PreViC: Predictive Visual Compression for Video Vision-Language Models

Anonymous ACL ARR submission

Abstract

Video vision-language models (VLMs) suffer from high inference cost because videos produce dense visual token sequences. These sequences dominate LLM prefill, making token compression essential for efficient video understanding. For deployment-oriented acceleration, compression should be training-free and pre-LLM, avoiding model-specific retraining and intrusive LLM modifications while reducing visual tokens before costly prefill begins. However, existing methods in this setting rely on static saliency, diversity, or temporal similarity, without directly modeling whether a token is predictable from retained visual context. We propose **PreViC** (**Pre**dictive **Vi**sual **C**ompression), a training-free, prompt-agnostic, pre-LLM compression method that retains visual tokens according to their predictive residuals. PreViC builds a compact visual memory from temporally novel anchor frames, then reconstructs each non-anchor token from nearby anchor tokens via ridge-regularized linear prediction. Tokens with large reconstruction errors are retained under a hierarchical budget. Across five video benchmarks on LLaVA-OneVision, PreViC consistently surpasses competing pre-LLM compression methods, with the largest gains under tight budgets and on long videos, while remaining competitive with the vanilla model on most benchmarks. These results support the principle — retain what visual memory cannot predict — as a practical foundation for video token compression.

1 Introduction

Video vision-language models (VLMs) extend multimodal reasoning from images to temporally rich videos, supporting question answering and reasoning over complex inputs (Xu et al., 2024; Hong et al., 2024; Ren et al., 2024). However, this capability introduces substantial inference cost (Tang et al., 2025). Even with only 16 sampled frames, LLaVA-OneVision produces 3137 visual tokens, occupying 97.9% of the input context in our profiling setup. This visual-token-dominated context occupies 23.8 GB of context-driven memory and accounts for 61.5% of the measured peak memory in LLaVA-OneVision-7B. These observations make visual-token compression essential for practical video VLM deployment, especially in multi-query scenarios such as video chatbots, streaming analysis, and meeting assistants (Qian et al., 2024).

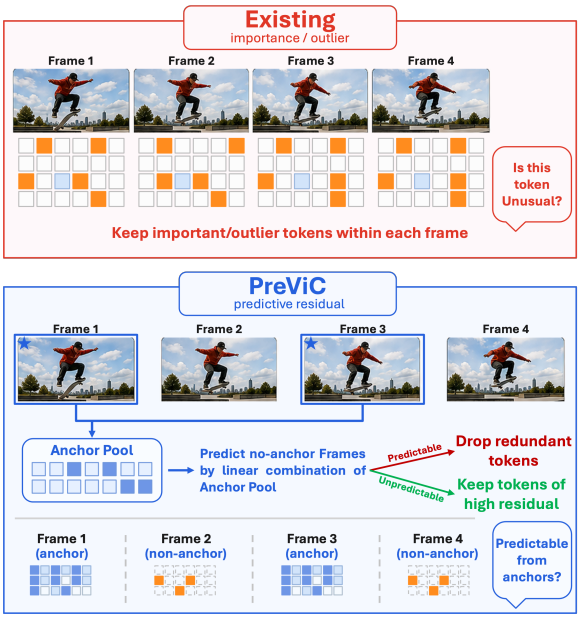


Figure 1: Comparison of PreViC and existing importance-based visual token compression methods.

For practical deployment, compression benefits from two properties. First, it should be training-free, allowing existing VLMs to apply the method without model-specific retraining or additional supervision. Second, it should operate before LLM prefill, since methods that prune after computing LLM attentions have already incurred much of the cost they aim to reduce. A pre-LLM design avoids intrusive language-model modifications and eases deployment across backbones.

Prior work on training-free compression selects visual tokens using saliency, diversity, temporal redundancy, or attention-based importance (Liu et al., 2025b; Alvar et al., 2025; Tao et al., 2025). These criteria work in many settings, but they treat token importance as an intrinsic or local property. They do not ask whether a token provides information beyond the retained visual context. A salient token may still be redundant if similar information appears across nearby frames (Fu et al., 2025b). Conversely, a subtle temporal change may matter even when it is not a strong outlier, because retained memory cannot predict such changes (Cai et al., 2024; Alexandre et al., 2023).

This limitation motivates a predictive view of video token compression. Classical video codecs reduce redundancy by keeping reference frames and encoding residuals that cannot be predicted from them (Richardson, 2011). Similarly, a video VLM should retain a token not merely because it looks salient in isolation, but because compact visual memory cannot predict it. Figure 1 illustrates this contrast: existing methods select distinctive tokens from the current visual stream, whereas PreViC preserves tokens that cannot be reconstructed from anchor memory. This shift reframes token compression from static importance estimation to reference-aware redundancy modeling.

We propose **PreViC** (**Pre**dictive **Vi**sual **C**ompression), a training-free, prompt-agnostic, pre-LLM method that retains visual tokens from videos according to their predictive residuals from an anchor-based visual memory. PreViC first selects temporally novel anchor frames to build a compact visual memory. It then reconstructs each non-anchor token from nearby anchor tokens through ridge-regularized linear prediction. The reconstruction error, or predictive residual, defines token importance. PreViC retains high-residual tokens, removes tokens that the anchor memory predicts well, and uses a hierarchical budget to preserve anchor information.

We evaluate PreViC on LLaVA-OneVision (Li et al., 2024a) across five video benchmarks. PreViC consistently surpasses competing pre-LLM compression methods under tight token budgets, and often approaches or exceeds the vanilla model despite using substantially fewer visual tokens. Efficiency profiling also shows reductions in prefill latency and memory usage. Because PreViC is prompt-agnostic, the same compressed visual representation can be reused across multiple queries for the same video. Overall, these results suggest that video token compression can be effectively framed as reference-aware redundancy modeling before LLM computation.

2 Related Work

2.1 Video Vision-Language Models

Video vision-language models (VLMs) extend image-based multimodal reasoning to temporal inputs by encoding sampled frames into visual token sequences (Fu et al., 2021; Jin et al., 2024; Alayrac et al., 2022; He et al., 2024). Even short videos produce thousands of visual tokens, which creates substantial bottlenecks in LLM prefill latency, memory consumption, and serving cost. These bottlenecks become especially pronounced in multi-query settings, where the same video may be queried repeatedly. PreViC targets this setting by compressing visual tokens before the LLM in a prompt-agnostic manner, producing reusable visual representations.

2.2 Predictive Compression in Video Codecs

Modern video codecs such as H.264/AVC and HEVC reduce bitrate by exploiting spatio-temporal redundancy: I-frames serve as reference pictures, while P- and B-frames encode only the residual between each block and its prediction from a reference frame (Lainema et al., 2012; Bross et al., 2021). Codecs thereby avoid encoding predictable content and reserve bits for unpredictable changes that carry new information. PreViC adapts this principle to visual token compression: anchor-frame tokens serve as reference memory, and predictive residuals serve as a training-free importance signal, without reproducing codec mechanisms such as motion vectors or entropy coding.

2.3 Pre-LLM Visual Token Compression

Pre-LLM compression methods reduce visual tokens before they enter the language model, directly lowering prefill cost (Shen et al., 2025; Liu et al., 2025b). Prior methods typically rely on temporal similarity, diversity, or static importance: DyCoke (Tao et al., 2025) removes temporally redundant tokens, DivPrune (Alvar et al., 2025) maximizes feature-space diversity, VidCom^2 (Liu et al., 2025b) retains visually distinctive tokens, and FastVID (Shen et al., 2025) combines multiple heuristic stages. PreViC shares the training-free pre-LLM setting but targets a different token selection criterion. PreViC scores by predictability from retained

anchor tokens—replacing per-token static importance with reference-aware redundancy modeling.

2.4 In-LLM and KV-Cache Compression

Another line of work compresses visual tokens inside the LLM or at the KV-cache level (Chen et al., 2024; Zhang et al., 2025b). FastV (Chen et al., 2024), SparseVLM (Zhang et al., 2025b), PruneVid (Huang et al., 2025), and HoliTom (Shao et al., 2025) prune or merge tokens using LLM-layer attention signals; LOOK-M (Wan et al., 2024) targets KV-cache compression. PreViC avoids these constraints by scoring tokens entirely before the LLM, eliminating the need for an LLM forward pass.

2.5 Prompt Conditioning in Compression

Visual token compression can be prompt-dependent or prompt-agnostic (Yang et al., 2025). Prompt-dependent methods select query-relevant frames or tokens using the input question, improving single-query accuracy but requiring recomputation for each query and limiting cache reuse (Zhang et al., 2025a; Di et al., 2025). Prompt-agnostic methods instead compress video content once and reuse the resulting representation across questions. PreViC deliberately targets the latter operating point: its residual scores depend only on visual tokens and anchor references, making the compression reusable but less adaptive to query-specific details.

2.6 Training-Based Visual Compression

Some methods train additional modules to compress or summarize visual tokens, such as reconstructive token compressors (Liu et al., 2025a) or language-aware dynamic compression (Ye et al., 2025). These approaches can learn semantic abstraction, but require extra training, parameters, or model-specific adaptation (Wang et al., 2025). PreViC targets a more deployable setting: it uses reconstruction error only as a training-free selection signal in the post-projector token space.

