PruneVid: Visual Token Pruning for Efficient Video Large Language Models

Xiaohu Huang[1] **Hao Zhou**[2] **Kai Han**[1]*
[1] Visual AI Lab, The University of Hong Kong
[2] Department of Computer Vision Technology (VIS), Baidu Inc.
huangxiaohu@connect.hku.hk zhouh156@mail.ustc.edu.cn kaihanx@hku.hk

Abstract

We introduce PruneVid, a training-free visual token pruning method designed to enhance the efficiency of multimodal video understanding. While Large Language Models (LLMs) have shown promising performance on video tasks due to their advanced visual comprehension capabilities, the substantial redundancy inherent in video data poses significant computational challenges. To address this issue, PruneVid (1) reduces intrinsic video redundancy by merging temporally static and spatially similar tokens, and (2) leverages LLMs' inherent ability to selectively prune visual tokens irrelevant to specific queries, thereby improving model efficiency. We validate our method across multiple video benchmarks, demonstrating that PruneVid can prune over 80% of tokens while maintaining competitive performance when combined with different video LLMs. Our results highlight PruneVid's superior effectiveness and efficiency compared to existing pruning methods.

1 Introduction

Large Language Models (LLMs) (Achiam et al., 2023; Yang et al., 2024; Touvron et al., 2023) have significantly advanced multi-modal understanding owing to their exceptional reasoning capabilities and proficiency in following instructions. Within the realm of video understanding, recent studies (Li et al., 2023c; Lin et al., 2023; Zhang et al., 2023; Li et al., 2024b, 2023a; Xu et al., 2024a; Wang et al., 2024a) have capitalized on the use of pre-trained LLMs as foundational models to address video question-answering tasks. However, the redundancy inherent in video content (Pan et al., 2021; Tong et al., 2022) can lead to significant computational expenses for LLMs due to the quadratic complexity of attention mechanisms. Consequently, effectively reducing the number of video tokens while preserving model performance emerges as an intriguing area of research.

Existing approaches to video token pruning, nonetheless, face notable limitations. For instance, LLaMA-VID (Li et al., 2023c) compresses a video into context and content tokens but necessitates expensive pretraining processes. LLaVA-PruMerge (Shang et al., 2024) utilizes CLIP (Radford et al., 2021) token correlations but overlooks the relevance of tokens to specific questions. KV cache methods like Look-M (Wan et al., 2024) and Elastic Cache (Liu et al., 2024d) merge multi-modal inputs by prioritizing text tokens, requiring full visual token encoding (Zhang et al., 2024; Liu et al., 2024c), which becomes inefficient for long sequences. While FastV (Chen et al., 2025) employs LLM attention patterns to prune tokens, it lacks video-specific optimizations and does not adequately address the reduction of video inputs.

In light of these challenges, we identify three essential criteria that an optimal token pruning method for multi-modal video understanding should satisfy. First, the method should be *training-free*, allowing seamless integration with existing models without necessitating extensive retraining or fine-tuning. Second, inherent video redundancy needs to be reduced to save computations on tokens with similar representations along both spatial and temporal dimensions. Third, it must retain visual tokens that are specifically relevant to the given queries, ensuring that the model maintains high performance and mitigates the risk of hallucinations when lacking pertinent information (Huang et al., 2024).

To achieve our objectives, we present PruneVid, a training-free approach for pruning video tokens to achieve efficient video understanding. As illustrated in Fig. 1 (a), our method first tackles intrinsic video redundancy by identifying static regions—areas with minimal variation due to motion or camera movements, often corresponding to

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 19959–19973

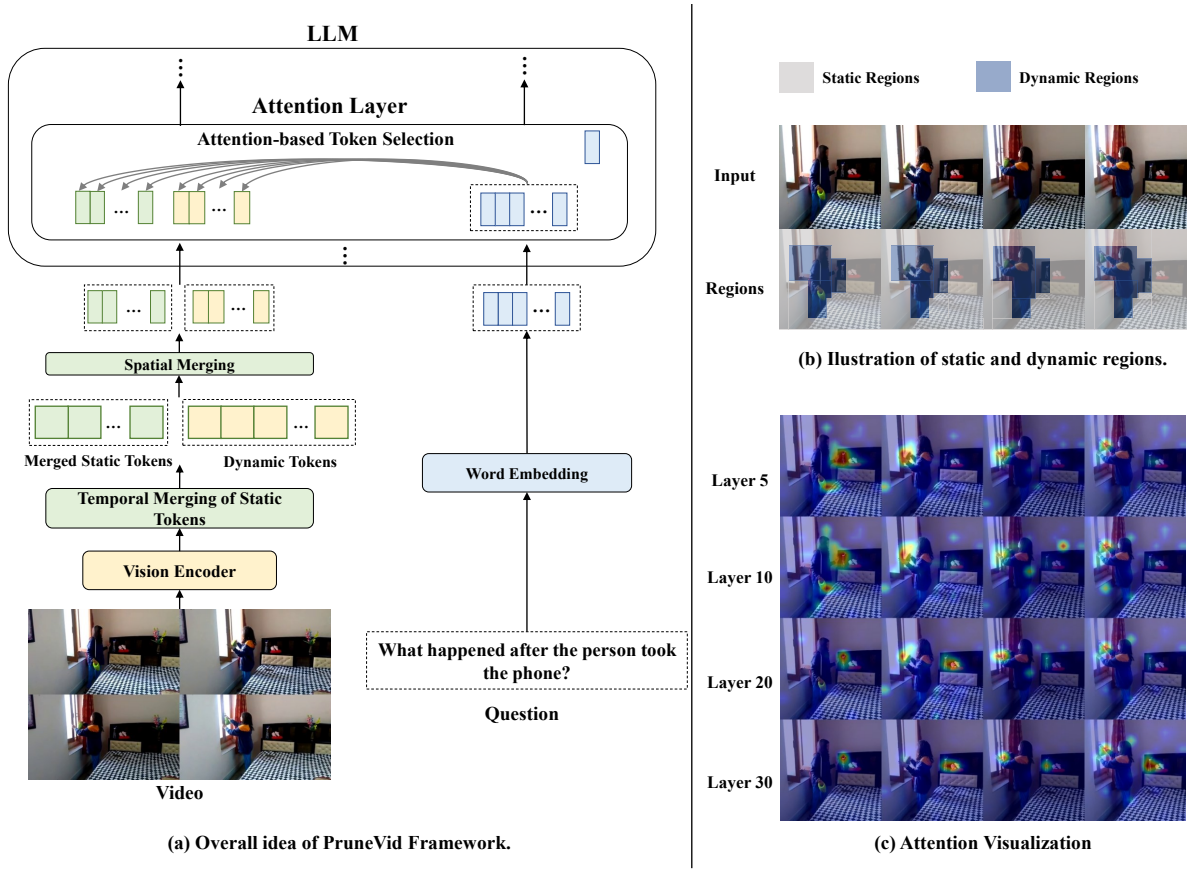

Figure 1: **An illustration of PruneVid.** (a) PruneVid begins by identifying static video regions with minimal temporal variation, reducing redundancy through temporal token merging. Then, spatial redundancy is compressed via clustering and merging, followed by question-guided attention in the LLM to select task-relevant visual tokens. (b) Static regions correspond to areas with minimal change, whereas dynamic regions display object or camera motion; thus, static regions can be compressed together temporally. (c) Attention map visualization across 32 layers shows persistent focus on question-relevant visual regions (*e.g.*, hands and window).

the background (see Fig. 1 (b)). We merge these static tokens along the temporal dimension, reducing computational burden by eliminating redundant temporal information. Next, we employ a clustering technique (Du et al., 2016) to merge spatially similar tokens in both static and dynamic regions, further compressing the input.

Subsequently, within the LLM, we utilize attention scores between the question and video tokens at an intermediate layer to distinguish and preserve discriminative visual tokens essential for answering the question, while pruning irrelevant ones. As depicted in Fig. 1 (c), attention visualizations reveal that the model consistently focuses on crucial features—such as hand movements and related objects (e.g., a window)—that are directly relevant to the question. This demonstrates that important visual regions can be effectively pinpointed using attention mechanisms, capitalizing on the LLM's reasoning and instruction-following strengths. Additionally, for the key-value caches from previous layers, we retain only the essential visual tokens, reducing computational demands during the decoding phase.

