL-8B** on **VideoMME** under the **25%** retention ratio. Total Latency denotes the end-to-end wall-clock time.

Methods	**Total Latency ↓ (s)**	**GPU Peak ↓ Memory (MB)**	**Performance ↑**
Qwen3-VL-8B	1536.6	23579.3	64.5
VisionZip (CVPR25)	1335.4 (↓13.1%)	19875.9 (↓15.7%)	60.1 (↓4.4)
HoliTom (NeurIPS25)	1274.3 (↓17.1%)	19875.9 (↓15.7%)	59.7 (↓4.8)
FastVID (NeurIPS25)	1298.7 (↓15.5%)	19263.3 (↓18.3%)	60.5 (↓4.0)
FlashVID (ICLR26)	1354.3 (↓11.9%)	40867.5 (↑73.3%)	62.3 (↓2.2)
Ours	**1176.3** (↓23.4%)	**19148.2** (↓18.8%)	**63.0** (↓1.5)

More experimental results are provided in Appendix A.4, A.5, A.6, A.7, A.8, A.9, including evaluations on InternVL3-8B, aggressive 5% token retention on LLaVA-OneVision-7B and Qwen3-VL-8B, long-video grounding on ActivityNet-Grounding, fine-grained video understanding on VNBench, and robustness analysis with more input frames.

5.3 Efficiency Analysis

Prefill-Time Efficiency on LLaVA-OneVision-7B. Table 3 reports the efficiency results on LLaVA-OneVision-7B at a 10% retention ratio. All pruning methods reduce the computational cost from **48.82** TFLOPs to around **4.2** TFLOPs. **CompVID** requires **68.3** ms for token compression and reduces the total prefill time from **466.4** ms to **126.4** ms. This results in a **3.7**× speedup. Meanwhile, **CompVID** retains **99.7%** of the original performance. Compared with PruneVID and FlashVID, it improves Acc by **5.0%** and **0.7%**, respectively, while requiring less total prefill time.

End-to-End Latency and Memory on Qwen3-VL-8B. Table 4 reports the efficiency comparison on the VideoMME benchmark using Qwen3-VL-8B under a 25% retention ratio. **CompVID** achieves the lowest end-to-end latency of **1176.3 s**, reducing latency by **23.4%** compared with the full-token Qwen3-VL-8B. It also achieves the best performance among all pruning methods, reaching **63.0** with

Table 5: Ablation study of SVD-guided Dimension Partition and SVD-guided Regional Allocation. In the first row, **binary tree** partitions the video feature volume based on feature variance.

Components			Performance				
Binary tree	SVD-guided Dimension Partition	SVD-guided Regional Allocation	VideoMME	LongVideo Bench	EgoSchema	MLVU	Acc (%)
✓	–	–	56.4	54.9	58.2	61.3	96.2
✓	✓	–	57.5	56.3	59.6	62.4	98.6
✓	✓	✓	**58.2**	**57.1**	**60.4**	**63.0**	**99.7**

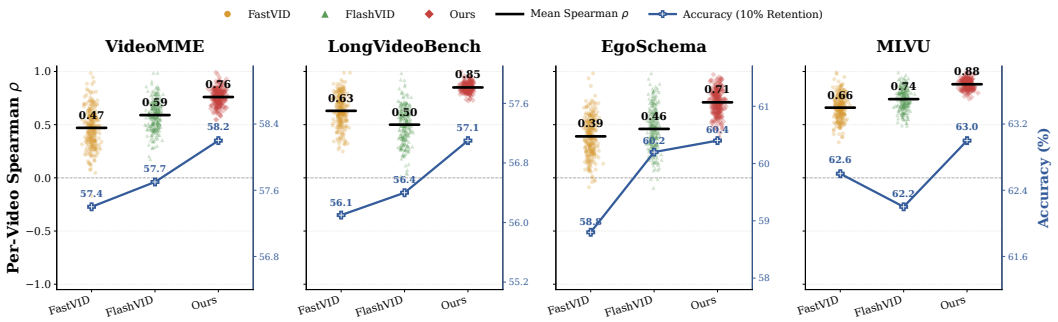


Figure 8: Spearman ρ analysis between regional complexity and token allocation on four benchmarks.

only a **1.5**-point drop from Vanilla. Moreover, **CompVID** reduces GPU peak memory from **23579.3 MB** to **19148.2 MB**, corresponding to an **18.8%** reduction. These results show that **CompVID** improves inference efficiency while preserving strong performance.

5.4 Ablation Study

In this section, we conduct ablation studies on LLaVA-OneVision-7B under a **10%** token retention ratio. Table 5 evaluates the two SVD-guided components of **CompVID**. The variance-guided binary tree baseline achieves **96.2%** Acc. SVD-guided dimension partition raises Acc to **98.6%**, a **2.4%** improvement. It identifies the most complex dimension of each region and partitions accordingly, preserving detailed variations in the video. SVD-guided regional allocation further improves Acc to **99.7%**. It assigns more tokens to complex regions and compresses simple regions more heavily, improving token allocation. Additional ablation studies are provided in Appendix A.1, A.2, A.3.

5.5 Correlation Analysis between Regional Complexity and Token Allocation

To assess whether regional complexity guides token allocation, we divide each video into eight video regions and compute the Spearman correlation between regional complexity and retained tokens. As shown in Fig. 8, **CompVID** achieves Spearman ρ values of **0.76**, **0.85**, **0.71**, and **0.88** on VideoMME, LongVideoBench, EgoSchema, and MLVU, outperforming FastVID (0.47, 0.63, 0.39, 0.66) and FlashVID (0.59, 0.50, 0.46, 0.74). Under a **10%** retention ratio on LLaVA-OneVision-7B, **CompVID** achieves **58.2**, **57.1**, **60.4**, and **63.0**. These results show stronger matching between token allocation and regional complexity while maintaining downstream performance. Details of Spearman ρ calculation are provided in Appendix E.1.