3 Method

3.1 Problem Formulation

A video VLM encodes sampled frames into visual tokens and feeds them into the LLM embedding space through a multimodal projector. We denote the projected visual-token tensor as $\mathbf{V} \in \mathbb{R}^{F \times T \times D}$, where F is the number of frames, T is the number of tokens per frame, and D is the LLM hidden dimension. Given a retain ratio $\alpha \in (0, 1]$, PreViC selects $m = \lceil \alpha F T \rceil$ tokens and feeds them to the LLM in the original temporal order.

PreViC is training-free and prompt-agnostic. It produces a keep mask $\mathbf{M} = f(\mathbf{V})$, where $\mathbf{M} \in \{0, 1\}^{F \times T}$ depends only on visual tokens and satisfies $\sum_{t=0}^{F-1} \sum_{p=1}^{T} \mathbf{M}_{t,p} = m$. Since $\mathbf{M}$ is independent of the text, the compressed visual feature can be cached and reused across multiple questions over the same video.

3.2 Overview of PreViC

PreViC adopts a predictive-coding view: instead of scoring tokens by static saliency or diversity, it asks whether each token can be predicted from a compact visual memory. As shown in Figure 2, PreViC consists of three components: R1 constructs an anchor-frame pool, R2 scores non-anchor tokens by predictive residuals, and R3 allocates the token budget between anchor tokens and high-residual non-anchor tokens. R2 is the main selection rule, while R1 and R3 provide the reference structure and an anchor-preservation floor. Appendix A provides the pseudocode.

3.3 R1: Anchor-Frame Pool

Let $\mathbf{v}_{t,p} \in \mathbb{R}^D$ denote the p-th visual token in frame t, where $t \in \{0, \ldots, F-1\}$ and $p \in \{1, \ldots, T\}$. To build a compact visual memory, PreViC first selects temporally novel anchor frames. For each frame $t > 0$, we compute a frame-level novelty score ν_t as the average token-wise change from the previous frame:

$$\nu_t = \frac{1}{T} \sum_{p=1}^{T} \|\mathbf{v}_{t,p} - \mathbf{v}_{t-1,p}\|_2^2 \qquad (1)$$

This score is a lightweight feature-space analogue of codec-style scene-cut detection (Sastre et al., 2006): frames with high novelty are less predictable from preceding frames and therefore serve as useful anchors for later residual prediction. We set $\nu_0 = +\infty$ so that the first frame is always included as an initial reference. Given the number of anchor frames a, we select the top-a frames according to ν_t and denote their indices by $\mathcal{A} = \mathrm{Top}_a(\nu_0, \ldots, \nu_{F-1})$. The anchor pool $\mathcal{P}$ is then defined as the set of all visual tokens from the selected anchor frames:

$$\mathcal{P} = \{\mathbf{v}_{t,p} \,|\, t \in \mathcal{A},\ p = 1, \ldots, T\}. \qquad (2)$$

The anchor pool serves as a compact reference memory, analogous to reference frames in predic-

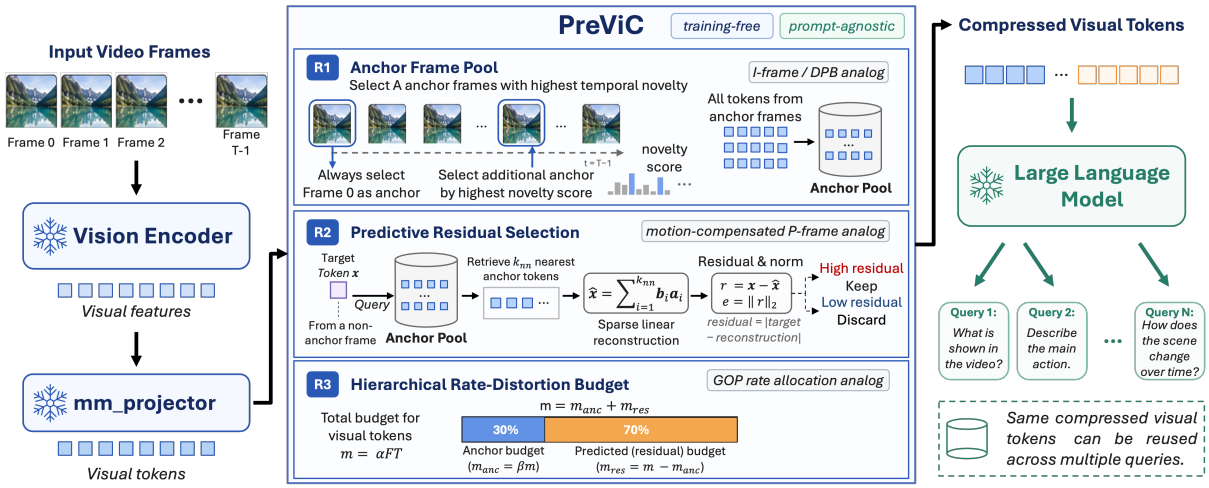


Figure 2: Overview of PreViC. Given post-projector visual tokens, R1 selects temporally novel anchor frames as visual memory, R2 scores non-anchor tokens by predictive residuals from nearby anchors, and R3 allocates the budget between preserved anchors and high-residual tokens before LLM prefill.

tive video coding. We use $a = 2$ by default; Appendix F shows anchor-count robustness.

3.4 R2: Predictive Residual Scoring

For each non-anchor token $\mathbf{x} = \mathbf{v}_{t,p} \in \mathbb{R}^D$ with $t \notin \mathcal{A}$, PreViC estimates how well $\mathbf{x}$ can be predicted from the anchor pool $\mathcal{P}$. It retrieves the k_{nn} nearest anchor tokens (default $k_{\text{nn}} = 4$) by cosine similarity, denoted $\mathcal{N}_{k_{\text{nn}}}(\mathbf{x})$. Let $\mathbf{B}_{\mathbf{x}} = [\mathbf{b}_1^\top; \ldots; \mathbf{b}_{k_{\text{nn}}}^\top] \in \mathbb{R}^{k_{\text{nn}} \times D}$ denote the row-wise anchor basis, where each row $\mathbf{b}_i^\top$ is a retrieved anchor token from $\mathcal{N}_{k_{\text{nn}}}(\mathbf{x})$. PreViC reconstructs $\mathbf{x}$ from this local anchor basis via ridge regression:

$$\begin{aligned} \mathbf{c}_{\mathbf{x}}^* &= \arg\min_{\mathbf{c}_{\mathbf{x}} \in \mathbb{R}^{k_{\text{nn}}}} \left\| \mathbf{B}_{\mathbf{x}}^\top \mathbf{c}_{\mathbf{x}} - \mathbf{x} \right\|_2^2 + \lambda \left\| \mathbf{c}_{\mathbf{x}} \right\|_2^2 \\ &= \left(\mathbf{B}_{\mathbf{x}} \mathbf{B}_{\mathbf{x}}^\top + \lambda \mathbf{I} \right)^{-1} \mathbf{B}_{\mathbf{x}} \mathbf{x}. \end{aligned} \tag{3}$$

The reconstructed token is $\hat{\mathbf{x}} = \mathbf{B}_{\mathbf{x}}^\top \mathbf{c}_{\mathbf{x}}^*$, and the predictive residual score is $\rho(\mathbf{x}) = \|\mathbf{x} - \hat{\mathbf{x}}\|_2$. A large residual indicates that $\mathbf{x}$ contains information not well explained by the anchor memory, whereas a small residual indicates that $\mathbf{x}$ is predictable from existing anchor tokens. We use the Euclidean norm for interpretability; the squared norm gives the same ranking.

Equation (3) should be interpreted as a local predictability model in the post-projector visual-token space, not as an assumption that LLM hidden states are globally linear. This follows the intuition of local linear reconstruction (Roweis and Saul, 2000): nearby tokens provide a compact basis, and the residual measures the component unexplained by that local neighborhood. This view is also consistent with prior analyses showing meaningful linear structure in multimodal representation spaces (Bhalla et al., 2024).

We use cosine similarity for scale-insensitive neighbor retrieval, while measuring the residual in the post-projector feature space consumed by the LLM. Cosine similarity retrieves semantically similar anchors regardless of token norm, whereas the L2 residual preserves both directional and magnitude differences that the anchor basis cannot explain. Appendix K shows that raw post-projector features outperform several normalized variants, supporting this default metric choice.

We choose the ridge coefficient adaptively for numerical stability:

$$\lambda = \max\left(\epsilon,\ \epsilon \cdot \frac{\operatorname{tr}(\mathbf{B}_{\mathbf{x}} \mathbf{B}_{\mathbf{x}}^\top)}{k_{\text{nn}}} \right), \quad \epsilon = 10^{-3}. \tag{4}$$

This adaptive choice scales the regularization with the average squared norm of retrieved anchor tokens and stabilizes the solution when nearest anchors are highly correlated. Appendix K shows that PreViC is robust to a wide range of λ values.

3.5 R3: Hierarchical Rate Allocation

R3 allocates the global token budget $m = \lceil \alpha FT \rceil$ between anchor tokens and residual-selected non-anchor tokens. Since anchor tokens are themselves the reference memory, their predictive residual is near zero by construction; a purely residual-based ranking would therefore discard anchor tokens and destroy the reference structure used in

Figure 3: Qualitative visualization of PreViC token selection at $\alpha = 0.3$. For each frame, we show the original image, the R2 predictive residual map, and the retained token locations. Anchor frames are marked in red.