We integrate PruneVid with three video LLMs: PLLaVA (Xu et al., 2024a), ST-LLM (Liu et al., 2024b), and LLaVA-OneVision (Li et al., 2024a), and evaluate their performance on multiple video benchmarks, including MVBench (Li et al., 2024b), Video-MME (Fu et al., 2024), Egoschema (Mangalam et al., 2023), and VideoChatGPT-Bench (Maaz et al., 2023). Our extensive experiments demonstrate that PruneVid can prune over 80% of visual tokens with only minimal performance degradation in certain cases. Notably, our method can occasionally enhance model performance. Furthermore, it achieves competitive results compared to the baseline model while reducing FLOPs by 74% to 80%, and reducing GPU memory usage.

The main contributions of this paper are as follows: (1) We introduce PruneVid, a training-free framework that efficiently prunes video tokens for video understanding tasks, seamlessly integrating with off-the-shelf video LLMs. (2) Our method minimizes video redundancy by merging static tokens over time and clustering spatially similar ones. Additionally, we leverage attention scores between the question and video tokens within the LLM to retain only the visual tokens pertinent to answering the questions. (3) Through extensive experiments across multiple benchmarks, we demonstrate that PruneVid consistently achieves superior efficiency and effectiveness with various video LLMs compared to existing approaches.

2 Related Work

2.1 Video Large Language Model

Recent Video LLMs fall into training-free and training-required paradigms. Training-free methods adapt image LLMs for videos through spatial compression: FreeVA (Wu, 2024) compacts frame features, IG-VLM (Kim et al., 2024) grids frames into images, and SF-LLaVA (Xu et al., 2024b) employs SlowFast (Feichtenhofer et al., 2019) networks for temporal modeling. While efficient, these methods handle only short clips due to limited temporal reasoning.

Training-required approaches enhance video understanding through dataset-driven adaptation. Video-ChatGPT (Maaz et al., 2023), Video-LLaVA (Lin et al., 2023), and PLLaVA (Xu et al., 2024a) extend image LLMs with video tuning. Others optimize tokenization (VideoChat2 (Li et al., 2024b), VILA (Lin et al., 2024)), integrate audio (Chat-UniVi (Jin et al., 2024)), or use dynamic masking (LLaMA-VID (Li et al., 2023c), ST-LLM (Liu et al., 2024b)). LLaVA-OneVision (Li et al., 2024a) expands LLaVA (Liu et al., 2024a) with multi-signal inputs. Unlike these methods, PruneVid enhances efficiency of *existing* Video LLMs without additional training.

2.2 Visual Token Pruning

Vision-centric token pruning methods reduce computational overhead through dynamic selection (DynamicViT (Rao et al., 2021)), architecture simplification (FastViT (Vasu et al., 2023)), or token merging (ToMe (Bolya et al., 2023), SPViT (Kong et al., 2022)). For multi-modal tasks, LLaVA-Prumerge uses CLIP attention (Radford et al., 2021) to prune visual tokens, while FastV (Chen et al., 2025) removes tokens with low attention weights relative to `[EOS]`. Our work focuses on video-specific token pruning across both paradigms.

3 Method

To efficiently process video data, our method strategically minimizes redundancy in visual tokens before inputting them into the LLM and effectively identifies question-relevant visual tokens within the LLM. In this section, we first introduce the necessary preliminaries and then provide a detailed explanation of our approach.

3.1 Preliminaries

Our model processes input sequences in two distinct phases: (1) *pre-filling*, where we combine the question and visual tokens to establish initial representations, and (2) *autoregressive decoding*, where we generate the answers. Specifically, let $\boldsymbol{X}_q \in \mathbb{R}^{N_q \times C}$ denote the question tokens, and $\tilde{\boldsymbol{X}}_v \in \mathbb{R}^{N_v' \times C}$ represent the merged visual tokens (as detailed in Sec. 3.2), where N_q, N_v', and C are the numbers of question tokens, visual tokens, and channels, respectively. These tokens are concatenated into a single sequence $\boldsymbol{X} \in \mathbb{R}^{(N_q+N_v') \times C}$.

In the transformer's L layers, each layer l computes queries $\boldsymbol{Q}^{(l)}$, keys $\boldsymbol{K}^{(l)}$, and values $\boldsymbol{V}^{(l)}$ via linear projections. To enforce autoregressive constraints, we apply causal attention:

$$\boldsymbol{A}^{(l)} = \text{Softmax}\left(\frac{\boldsymbol{Q}^{(l)}(\boldsymbol{K}^{(l)})^\top}{\sqrt{C}} + \boldsymbol{m}\right), \quad (1)$$

where the mask $\boldsymbol{m}$ ensures that each position only attends to previous positions, which is set as either 0 or $-\infty$. During the pre-filling phase, we store key-value (KV) caches $\boldsymbol{KV}^{(l)} = (\boldsymbol{K}^{(l)}, \boldsymbol{V}^{(l)})$ for all layers.

In the decoding phase, we generate tokens autoregressively using these caches, which avoids the recomputation of historical tokens and enhances efficiency. To optimize memory and computation, we compress the KV caches from early layers (as detailed in Sec. 3.3) by retaining only the relevant visual tokens selected via LLM-guided attention. For layers $1 \leq l \leq M$, the compressed caches are defined as $\tilde{\boldsymbol{K}}^{(l)} = \left[\tilde{\boldsymbol{K}}_v^{(l)}; \boldsymbol{K}_q^{(l)}\right]$ and $\tilde{\boldsymbol{V}}^{(l)} = \left[\tilde{\boldsymbol{V}}_v^{(l)}; \boldsymbol{V}_q^{(l)}\right]$, effectively reducing the sequence length to $|\mathcal{S}| + N_q$, where $\mathcal{S}$ is the set of

