CUEMINER: EFFICIENT AND ACCURATE VIDEO LARGE LANGUAGE MODELS VIA CONTEXT MINING

Anonymous authors
Paper under double-blind review

ABSTRACT

How can we minimize memory consumption during Video Large Language Model (VLLM) inference while preserving temporal reasoning accuracy? VLLMs achieve strong temporal reasoning performance in video understanding by processing rich visual information across video frames, yet retaining the full visual representation incurs considerable memory overhead. To alleviate this overhead, existing approaches compress video representations by selectively retaining only visual information deemed important. Specifically, they excavate information associated with key events. However, these approaches do not consider preserving the visual evidence that temporally precedes such events, even though this evidence provides critical context for interpreting temporally subsequent key events. Consequently, aggressive compression degrades temporal reasoning accuracy in VLLMs. In this paper, we propose CUEMINER, an efficient video compression method that preserves temporal reasoning accuracy even under aggressive compression. To preserve accuracy, CUEMINER not only captures salient events, but also mines the temporally antecedent visual context that enables VLLMs to interpret how subsequent events develop over time. On MLVU with LLaVA-OneVision, CUEMINER retains 99.4% of the accuracy at a retention ratio of 5%.

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

1 INTRODUCTION

Given a Video Large Language Model (VLLM), a video, and a user query, how can we minimize inference-time memory consumption without sacrificing temporal reasoning accuracy? Video large language models (VLLMs) achieve strong video understanding performance by representing each video frame with dense visual tokens (An et al., 2026; Lin et al., 2024; Li et al., 2025b; Zhang et al., 2025c), but processing all visual tokens generated from the frames requires substantial memory usage (Man et al., 2025) (Figure 1(a)).

To reduce this memory consumption, existing methods compress visual tokens by retaining only those deemed important (Tao et al., 2025). Specifically, since existing methods regard visual content that substantially deviates from the preceding temporal context as salient, they extract visual tokens that are markedly distinctive from visual tokens observed in preceding frames (Tang et al., 2026; Luo et al., 2026; Wang et al., 2026). However, previous approaches do not account for preserving visual tokens that temporally precede such distinctive tokens, even though these antecedent tokens provide contextual cues essential for interpreting how subsequent key events unfold (Figure 1(b)). Therefore, existing visual-token compression methods degrade temporal reasoning accuracy in VLLMs.

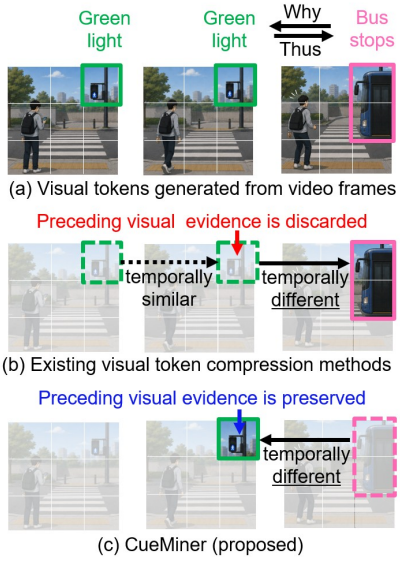

Figure 1: (a) Visual tokens generated from video frames. (b) Existing methods fail to preserve the visual cue that precedes the event. (c) In contrast, CUEMINER preserves event-preceding visual evidence.

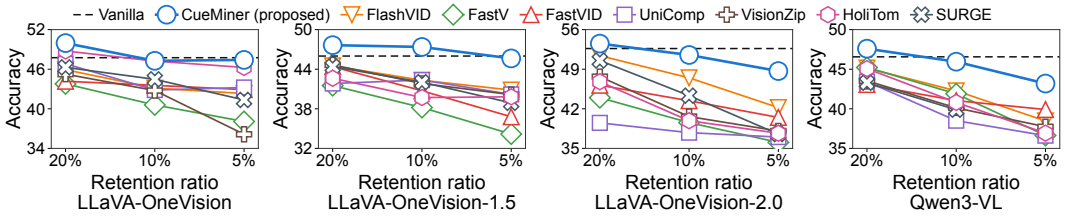

Figure 2: MLVU accuracy across retention ratios for LLaVA-OneVision 1, 1.5, 2.0, and Qwen3-VL.