R2. R3 resolves this by reserving a fraction of the budget for anchor preservation. We first define a nominal anchor budget $\widehat{m}_{\text{anc}} = \lfloor \beta m \rfloor$, with $\beta = 0.3$ by default. The anchor pool contains exactly $N_{\text{anc}} = aT$ tokens, and we set $N_{\text{res}} = (F - a)T$. The actual anchor budget m_{anc} is obtained by clipping $\widehat{m}_{\text{anc}}$ to the feasible range $\big[\max(0, m - N_{\text{res}}),\ \min(m, N_{\text{anc}})\big]$, and the residual budget is $m_{\text{res}} = m - m_{\text{anc}}$. Thus, β specifies a target anchor fraction rather than a guaranteed final fraction. Algorithm 1 details the clipping rules for edge cases. When $m_{\text{anc}} = N_{\text{anc}}$, all anchors are kept and the surplus goes to residual-selected tokens; otherwise, anchor tokens are ranked by distance to the anchor-pool mean $s(\mathbf{q}) = \|\mathbf{q} - \bar{\mathbf{q}}\|_2$ and the top-m_{anc} anchor tokens are retained. PreViC then retains the top-m_{res} non-anchor tokens by $\rho(\mathbf{x})$.

R3 acts as an anchor-preservation floor rather than the primary ranking criterion. Residuals are computed using the full anchor pool $\mathcal{P}$ as the scoring memory, while R3 determines which anchor tokens are emitted in the final compressed sequence. Appendix K shows that alternative anchor-overflow rules yield similar results.

3.6 Qualitative Visualization of PreViC

Figure 3 illustrates PreViC on a 16-frame video: R1 selects the first and the 15th frames as anchors, while R2 concentrates retained tokens on moving objects and human-object interactions rather than static backgrounds. This confirms the intended behavior: anchor frames provide reference memory, and residual scoring preserves temporally novel regions. See Appendix J for more detail.

4 Experiments

We investigate two central questions: (i) whether PreViC is competitive with strong baselines across token budgets (Table 1), and (ii) whether the ad-

Table 1: Main budget-aware comparison on LLaVA-OV-7B across video understanding benchmarks. Target Budget denotes the approximate retained-token ratio. Rows are grouped by approximate retained-token ratio and compression locus. Pre-LLM methods select tokens before any LLM forward pass, whereas in-LLM or hybrid methods may use LLM-layer signals for selection. Bold indicates the best score within each budget and locus group.

Target Budget	Locus	Method	MVBench	Video-MME	EgoSchema	LVB	MLVU
Full tokens		Vanilla	54.12	69.80	61.60	55.72	62.47
∼30% retention	in-LLM	PruneVid	53.40	64.60	61.00	55.42	62.14
		HoliTom	**54.58**	**67.00**	**63.40**	56.84	**64.81**
		AIM	53.62	65.40	61.00	**58.27**	63.66
	pre-LLM	DivPrune	52.88	66.20	62.00	55.12	62.88
		DyCoke	53.58	65.60	61.20	54.15	62.56
		FastVID	52.08	64.60	60.60	55.27	61.18
		VidCom^2	52.95	65.40	61.00	55.12	61.82
		PreViC	**53.85**	**69.40**	**62.60**	**57.74**	**64.21**
∼50% retention	in-LLM	PruneVid	53.38	64.60	61.00	55.42	62.14
		HoliTom	**55.17**	**70.60**	**63.20**	56.55	**65.09**
		AIM	53.73	65.40	60.80	**58.27**	63.66
	pre-LLM	DivPrune	53.45	66.00	61.20	55.20	63.16
		DyCoke	**54.17**	**70.40**	61.80	55.12	62.28
		FastVID	52.75	67.00	61.60	55.80	61.59
		VidCom^2	53.92	66.40	61.00	55.12	61.82
		PreViC	53.62	**70.40**	**62.60**	**57.07**	**65.13**
∼70% retention	in-LLM	PruneVid	53.48	67.40	61.00	55.42	62.14
		HoliTom	**55.08**	69.20	**62.40**	56.69	**65.50**
		AIM	53.62	**69.40**	60.60	**58.27**	63.66
	pre-LLM	DivPrune	53.95	69.80	62.00	54.90	**62.37**
		DyCoke	53.70	66.20	61.60	55.27	61.27
		FastVID	53.25	67.20	**62.60**	**55.72**	61.73
		VidCom^2	54.15	**72.20**	61.00	55.12	61.82
		PreViC	**54.17**	70.80	**62.60**	55.12	62.01

vantage of predictive residual scoring grows as the budget tightens (long-video columns of Table 1 and the component ablation in Table 5).

4.1 Experimental Setup

Models and benchmarks. We evaluate PreViC on LLaVA-OneVision (Li et al., 2024a) 0.5B and 7B. The 0.5B backbone allows us to examine whether removing predictable visual redundancy benefits smaller video VLMs, whereas the 7B backbone evaluates efficiency gains in a regime where LLM prefill is substantially more expensive. We report results on MVBench (Li et al., 2024b), Video-MME (Fu et al., 2025a), EgoSchema (Mangalam et al., 2023), LongVideoBench (Wu et al., 2024), and MLVU (Zhou et al., 2025). MVBench is evaluated over 20 tasks with 200 samples per task, and Video-MME is evaluated on the short-video subset with a 500-sample limit.

Baselines. We compare PreViC with DivPrune (Alvar et al., 2025), DyCoke (Tao et al., 2025), VidCom^2 (Liu et al., 2025b), FastVID (Shen et al., 2025), PruneVid (Huang et al., 2025), AIM (Zhong et al., 2025), and HoliTom (Shao et al., 2025). These baselines cover diversity-based pruning, temporal merging, outlier-based scoring, dynamic pruning, and hybrid pre-/in-LLM compression. Vanilla denotes the uncompressed model. We also report each method's compression locus, since in-LLM or hybrid methods may require additional LLM-layer computation to obtain pruning signals.

PreViC settings. Unless otherwise specified, PreViC uses $a = 2$, $k_{\mathrm{nn}} = 4$, $\beta = 0.3$. The main comparison reports $\alpha \in \{0.3, 0.5, 0.7\}$, while full retain-ratio sweeps are provided in Appendix D. Anchor-count analysis appears in Appendix F, nearest-neighbor size sensitivity in Appendix G, and feature-space and residual-scoring sensitivity analyses in Appendix K. A complete summary of notation is provided in Appendix B.

4.2 Budget-Aware Benchmark Comparison

Table 1 reports the main budget-aware comparison on LLaVA-OV-7B. Methods are grouped by retained-token ratio and compression locus, allowing pre-LLM methods to be compared with in-LLM

Table 2: Accuracy–latency Pareto on LLaVA-OV-0.5B Video-MME. TTFT includes preprocessing.

Method	Ret.	Score	Δ	TTFT (ms)
Vanilla	100%	53.60	0.00	350.19
DivPrune	50%	54.20	+0.60	508.92
DyCoke	30%	53.00	-0.60	318.30
FastVID	30%	51.60	-2.00	336.54
VidCom^2	∼68%	55.00	+1.40	333.60
PreViC	30%	**57.20**	**+3.60**	**300.95**
PreViC	50%	56.00	+2.40	318.47
PreViC	70%	55.60	+2.00	331.65

or hybrid methods under similar budgets. Among pre-LLM compressors, PreViC is most effective under tight budgets: at the ∼30% budget, it achieves the best pre-LLM score on all five benchmarks, and at the ∼50% budget, it remains the best or tied for the best on four benchmarks. This supports our hypothesis that predictive residual scoring is particularly useful when only a small number of visual tokens can be retained.

Compared with in-LLM or hybrid methods such as HoliTom and AIM, PreViC does not use LLM-layer attention or hidden states for selection. Instead, it scores tokens before the LLM, directly shortens the prefill sequence, and produces prompt-agnostic visual tokens that can be reused across queries. The corresponding 0.5B comparison is provided in Appendix C.

Appendix E further provides task-wise MVBench results, showing that this pattern is not uniform across task types. PreViC consistently improves performance on object- and event-centric tasks where task-relevant evidence appears as temporally distinctive changes, but can degrade performance on counting or permutation-style tasks that require dense frame coverage.

4.3 Accuracy–Latency Pareto on Video-MME

Table 2 reports the accuracy–latency Pareto comparison on LLaVA-OV-0.5B Video-MME. For baselines, we report the strongest available setting under comparable retained-token budgets. At 30% retention, PreViC improves accuracy from 53.60 to 57.20 while reducing TTFT (Time-To-First-Token) from 350.19 ms to 300.95 ms, demonstrating simultaneous gains in accuracy and latency.

4.4 Prefill Acceleration and Memory Savings

Table 3 reports the efficiency comparison on LLaVA-OV-7B. At 70% retention, PreViC reduces prefill latency from 263.26 ms to 184.60 ms and E2E latency from 471.62 ms to 381.70 ms while slightly improving MVBench accuracy. At 30% retention, it further reduces prefill latency to 83.20 ms and E2E latency to 265.20 ms, while lowering peak memory from 38.69 GB to 23.54 GB. These improvements stem from shortening the visual sequence before LLM prefill, without incurring additional in-LLM scoring overhead. Appendix I further provides a stage-wise timing breakdown, showing that the full PreViC compression stage accounts for only 0.42–0.68% of E2E latency.

Table 3: Detailed efficiency on LLaVA-OV-7B. Prefill denotes LLM prefill latency after visual-token compression; E2E includes visual preprocessing, compression, prefill, and decoding. Latency is in milliseconds and memory in GB. Acc. denotes MVBench accuracy.