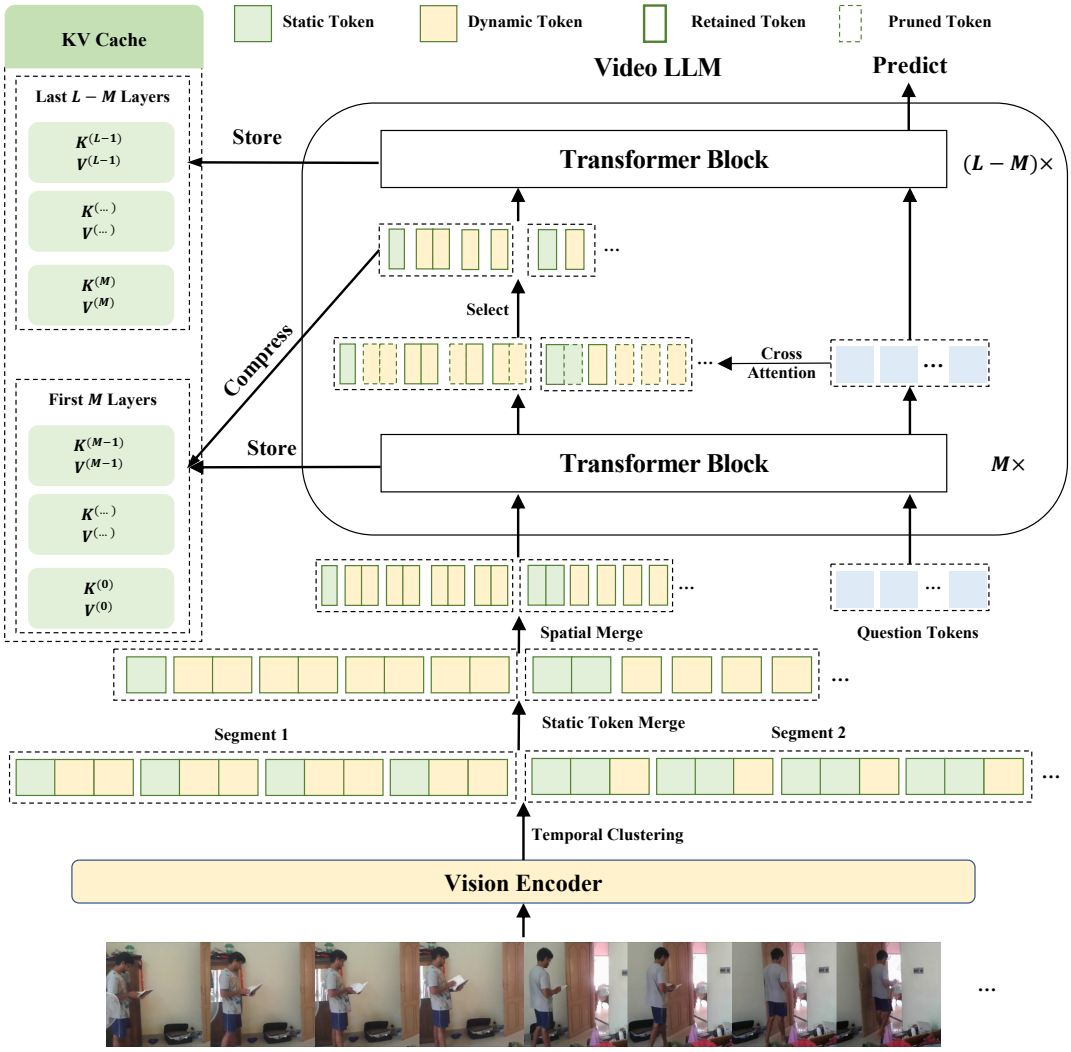

Figure 2: **Illustration of the PruneVid framework**. The framework starts by segmenting the input video into distinct scenes to better capture content variations. Within each scene, we decompose the video tokens into static and dynamic ones based on temporal changes. Static tokens, representing regions with minimal motion, are compressed along the temporal dimension to reduce redundancy. Both static and dynamic tokens are then further compressed by merging spatially similar tokens in the spatial dimension. Afterward, by using the question-to-video attention weights learned from an intermediate layer, we determine which tokens should be pruned to improve efficiency.

indices of the selected visual tokens (see Sec. 3.3). This compression preserves inference quality while significantly enhancing efficiency.

3.2 Spatial-Temporal Token Merging

As depicted in Fig. 2, given an input video consisting of T frames, we first extract visual tokens from each frame using a visual encoder. Let $\boldsymbol{X}_v^{(t)} \in \mathbb{R}^{N_v \times C}$ denote the visual tokens for frame t, where N_v is the number of tokens per frame. The complete set of visual tokens for the video is $\boldsymbol{X}_v = \{\boldsymbol{X}_v^{(1)}, \boldsymbol{X}_v^{(2)}, \dots, \boldsymbol{X}_v^{(T)}\}$.

Recognizing that videos often consist of multiple distinct scenes, it is beneficial to segment the video into separate scenes before identifying static tokens. This preliminary step allows for more precise localization of static regions within each scene, as static tokens may vary significantly from one scene to another. To achieve this, we perform temporal clustering based on the visual content of the frames. Specifically, we begin by computing an average pooled feature vector $\boldsymbol{f}^{(t)} \in \mathbb{R}^C$ for each frame t, obtained by averaging over its tokens. This provides a compact representation of the visual content in each frame. Utilizing the sequence of features $\{\boldsymbol{f}^{(1)}, \dots, \boldsymbol{f}^{(T)}\}$, we then apply the Density Peaks Clustering with k-Nearest Neighbors (DPC-KNN) (Du et al., 2016) algorithm to group the frames into B temporal segments $\{\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_B\}$. To ensure that consecutive frames are clustered to-

gether, we constrain the clustering process to accept only continuous frames as members of the same cluster. If a frame falls into a non-consecutive (discontinuous) cluster, it is reassigned to a neighboring cluster whenever possible. If no neighboring cluster is available, that frame is then assigned to a new cluster. Each segment $\mathcal{T}_b$ comprises a subset of consecutive frames that exhibit similar visual content, effectively delineating different scenes within the video.

Within each temporal segment $\mathcal{T}_b$, we analyze the spatial tokens across the frames to identify *static tokens*—those that remain largely unchanged throughout the segment. For each spatial location i (where $1 \leq i \leq N_v$), we extract the sequence of tokens $\{\boldsymbol{X}_v^{(t)}(i) \mid t \in \mathcal{T}_b\}$ and compute the feature similarities between every pair of tokens in this sequence. Specifically, for tokens at times t and t' within $\mathcal{T}_b$, the similarity is measured using cosine similarity $s_i^{(t,t')}$:

$$s_i^{(t,t')} = \frac{\boldsymbol{X}_v^{(t)}(i)^\top \boldsymbol{X}_v^{(t')}(i)}{\|\boldsymbol{X}_v^{(t)}(i)\| \|\boldsymbol{X}_v^{(t')}(i)\|}. \tag{2}$$

We then compute the average similarity for each spatial location i within the segment:

$$\bar{s}_i = \frac{2}{|\mathcal{T}_b|(|\mathcal{T}_b|-1)} \sum_{\substack{t,t' \in \mathcal{T}_b \\ t<t'}} s_i^{(t,t')}. \tag{3}$$

Tokens with an average similarity above a threshold τ are considered *static*: $\mathcal{I}_{\text{static}} = \{i \mid \bar{s}_i \geq \tau\}$.

For these static tokens, we perform temporal averaging within the segment to compress temporal redundancy:

$$\tilde{\boldsymbol{X}}_v^{(b)}(i) = \frac{1}{|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} \boldsymbol{X}_v^{(t)}(i), \ \forall i \in \mathcal{I}_{\text{static}}. \tag{4}$$

The dynamic tokens, corresponding to $\mathcal{I}_{\text{dynamic}} = \{1, \ldots, N_v\} \setminus \mathcal{I}_{\text{static}}$, are retained without temporal averaging to preserve important motion information.

To further reduce spatial redundancy, we perform spatial clustering within each scene, again using the DPC-KNN algorithm. We apply this clustering separately to both the static and dynamic tokens. For frame t, we cluster the static tokens $\{\boldsymbol{X}_v^{(t)}(i) \mid i \in \mathcal{I}_{\text{static}}\}$ and dynamic tokens $\{\boldsymbol{X}_v^{(t)}(i) \mid i \in \mathcal{I}_{\text{dynamic}}\}$ into respective clusters. We then average the tokens within each cluster to represent them with a single token.

After completing these merging operations, we obtain a reduced set of visual tokens $\tilde{\boldsymbol{X}}_v$ by collecting and concatenating the merged visual tokens for the entire video with significantly less redundancy.

3.3 LLM-Guided Token Selection

We further reduce the visual tokens by leveraging the LLM's internal attentions to select the most relevant tokens with respect to the given question.

Consider the LLM with L layers. During the pre-filling stage, we target the M-th layer, where $1 \leq M \leq L$, to compute cross-attention weights between the question tokens and the merged visual tokens to obtain a measure of relevance.

At the M-th layer, we calculate the attention matrix $\boldsymbol{A}^{(M)} \in \mathbb{R}^{(N_v'+N_q)\times(N_v'+N_q)}$. To extract the cross-attention scores between question and visual tokens, we select the submatrix $\boldsymbol{A}_{qv}^{(M)} \in \mathbb{R}^{N_q \times N_v'}$:

$$\boldsymbol{A}_{qv}^{(M)} = \boldsymbol{A}^{(M)}[N_q{:}, \ :N_v'], \tag{5}$$

where $\boldsymbol{A}^{(M)}[N_q{:}, \ :N_v']$ corresponds to the attention scores from the question tokens to the visual tokens.

Next, we compute the maximum attention values $\boldsymbol{a}_v \in \mathbb{R}^{N_v'}$ for each visual token by applying max pooling over all question tokens. This approach captures the most informative tokens, as not all question tokens are equally important. We then sort the attention scores in descending order and select the top $\alpha\%$ of visual tokens. The set of indices for the selected tokens is represented by $\mathcal{S}$.