In this paper, we propose CUEMINER, an efficient visual token compression method that maintains temporal reasoning accuracy. To sustain accuracy, CUEMINER not only extracts key events, but also excavates the pre-event visual evidence that provides context for interpreting the subsequent key events. To extract such event-preceding visual evidence, beyond mining visual tokens that are temporally distinctive in the forward direction, CUEMINER additionally identifies tokens that are temporally prominent in the backward direction. By reversing the temporal direction, CUEMINER mines pivotal visual evidence that immediately precedes the key events (Figure 1(c)). Such evidence provides contextual cues that enable VLLMs to better understand how those events unfold over time. Across diverse VLLMs, including LLaVA-OneVision 1, 1.5, and the latest 2.0, as well as Qwen3-VL, CUEMINER maintains robust accuracy even at aggressive retention ratios (Figure 2). This enables VLLMs to achieve substantial memory savings during inference.

The main contributions are as follows:

- **Performance.** Across various VLLMs, CUEMINER consistently maintains accuracy even at very low retention ratios, thereby substantially lowering memory consumption.
- **Scalability.** CUEMINER compresses visual tokens independently of any specific question, making it effective for serving large volumes of questions over a video's lifetime.
- **Practicality.** CUEMINER requires no retraining or fine-tuning of any module of the VLLM, allowing seamless integration with off-the-shelf VLLMs.

The codes are available at `https://anonymous.4open.science/r/CueMiner-1C32`. Notations used throughout the paper are provided in Appendix A.1.

2 PRELIMINARIES AND RELATED WORK

2.1 PROBLEM DEFINITION

Problem 1 (Video compression)**.** Given a VLLM $\mathcal{M}$, an input video $\mathcal{V}$, and a user query q, minimize inference-time memory consumption while preserving temporal reasoning accuracy.

2.2 VIDEO LARGE LANGUAGE MODELS

Given an input video $\mathcal{V}$ and a user query q, a VLLM $\mathcal{M}$ generates a textual response y. Specifically, the VLLM first samples T frames from $\mathcal{V}$, which we index by $r = 1, \ldots, T$. Then, a vision encoder transforms each sampled frame into an $H \times W$ grid of visual tokens. We denote the d-dimensional visual-token embedding at spatial position (h, w) in sampled frame r by $\mathbf{z}_{r,(h,w)} \in \mathbb{R}^d$, where $1 \le r \le T$, $1 \le h \le H$, $1 \le w \le W$. Accordingly, the T sampled frames from $\mathcal{V}$ produce a total of $N_{\mathcal{V}} = THW$ visual tokens. In parallel, a textual tokenizer converts the user query q into N_q textual tokens. Let $\mathbf{x}_m^{\text{text}} \in \mathbb{R}^d$, $m = 1, \ldots, N_q$, denote the embedding of the m-th textual token. The visual tokens from all sampled frames are flattened and concatenated with the textual tokens to form the input token sequence $\mathbf{X} = (\mathbf{z}_{1,(1,1)}, \ldots, \mathbf{z}_{T,(H,W)}, \mathbf{x}_1^{\text{text}}, \ldots, \mathbf{x}_{N_q}^{\text{text}}) \in \mathbb{R}^{(N_{\mathcal{V}}+N_q)\times d}$, which is fed into the LLM of the VLLM $\mathcal{M}$. Given $\mathbf{X}$, the LLM autoregressively generates the textual response $y = (y_1, \ldots, y_{N_y})$ according to the autoregressive distribution: $p_{\mathcal{M}}(y \mid \mathbf{X}) = \prod_{k=1}^{N_y} p_{\mathcal{M}}(y_k \mid \mathbf{X}, y_{<k})$, where N_y denotes the number of generated textual tokens.

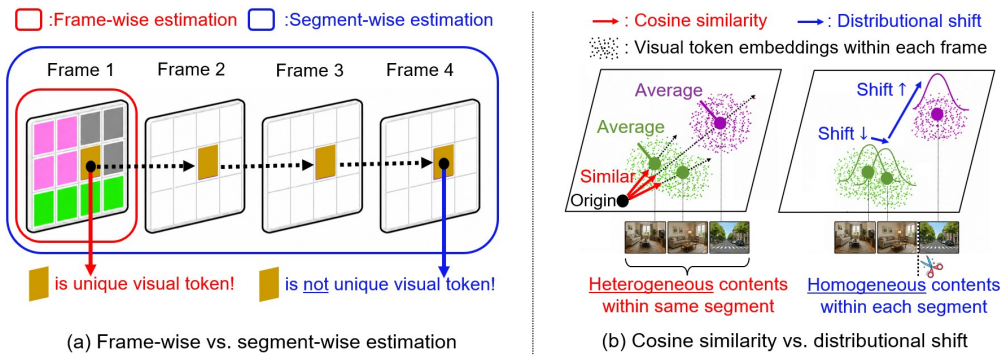

Figure 3: For both (a) and (b), comparison of the naive approach and the proposed design.