Method	Ret.	Prefill	E2E	Mem.	Acc.
Vanilla	100%	263.26	471.62	38.69	54.12
DyCoke	51.6%	143.12	335.25	27.98	54.17
VidCom^2	70.0%	184.64	383.06	31.73	54.15
FastVID	69.9%	183.78	382.41	31.70	53.25
DivPrune	70.0%	183.12	524.93	31.91	53.95
PruneVid	50.0%	139.29	346.37	27.69	53.70
PreViC	70.0%	184.60	381.70	31.73	**54.17**
PreViC	50.0%	136.30	321.90	27.49	53.62
PreViC	30.0%	**83.20**	**265.20**	**23.54**	53.85

Table 4: Empirical multi-query reuse on one video with LLaVA-OV-7B and PreViC at $\alpha = 0.7$. n indicates the number of questions asked about the same video.

n	Naive (ms)	PreViC reuse (ms)	Speedup
1	960	960	1.00×
2	1921	1583	1.21×
4	3841	2830	1.36×
8	7683	5322	1.44×

4.5 Prompt-Agnostic Multi-Query Reuse

We directly validate the prompt-agnostic reuse motivation from the introduction by measuring multi-query speedup: because PreViC derives its keep mask from the video alone, the compressed visual representation can be reused across different questions. Table 4 reports an empirical reuse setting in which preprocessing and compression are performed for the first query, while subsequent queries reuse the cached visual preprocessing. The speedup reaches 1.44× for eight questions on the same video. Appendix L details the caching assumptions, empirical reuse formula, and analytical upper-bound scenario.

Table 5: Summary of R1/R2/R3 component ablation on LLaVA-OV-0.5B MVBench. R2-on/off averages are computed over all component configurations with or without R2, respectively.

Setting	$\alpha = 0.7$	$\alpha = 0.5$	$\alpha = 0.3$
R2-off average	46.15	45.21	44.38
R2-on average	45.93	46.42	45.91
R2 contribution	-0.23	**+1.21**	**+1.54**
Full PreViC	**46.08**	**46.58**	45.75

4.6 Component Analysis

Table 5 summarizes the R1/R2/R3 component ablation. R2 is the primary component: configurations that include R2 improve over those without it by $+1.21$ at 50% retention and $+1.54$ at 30% retention; at 70% retention, where budget constraints are loose, R2's contribution is slightly negative (-0.23). The R1 novelty-based anchor selection contributes, especially on long-video benchmarks (Table 14), while R3 provides a stabilization floor. Details are provided in Appendices F, H, and K. Together with the long-video pattern in Table 1, this answers question (ii): the predictive-residual advantage grows monotonically as the visual-token budget tightens.

5 Discussion

Pre-LLM efficiency, composability, and multi-query reuse. PreViC is architecturally compatible with in-LLM methods rather than a replacement for them. PreViC removes tokens that visual memory can predict, whereas in-LLM methods can remove tokens that the query deems irrelevant. Because these criteria are computed at different stages, they could be combined without joint training. We leave a systematic evaluation of such combined pipelines to future work. Because selection depends only on visual tokens, the compressed representation can be cached and reused across queries. The $1.44\times$ speedup at n=8 queries in a representative reuse setting (Table 4) demonstrates the practical benefit of caching video-side preprocessing and compression. In the current implementation, this reuse skips repeated visual preprocessing; stronger reuse through visual KV-cache passthrough would require additional serving-system support, as discussed in Appendix L.

Predictive residuals: scope and limits. PreViC's predictive criterion succeeds in specific task regimes that align with its temporal selection bias. Task-wise analysis (Appendix E) reveals when this selection criterion succeeds and when it fails: PreViC consistently improves performance on object- and event-centric tasks, where task-relevant evidence appears as temporally novel changes, but degrades performance on counting and permutation tasks, where evidence is distributed uniformly across frames. These results show that the criterion is structurally bounded by the temporal properties of the video—effective where temporal change is concentrated and limited where it is not.

Why predictive residuals matter more under tight budgets. Predictive residuals are most effective when temporal redundancy is high and the budget is tight: PreViC shows the largest gains on long-video benchmarks (LVB $+2.02$, MLVU $+1.74$, EgoSchema $+1.00$ at 30% budget in Table 1), while short-video tasks show smaller or negative gaps (MVBench -0.27, Video-MME -0.40). The component ablation confirms this trend: R2's contribution grows from $+1.21$ at $\alpha = 0.5$ to $+1.54$ at $\alpha = 0.3$, but becomes slightly negative (-0.23) at $\alpha = 0.7$, where the budget is large enough that residual scoring is no longer decisive.

6 Conclusion

We presented PreViC, a training-free and prompt-agnostic pre-LLM compression method for video VLMs. Instead of estimating token importance in isolation, PreViC retains visual tokens that cannot be predicted from an anchor-based visual memory, reframing compression as reference-aware redundancy modeling. Across five video benchmarks, PreViC shows strong accuracy–efficiency trade-offs under explicit token budgets. It performs particularly well among pre-LLM methods under aggressive compression, improves the accuracy–latency Pareto on 0.5B Video-MME, and reduces 7B prefill latency and memory usage without relying on LLM-layer signals. Ablations show that predictive residual scoring is the main contributor under tighter token budgets.

These results suggest a simple principle for efficient video VLMs: visual tokens should be scored by the information they add beyond retained visual memory. We believe this predictive-residual perspective provides a lightweight foundation for pre-LLM video token compression and suggests a natural interface with later prompt-aware, in-LLM, or KV-cache compression stages.

Limitations

Prompt-agnostic design constrains query-specific selection. PreViC selects tokens based solely on visual structure, without access to the query or LLM signals. This design choice enables reuse across multiple queries but prevents token selection from adapting to query-specific content: for questions targeting specific objects, brief events, or fine-grained spatial details, a prompt-aware selection stage could recover tokens that PreViC cannot identify from visual structure alone. PreViC therefore functions as a reusable first stage that removes temporal redundancy, complementing rather than replacing query-driven selection methods.

Residual scoring measures feature-space predictability, not semantic relevance. A low predictive residual indicates only that a token is reconstructible in the post-projector feature space—it does not imply semantic irrelevance for the downstream task. Conversely, a high residual may reflect background motion, camera changes, or feature-space artifacts rather than task-relevant new content. Our sensitivity analyses show that raw projected features perform well across evaluated benchmarks, suggesting that post-projector feature geometry aligns sufficiently with task-relevant semantics in the tested regimes; however, this alignment may not hold across all backbone architectures.

Local predictability and retrieval assumptions. R2 uses local reconstruction in the post-projector feature space as a proxy for visual predictability, with neighbors retrieved by feature similarity. This proxy can be imperfect when feature-nearest anchors are spatially or temporally mismatched, or when strong camera motion, occlusion, or nonlinear appearance changes distort local correspondence. Future work could incorporate spatial-temporal constraints or adaptive retrieval while preserving the same residual-scoring principle. Performance remains stable within the swept anchor and residual-scoring hyperparameters (Appendices F and K), but we did not evaluate extreme long-form or high-motion videos.

Preprocessing cost may grow for very long videos. The preprocessing cost of PreViC—nearest-anchor search and ridge reconstruction—scales with token count, anchor-pool size, and feature dimension. This overhead is negligible in our evaluated settings (0.42–0.68% of E2E latency), but grows with video length and backbone size; for very long videos, approximate nearest-neighbor search, temporal blocking, or other scalable retrieval strategies may be needed.

Hyperparameter sensitivity and adaptivity. The best retain ratio varies across benchmarks and tasks: aggressive compression benefits videos with high temporal redundancy, while tasks requiring dense visual coverage may benefit from more tokens. The fixed hyperparameters a=2, k_{nn}=4, and β=0.3 are validated over our benchmark suite but may not generalize to all video types or backbone architectures.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, and 1 others. 2022. Flamingo: a visual language model for few-shot learning. In *Proceedings of the Advances in neural information processing systems*, volume 35, pages 23716–23736.

David Alexandre, Hsueh-Ming Hang, and Wen-Hsiao Peng. 2023. Hierarchical b-frame video coding using two-layer canf without motion coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10249–10258.

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. 2025. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 9392–9401.

Usha Bhalla, Alex Oesterling, Suraj Srinivas, Flavio P Calmon, and Himabindu Lakkaraju. 2024. Interpreting clip with sparse linear concept embeddings (splice). *Proceedings of the Advances in Neural Information Processing Systems*, 37:84298–84328.

Benjamin Bross, Jianle Chen, Jens-Rainer Ohm, Gary J Sullivan, and Ye-Kui Wang. 2021. Developments in international video coding standardization after avc, with an overview of versatile video coding (vvc). *Proceedings of the IEEE*, 109(9):1463–1493.

Mu Cai, Reuben Tan, Jianrui Zhang, Bocheng Zou, Kai Zhang, Feng Yao, Fangrui Zhu, Jing Gu, Yiwu Zhong, Yuzhang Shang, and 1 others. 2024. Temporalbench: Benchmarking fine-grained temporal understanding for multimodal video models. *arXiv preprint arXiv:2410.10818*.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2024. 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*, pages 19–35. Springer.

Shangzhe Di, Zhelun Yu, Guanghao Zhang, Haoyuan Li, Hao Cheng, Bolin Li, Wanggui He, Fangxun Shu, and Hao Jiang. 2025. Streaming video question-answering with in-context video kv-cache retrieval. In *Proceedings of the International Conference on Learning Representations*, volume 2025, pages 42115–42127.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, and 1 others. 2025a. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24108–24118.