By focusing on the top $\alpha\%$ tokens, we align the model's attention with the most question-relevant visual information. To finalize the pre-filling stage, we combine the selected visual tokens with the question tokens, enabling processing in the remaining $(L-M)$ layers of the LLM. The KV vectors derived from the retained visual tokens and question tokens, calculated in the last $(L-M)$ layers, are stored in the KV cache for the decoding process.

To reduce memory and computation, we compress KV caches from layers $1 \leq l \leq M$ by retaining only the $\alpha\%$ selected visual tokens (indices $\mathcal{S}$). For each layer l, we compress the visual token keys and values via row selection:

$$\tilde{\boldsymbol{K}}_v^{(l)} = \boldsymbol{K}_v^{(l)}[\mathcal{S}, :], \quad \tilde{\boldsymbol{V}}_v^{(l)} = \boldsymbol{V}_v^{(l)}[\mathcal{S}, :], \tag{6}$$

and concatenate them with question token matrices $\boldsymbol{K}_q^{(l)}$, $\boldsymbol{V}_q^{(l)}$ to form the final caches:

$$\tilde{\boldsymbol{K}}^{(l)} = [\tilde{\boldsymbol{K}}_v^{(l)}; \boldsymbol{K}_q^{(l)}], \quad \tilde{\boldsymbol{V}}^{(l)} = [\tilde{\boldsymbol{V}}_v^{(l)}; \boldsymbol{V}_q^{(l)}]. \tag{7}$$

Table 1: **Comparison with the state-of-the-arts.** Performance and efficiency comparison are measured across different methods and benchmarks, where the retained ratio and FLOPs are averaged across all the benchmarks. The best results of pruning methods are **bolded**.

Method	Retained Ratio	FLOPs (×)	MVBench	VideoMME	EgoSchema	VideoChatGPT-Bench					
					Subset / Fullset	TU	CU	CO	DO	CI	Avg
PLLaVA	100.0%	1.00×	46.6	44.4	47.8 / 42.6	2.33	3.62	2.93	2.86	3.21	2.99
PLLaVA w/ FastV	30.0%	0.33×	46.1	43.6	46.2 / 41.0	2.38	3.49	2.89	2.76	3.14	2.93
PLLaVA w/ Prumerge	55.7%	0.53×	45.6	43.8	45.2 / 40.4	2.34	3.52	2.90	2.76	3.15	2.93
PLLaVA w/ Look-M	20.0%	1.00×	46.6	44.3	47.0 / 42.3	2.28	3.41	2.75	2.65	3.00	2.82
PLLaVA w/ Ours	**16.2%**	**0.23**×	**47.6**	**45.0**	**49.0 / 42.6**	**2.44**	**3.51**	**2.99**	**2.78**	**3.20**	**2.98**
ST-LLM	100.0%	1.00×	54.9	42.0	56.2 / 45.6	2.46	3.46	2.66	2.63	3.08	2.86
ST-LLM w/ FastV	30.0%	0.37×	42.9	34.5	48.0 / 38.5	2.01	2.23	1.55	1.94	1.69	1.88
ST-LLM w/ Look-M	20.0%	1.00×	54.0	40.6	54.0 / 44.5	2.35	3.41	2.60	2.51	3.01	2.78
ST-LLM w/ Ours	**15.1%**	**0.26**×	**54.3**	**41.4**	**54.6 / 44.7**	**2.40**	**3.43**	**2.63**	**2.60**	**3.04**	**2.82**
LLaVA-OneVision	100.0%	1.00×	58.0	58.2	62.0 / 60.0	2.75	3.70	3.39	2.97	3.50	3.26
LLaVA-OneVision w/ FastV	30.0%	0.30×	57.2	57.6	**62.6 / 60.0**	2.65	3.61	3.28	2.85	3.39	3.16
LLaVA-OneVision w/ Prumerge	55.2%	0.49×	52.9	56.7	62.2 / 60.0	2.72	3.64	**3.32**	**2.94**	3.44	3.21
LLaVA-OneVision w/ Look-M	20.0%	1.00×	57.0	58.0	62.0 / 59.8	2.71	3.70	3.29	2.89	3.44	3.21
LLaVA-OneVision w/ Ours	**17.0%**	**0.20**×	**57.5**	**58.6**	**62.6** / 59.5	**2.73**	**3.72**	3.28	**2.94**	**3.51**	**3.24**

This reduces the sequence length from $N_v' + N_q$ to $|\mathcal{S}| + N_q$, cutting memory and FLOPs proportionally while maintaining decoding efficiency.

4 Experiment

4.1 Datasets and Evaluation Metrics

Generic Multi-Choice VideoQA. MVbench (Li et al., 2024b) encompasses 20 temporally challenging tasks that cannot be addressed using a single frame. Each task includes 200 test samples, formatted as multiple-choice VideoQA. These samples require the model to choose the correct answer from several provided options.

Long-form Multi-Choice VideoQA. We conduct evaluations of our models using two well-regarded benchmarks for long-form video benchmarks: Video-MME (Fu et al., 2024) and Egoschema (Mangalam et al., 2023). In these evaluations, the models are tasked with selecting the correct answer from multiple-choice options.

Text Generation. VideoChatGPT-Bench, introduced by (Maaz et al., 2023), focuses on five aspects: Correctness of Information (CI), Detail Orientation (DO), Contextual Understanding (CU), Temporal Understanding (TU), and Consistency (CO). For evaluation, we use `GPT-3.5-Turbo-0125` for scoring.

4.2 Baselines

We benchmark against three visual token pruning approaches: (1) **LLaVA-PruMerge** (Shang et al., 2024) leverages CLIP attention sparsification with adaptive ratio tuning via outlier detection; (2) **Look-M** (Wan et al., 2024) extends text-only KV cache compression to multi-modal scenarios via pivotal merging; (3) **FastV** (Chen et al., 2025) prunes tokens with low attention scores. All comparisons use official implementations on video benchmarks for fairness.

4.3 Implementation Details

We use NVIDIA A100 GPUs with 80GB of memory for all experiments. We implement PruneVid, LLaVA-PruMerge, Look-M, and FastV on three 7B video LLMs: PLLaVA (Xu et al., 2024a), ST-LLM (Wang et al., 2024a), and LLaVA-OneVision (Li et al., 2024a). LLaVA-PruMerge is incompatible with ST-LLM, so it is excluded from related comparisons. As per the official settings, the input frames are set to 16 for both PLLaVA and ST-LLM, and 32 for LLaVA-OneVision. For the VideoChatGPT-Bench, ST-LLM uses 64 input frames. Besides, for spatial-temporal merging, the threshold τ is set to 0.8, the temporal segment ratio is 0.25, and the spatial merging ratio is 0.5. Across all benchmarks, the token selection ratio α is set as 0.4, and attention calculations use the 10th layer (M). For FastV, we prune the tokens at the 2nd layer and set the retained ratio to 0.3 to achieve roughly comparable FLOPs to our method. Additionally, the FLOPs in the experiments are measured in relation to the visual tokens in the LLM. For the diagnostic study in Sec. 4.5, we conduct experiments based on PLLaVA.

4.4 Main Result

As shown in Tab. 1, our method consistently achieves superior performance compared to other pruning methods while retaining fewer tokens and achieving lower FLOPs. For instance, us-

Table 2: **Ablation study of the proposed modules.** The displayed retained ratio and FLOPs are averaged results measured on MVBench and VideoMME.

No.	Baseline	Static Token Merge	Spatial Merge	Attention-based Selection	Retained Ratio	FLOPs (×)	MVBench	VideoMME
1	✓				100.0%	1.00×	46.6	44.4
2	✓	✓			71.5%	0.69×	46.9	43.6
3	✓		✓		50.0%	0.48×	**47.6**	44.6
4	✓	✓	✓		36.1%	0.34×	47.2	44.7
5	✓			✓	40.0%	0.59×	47.1	44.4
6	✓	✓	✓	✓	**14.1%**	**0.20×**	**47.6**	**45.0**

Table 3: **Efficiency comparison for visual token pruning methods.** TTFT stands for time-to-first-token, which is commonly used for evaluating the efficiency of LLMs. TPS stands for tokens-per-second. The accuracy is measured on MVBench.