2.3 Frame Selection for Video Large Language Models

Although sampling more frames improves accuracy, it proportionally increases the number of visual tokens processed by the VLLM, causing substantial memory consumption (Zhang et al., 2025d). To reduce this cost, key-frame selection methods retain only a subset of sampled frames, reducing the visual tokens fed into the LLM of $\mathcal{M}$ (Buch et al., 2025; Hu et al., 2025; Liu et al., 2025; Zhu et al., 2026). However, understanding a key event requires not only the frame depicting the key event, but also event-preceding frames that capture the context surrounding it (Fang et al., 2026).

Context-aware methods address this by selecting temporally continuous frames rather than isolated key frames, thereby preserving contextual evidence around important events (Sun et al., 2025; Yao et al., 2025). Nevertheless, frame selection remains coarse-grained: once a frame is selected, all HW visual tokens generated from it are retained regardless of individual importance (Fan et al., 2026a).

2.4 Visual Token Compression for Video Large Language Models

Visual-token compression instead selectively prunes or merges visual tokens. Existing methods broadly fall into three branches: attention-based, redundancy-based, and surprisal-based compression. Attention-based methods estimate token importance from attention scores and retain highly attended tokens (Chen et al., 2024; Yang et al., 2025), but attention scores are potentially biased proxies, leading to inaccurate identification of informative tokens (Zhang et al., 2025a; Alvar et al., 2025). Redundancy-based methods instead exploit similarities among tokens to remove or merge repeated content (Shao et al., 2025; Shen et al., 2025; Fan et al., 2026b), but relying on similarity loses visual tokens informative for video understanding (Yuan et al., 2026; Zhang et al., 2025b).

Surprisal-based methods retain visual tokens that are distinctive relative to other visual observations (Zhang et al., 2026). Specifically, the methods identify newly emerging visual evidence with respect to preceding observations, capturing tokens associated with key events (Tang et al., 2026; Luo et al., 2026). However, despite the emphasis on temporal continuity essential for understanding key events (section 2.3), previous methods do not preserve visual evidence that precedes key events.

3 Proposed Method

3.1 Overview

We propose CueMiner, an efficient visual token compression method for VLLMs that preserves temporal reasoning accuracy. CueMiner tackles the following key challenges (C1–C3); refer to Figure 1(b), Figure 3(a), and Figure 3(b) for C1, C2, and C3, respectively.

C1. How can we mine antecedent visual evidence preceding key events? Existing surprisal-based compression methods retain only visual tokens associated with key events. However, they do not preserve visual evidence preceding key events, thereby losing earlier states or actions needed to understand how an event unfolds (Figure 1(b)).

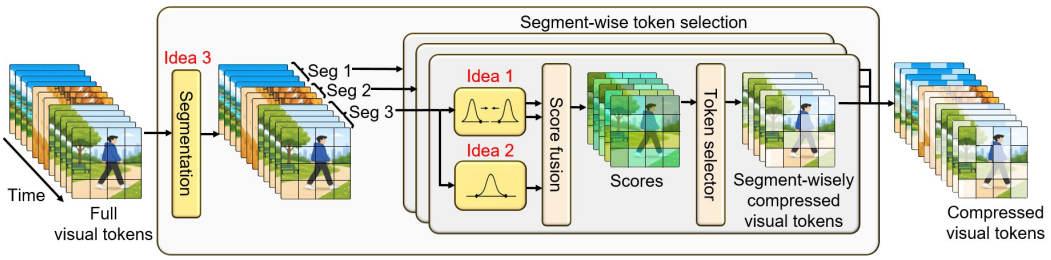

Figure 4: Overall process of CUEMINER.