6 Conclusion

In this work, we present **CompVID**, a training-free framework that allocates visual tokens according to local complexity. **CompVID** uses dimension-wise SVD to estimate regional and dimension-wise complexity, and introduces **SVDtree** to recursively partition video regions along their complexity-dominant dimensions and allocate region-specific token budgets. Complex regions therefore receive finer representations and more tokens, while simple regions are compressed more aggressively. Experiments on **4** VLLMs and **7** benchmarks show that **CompVID** improves the efficiency-accuracy trade-off across different token budgets. Further analyses of token allocation, partition direction, and input length validate the effectiveness of complexity-aware video token compression.

AI USE STATEMENT

In preparing this work, we employed large language models (LLMs) to assist with polishing the writing, conducting literature reviews, improving the clarity of presentation, and refining the organization of the manuscript. All generated content was carefully reviewed and verified by the authors.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of **CompVID** in §4 and experimental settings in §5.1 and Appendix C. All key configurations, including VLLM backbones, frame sampling, token budgets, partition settings, hardware, and evaluation protocols, are reported. Code and configuration files will be released to support reproducibility for public research use.

ETHICS STATEMENT

In this work, we investigate complexity-aware visual token compression for improving the inference efficiency of Video Large Language Models. Our experiments are conducted using publicly available VLLMs and video understanding benchmarks. We do not collect new human-subject data or private information. Our research focuses exclusively on improving VLLM inference efficiency and does not involve safety-critical applications.

REFERENCES

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

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 2022.

Shashank Raj Alvar, Gaurav Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9392–9401, 2025.

Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, et al. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

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

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2015.

Keshigeyan Chandrasegaran, Agrim Gupta, Lea M Hadzic, Taran Kota, Jimming He, Cristóbal Eyzaguirre, Zane Durante, et al. Hourvideo: 1-hour video-language understanding. *Advances in Neural Information Processing Systems*, 2024.

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 *Proceedings of the European Conference on Computer Vision*, 2024.

Janghoon Cho, Jungsoo Lee, Munawar Hayat, Kyuwoong Hwang, Fatih Porikli, and Sungha Choi. Floc: Facility location-based efficient visual token compression for long video understanding. In *Proceedings of the International Conference on Learning Representations*, 2026.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Zhengfang Duanmu, Kede Ma, and Zhou Wang. Quality-of-experience of adaptive video streaming: Exploring the space of adaptations. In *Proceedings of the ACM international conference on Multimedia*, 2017.

Ziyang Fan, Keyu Chen, Ruilong Xing, Yulin Li, Li Jiang, and Zhuotao Tian. Flashvid: Efficient video large language models via training-free tree-based spatiotemporal token merging. In *Proceedings of the International Conference on Learning Representations*, 2026.

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

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

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *Transactions on Machine Learning Research*, 2025a.

Jinlong Li, Liyuan Jiang, Haonan Zhang, and Nicu Sebe. Token reduction via local and global contexts optimization for efficient video large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

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 *Proceedings of the International Conference on Machine Learning*, 2023.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Yulin Li, Haokun Gui, Ziyang Fan, Junjie Wang, Bin Kang, Bin Chen, and Zhuotao Tian. Less is more, but where? dynamic token compression via llm-guided keyframe prior. *Advances in Neural Information Processing Systems*, 2025b.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

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

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in Neural Information Processing Systems*, 2024b.

Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv preprint arXiv:2306.05424*, 2023.

Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. *Advances in Neural Information Processing Systems*, 2023.

Bohao Peng, Zhuotao Tian, Shu Liu, Mingchang Yang, and Jiaya Jia. Scalable language model with generalized continual learning. *arXiv preprint arXiv:2404.07470*, 2024.

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.

Kele Shao, Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. Holitom: Holistic token merging for fast video large language models. *Advances in Neural Information Processing Systems*, 2025.

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. *Advances in Neural Information Processing Systems*, 2025.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 2024.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Chao Yuan, Shimin Chen, Minliang Lin, Limeng Qiao, Guanglu Wan, and Lin Ma. Unicomp: Rethinking video compression through informational uniqueness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, et al. Lmms-eval: Reality check on the evaluation of large multimodal models. In *Findings of the Association for Computational Linguistics: NAACL*, 2025a.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. In *Proceedings of the International Conference on Machine Learning*, 2025b.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data. *Transactions on Machine Learning Research*, 2024.

Zijia Zhao, Haoyu Lu, Yuqi Huo, Yifan Du, Tongtian Yue, Longteng Guo, Bingning Wang, et al. Needle in a video haystack: A scalable synthetic evaluator for video mllms. In *Proceedings of the International Conference on Learning Representations*, 2024.

Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Shitao Xiao, Xi Yang, Yongping Xiong, et al. Mlvu: A comprehensive benchmark for multi-task long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenlong Ye, Lixin Gu, Hao Tian, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

APPENDIX

A ADDITIONAL EXPERIMENTAL RESULTS

A.1 ADDITIONAL ABLATION STUDY ON PARTITION AND COMPLEXITY ESTIMATION

Ablation on Partition and Complexity Estimation. Table 6 compares different partition strategies, complexity metrics, and regional complexity calculation methods. For the partition strategy, SVDtree partition achieves an Acc of **99.7%**, compared with **95.7%** for temporal partition, with consistent improvements on all four benchmarks. In particular, SVDtree partition improves the scores on VideoMME, LongVideoBench, EgoSchema, and MLVU from 56.0, 54.4, 57.8, and 60.9 to **58.2**, **57.1**, **60.4**, and **63.0**, respectively. This suggests that restricting partitioning to the temporal dimension cannot fully capture the heterogeneous variations of video features, whereas dimension-adaptive partitioning can better preserve informative variations distributed along the temporal, height, and width dimensions.