Method	FLOPs (×)	TTFT Speed Up (×)	TPS	GPU Mem	Accuracy
Baseline	1.00 ×	1.00 ×	167.3	20G	46.6
Baseline w/FastV	0.33 ×	1.15 ×	183.8	19G	46.1
Baseline w/Prumerge	0.53 ×	1.32 ×	188.2	19G	45.6
Baseline w/Look-M	1.00 ×	0.15 ×	149.3	35G	46.6
Baseline w/Ours	**0.23** ×	**1.55** ×	**191.5**	**17G**	**47.6**

ing PLLaVA, our approach retains only 16.2% of tokens yet surpasses the performance of other pruning methods and even the baseline model on MVBench, VideoMME, and Egoschema. Similar patterns are observed with ST-LLM and LLaVA-OneVision, where our method maintains robust performance with retained ratios as low as 15.1% and 17.0%, respectively, across all benchmarks. This underscores our approach's effectiveness in balancing accuracy with significant reductions in computational overhead.

In contrast, while Prumerge maintains competitive accuracy on some models, it fails to do so with substantially reduced token budgets. Look-M achieves decent performance but requires the vanilla attention implementation for all layers, resulting in lower efficiency. Additionally, FastV struggles to maintain consistent performance across different models; although it performs well on PLLaVA and LLaVA-OneVision, its accuracy on ST-LLM is unsatisfactory, indicating a lack of robustness across diverse architectures. Our method effectively adapts by identifying and preserving only the most informative tokens, delivering strong overall performance with significantly reduced computational costs.

4.5 Diagnostic Study

Ablation of Module Designs. As presented in Tab. 2, we conduct comprehensive ablation experiments to evaluate the efficiency and performance of the proposed modules. Our analysis reveals that merging static tokens along the temporal dimension (**Row 2**) eliminates 28.5% of redundant tokens while maintaining baseline performance (**Row 1**). This demonstrates that over one-fourth of static tokens can be merged to enhance efficiency without sacrificing accuracy. Furthermore, by applying spatial merging (**Row 3**), we reduce spatial redundancy by half, achieving notable gains in efficiency and even improving accuracy on MVBench compared to the baseline. When combining both spatial and temporal merging techniques (**Row 4**), the spatial-temporal redundancy is reduced by 63.9%, highlighting the complementary nature of these methods. Additionally, incorporating attention-based token selection (**Row 5**) captures question-relevant visual tokens and eliminates 60% of irrelevant ones, thereby refining the model's focus. The integration of all these techniques (**Row 6**) achieves the best balance between efficiency and performance, significantly reducing computational complexity with a retained ratio of only 14.1% and achieving the highest accuracy across both MVBench and VideoMME. These results validate the synergistic effects of the proposed modules in effectively mitigating video redundancy.

Efficiency Analysis. As shown on Tab. 3, We compare multiple visual token pruning methods with respect to FLOPs, TTFT speed-up, TPS, GPU

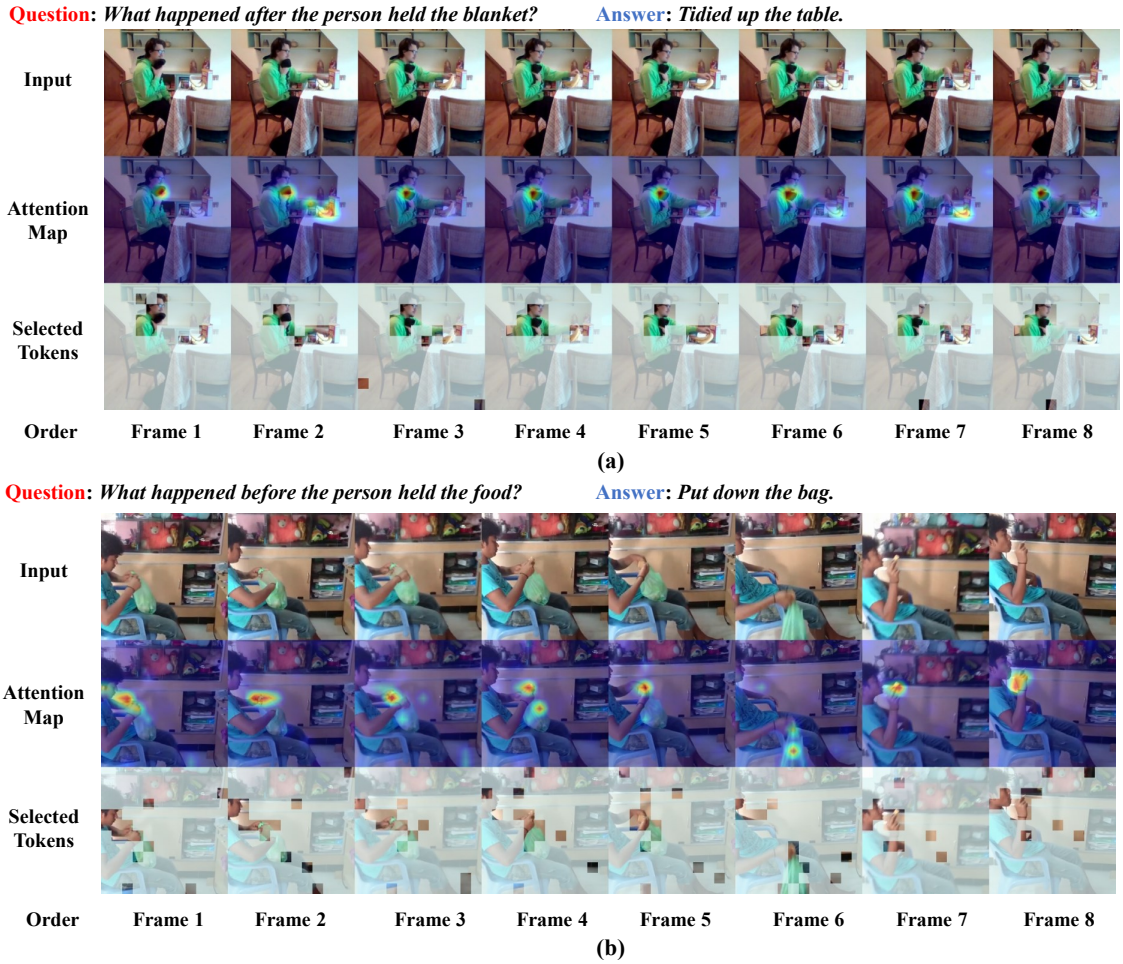

Figure 3: Visualization of the question-to-visual attentions and token selection with the guidance of the attentions.

memory usage, and accuracy. Overall, FastV and Prumerge both demonstrate notable FLOPs reduction and moderate speed gains, though their influence on accuracy remains slightly adverse. By contrast, Look-M manages to preserve accuracy comparable to the baseline but incurs very low TTFT and TPS, and escalates GPU memory to a high level, suggesting that its underlying layer-wise strategy increases overhead. In contrast, our proposed method achieves the most efficient balance by yielding the best efficiency and the highest accuracy, clearly underscoring the advantages of our token pruning approach over existing techniques.

Attention-Guided Token Selection. Fig. 5 illustrates our model's ability to localize question-relevant regions through LLM capacity, focusing on latent visual semantics: **(a)** The model identifies the key object mentioned in the question, the *blanket*, and uses the temporal cue (*after*) from the question to locate additional relevant visual elements, such as objects on the table and the person's hand movements in frames 2, 6, and 7. These details are not directly provided in the question and are inferred through the model's reasoning abilities. **(b)** The model accurately detects the action of the person holding food in frames 7 and 8, and infers that the presence of a *bag* the person puts down is relevant for answering the question, even though the *bag* is not mentioned. This demonstrates the model's ability to reason about relevant objects based on contextual cues.