Tianyu Fu, Tengxuan Liu, Qinghao Han, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang. 2025b. Framefusion: Combining similarity and importance for video token reduction on large vision language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22654–22663.

Tsu-Jui Fu, Linjie Li, Zhe Gan, Kevin Lin, William Yang Wang, Lijuan Wang, and Zicheng Liu. 2021. Violet: End-to-end video-language transformers with masked visual-token modeling. *arXiv preprint arXiv:2111.12681*.

Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. 2024. Ma-lmm: Memory-augmented large multimodal model for long-term video understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13504–13514.

Wenyi Hong, Weihan Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yan Wang, Yean Cheng, Shiyu Huang, Junhui Ji, Zhao Xue, and 1 others. 2024. Cogvlm2: Visual language models for image and video understanding. *arXiv preprint arXiv:2408.16500*.

Xiaohu Huang, Hao Zhou, and Kai Han. 2025. Prunevid: Visual token pruning for efficient video large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 19959–19973.

Yang Jin, Zhicheng Sun, Kun Xu, Liwei Chen, Hao Jiang, Quzhe Huang, Chengru Song, Yuliang Liu, Di Zhang, Yang Song, and 1 others. 2024. Video-lavit: Unified video-language pre-training with decoupled visual-motional tokenization. *arXiv preprint arXiv:2402.03161*.

Jani Lainema, Frank Bossen, Woo-Jin Han, Junghye Min, and Kemal Ugur. 2012. Intra coding of the hevc standard. *IEEE transactions on circuits and systems for video technology*, 22(12):1792–1801.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and 1 others. 2024a. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

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

Xiangrui Liu, Yan Shu, Zheng Liu, Ao Li, Yang Tian, and Bo Zhao. 2025a. Video-xl-pro: Reconstructive token compression for extremely long video understanding. *arXiv preprint arXiv:2503.18478*.

Xuyang Liu, Yiyu Wang, Junpeng Ma, and Linfeng Zhang. 2025b. Video compression commander: Plug-and-play inference acceleration for video large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1910–1924.

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

Rui Qian, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Shuangrui Ding, Dahua Lin, and Jiaqi Wang. 2024. Streaming long video understanding with large language models. *Proceedings of the Advances in Neural Information Processing Systems*, 37:119336–119360.

Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. 2024. Timechat: A time-sensitive multimodal large language model for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14313–14323.

Iain E Richardson. 2011. *The H. 264 advanced video compression standard*. John Wiley & Sons.

Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326.

Jorge Sastre, Pau Usach, A Moya, Valery Naranjo, and JM López. 2006. Shot detection method for low bitrate h. 264 video coding. In *2006 14th European Signal Processing Conference*, pages 1–5. IEEE.

Kele Shao, Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. 2025. Holitom: Holistic token merging for fast video large language models. *Proceedings of the Advances in Neural Information Processing Systems*, 38:135547–135570.

Leqi Shen, Guoqiang Gong, Tao He, Yifeng Zhang, Pengzhang Liu, Sicheng Zhao, and 1 others. 2025. Fastvid: Dynamic density pruning for fast video large language models. *Proceedings of the Advances in Neural Information Processing Systems*, 38:123553–123581.

Yunlong Tang, Jing Bi, Siting Xu, Luchuan Song, Susan Liang, Teng Wang, Daoan Zhang, Jie An, Jingyang Lin, Rongyi Zhu, and 1 others. 2025. Video understanding with large language models: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*.

Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. 2025. Dycoke: Dynamic compression of tokens for fast video large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18992–19001.

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. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4065–4078.

Xiangchen Wang, Jinrui Zhang, Teng Wang, Haigang Zhang, and Feng Zheng. 2025. Seeing more, saying more: Lightweight language experts are dynamic video token compressors. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 541–558.

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

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

Yanlai Yang, Zhuokai Zhao, Satya Narayan Shukla, Aashu Singh, Shlok Kumar Mishra, Lizhu Zhang, and Mengye Ren. 2025. Streammem: Query-agnostic kv cache memory for streaming video understanding. *arXiv preprint arXiv:2508.15717*.

Xubing Ye, Yukang Gan, Xiaoke Huang, Yixiao Ge, and Yansong Tang. 2025. Voco-llama: Towards vision compression with large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29836–29846.

Shaojie Zhang, Jiahui Yang, Jianqin Yin, Zhenbo Luo, and Jian Luan. 2025a. Q-frame: Query-aware frame selection and multi-resolution adaptation for video-llms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22056–22065.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis A Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and 1 others. 2025b. Sparsevlm: Visual token sparsification for efficient vision-language model inference. In *Proceedings of the International Conference on Machine Learning*, pages 74840–74857. PMLR.

Yiwu Zhong, Zhuoming Liu, Yin Li, and Liwei Wang. 2025. Aim: Adaptive inference of multi-modal llms via token merging and pruning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20180–20192.

Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Zhengyang Liang, Shitao Xiao, Minghao Qin, Xi Yang, Yongping Xiong, Bo Zhang, and 1 others. 2025. Mlvu: Benchmarking multi-task long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13691–13701.

A Pseudocode

Algorithm 1 PreViC

Require: Visual tokens $\mathbf{V} \in \mathbb{R}^{F\times T\times D}$, retain ratio α, anchor count a, anchor budget fraction β, neighbor count k_{nn}

Ensure: Compressed visual token sequence $\tilde{\mathbf{V}}$

$m \leftarrow \lceil \alpha F T \rceil$

Compute frame novelty $\nu_t = \frac{1}{T}\sum_{p=1}^{T} \|\mathbf{v}_{t,p} - \mathbf{v}_{t-1,p}\|_2^2$ for $t > 0$

Set $\nu_0 \leftarrow +\infty$

$\mathcal{A} \leftarrow \text{Top}_a(\nu_0, \ldots, \nu_{F-1})$

$\mathcal{P} \leftarrow \{\mathbf{v}_{t,p} \mid t \in \mathcal{A},\ p = 1, \ldots, T\}$

$N_{\text{anc}} \leftarrow |\mathcal{P}| = aT$

$N_{\text{res}} \leftarrow (F - a)T$

$\widehat{m}_{\text{anc}} \leftarrow \lfloor \beta m \rfloor$

$m_{\text{anc}}^{\min} \leftarrow \max(0,\ m - N_{\text{res}})$

$m_{\text{anc}}^{\max} \leftarrow \min(m,\ N_{\text{anc}})$

$m_{\text{anc}} \leftarrow \text{clip}\big(\widehat{m}_{\text{anc}},\ m_{\text{anc}}^{\min},\ m_{\text{anc}}^{\max}\big)$ ▷ clamp to $[m_{\text{anc}}^{\min},\ m_{\text{anc}}^{\max}]$

$m_{\text{res}} \leftarrow m - m_{\text{anc}}$

if $m_{\text{anc}} = N_{\text{anc}}$ **then**

  $\mathcal{S}_{\text{anc}} \leftarrow \mathcal{P}$

else

  $\bar{\mathbf{q}} \leftarrow \frac{1}{N_{\text{anc}}}\sum_{\mathbf{q}\in\mathcal{P}} \mathbf{q}$

  Score each anchor token by $s(\mathbf{q}) = \|\mathbf{q} - \bar{\mathbf{q}}\|_2$

  $\mathcal{S}_{\text{anc}} \leftarrow \text{Top}_{m_{\text{anc}}}\{s(\mathbf{q}) \mid \mathbf{q} \in \mathcal{P}\}$

end if

for all $\mathbf{x} = \mathbf{v}_{t,p}$ where $t \notin \mathcal{A}$ **do**

  $\mathcal{N}_{k_{\text{nn}}}(\mathbf{x}) \leftarrow \text{Top}_{k_{\text{nn}}}\{\cos(\mathbf{x}, \mathbf{q}) \mid \mathbf{q} \in \mathcal{P}\}$

  Form $\mathbf{B}_{\mathbf{x}} = [\mathbf{b}_1^\top; \ldots; \mathbf{b}_{k_{\text{nn}}}^\top]$ from $\mathcal{N}_{k_{\text{nn}}}(\mathbf{x})$

  $\lambda \leftarrow \max\big(\epsilon,\ \epsilon \cdot \text{tr}(\mathbf{B}_{\mathbf{x}}\mathbf{B}_{\mathbf{x}}^\top)/k_{\text{nn}}\big)$ ▷ $\epsilon = 10^{-3}$, Eq. (4)

  $\mathbf{c}_{\mathbf{x}}^* \leftarrow (\mathbf{B}_{\mathbf{x}}\mathbf{B}_{\mathbf{x}}^\top + \lambda \mathbf{I})^{-1}\mathbf{B}_{\mathbf{x}}\mathbf{x}$

  $\hat{\mathbf{x}} \leftarrow \mathbf{B}_{\mathbf{x}}^\top \mathbf{c}_{\mathbf{x}}^*$

  $\rho(\mathbf{x}) \leftarrow \|\mathbf{x} - \hat{\mathbf{x}}\|_2$

end for

$\mathcal{S}_{\text{res}} \leftarrow$ top-m_{res} non-anchor tokens by $\rho(\mathbf{x})$

$\tilde{\mathbf{V}} \leftarrow \mathcal{S}_{\text{anc}} \cup \mathcal{S}_{\text{res}}$ flattened in original temporal order

return $\tilde{\mathbf{V}}$

Table 6: Notation summary throughout the paper.