We further compare three alternatives for complexity estimation. Feature variance and feature entropy achieve **96.2%** and **96.9%** Acc, respectively, while SVD improves the Acc to **99.7%** and consistently yields the best performance across all benchmarks. This indicates that singular-value decay provides a more effective characterization of the low-rank structure and variation patterns within regional video features than simple statistical measures such as variance or entropy. Finally, we examine the formulation of regional complexity. Directly averaging the dimension-wise complexities achieves **99.3%** Acc, while further incorporating the region size improves the Acc to **99.7%**. The improvement shows that regional complexity should account for not only the intensity of feature variation but also the spatiotemporal extent over which such variation occurs. Overall, these ablations validate the effectiveness of both SVD-guided partitioning and the proposed size-aware complexity formulation in **`CompVID`**.

A.2 ADDITIONAL ABLATION STUDY ON PARTITIONING AND REGIONAL TOKEN ALLOCATION UNDER MAX-MIN DIVERSITY

To examine whether the performance gain mainly comes from our partition and allocation strategy rather than Max-Min Diversity, we compare different video partition and token allocation schemes while using the same Max-Min Diversity token selection method. Specifically, the video-level variant treats the entire video as a whole and selects tokens globally. The frame-level and segment-level variants partition the video by individual frames and temporal segments, respectively, with token budgets uniformly distributed across the resulting units. Uniform region-level further divides the video into regular spatiotemporal regions and assigns an equal budget to each region. As shown in Table 7, these variants achieve 93.6%, 95.8%, 96.5%, and 96.7% Acc, respectively. In contrast, our region-level strategy adaptively partitions the video and allocates tokens according to regional complexity, achieving 99.7% Acc and the best performance on all four benchmarks. These results indicate that the improvement primarily comes from complexity-aware regional partitioning and token allocation.

A.3 SENSITIVITY TO THE PARTITION NUMBER

Fig. 9 studies the sensitivity to the partition number K. Performance improves as K increases from **4** to **16**. A small K produces coarse regions and limits adaptive token allocation. The overall performance becomes stable when K reaches **16**. Increasing K to **32** or **64** brings no clear gain and slightly reduces performance. A large K divides the video into overly small regions, which may break semantic continuity. We therefore set $K = 16$ in the main experiments.

A.4 ADDITIONAL EXPERIMENTS ON INTERNVL3-8B

We further apply **`CompVID`** to one of the most advanced MLLMs, InternVL3-8B, to evaluate its generalization across different model architectures. As shown in Table 8, under a **15%** retention ratio, **`CompVID`** achieves the highest Acc of **95.3%**, outperforming FlashVID, FastVID, and PruneVID by **1.2%**, **3.3%**, and **8.1%**, respectively. In terms of average score, our method also obtains the best result of **63.1**, exceeding FlashVID and FastVID by **0.7** and **2.2** points. Consistent improvements

Table 6: Ablation studies on partition strategies, complexity metric methods, and regional complexity calculation methods. **Temporal Partition** partitions the feature volume only along the temporal dimension. **Feature Variance** estimates complexity using the variance of regional visual features, while **Feature Entropy** measures complexity based on the entropy of regional feature distributions.

Variant	VideoMME	LongVideo Bench	EgoSchema	MLVU	Acc (%)
Partition Strategy					
Temporal Partition	56.0	54.4	57.8	60.9	95.7
SVDtree partition	**58.2**	**57.1**	**60.4**	**63.0**	**99.7**
Complexity Metric Methods					
Feature Variance	56.4	54.9	58.2	61.3	96.2
Feature Entropy	56.8	55.5	58.6	61.1	96.9
SVD	**58.2**	**57.1**	**60.4**	**63.0**	**99.7**
Regional Complexity Calculation Methods					
$(c_T(\mathcal{R}) + c_H(\mathcal{R}) + c_W(\mathcal{R}))/3$	58.0	56.8	60.1	62.8	99.3
$\lvert\mathcal{R}\rvert \cdot (c_T(\mathcal{R}) + c_H(\mathcal{R}) + c_W(\mathcal{R}))/3$	**58.2**	**57.1**	**60.4**	**63.0**	**99.7**

Table 7: Ablation study of different video partition and token allocation strategies under the same Max-Min Diversity token selection. All experiments are conducted on LLaVA-OneVision-7B with a 10% token retention ratio.

Methods	VideoMME	LongVideo Bench	EgoSchema	MLVU	Acc (%)
Video-level	56.2	54.8	58.1	60.9	93.6
Frame-level	56.9	55.4	59.7	61.6	95.8
Segment-level	57.6	56.0	59.3	62.4	96.5
Uniform Region-level	57.3	56.3	59.5	62.2	96.7
Our Region-level	**58.2**	**57.1**	**60.4**	**63.0**	**99.7**

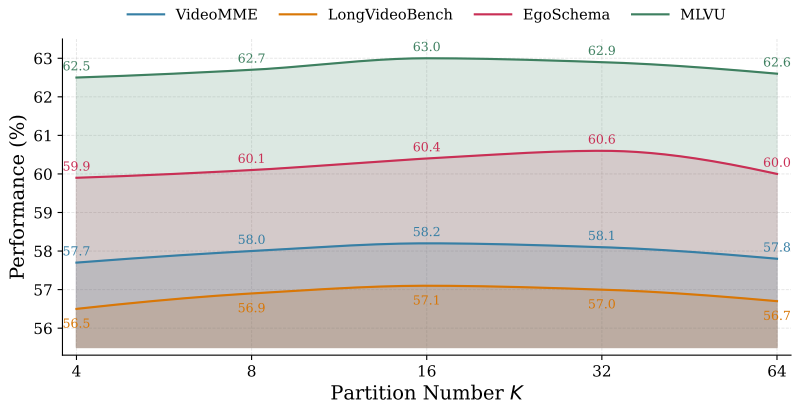


Figure 9: Sensitivity analysis of the partition number K.

are observed across VideoMME and LongVideoBench, while maintaining competitive performance on MLVU. These results indicate that the proposed complexity-guided compression strategy is not tailored to a specific VLLM backbone, but remains effective across different multimodal architectures. This further confirms the transferability and robustness of **CompVID** across different MLLMs.