Performance on Fast-Changing Samples. We conduct experiments on the video-MME, MVBench, and Ego-schema (fullset) datasets, where we utilize RAFT (Teed and Deng, 2020) for optical flow estimation to identify the 10% most dynamic samples in each category, forming our "fast-changing" subset. The following table presents a comparison between the full dataset and this fast-changing subset in terms of retained ratio and performance metrics, using PLLaVA as the baseline.

The results demonstrate that our method maintains robust performance even on fast-changing

Table 4: Performance comparison on the full dataset ("Full") and the fast-changing video subset ("Fast"). The retained ratio is averaged across three datasets.

Method	Retained Ratio (ST-Merge)		Retained Ratio (Final)		MVBench		Video-MME		Ego-schema	
	Full	Fast	Full	Fast	Full	Fast	Full	Fast	Full	Fast
Baseline	100%	100%	100%	100%	46.6	47.5	44.4	44.0	42.6	43.9
Baseline w/Ours	40.2%	44.0%	16.1%	17.6%	47.6	49.6	45.0	43.4	42.6	43.4

Table 5: Performance comparison using different retained ratios across multiple benchmarks.

Model	Retained Ratio	FLOPs(×)	MVBench	VideoMME	Egoschema (Sub/Full)	VCG Bench
PLLaVA	100.0%	1.00	46.6	44.4	47.8/42.6	2.99
PLLaVA w/Ours	24.3%	0.27	47.4	45.0	48.9/42.4	2.98
PLLaVA w/Ours	16.2%	0.23	47.6	45.0	49.0/42.6	2.98
PLLaVA w/Ours	8.1%	0.17	46.9	44.6	48.2/42.3	2.93
ST-LLM	100.0%	1.00	54.9	42.0	56.2/45.6	2.86
ST-LLM w/Ours	22.7%	0.31	54.3	41.3	54.6/45.2	2.83
ST-LLM w/Ours	15.1%	0.26	54.3	41.4	54.6/44.7	2.82
ST-LLM w/Ours	7.6%	0.21	53.9	41.6	54.2/45.0	2.76
LLaVA-OV	100.0%	1.00	58.0	58.2	62.0/60.0	3.26
LLaVA-OV w/Ours	25.5%	0.27	57.5	58.1	61.8/59.8	3.25
LLaVA-OV w/Ours	17.0%	0.20	57.5	58.6	62.6/59.5	3.24
LLaVA-OV w/Ours	8.5%	0.17	57.1	56.9	60.2/58.6	3.11

video content. Specifically, for videos with more dynamic content, the spatial-temporal merging module yields a slightly higher retained ratio (44.0% vs. 40.2% on the full set) since fewer static regions can be merged. However, after applying our attention-based pruning strategy, the final retained ratios become very similar (17.6% for fast-changing vs. 16.1% for the full set).

Overall, our approach achieves excellent efficiency even on rapidly changing samples, with only minimal performance degradation compared to the baseline. These results validate the versatility and effectiveness of our method across different types of video content, including those with substantial frame-to-frame variations.

Performance with Different Retained Ratios. To further illustrate the generalization of our method, we have extended our experiments by including additional results with various retained ratios, as detailed in Table 5. Specifically, by adjusting α while keeping all hyperparameters constant across the different models, we demonstrate that even with a pruning ratio exceeding 90%, our method can still achieve performance comparable to the original models. This result underscores the robustness and generalization capability of our approach.

More ablation studies on hyper-parameters and visualizations are provided in Appendix A.

5 Conclusion

We propose PruneVid, a training-free visual token pruning method for efficient multi-modal video understanding. By merging redundant tokens and using LLM attention to select relevant visual information, PruneVid significantly reduces computational cost while maintaining or even improving performance. Experiments show that PruneVid can prune over 80% of visual tokens with minimal impact on accuracy. Our method integrates easily with existing video LLMs and provides a practical solution for efficient long-video processing.

6 Limitations

Our method only prunes tokens based on a single layer of attention, without considering the differences and redundancy across different layers. Exploring layer-wise pruning strategies that leverage multi-layer attention information is left for future work.

Acknowledgments

This work is supported by Hong Kong Research Grant Council - Early Career Scheme (Grant No. 27208022), National Natural Science Foundation of China (Grant No. 62306251), and HKU Seed Fund for Basic Research.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *Arxiv e-prints*.

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

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

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2025. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *eccv*.

Mingjing Du, Shifei Ding, and Hongjie Jia. 2016. Study on density peaks clustering based on k-nearest neighbors and principal component analysis. *Knowledge-Based Systems*.

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. 2019. Slowfast networks for video recognition. In *ICCV*.

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

Wen Huang, Hongbin Liu, Minxin Guo, and Neil Zhenqiang Gong. 2024. Visual hallucinations of multimodal large language models. *Arxiv e-prints*.

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

Wonkyun Kim, Changin Choi, Wonseok Lee, and Wonjong Rhee. 2024. An image grid can be worth a video: Zero-shot video question answering using a vlm. *Arxiv e-prints*.

Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Wei Niu, Mengshu Sun, Xuan Shen, Geng Yuan, Bin Ren, Hao Tang, et al. 2022. Spvit: Enabling faster vision transformers via latency-aware soft token pruning. In *ECCV*.

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

KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. 2023a. Videochat: Chat-centric video understanding. *Arxiv e-prints*.

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

Kunchang Li, Yali Wang, Yizhuo Li, Yi Wang, Yinan He, Limin Wang, and Yu Qiao. 2023b. Unmasked teacher: Towards training-efficient video foundation models.

Yanwei Li, Chengyao Wang, and Jiaya Jia. 2023c. Llama-vid: An image is worth 2 tokens in large language models. In *CVPR*.

Bin Lin, Bin Zhu, Yang Ye, Munan Ning, Peng Jin, and Li Yuan. 2023. Video-llava: Learning united visual representation by alignment before projection. *Arxiv e-prints*.

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

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024a. Visual instruction tuning. *nips*.

Ruyang Liu, Chen Li, Haoran Tang, Yixiao Ge, Ying Shan, and Ge Li. 2024b. St-llm: Large language models are effective temporal learners. *Arxiv e-prints*.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. 2024c. Scissorhands: Exploiting the persistence of importance hypothesis for llm kv cache compression at test time. *NeurIPS*.

Zuyan Liu, Benlin Liu, Jiahui Wang, Yuhao Dong, Guangyi Chen, Yongming Rao, Ranjay Krishna, and Jiwen Lu. 2024d. Efficient inference of vision instruction-following models with elastic cache. *Arxiv e-prints*.

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

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

Bowen Pan, Rameswar Panda, Camilo Fosco, Chung-Ching Lin, Alex Andonian, Yue Meng, Kate Saenko, Aude Oliva, and Rogerio Feris. 2021. Va-red2: Video adaptive redundancy reduction. *arXiv*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. 2021. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *NeurIPS*.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. 2024. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *Arxiv e-prints*.

Zachary Teed and Jia Deng. 2020. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. 2022. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *nips*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *Arxiv e-prints*.

Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. 2023. Fastvit: A fast hybrid vision transformer using structural reparameterization. In *ICCV*.

Zhongwei Wan, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. 2024. Look-m: Look-once optimization in kv cache for efficient multimodal long-context inference. *Arxiv e-prints*.

Jiawei Wang, Liping Yuan, and Yuchen Zhang. 2024a. Tarsier: Recipes for training and evaluating large video description models. *Arxiv e-prints*.

Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng, Jilan Xu, Zun Wang, et al. 2024b. Internvideo2: Scaling video foundation models for multimodal video understanding. *Arxiv e-prints*.

Wenhao Wu. 2024. Freeva: Offline mllm as training-free video assistant. *Arxiv e-prints*.

Lin Xu, Yilin Zhao, Daquan Zhou, Zhijie Lin, See Kiong Ng, and Jiashi Feng. 2024a. Pllava: Parameter-free llava extension from images to videos for video dense captioning. *Arxiv e-prints*.

Mingze Xu, Mingfei Gao, Zhe Gan, Hong-You Chen, Zhengfeng Lai, Haiming Gang, Kai Kang, and Afshin Dehghan. 2024b. Slowfast-llava: A strong training-free baseline for video large language models. *Arxiv e-prints*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *Arxiv e-prints*.