Symbol	Meaning	Symbol	Meaning
$\mathbf{V} \in \mathbb{R}^{F\times T\times D}$	Projected visual-token tensor after the multimodal projector	$\mathbf{v}_{t,p} \in \mathbb{R}^D$	p-th projected visual token in frame t
F	Number of sampled video frames	T	Number of visual tokens per frame
D	Hidden dimension of each projected visual token	t	Frame index, $t \in \{0, \ldots, F-1\}$
p	Token index within a frame, $p \in \{1, \ldots, T\}$	$\alpha \in (0, 1]$	Global retain ratio
$m = \lceil \alpha F T \rceil$	Total number of visual tokens retained after compression	$\mathbf{M} \in \{0, 1\}^{F\times T}$	Binary keep mask over visual tokens
a	Number of anchor frames	$\mathcal{A}$	Set of selected anchor-frame indices
ν_t	Frame-level temporal novelty score for frame t	$\mathcal{P}$	Anchor-token pool constructed from frames in $\mathcal{A}$
$\mathbf{x}$	Non-anchor target token to be scored	k_{nn}	Number of nearest anchor tokens used for local reconstruction
$\mathcal{N}_{k_{\text{nn}}}(\mathbf{x})$	Set of k_{nn} nearest anchor tokens for $\mathbf{x}$	$\mathbf{B}_{\mathbf{x}} \in \mathbb{R}^{k_{\text{nn}}\times D}$	Local anchor basis retrieved for token $\mathbf{x}$
$\mathbf{b}_i \in \mathbb{R}^D$	i-th anchor token in the local basis $\mathbf{B}_{\mathbf{x}}$	$\mathbf{c}_{\mathbf{x}} \in \mathbb{R}^{k_{\text{nn}}}$	Reconstruction coefficient vector for $\mathbf{x}$
$\hat{\mathbf{x}} \in \mathbb{R}^D$	Reconstructed token $\mathbf{B}_{\mathbf{x}}^\top \mathbf{c}_{\mathbf{x}}^*$ predicted from the local anchor basis	$\rho(\mathbf{x})$	Predictive residual score of token $\mathbf{x}$
$\beta \in [0, 1]$	Anchor budget fraction	$\widehat{m}_{\text{anc}} = \lfloor \beta m \rfloor$	Nominal anchor budget before clipping
λ	Ridge regularization coefficient	m_{anc}	Final anchor budget after clipping
$m_{\text{res}} = m - m_{\text{anc}}$	Token budget assigned to residual-scored non-anchor tokens	$s(\mathbf{q})$	Anchor-token score used when the anchor pool exceeds m_{anc}
$N_{\text{anc}} = aT$	Maximum anchor pool size (number of anchor tokens)	$N_{\text{res}} = (F - a)T$	Number of non-anchor tokens
ϵ	Ridge regularization scale (10^{-3} by default)	$\bar{\mathbf{q}} \in \mathbb{R}^D$	Centroid of the anchor pool $\mathcal{P}$
$m_{\text{anc}}^{\min} = \max(0, m - N_{\text{res}})$	Lower clipping bound for anchor budget	$m_{\text{anc}}^{\max} = \min(m, N_{\text{anc}})$	Upper clipping bound for anchor budget
$\mathcal{S}_{\text{anc}}$	Selected anchor token set	$\mathcal{S}_{\text{res}}$	Selected residual token set
$\tilde{\mathbf{V}}$	Compressed output visual token sequence	T_{full}	Total processing time per query
T_{prep}	Pre-LLM preparation time (amortized across queries)	T_{LLM}	LLM inference time per query
$\mathbf{q} \in \mathbb{R}^D$	Anchor token; generic element of the anchor pool $\mathcal{P}$	n	Number of queries about the same video
$T_{\text{reuse}}(n) = T_{\text{prep}} + n \cdot T_{\text{LLM}}$	Multi-query reuse cost for n queries		

B Notation

Table 6 summarizes the notation used throughout the paper. We use lowercase letters for scalars, bold lowercase letters for vectors, bold uppercase letters for matrices or tensors, and calligraphic letters for sets. In particular, we distinguish the global retain ratio α from the actual token budget m, and the nearest-neighbor size k_{nn} from both quantities. The retain ratio α controls the overall compression level, whereas m denotes the resulting discrete token budget. The anchor budget fraction β controls how this budget is split between anchor tokens and

Table 7: Budget-aware comparison on LLaVA-OV-0.5B across video understanding benchmarks. Target Budget denotes the approximate retained-token ratio. Rows are grouped by comparable token budgets, and PreViC rows are separated from baselines. Bold marks the best score within each budget group.

Target Budget	Locus	Method	MVBench	Video-MME	EgoSchema	LVB	MLVU
Full tokens		Vanilla	46.45	53.60	25.20	44.95	46.09
∼30% retention	in-LLM	PruneVid	**46.42**	50.60	**25.80**	45.40	**47.75**
		HoliTom	43.83	**53.00**	**25.80**	45.32	46.96
		AIM	42.27	48.60	25.60	**45.77**	46.09
	pre-LLM	DivPrune	44.75	51.80	26.20	45.25	43.88
		DyCoke	45.73	52.00	24.80	44.58	45.68
		FastVID	44.25	49.20	**28.00**	45.40	43.28
		VidCom^2	**45.90**	55.00	27.40	45.62	**46.00**
		PreViC	45.75	**57.20**	27.60	**45.77**	44.25
∼50% retention	in-LLM	PruneVid	**46.42**	50.40	25.80	45.32	**47.61**
		HoliTom	43.75	**53.60**	**26.80**	45.32	46.55
		AIM	42.30	48.80	25.60	**45.77**	46.09
	pre-LLM	DivPrune	45.27	54.20	24.80	45.48	44.16
		DyCoke	45.98	51.80	26.80	45.03	43.33
		FastVID	44.88	48.40	**27.80**	**45.77**	45.35
		VidCom^2	45.90	54.40	27.00	45.10	46.23
		PreViC	**46.58**	**56.00**	25.80	45.70	**47.29**
∼70% retention	in-LLM	PruneVid	**44.85**	50.40	25.20	43.60	45.08
		HoliTom	44.08	**52.80**	**25.80**	45.55	44.25
		AIM	42.38	48.80	25.40	**45.70**	**46.09**
	pre-LLM	DivPrune	45.58	52.80	25.00	44.58	43.84
		DyCoke	45.40	53.00	25.80	**46.60**	**46.96**
		FastVID	45.28	51.60	**26.80**	44.13	46.04
		VidCom^2	**46.12**	55.00	25.60	45.10	45.81
		PreViC	46.08	**55.60**	25.40	46.45	46.69

residual-scored non-anchor tokens. The nearest-neighbor size k_{nn} is used only in predictive residual scoring and is independent of both α and m.

C Budget-Aware Short-Video Comparison on 0.5B

Table 7 reports the same budget-aware comparison on LLaVA-OV-0.5B. We place this table in the appendix to keep the main paper focused on the 7B setting, where prefill latency and memory savings are more impactful. The 0.5B results nevertheless show that PreViC's predictive residual criterion is effective for smaller video VLMs, especially on Video-MME under aggressive compression. At 30% retention, PreViC achieves the strongest Video-MME score while keeping only a small fraction of visual tokens. At 50% retention, it also gives the best MVBench score among the compared methods. These results suggest that smaller models can benefit from removing predictable visual redundancy, since such redundancy may otherwise act as distracting context during reasoning.

Table 8: PreViC retain-ratio sweep on LLaVA-OV-0.5B Video-MME. TTFT includes preprocessing.

Setting	Ret.	Score	Δ	TTFT (ms)
Vanilla	100%	53.60	0.00	350.19
PreViC, $\alpha = 0.3$	30%	**57.20**	**+3.60**	**300.95**
PreViC, $\alpha = 0.5$	50%	56.00	+2.40	318.47
PreViC, $\alpha = 0.6$	60%	54.60	+1.00	318.45
PreViC, $\alpha = 0.7$	70%	55.60	+2.00	331.65
PreViC, $\alpha = 0.8$	80%	54.40	+0.80	336.67

D Full Retain-Ratio Sweeps

0.5B Video-MME sweep. Table 8 reports the full PreViC sweep on LLaVA-OV-0.5B Video-MME. All measured retain ratios improve over vanilla, and the best result appears at $\alpha = 0.3$, where PreViC keeps only 30% of visual tokens.

7B benchmark sweep. Table 9 reports the full PreViC sweep on LLaVA-OV-7B. Different benchmarks prefer different retain ratios.

Table 9: Full retain-ratio sweep of PreViC on LLaVA-OV-7B. Average Δ is computed against vanilla across the five benchmarks.

Setting	MVBench	Video-MME	EgoSchema	LongVideo Bench	MLVU	Avg. Δ
Vanilla	54.12	69.80	61.60	55.72	62.47	0.00
PreViC, $\alpha = 0.3$	53.85	69.40	62.60	57.74	64.21	+0.82
PreViC, $\alpha = 0.4$	54.02	69.80	**63.00**	**57.82**	65.13	**+1.21**
PreViC, $\alpha = 0.5$	53.62	70.40	62.60	57.07	65.13	+1.02
PreViC, $\alpha = 0.6$	53.80	**71.00**	62.20	55.50	61.78	+0.11
PreViC, $\alpha = 0.7$	**54.17**	70.80	62.60	55.12	62.01	+0.20
PreViC, $\alpha = 0.8$	54.02	70.40	62.60	56.99	**65.32**	+1.12

Table 10: Task-wise analysis on MVBench under a fixed retain ratio $\alpha = 0.5$. We compare Vanilla and PreViC on the representative MVBench tasks reported in the detailed task table. Δ denotes the absolute change over Vanilla. The overall score is the full 20-task MVBench average.