A.5 Additional Experiments on LLaVA-OneVision-7B

Under the extremely low **5%** retention ratio on LLaVA-OneVision-7B, **CompVID** still achieves the best overall accuracy among all pruning methods. As shown in Table 9, **CompVID** obtains **96.3%** Acc, outperforming the strongest baseline FlashVID by **0.7%**. It also exceeds HoliTom and AOT by **2.6%**, and outperforms FastV and VisionZip by **8.3%** and **6.4%**, respectively. These results show that **CompVID** remains robust under aggressive token pruning and preserves useful visual information even with a highly limited token budget.

Table 8: Performance comparison under 15% retention ratio on **InternVL3-8B**.

Method	Retention Ratio	VideoMME	LongVideo Bench	MLVU	Avg.	
					Score	Acc (%)
InternVL3-8B						
Vanilla	100%	66.6	59.4	72.7	66.2	100
PruneVID (ACL25)	15%	57.4	53.5	62.1	57.7	87.2
FastVID (NeurIPS25)		62.7	54.6	65.3	60.9	92.0
FlashVID (ICLR26)		63.6	54.8	**68.7**	62.4	94.1
Ours		**65.0**	**55.7**	68.6	**63.1**	**95.3**

Table 9: Performance comparison under **5%** retention ratio on **LLaVA-OneVision-7B**.

Method	Retention Ratio	VideoMME				EgoSchema		LongVideo Bench	MLVU	Avg.	
		Short	Medium	Long	Overall	Subset	Total			Score	Acc (%)
LLaVA-OneVision-7B											
Vanilla	100%	69.7	56.7	48.8	58.5	62.1	60.3	56.4	64.7	59.7	100.0
FastV (ECCV24)	5%	58.6	49.7	46.8	51.7	56.1	54.3	48.1	54.8	52.5	88.0
VisionZip (CVPR25)		60.7	49.8	45.5	52.0	59.3	56.5	48.2	57.4	53.7	89.9
PruneVID (ACL25)		64.4	50.3	47.9	54.2	56.6	55.2	53.5	59.3	55.2	92.4
FastVID (NeurIPS25)		66.1	51.7	48.1	55.3	56.1	55.6	53.2	57.7	55.5	92.9
HoliTom (NeurIPS25)		65.4	52.6	47.3	55.1	58.4	56.7	53.6	58.6	56.0	93.7
AOT (CVPR26)		64.7	53.4	46.6	54.9	59.6	58.3	50.9	59.2	56.0	93.7
FlashVID (ICLR26)		68.5	**54.2**	46.5	56.4	59.2	**58.5**	54.7	58.6	57.1	95.6
Ours		**69.4**	52.2	**48.5**	**56.7**	**60.4**	58.0	**55.5**	**59.4**	**57.5**	**96.3**

Table 10: Performance comparison under a **5%** retention ratio on **Qwen3-VL-8B**.

Method	Retention Ratio	VideoMME				MVBench	LongVideo Bench	MLVU	Avg.	
		Short	Medium	Long	Overall				Score	Acc (%)
Qwen3-VL-8B										
Vanilla	100%	76.0	60.4	57.0	64.5	68.6	60.3	63.5	64.3	100.0
VisionZip (CVPR25)	5%	59.2	54.9	50.0	54.7	53.5	50.2	57.0	54.2	84.6
HoliTom (NeurIPS25)		61.8	56.7	50.7	56.4	55.3	51.4	57.7	55.7	86.9
FastVID (NeurIPS25)		64.2	57.4	51.2	57.6	60.5	53.4	58.1	57.5	89.6
FlashVID (ICLR26)		65.6	56.8	**52.2**	58.2	**61.7**	54.6	57.2	58.0	90.4
Ours		**67.1**	**57.6**	52.0	**58.9**	61.5	**57.0**	**59.4**	**59.1**	**92.0**

A.6 Additional Experiments on Qwen3-VL-8B

Table 10 reports the results on Qwen3-VL-8B under a **5%** token retention ratio. **CompVID** achieves the highest Acc of **92.0%**, exceeding FlashVID and FastVID by **1.6%** and **2.4%**, respectively. The gains over HoliTom and VisionZip are **5.1%** and **7.4%**. **CompVID** also obtains the best results on several individual benchmarks, including an overall VideoMME score of **58.9**, a LongVideoBench score of **57.0**, and an MLVU score of **59.4**. Even when only **5%** of the visual tokens are retained, **CompVID** preserves **92.0%** of the full-token performance, demonstrating its effectiveness under a highly constrained token budget.

A.7 Comparisons on Long-Video Grounding Task

We further evaluate **CompVID** on ActivityNet-Grounding [Caba Heilbron et al., 2015], a benchmark for temporal grounding in long videos. As shown in Table 11, FastVID achieves a competitive Acc of **77.4%** using temporal segmentation. **CompVID** allocates tokens based on the complexity of each video region. This helps retain details that are useful for temporal localization. As a result, **CompVID** achieves the highest Acc of **79.8%**, outperforming FastVID and FlashVID by **2.4%** and **5.8%**, respectively.

A.8 Comparisons on Fine-Grained Video Understanding Tasks

Table 12 reports the results on VNBench, which tests fine-grained video understanding. With only **25%** of the visual tokens retained, **CompVID** achieves the highest score of **66.3**. It outperforms FlashVID and FastVID by **3.4** and **4.9** points, respectively. **CompVID** preserves more visual details in complex regions, which benefits fine-grained video understanding.

Table 11: Performance comparison on **long-video grounding tasks**.

Method	Retention Ratio	ActivityNet-Grounding				
		R1@0.3	R1@0.5	R1@0.7	mIoU (%)	Acc (%)
Qwen3-VL-8B						
Vanilla	100%	52.9	35.8	22.9	39.1	100
FastVID (NeurIPS25)	25%	41.7	27.2	**16.9**	31.6	77.4
FlashVID (ICLR26)		41.0	25.3	16.1	30.3	74.0
Ours		**42.6**	**28.9**	**16.9**	**32.9**	**79.8**

Table 12: Performance comparison on **fine-grained tasks**.