C2. How can we accurately estimate the information content of each visual token? Conventional approaches estimate the information content of visual tokens at the frame level. However, they ignore temporal redundancy across frames, overestimating the information content of tokens that are rare within one frame but recur over time (see red in Figure 3(a)).

C3. How can we accurately separate visually discriminative frames into different segments? Previous approaches use cosine similarity between adjacent frames' averaged visual-token representations to partition video frames into segments, placing segment boundaries where it drops sharply. However, since cosine similarity considers only direction, using it groups frames into a single segment even when the visual-token distributions generated by individual frames differ substantially (red in Figure 3(b)). This complicates the approximation of the distribution over all visual tokens generated across the frames in each segment.

To address these challenges, we propose the following three ideas:

I1. Bidirectional scoring (Section 3.2). CUEMINER additionally identifies tokens temporally distinctive in the backward direction to mine visual cues preceding key events (Figure 1(c)).

I2. Segment-wise estimation (Section 3.3). CUEMINER estimates the information content of each token over consecutive frames (blue in Figure 3(a)). This segment-wise approach provides accurate estimates of the content, reflecting temporal redundancy across frames.

I3. Distributional shift (Section 3.4). CUEMINER places boundaries at pronounced distributional shifts between adjacent frames (blue in Figure 3(b)). This enables a more accurate approximation of the segment-wise visual-token distributions with a simple density model.

Figure 4 presents an overview of CUEMINER, which partitions sampled frames into segments and performs visual-token compression for each segment. **I1** takes the segment's visual tokens as input and estimates two scores for each token: the first measuring how temporally surprising it is relative to preceding visual content, while the second measures how temporally surprising it is relative to subsequent visual content (refer to Figure 1(b) and Figure 1(c), respectively). **I2** takes visual tokens from the same segment as input and estimates the information content of each token. To partition sampled frames into segments, **I3** takes video frames as input and produces segment boundaries.

3.2 BIDIRECTIONAL SCORING

How can we mine visual evidence that precedes key events? Existing surprisal-based compression methods retain only tokens that are temporally distinctive in the forward direction to extract visual tokens representing key events in a video. However, they do not preserve visual evidence preceding key events, although such cues provide the contextual information needed to understand how an event unfolds (Figure 1(b)). As a result, conventional methods lose states or actions required for temporal reasoning. To address this limitation, CUEMINER additionally identifies temporally distinctive tokens in the backward direction to mine evidence preceding key events (Figure 1(c)). To identify key-event and pre-event tokens, respectively, CUEMINER estimates two scores for each token: a forward score that measures how temporally distinctive the token is relative to preceding visual evidence, whereas a backward score indicates how temporally different it is relative to subsequent visual representations.

To estimate these scores, we use following notations. Let a video comprise T sampled frames. After segmentation (Section 3.4), the frames form K segments, denoted by $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_K\}$, each

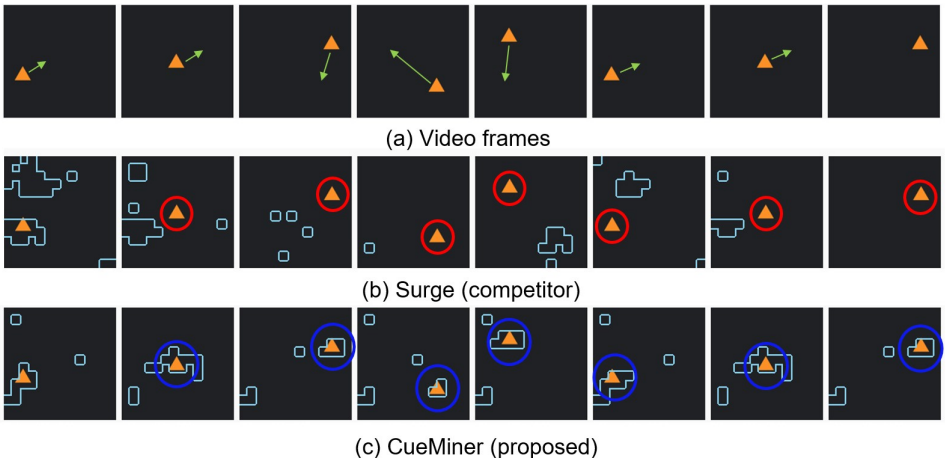

Figure 5: (a) Green arrows indicate the movement path to the next frame. For both (b) and (c), sky-blue regions visualize the retained visual tokens after compression.