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

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. 2024. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *NeurIPS*.

A Appendix

A.1 Quantitive Results

Ablation Study on Token Selection Ratio α and the Position of Pruning Layer M. As shown in Fig. 4 (a), we observe that when pruning attention from the 10th layer onward, model accuracy, despite minor fluctuations, gradually saturates. Therefore, selecting M as 10 for token pruning results in lower computational costs compared to pruning at later layers. Additionally, we find that using a larger α does not necessarily yield better results. For instance, when M is 10, an α of 0.4 achieves better accuracy than 0.5, as retaining more tokens may introduce irrelevant information, adversely affecting the outcome.

Ablation Study on threshold τ and temporal segment ratio. As depicted in Fig. 4 (b), we observe that performance consistently improves as τ increases from 0.6 to 0.8, while the performance between $\tau = 0.8$ and $\tau = 0.9$ remains similar. Given that setting $\tau = 0.8$ allows the model to merge more tokens along the temporal dimension, resulting in a higher compression ratio, we select $\tau = 0.8$. Concerning the temporal segment ratio, the variation in its values does not significantly affect performance. We found that setting it as either 0.25 or 1.0 delivers good results across the two datasets. However, since setting it as 1.0 treats each input frame as an individual segment, which hinders the effective temporal merging of static tokens, we choose to set 0.25.

Ablation Study on threshold τ and spatial merging ratio. As shown in Fig. 4 (c), the performance is comparable for τ values of 0.8 and 0.9, with $\tau = 0.8$ being slightly superior, which is a similar phenomenon as in Fig. 4 (b). Regarding the spatial merging ratio, setting it to 0.5 provides the optimal accuracy. This is because a smaller value leads to an overly aggressive merging of spatial

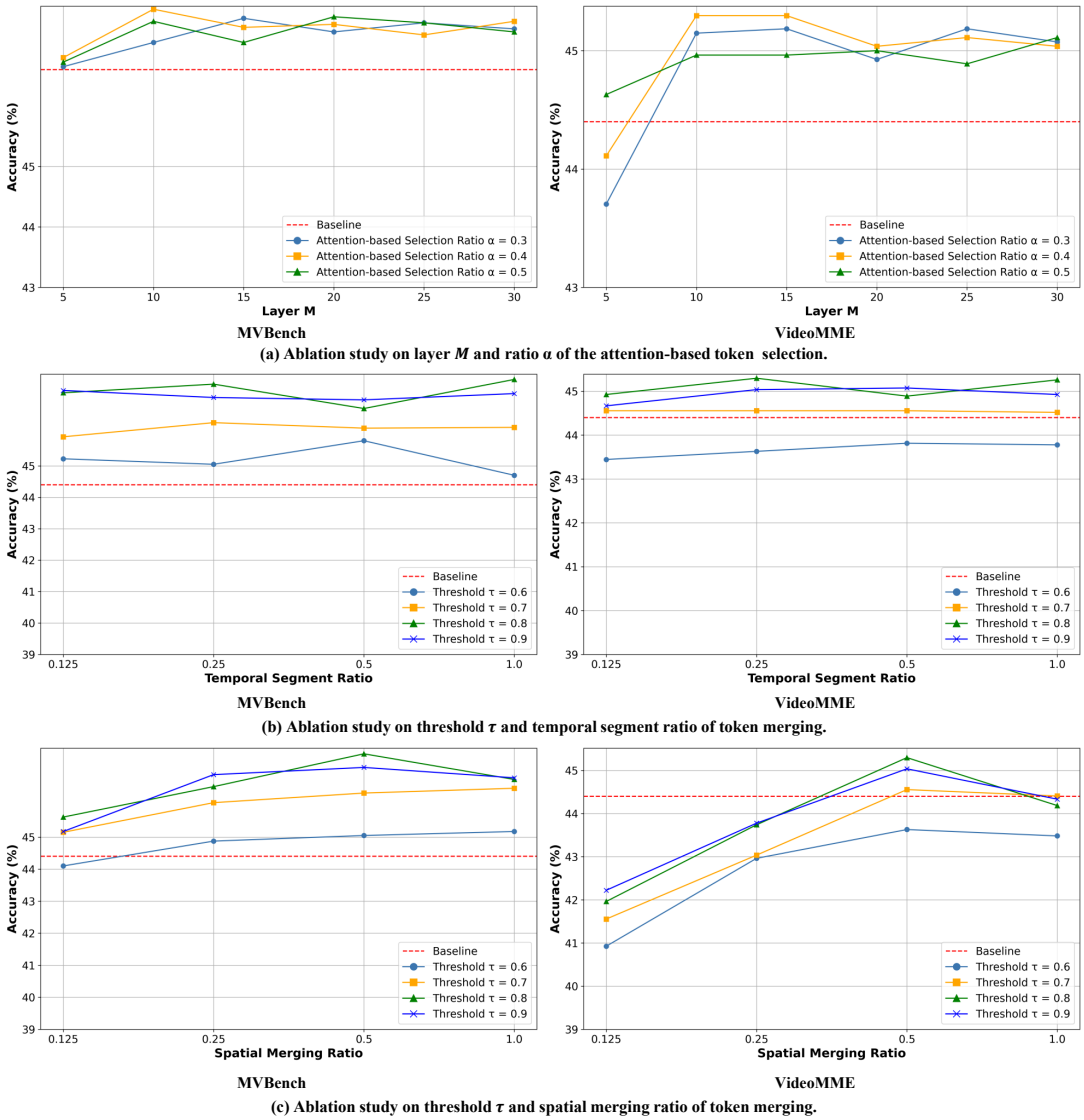

Figure 4: The ablation study of hyper-parameters on MVBench and VideoMME.

tokens, which degrades performance. On the other hand, setting it to 1.0 is unable to merge redundant tokens, which also adversely affects efficiency.

Combined with State-of-the-Art Models. We further evaluated our method on Qwen2.5 VL (Bai et al., 2025), a recent state-of-the-art MLLM. The results are summarized in Table 6. Our method achieves a significant reduction in FLOPs (by over 78%) while maintaining competitive performance on all benchmarks, demonstrating compatibility and scalability with recent SOTA models.

These results confirm that our approach can be effectively extended to SOTA models such as Qwen2.5-VL, maintaining strong performance while significantly reducing computational requirements. We additionally conduct ablation studies on Qwen2.5 VL, which further verify the compatibility of our modules with the latest SOTA architectures.

A.2 Qualitative Results

Attention-Guided Token Selection. Fig. 5 (a): Although the question does not explicitly mention a *book*, the model successfully identifies key visual regions associated with the *book* by reasoning over the visual content and contextual information provided. (b): The model effectively focuses on the person's hand movements, which are essential for answering the question. Despite the lack of explicit emphasis on hand motions in the question, the model infers their significance through contextual reasoning. (c): The model accurately attends to the action of the man taking the bag, even though the question only references the per-

Table 6: Performance on Qwen2.5 VL.

Model	Retained Ratio	FLOPs(×)	MVBench	VideoMME	Egoschema (Sub/Full)	VCGBench
Qwen 2.5 VL	100.0%	1.00	65.5	65.3	64.0/61.0	3.33
Qwen 2.5 VL w/Ours	18.1%	0.21	64.9	64.8	63.0/60.3	3.28

son on the couch. This demonstrates its ability to reason beyond the explicit scope of the question. (d): The model precisely focuses on the person's hand movements, which are highly relevant to answering the question, while disregarding irrelevant content and regions. This highlights the model's strong capacity to comprehend the question and extract critical visual information effectively.

Attention Map Comparison. In Fig. 6, we include comparisons between our LLM's attention maps and those of several strong video encoders, including UMT (Li et al., 2023b) and InternVideo2 (Wang et al., 2024b). The results show that, unlike these models, the question-to-vision attentions in the LLM accurately focus on visual tokens that are pertinent to the question. In contrast, the other models often struggle to pinpoint key tokens and may focus on irrelevant objects or background elements. These observations suggest that LLMs possess a unique ability to align visual information with linguistic context through their reasoning capabilities, which is not simply a byproduct of standard attention mechanisms in typical video encoders.