Method	Retention Ratio	VNBench
LLaVA-OneVision-7B		
Vanilla	100%	68.9
FastVID (NeurIPS25)	25%	61.4
FlashVID (ICLR26)		62.9
Ours		**66.3**

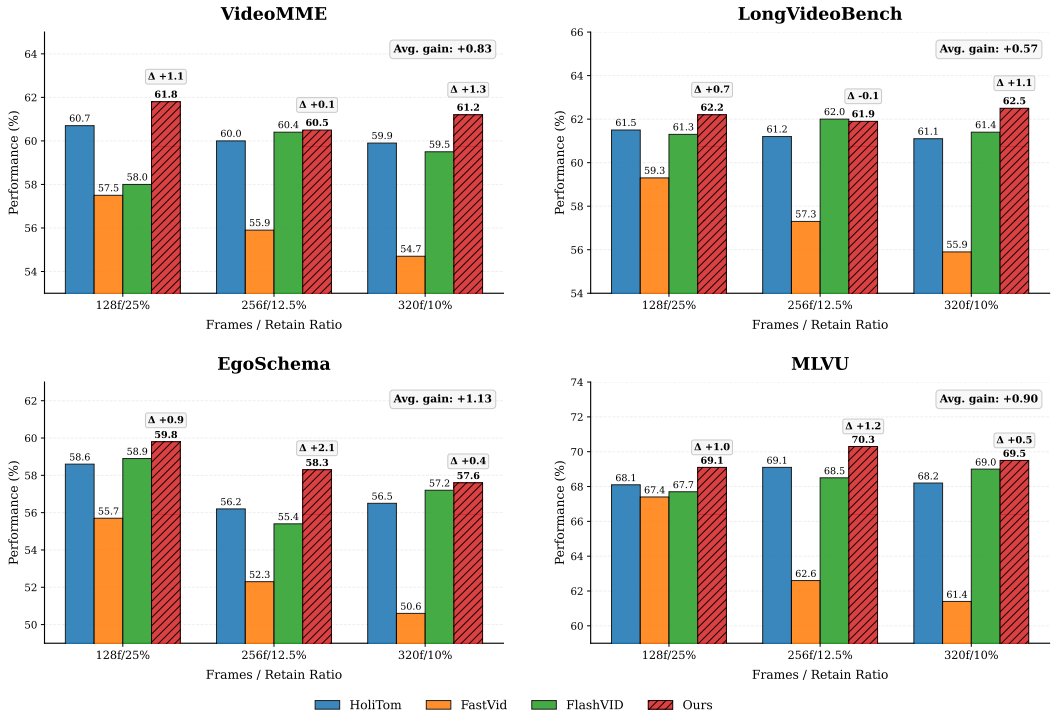


Figure 10: Performance comparison across different frame settings and retention ratios on four video understanding benchmarks, showing that our method consistently achieves gains over existing token pruning approaches.

A.9 Robustness Analysis on More Frames

Fig. 10 further evaluates the robustness of **CompVID** under different input lengths and token budgets. Specifically, we compare four methods using 128, 256, and 320 input frames, with retention ratios of 25%, 12.5%, and 10%, respectively. Across the four benchmarks and three settings, **CompVID** achieves the best performance in 11 of the 12 cases, while falling only **0.1** points behind the best result on LongVideoBench with 256 input frames. Compared with the strongest competing method in each setting, **CompVID** obtains average gains of **0.83**, **0.57**, **1.13**, and **0.90** points on VideoMME, LongVideoBench, EgoSchema, and MLVU, respectively. Notably, the advantage is maintained as the number of input frames increases from 128 to 320 while the retention ratio decreases from 25% to 10%. This indicates that increasing video length does not diminish the effectiveness of our complexity-guided compression strategy. Instead, **CompVID** consistently identifies and preserves informative visual tokens under increasingly constrained token budgets, avoiding excessive token allocation to redundant regions. These results demonstrate that **CompVID** remains effective across different input lengths and compression levels, highlighting its robustness for efficient long-video understanding.

B Complexity-Level Calculation

To compare how different pruning methods allocate tokens across regions with different complexity levels, we group the video regions into three categories: *Low*, *Medium*, and *High*. For each video, we first compute the regional complexity $C(\mathcal{R}_k)$ for all K regions. Since the complexity range may vary across videos, we use the $1/3$ and $2/3$ quantiles as video-specific thresholds:

$$q_{1/3} = \mathcal{Q}_{1/3}\left(\{C(\mathcal{R}_k)\}_{k=1}^{K}\right), \qquad q_{2/3} = \mathcal{Q}_{2/3}\left(\{C(\mathcal{R}_k)\}_{k=1}^{K}\right), \tag{9}$$

where $\mathcal{Q}_p(\cdot)$ denotes the p-th quantile. The complexity level of each region is then determined by

$$L(\mathcal{R}_k) = \begin{cases} \text{Low}, & C(\mathcal{R}_k) \le q_{1/3}, \\ \text{Medium}, & q_{1/3} < C(\mathcal{R}_k) \le q_{2/3}, \\ \text{High}, & C(\mathcal{R}_k) > q_{2/3}. \end{cases} \tag{10}$$

This quantile-based grouping measures the relative complexity of regions within each video and avoids using fixed thresholds across videos with different complexity ranges. For each pruning method, we then count the retained visual tokens belonging to the three complexity levels and normalize them by the total number of retained tokens. The resulting proportions show how the token budget is distributed across low-, medium-, and high-complexity regions, as reported in Fig. 3.

C Implementation Details

C.1 Reproduction Details of Compared Baselines

All experiments are conducted within the LMMs-Eval [Zhang et al., 2025a] evaluation framework. The results of the original LLaVA-OneVision-7B [Li et al., 2025a], Qwen3-VL-8B [Bai et al., 2025a], Qwen3-VL-32B [Bai et al., 2025a], and InternVL3-8B [Zhu et al., 2025] show slight deviations from those reported in their respective papers, which remain within a reasonable margin of error. For a fair comparison, all baseline methods are reimplemented in LMMs-Eval according to their official implementations and evaluation settings.