Task	LLaVA-OV-0.5B			LLaVA-OV-7B		
	Vanilla	PreViC	Δ	Vanilla	PreViC	Δ
Action Prediction	56.5	58.5	+2.0	69.5	69.0	-0.5
Action Sequence	53.5	51.5	-2.0	63.0	67.0	+4.0
Character Order	59.0	58.0	-1.0	68.5	68.5	0.0
Episodic Reasoning	41.5	42.5	+1.0	50.5	50.5	0.0
Moving Attribute	53.5	56.0	+2.5	66.5	68.0	+1.5
Moving Count	41.0	40.5	-0.5	45.0	39.5	-5.5
Object Interaction	70.0	73.5	+3.5	77.0	78.5	+1.5
Object Shuffle	38.0	37.0	-1.0	38.0	35.0	-3.0
Scene Transition	88.5	90.0	+1.5	94.0	94.5	+0.5
Action Localization	27.5	30.0	+2.5	39.5	37.5	-2.0
Fine-grained Action	37.0	39.5	+2.5	45.0	46.0	+1.0
Object Existence	52.0	53.0	+1.0	53.5	55.5	+2.0
Overall, 20-task avg.	46.45	46.58	+0.13	54.12	53.62	-0.50

Table 11: Anchor-count sensitivity on long-video benchmarks with LLaVA-OV-7B. All results use $F = 32$ frames and $\alpha = 0.5$.

Anchor Frames	Anchor Pool Size	Ego Schema	Long Video Bench	MLVU	Avg.
$a = 1$	196	62.00	56.77	64.95	61.24
$a = 2$	392	**62.60**	**57.07**	65.13	**61.60**
$a = 4$	784	62.40	56.84	64.63	61.29
$a = 8$	1568	62.00	**57.22**	**65.32**	61.51

E Task-wise Analysis on MVBench

To examine when prompt-agnostic predictive residual selection helps or hurts, we analyze MVBench performance at the task level. For a controlled cross-scale comparison, we use the same retain ratio $\alpha = 0.5$ for both LLaVA-OV-0.5B and LLaVA-OV-7B. This analysis is diagnostic: the best headline operating point may differ across backbones, but fixing α allows us to compare task-level behavior under the same compression budget.

Table 10 shows that PreViC affects task types differently. At $\alpha = 0.5$, PreViC improves several object- and event-centric tasks across both model scales, including moving attribute, object interaction, scene transition, fine-grained action, and object existence. This supports the predictive-residual view that tokens difficult to reconstruct from anchor memory often capture distinctive visual changes or object-level evidence.

The main failure modes are also consistent across scales. Moving count and object shuffle degrade for both 0.5B and 7B, suggesting that counting and permutation-style tasks require broad token coverage or repeated visual evidence. Such evidence may be predictable from anchors but still useful for verifying counts or object arrangements. Overall, this analysis suggests that PreViC is effective as a reusable pre-LLM compressor, while counting, object-shuffle, and localization-sensitive tasks may benefit from higher budgets or lightweight prompt-aware refinement.

Table 12: Nearest-neighbor size sensitivity on LLaVA-OV-0.5B MVBench. All results use $F = 32$ frames and $a = 2$ anchor frames. Bold indicates the best per column.

k_{nn}	$\alpha = 0.5$	$\alpha = 0.7$
1	46.25	46.10
2	46.23	45.92
4 (default)	**46.58**	46.08
8	46.55	**46.17**
16	46.45	46.05

Table 13: Component ablation of PreViC on LLaVA-OV-0.5B MVBench.

R1	R2	R3	$\alpha = 0.7$	$\alpha = 0.5$	$\alpha = 0.3$
N	N	N	46.15	45.30	44.08
Y	N	N	46.15	45.12	44.40
N	Y	N	45.77	46.25	45.80
N	N	Y	46.15	45.30	44.38
Y	Y	N	46.08	**46.58**	46.02
Y	N	Y	46.15	45.12	44.65
N	Y	Y	45.77	46.25	**46.08**
Y	**Y**	**Y**	**46.08**	**46.58**	45.75

R1: novelty anchor selection; R2: predictive residual scoring; R3: hierarchical rate allocation.

F Anchor Sensitivity

Table 11 shows that performance is largely stable across anchor counts on long-video benchmarks. Although different benchmarks prefer different values, the default $a = 2$ is competitive with larger anchor pools while keeping the anchor memory small. Larger anchor pools may provide dataset-specific gains when broader temporal coverage helps residual reconstruction.

G Nearest-Neighbor Size Sensitivity

Table 12 shows that LLaVA-OV-0.5B MVBench accuracy is stable across $k_{\text{nn}} \in \{1, 2, 4, 8, 16\}$, with all values within ± 0.35 pp of the default. $k_{\text{nn}} = 4$ achieves the best score at the headline operating point ($\alpha = 0.5$), while $k_{\text{nn}} = 8$ provides a marginal gain of $+0.09$ pp at $\alpha = 0.7$ only. We adopt $k_{\text{nn}} = 4$ as the default.

H Component Ablations

R1/R2/R3 ablation. Table 13 reports the full component ablation. R2 is the only component that consistently separates strong from weak configurations under tighter budgets. R1 and R3 are structural: R1 provides the reference memory, and R3 prevents destructive anchor loss.

Table 14: Long-video R1 ablation on LLaVA-OV-7B at 50% retention.

Setting	EgoSchema	LongVideo Bench	MLVU
Full PreViC	**62.60**	**57.07**	**65.13**
w/o R1 random anchors	61.80	56.32	63.89
w/o R1 uniform anchors	62.00	56.55	64.21

Table 15: Stage-wise timing breakdown of PreViC. Rows report pipeline stages with separate 0.5B and 7B measurements. Columns correspond to retain ratios $\alpha \in \{0.3, 0.5, 0.7\}$. TTFT includes vision encoding/projection, PreViC compression, and LLM prefill. All time values are in milliseconds. Measurements use a representative MVBench sample (16 frames, 3137 visual tokens); TTFT values are lower than those in Table 2, which uses a Video-MME sample (32 frames, 6273 visual tokens).

Stage	Backbone	α		
		0.3	0.5	0.7
Vision Enc. +Proj.	0.5B	124.8	127.4	128.0
	7B	121.6	121.6	123.0
R1 Anchor	0.5B	0.24	0.19	0.31
	7B	0.28	0.24	0.22
Exact k-NN	0.5B	0.29	0.29	0.37
	7B	0.54	0.54	0.54
Ridge Solve	0.5B	0.40	0.37	0.48
	7B	0.82	0.81	0.81
PreViC Total	0.5B	**0.95**	**0.87**	**1.19**
	7B	**1.66**	**1.62**	**1.59**
LLM Prefill	0.5B	14.0	14.6	17.5
	7B	83.2	136.3	184.6
TTFT	0.5B	149.8	156.3	162.9
	7B	246.2	301.7	361.7
E2E	0.5B	**161.1**	**166.4**	**173.7**
	7B	**265.2**	**321.9**	**381.7**
Peak Mem. (GB)	0.5B	3.68	3.68	3.85
	7B	17.25	17.25	17.55
PreViC % E2E	0.5B	0.59%	0.52%	0.68%
	7B	0.63%	0.50%	0.42%

Long-video R1 ablation. Table 14 evaluates anchor construction on long-video benchmarks. Unlike short-video MVBench, long-video tasks are more sensitive to the reference memory, and removing novelty-based anchors consistently hurts performance.

I Compression Overhead Breakdown

Table 15 reports a stage-wise timing breakdown of PreViC on LLaVA-OV-0.5B and LLaVA-OV-7B. This analysis addresses the concern that exact k-NN retrieval and ridge reconstruction may offset the efficiency gains from token compression. Across all retain ratios, the full PreViC stage takes about 1 ms on 0.5B and less than 2 ms on 7B, accounting for only 0.42–0.68% of end-to-end latency. Exact k-NN remains below 0.6 ms, while ridge reconstruction remains below 0.9 ms even for the 7B backbone.

The overhead is nearly constant across α because residual scores are computed before the final top-m selection. Thus, changing the retain ratio mainly affects the number of tokens passed to the LLM, not the cost of PreViC scoring. These results indicate that exact full-dimensional residual scoring is practical in our setting, and that the dominant efficiency gain comes from reducing the sequence length processed during LLM prefill.

J Qualitative Visualization of R1 and R2

We visualize the token selection behavior of PreViC on a 16-frame video at $\alpha = 0.3$. Figures 3 and 4 provide a qualitative view of how R1 and R2 operate together. In Figure 3, we show the original frames, R2 predictive residual maps, and retained token locations for all frames. Anchor frames are marked in red, while non-anchor frames are selected by residual scores. The retained token locations are concentrated around temporally changing regions, such as the person, moving cloth, and human-object interactions, rather than uniformly covering static background areas.

Figure 4 (a) shows the R1 novelty score for each frame. PreViC always includes the first frame as an initial reference and selects the additional anchor frame with high temporal novelty. Figure 4 (b) shows the resulting kept-token distribution across frames. Anchor frames retain many tokens as reference memory, while non-anchor frames receive fewer tokens selected by R2 residual scores. Taken together, these visualizations illustrate the intended behavior of PreViC: R1 builds a compact visual memory from temporally distinctive frames, while R2 allocates the remaining budget to unpredictable visual regions.