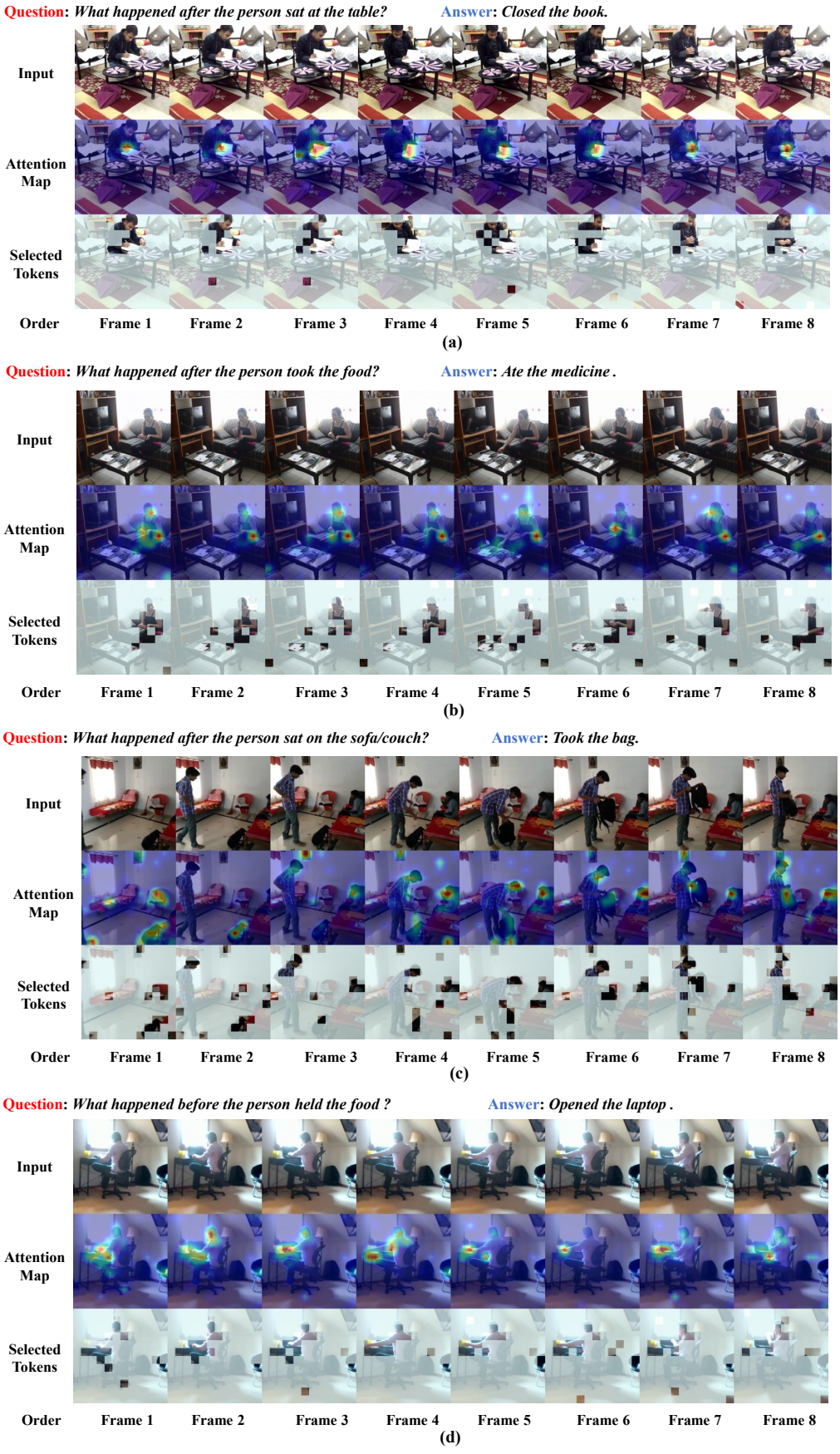

Figure 5: More examples of visualizations of the question-to-visual attentions and token selection.

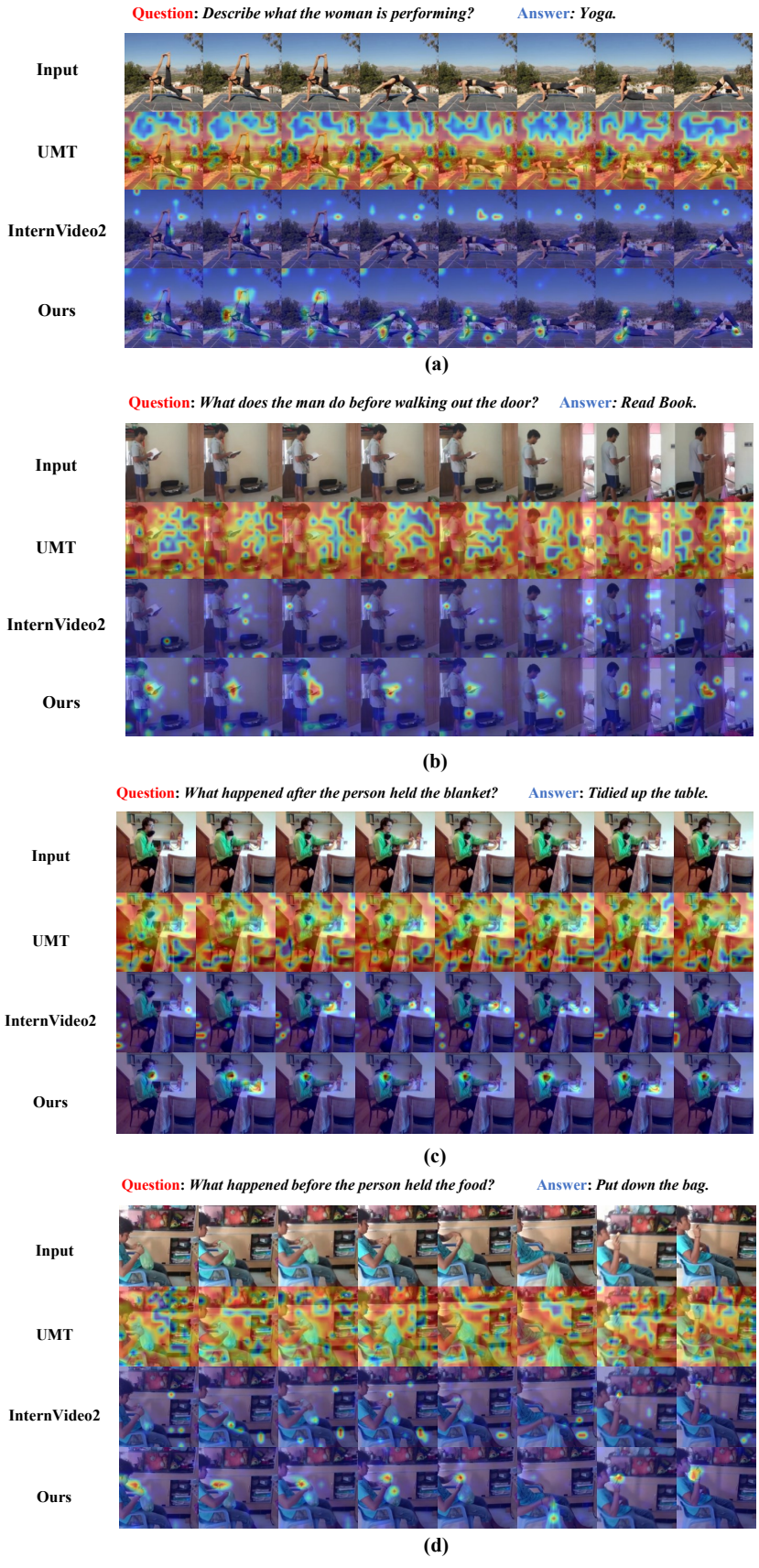

Figure 6: Attention map comparison of video encoders and our method.