FastV [Chen et al., 2024]. FastV is implemented by pruning visual tokens according to attention scores from early LLM layers. Following its official setting, FastV performs token pruning at the K-th LLM layer, where K is set to 2, and retains the top-ranked visual tokens under each retention ratio.

VisionZip [Yang et al., 2025]. VisionZip compresses visual tokens by retaining dominant and contextual tokens in each frame, with a dominant-to-contextual token ratio of 54:10 following its official setting. Since the original VisionZip prunes tokens at the output of the vision encoder, it is not directly compatible with the pooling operations in video LLMs. We therefore adopt VisionZip*, which performs token pruning after the pooling stage.

PruneVID [Huang et al., 2024]. PruneVID reduces video redundancy by merging temporally static tokens and clustering spatially similar tokens, followed by LLM-guided token selection based on question–video attention scores. Following its official setting, we perform pruning at the 10th LLM layer with $\alpha = 0.4$, $\tau = 0.8$, and temporal segment ratio of 0.25.

HoliTom [Shao et al., 2025]. HoliTom performs holistic token merging through both outer-LLM and inner-LLM compression. Specifically, it first conducts global redundancy-aware temporal segmentation and spatiotemporal merging before feeding visual tokens into the LLM, and then applies similarity-based token merging inside the LLM. Following its official setting, we set the temporal merging threshold to 0.8 by default and use 0.65 for the 10% retention ratio.

FastVID [Shen et al., 2025]. FastVID reduces video redundancy through dynamic temporal segmentation and density-based spatiotemporal pruning. Specifically, it first partitions video tokens into temporally ordered segments, and then preserves representative anchor tokens and salient tokens within each segment. Following its official setting, we set the dynamic segmentation number $c = 8$, segmentation threshold $\tau = 0.9$, salient-token ratio $d = 0.4$, temporal merging period $p = 4$, and merging balance factor $\beta = 0.6$. The retention ratio is adjusted according to each target compression setting.

FlashVID [Fan et al., 2026]. FlashVID compresses video tokens through Attention and Diversity-based Token Selection (ADTS) and Tree-based Spatiotemporal Token Merging (TSTM). Specifically, ADTS selects representative tokens by jointly considering attention saliency and feature diversity, while TSTM further merges redundant spatiotemporal tokens using a tree-based structure. Following its official setting, we set the segmentation threshold to 0.9, the minimum number of segments to 8, the ADTS balance factor to 0.7, the temporal merging threshold to 0.8, the expansion factor to 1.25, and the inner-LLM pruning layer to 20.

AOT [Li et al., 2026]. AOT reduces video tokens by aggregating local and global contexts through optimal transport. Specifically, it first establishes local- and global-aware token anchors within each frame, and then performs intra-frame and inter-frame optimal transport to aggregate informative cues from pruned tokens. Following its official setting, we use 100 Sinkhorn–Knopp iterations with entropy parameter $\lambda = 0.1$. For LLaVA-OneVision-7B, the number of intra-frame token anchors is set to 126, 144, 196, and 205 under 10%, 15%, 20%, and 25% token retention budgets, respectively.

C.2 Model Details

LLaVA-OneVision-7B [Li et al., 2025a]. LLaVA-OneVision is a unified vision-language model designed for both image and video understanding. In our experiments, we use LLaVA-OneVision-7B as one of the representative VLLM backbones. Following the official setting, each video is uniformly sampled into 32 frames, and each frame is encoded into 196 visual tokens before being fed into the LLM.

Qwen3-VL-8B [Bai et al., 2025a]. Qwen3-VL-8B is a representative model from the Qwen3-VL series, which improves multimodal reasoning through enhanced visual perception, spatiotemporal modeling, and long-context understanding. In our experiments, we use Qwen3-VL-8B as a video-language backbone and adopt the 32-frame input setting for evaluation.

Qwen3-VL-32B [Bai et al., 2025a]. Qwen3-VL-32B is a larger-scale model from the Qwen3-VL family, designed for advanced multimodal reasoning, long-context understanding, and video comprehension. Compared with smaller variants, it provides stronger visual-language reasoning capacity while supporting native long-context multimodal inputs. In our experiments, we adopt Qwen3-VL-32B-Instruct as a large-scale VLLM backbone and use the 32-frame video input setting for evaluation.

InternVL3-8B [Zhu et al., 2025]. InternVL3-8B is a representative model from the InternVL3 series, which introduces native multimodal pre-training and improves multimodal perception and reasoning capabilities. We adopt InternVL3-8B as a video-language backbone to evaluate the generality of our method across different VLLM architectures. Following our experimental setting, we use the 32-frame video input setting for evaluation.

C.3 Benchmark Details

This section briefly introduces the video understanding benchmarks used in our evaluation.

VideoMME. VideoMME [Fu et al., 2024] is a comprehensive benchmark for evaluating MLLMs in video understanding. It contains 900 videos, totaling 254 hours, with 2,700 QA pairs across 6 visual domains and 30 subfields. Video lengths range from 11 seconds to 1 hour, covering short, medium, and long-term reasoning.

LongVideoBench. LongVideoBench [Wu et al., 2024] is a long-context video-language benchmark for evaluating VLLMs. It contains 3,763 videos and 6,678 multiple-choice questions, covering 17 fine-grained categories. Videos can be up to 1 hour long.

MLVU. MLVU [Zhou et al., 2025] contains 3,102 questions across 9 task categories. It is split into a dev set with 2,593 questions and a test set with 509 questions. The benchmark uses long videos ranging from 3 minutes to 2 hours, covering diverse scenarios such as movies, surveillance, egocentric videos, cartoons, and game videos.