Table 16: MVBench sweep comparing random projection with full-dimensional residual scoring.

Backbone	Setting	JL on	Full-D	Δ
0.5B	$\alpha = 0.3$	45.88	45.75	-0.13
0.5B	$\alpha = 0.5$	46.30	**46.58**	+0.28
0.5B	$\alpha = 0.6$	45.92	46.15	+0.23
0.5B	$\alpha = 0.7$	46.12	46.08	-0.04
0.5B	$\alpha = 0.8$	46.12	46.05	-0.07
7B	$\alpha = 0.3$	53.83	53.85	+0.02
7B	$\alpha = 0.5$	53.80	53.62	-0.18
7B	$\alpha = 0.6$	53.75	53.80	+0.05
7B	$\alpha = 0.7$	**54.33**	54.17	-0.16
7B	$\alpha = 0.8$	53.95	54.02	+0.07

Table 17: Feature normalization and ridge sensitivity on LLaVA-OV-0.5B MVBench at $\alpha = 0.5$.

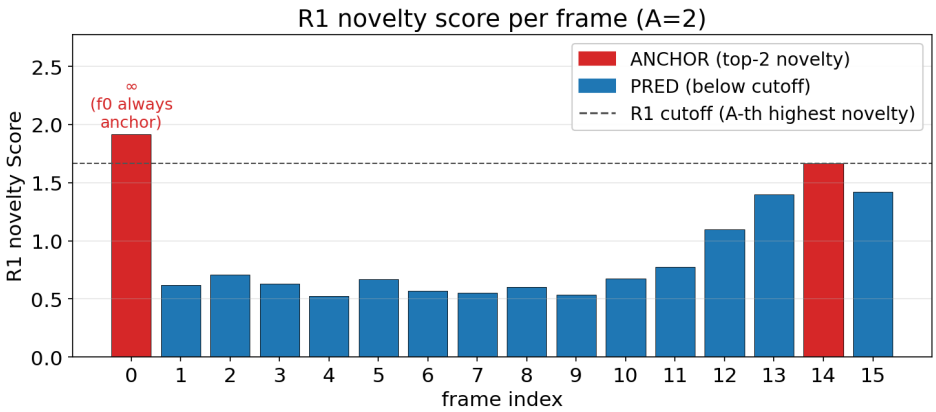
Group	Variant	MVBench	Δ vs. default
Normalization	raw default	**46.58**	0.00
Normalization	L2-normalize after projector	46.10	-0.48
Normalization	video-level mean centering	46.42	-0.16
Normalization	channel standardization	46.23	-0.35
Ridge λ	adaptive default	**46.58**	0.00
Ridge λ	10^{-5}	46.55	-0.03
Ridge λ	10^{-2}	46.55	-0.03
Ridge λ	10^{-1}	46.55	-0.03

K Feature-Space and Residual-Scoring Sensitivity

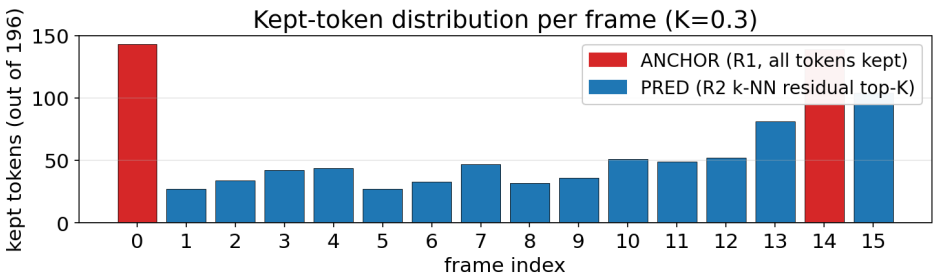
Full-D scoring versus random projection. Table 16 compares random-projected residual scoring with full-dimensional scoring. Full-D is not uniformly better at every α, but it gives the best 0.5B headline result and avoids projection-induced information loss.

Normalization and ridge sensitivity. Table 17 addresses two implementation concerns. Raw post-projector features outperform normalized variants, suggesting that token magnitudes carry useful information. The method is also insensitive to λ over several orders of magnitude.

Anchor overflow variants. When $N_{\text{anc}} > m_{\text{anc}}$, PreViC ranks anchor tokens by distance to the anchor-pool mean. Table 18 shows that k-center and farthest-point alternatives behave identically at the measured setting, so we keep the simpler mean-distance rule.



(a) R1 novelty scores



(b) Kept-token distribution

Figure 4: Qualitative visualization of PreViC token selection at $\alpha = 0.3$. (a) R1 selects anchor frames using frame-level novelty scores. (b) The kept-token distribution shows that anchor frames preserve reference tokens, while non-anchor frames receive residual-selected tokens. Overall, PreViC retains high-residual regions around temporally changing objects and interactions, such as the person and moving cloth.

Table 18: Anchor overflow scoring variants on LLaVA-OV-0.5B MVBench at $\alpha = 0.5$.

Anchor overflow method	MVBench	Δ
mean-distance default	**46.58**	0.00
k-center	46.58	0.00
farthest-point sampling	46.58	0.00

Table 19: Empirical timing components for multi-query reuse. Times are in milliseconds.

Backbone	T_{full}	T_{prep}	T_{LLM}
0.5B PreViC, $\alpha = 0.7$	517.76	278.94	238.82
7B PreViC, $\alpha = 0.7$	960.32	337.21	623.11

L Multi-Query Reuse Details

The empirical multi-query experiment in the main paper uses the current LLaVA-OV API and skips only repeated video preprocessing. Let T_{full} be the time for re-encoding and answering one query, T_{prep} the video preprocessing plus PreViC time, and T_{LLM} the derived LLM-only time. Then visual-preprocessing reuse is computed as $T_{\text{reuse}}(n) = T_{\text{prep}} + n \cdot T_{\text{LLM}}$, while naive serving costs $n \cdot T_{\text{full}}$, where n is the number of questions asked about the same video. Table 19 reports the empirical timing components used to derive this formula. Table 20 reports an analytical upper bound that assumes visual KV-cache passthrough. This passthrough assumption requires serving-system support beyond the current API, but it clarifies the potential benefit of prompt-agnostic visual compression.

M Time Complexity

The cost of anchor selection is $O(FTD)$. For residual scoring, exact nearest-neighbor retrieval costs $O((F-a)TN_{\text{anc}}D)$, and the batched ridge solve costs $O((F-a)Tk_{\text{nn}}^3)$. The total preprocessing cost is therefore $O(FTD)+O((F-a)TN_{\text{anc}}D)+O((F-a)Tk_{\text{nn}}^3)$. Since a and k_{nn} are small, this overhead is lightweight and is outweighed by the reduction in LLM prefill cost. For very long videos (e.g., $F > 64$), the exact k-NN cost scales linearly in F as $O((F-a)T \cdot aT \cdot D)$; approximate nearest-neighbor methods (e.g., IVF-indexed retrieval or temporal blocking) can reduce per-token search cost from $O(aT \cdot D)$ to $O(D \log(aT))$ without

Table 20: Analytical multi-query reuse projection on LLaVA-OV-7B. Time is in milliseconds.

Queries n	Vanilla naive	**PreViC reuse**	HoliTom partial reuse	AIM/ PruneVid no reuse
1	472	383	350	472
2	944	588	700	944
4	1888	998	1400	1888
8	3776	1818	2800	3776
16	7552	3460	5600	7552
32	15104	6743	11200	15104
128	60367	26437	44800	60367
512	241470	105215	179200	241470

changing the scoring principle.

N Implementation Details

All PreViC operations are implemented in PyTorch and executed on GPU unless otherwise stated. The k-NN retrieval is computed by batched cosine similarity between non-anchor tokens and the anchor pool. Ridge reconstruction is also batched on GPU using `torch.linalg.solve`. We compute the linear solve in float32 for numerical stability, while the backbone inference uses bf16. No CPU offloading is used for k-NN search or ridge solving. Latency measurements include the full PreViC preprocessing step, including anchor selection, k-NN retrieval, ridge reconstruction, top-K selection, and LLM prefill/decode. We report replicate medians after warmup and synchronize CUDA events before and after timing. Experiments are run on NVIDIA RTX PRO 6000 Blackwell GPUs with PyTorch 2.11.0 and Transformers 4.40.2.

O Artifact, Data, and Privacy Considerations

Artifact Licenses and Terms of Use. We use publicly available pretrained models, benchmark datasets, evaluation frameworks, and baseline implementations according to their original licenses, terms of use, and evaluation protocols. Our experiments rely on existing benchmark data and publicly released model or code artifacts for research evaluation only. We do not redistribute raw videos, benchmark annotations, pretrained model weights, or third-party baseline code. Released artifacts from this work will contain only our implementation code, configuration files, and aggregate evaluation results, subject to the licenses and access conditions of the underlying dependencies.

Data and Privacy. We use only publicly available video understanding benchmarks and collect no new human-subject data. Since public video benchmarks may contain people or real-world scenes, we do not release raw videos, personal annotations, or any additional information that could identify individuals beyond the original benchmark content. All reported results are aggregate benchmark statistics.

P Use of AI Assistants

The authors used AI-based assistants solely for language editing, grammar checking, and code debugging. All scientific ideas, experiments, analyses, implementation choices, and final text were reviewed and approved by the authors. No AI assistant was listed as an author, and no confidential or non-public third-party data were intentionally provided to AI systems.