EgoSchema. EgoSchema [Mangalam et al., 2023] is a diagnostic benchmark for very long-form video-language understanding. Derived from Ego4D, it contains over 5,000 multiple-choice QA

pairs, spanning 250+ hours of real egocentric video. Each question is based on a 3-minute video clip with 5 answer options, focusing on long-term temporal reasoning and natural human activities.

MVBench. MVBench [Li et al., 2024] is a comprehensive multi-modal video understanding benchmark for evaluating temporal reasoning. It includes 4,000 multiple-choice QA samples across 20 challenging video tasks, covering temporal perception and cognition.

ActivityNet-Grounding. ActivityNet-Grounding [Caba Heilbron et al., 2015] is a temporal sentence grounding benchmark based on ActivityNet captions. It contains about 20K untrimmed videos, 849 hours of video, and 100K temporally localized descriptions, each with a start and end timestamp. Each video has 3.65 annotated sentences on average, requiring models to localize text-described events in long videos.

VNBench. VNBench [Zhao et al., 2024] is a fine-grained synthetic video understanding benchmark built with the VideoNIAH framework. It contains 1,350 samples across 3 tasks: retrieval, ordering, and counting. It evaluates fine-grained temporal perception, chronological ordering, and spatiotemporal modeling ability.

D Related Work

Video Large Language Models. Recent advances in large language models (LLMs) (Touvron et al., 2023; Yang et al., 2024; Achiam et al., 2023; Anil et al., 2025; Peng et al., 2024; Lai et al., 2024) have significantly improved multimodal understanding capabilities. Building upon these successes, Video Large Language Models (VLLMs) (Li et al., 2025a; Zhang et al., 2024; Bai et al., 2025b; Bai et al., 2025a; Comanici et al., 2025) have made remarkable progress in expanding the capabilities of video understanding. Early VLLMs extend image-based multimodal models (Liu et al., 2024a; Li et al., 2023; Alayrac et al., 2022; Liu et al., 2024b) to videos by incorporating temporal information. For instance, Video-LLaMA (Zhang et al., 2023) introduces a temporal-aware video encoder to align video representations with LLMs, while VideoChatGPT (Maaz et al., 2023) leverages frame-level visual tokens for conversational video understanding. Subsequent models further enhance video-language alignment capabilities. Video-LLaVA (Lin et al., 2023) improves instruction-following and multimodal alignment with stronger vision encoders, while more recent models such as LLaVA-OneVision (Li et al., 2025a) and Qwen2.5-VL (Bai et al., 2025b) extend multimodal LLMs to support unified image-video understanding. Despite these advances, videos inherently produce a large number of visual tokens due to high spatial and temporal resolution, leading to substantial computational overhead during LLM inference and motivating the development of video token pruning techniques.

Visual Token Pruning. Visual token pruning has emerged as an effective technique for reducing the computational overhead of multimodal models. Early works mainly focus on image token pruning (Chen et al., 2024; Bolya et al., 2023; Shang et al., 2024; Yang et al., 2025; Zhang et al., 2025b), where redundant tokens are removed based on token importance, token diversity, or a combination of both, significantly reducing computational cost while maintaining competitive performance. Extending token pruning from images to videos is more challenging because video representations contain substantial redundancy both within individual frames and across time. Existing VLLM token pruning methods have explored compression at different video granularities. Video-level approaches jointly remove inter-frame and intra-frame redundancy to construct compact representations over the entire video (Fan et al., 2026; Cho et al., 2026; Li et al., 2026). Segment-level approaches first group temporally related frames and then perform spatiotemporal compression within each segment (Shen et al., 2025; Yuan et al., 2026). Frame-level approaches identify keyframes and assign different compression ratios according to frame importance (Li et al., 2025b). Beyond before-LLM compression, several methods progressively prune visual tokens inside the LLM based on their evolving relevance during inference (Huang et al., 2024; Shao et al., 2025; Tao et al., 2025). Despite differences in pruning granularity and execution stage, existing approaches primarily focus on determining which visual representations should be preserved, merged, or removed, while the region-specific representation demand induced by video complexity remains largely unexplored.

E Details about the calculation of Spearman ρ

E.1 Regional Complexity and Token Allocation

To measure the relationship between regional complexity and token allocation, we calculate the Spearman correlation for each video. Specifically, each video feature volume is uniformly divided into $K = 8$ spatiotemporal regions, denoted as $\{\mathcal{R}_{v,1}, \ldots, \mathcal{R}_{v,K}\}$, where v denotes the video index and k denotes the region index. For each region $\mathcal{R}_{v,k}$, we calculate its regional complexity $C(\mathcal{R}_{v,k})$ and record the number of retained tokens $b_{v,k}$ after token compression. Therefore, each video contains K pairs of regional complexity and retained token numbers, denoted as

$$\{(C(\mathcal{R}_{v,k}), b_{v,k})\}_{k=1}^{K}. \tag{11}$$

We then sort the regional complexity values of the K regions from small to large and assign each region a rank $r_{v,k}^{C}$. A region with a larger complexity value receives a larger rank. Similarly, we sort the retained token numbers from small to large and assign each region a rank $r_{v,k}^{b}$, where a larger rank corresponds to a larger number of retained tokens. If multiple regions have the same complexity value or the same number of retained tokens, their average rank is used. Thus, $r_{v,k}^{C}$ represents the relative complexity of region $\mathcal{R}_{v,k}$, while $r_{v,k}^{b}$ represents its relative number of retained tokens. For video v, we first calculate the average ranks of regional complexity and retained token numbers as

$$\bar{r}_{v}^{C} = \frac{1}{K}\sum_{k=1}^{K} r_{v,k}^{C}, \qquad \bar{r}_{v}^{b} = \frac{1}{K}\sum_{k=1}^{K} r_{v,k}^{b}. \tag{12}$$

Here, $\bar{r}_{v}^{C}$ is the average complexity rank of the K regions in video v, while $\bar{r}_{v}^{b}$ is the average retained-token rank of these regions. The Spearman correlation for video v is then calculated as

$$\rho_v = \frac{\sum_{k=1}^{K}\left(r_{v,k}^{C} - \bar{r}_{v}^{C}\right)\left(r_{v,k}^{b} - \bar{r}_{v}^{b}\right)}{\sqrt{\sum_{k=1}^{K}\left(r_{v,k}^{C} - \bar{r}_{v}^{C}\right)^{2}}\sqrt{\sum_{k=1}^{K}\left(r_{v,k}^{b} - \bar{r}_{v}^{b}\right)^{2}}}. \tag{13}$$

Here, ρ_v denotes the Spearman correlation for video v. The term $r_{v,k}^{C} - \bar{r}_{v}^{C}$ measures the difference between the complexity rank of region $\mathcal{R}_{v,k}$ and the average complexity rank, while $r_{v,k}^{b} - \bar{r}_{v}^{b}$ measures the difference between its retained-token rank and the average retained-token rank. In the numerator, the product is positive when the complexity rank and retained-token rank are both above their corresponding averages or both below their corresponding averages. The product is negative when one rank is above its average and the other is below its average. Therefore, the numerator measures whether the two rankings vary in the same direction across the K regions. The two terms in the denominator account for the overall differences of the complexity ranks and retained-token ranks from their corresponding averages and normalize ρ_v to the range $[-1, 1]$.

A value of ρ_v close to 1 indicates that regions with higher complexity tend to retain more visual tokens. A value close to 0 indicates a weak relationship between regional complexity and token allocation, while a negative value indicates that regions with higher complexity tend to retain fewer visual tokens. For a benchmark containing N evaluated videos, we calculate ρ_v independently for each video and report the mean Spearman correlation over all videos as

$$\bar{\rho} = \frac{1}{N}\sum_{v=1}^{N} \rho_v, \tag{14}$$

where N denotes the number of evaluated videos in the benchmark and $\bar{\rho}$ denotes the mean Spearman correlation reported in Fig. 8. A larger $\bar{\rho}$ indicates a stronger relationship between regional complexity and token allocation across the benchmark, meaning that regions with higher complexity are more consistently assigned more visual tokens. We use the same calculation for FastVID, FlashVID, and **CompVID** under the same 10% token retention ratio on VideoMME, LongVideoBench, EgoSchema, and MLVU.

E.2 Regional Complexity and Logit Margin Gain

To examine whether regional complexity reflects representation demand, we measure the change in prediction confidence when the token budget is reallocated toward individual video regions. We conduct the analysis on the 900 videos in VideoMME and uniformly divide each video feature volume into $K = 8$ non-overlapping spatiotemporal regions using a $2 \times 2 \times 2$ partition, yielding $900 \times 8 = 7,200$ region-level samples. We set the global token budget to $B = 640$, with an initial budget of 80 tokens for each region. For each region $\mathcal{R}_{i,k}$ of video i, we compute its regional complexity defined in Eq. (4):

$$C(\mathcal{R}_{i,k}) = |\mathcal{R}_{i,k}| \cdot \frac{c_T(\mathcal{R}_{i,k}) + c_H(\mathcal{R}_{i,k}) + c_W(\mathcal{R}_{i,k})}{3}, \tag{15}$$

where c_T, c_H, and c_W denote the complexity along the temporal, height, and width dimensions, respectively. VideoMME uses four-way multiple-choice questions with candidate answers $\{A, B, C, D\}$. For each question q, let $y^*_{i,q}$ denote the correct answer and $z_{i,q,y}$ denote the output logit of candidate y. We define the logit margin under the uniform baseline allocation as

$$M^{\text{base}}_{i,q} = z^{\text{base}}_{i,q,y^*_{i,q}} - \max_{y \neq y^*_{i,q}} z^{\text{base}}_{i,q,y}. \tag{16}$$

For each target region $\mathcal{R}_{i,k}$, we increase its token budget by $\Delta b = 70$, from 80 to 150 tokens, while reducing the budget of each of the remaining seven regions by 10 tokens, from 80 to 70. Thus, the total token budget remains fixed at $B = 640$. We then run the model under this reallocated budget and obtain the resulting logit margin:

$$M^{\text{inc}}_{i,k,q} = z^{\text{inc}}_{i,k,q,y^*_{i,q}} - \max_{y \neq y^*_{i,q}} z^{\text{inc}}_{i,k,q,y}. \tag{17}$$

The logit margin gain for region $\mathcal{R}_{i,k}$ is averaged over all Q_i questions associated with video i:

$$G_{i,k} = \frac{1}{Q_i} \sum_{q=1}^{Q_i} \left(M^{\text{inc}}_{i,k,q} - M^{\text{base}}_{i,q} \right). \tag{18}$$

A larger $G_{i,k}$ indicates that reallocating more tokens to region $\mathcal{R}_{i,k}$ produces a larger improvement in the model's confidence toward the correct answer. For visualization, regional complexity and logit margin gain are independently normalized to $[0, 1]$ over all 7,200 regions:

$$\widetilde{C}_{i,k} = \frac{C(\mathcal{R}_{i,k}) - C_{\min}}{C_{\max} - C_{\min}}, \qquad \widetilde{G}_{i,k} = \frac{G_{i,k} - G_{\min}}{G_{\max} - G_{\min}}. \tag{19}$$

We compute the Spearman correlation between $\widetilde{C}_{i,k}$ and $\widetilde{G}_{i,k}$. Let $r^C_{i,k}$ and $r^G_{i,k}$ denote their ranks over the 7,200 samples, with tied values assigned their average ranks. The correlation is computed as

$$\rho = \frac{\sum_{i,k} (r^C_{i,k} - \bar{r}^C)(r^G_{i,k} - \bar{r}^G)}{\sqrt{\sum_{i,k} (r^C_{i,k} - \bar{r}^C)^2} \sqrt{\sum_{i,k} (r^G_{i,k} - \bar{r}^G)^2}}. \tag{20}$$

As shown in Fig. 5, the resulting Spearman correlation is $\rho = 0.56$, indicating a positive association between regional complexity and the benefit of reallocating more tokens to that region.