containing consecutive frames. Let T_s be the number of frames in segment $\mathcal{G}_s$, whose frames we index by $t = 1, \ldots, T_s$. Let $\mathbf{z}_{s,t,(h,w)} \in \mathbb{R}^d$ denote the d-dimensional visual token at spatial position (h, w) in the t-th frame of $\mathcal{G}_s$, where each frame has an $H \times W$ grid of visual tokens with $1 \leq h \leq H$ and $1 \leq w \leq W$. Procedures for computing the forward score $S^F_{s,t,(h,w)}$ (Equation 3) and backward score $S^B_{s,t,(h,w)}$ (Equation 6) are described in Sections 3.2.1 and 3.2.2, respectively.

3.2.1 Forward Scoring via Spatio-Temporal Forward Memory

To detect key events, existing surprisal-based methods identify visual tokens temporally surprising with respect to preceding visual observations. However, such temporal-only methods do not account for the spatial dimension. Consequently, they are less effective at capturing visual content undergoing spatial displacement across frames (Figure 5(a) and Figure 5(b)). To address this limitation, CueMiner jointly considers the temporal and spatial context of visual tokens, thereby making their detection more robust to changes in spatial location across frames (Figure 5(a) and Figure 5(c)).

To achieve this, we propose a **Spatio-Temporal Forward Memory (STFMem)**, a frame-specific visual memory that summarizes the visual content observed in frames preceding the specified frame. This memory is used to estimate the forward score (Equation 3), measuring how surprising the specified frame's visual content is relative to the summarized visual content (refer to Figure 1(b)).

To construct STFMem for target frame t in segment $\mathcal{G}_s$, we collect the visual-token embeddings generated from frames preceding target frame t within segment $\mathcal{G}_s$ into a set of visual observations $\mathcal{Z}^{(s)}_{<t}$, defined as:

$$\mathcal{Z}^{(s)}_{<t} = \left\{ \mathbf{z}_{s,\tau,(h,w)} \in \mathbb{R}^d \;\middle|\; 1 \leq \tau < t,\ 1 \leq h \leq H,\ 1 \leq w \leq W \right\}. \tag{1}$$

CueMiner approximates the distribution of the visual-token embeddings in $\mathcal{Z}^{(s)}_{<t}$ (Equation 1), defining the resulting distribution as STFMem $\hat{p}^F_{s,<t}$ for target frame t in segment $\mathcal{G}_s$:

$$\hat{p}^F_{s,<t}\left(\mathbf{z}; \mathcal{Z}^{(s)}_{<t}\right), \qquad \mathbf{z} \in \mathbb{R}^d. \tag{2}$$

CueMiner measures the forward score $S^F_{s,t,(h,w)}$ of the visual token at spatial position (h, w) in target frame t of segment $\mathcal{G}_s$ using STFMem $\hat{p}^F_{s,<t}$. Specifically, CueMiner evaluates how unique the token is within the STFMem $\hat{p}^F_{s,<t}$ and defines this uniqueness as the forward score $S^F_{s,t,(h,w)}$:

$$S^F_{s,t,(h,w)} = -\log \hat{p}^F_{s,<t}\left(\mathbf{z}_{s,t,(h,w)}; \mathcal{Z}^{(s)}_{<t}\right). \tag{3}$$

Visual tokens that are unlikely under $\hat{p}^F_{s,<t}$ receive high forward scores $S^F_{s,t,(h,w)}$ and are regarded as spatio-temporally pivotal visual tokens.

3.2.2 BACKWARD SCORING VIA SPATIO-TEMPORAL BACKWARD MEMORY

Furthermore, to identify temporally antecedent evidence that contextualizes temporally subsequent key events, we propose a **Spatio-Temporal Backward Memory (STBMem)**. In contrast to STFMem, STBMem is a frame-specific visual memory that summarizes the visual content observed in frames subsequent to the specified frame within the same segment. This memory is used to estimate a backward score (Equation 6), assessing how unexpected the specified target frame's visual content is relative to the visual content summarized by STBMem (refer to Figure 1(c)).

To construct STBMem for target frame t in segment $\mathcal{G}_s$, we define $\mathcal{Z}^{(s)}_{>t}$ as the set of visual-token embeddings generated from frames subsequent to target frame t within segment $\mathcal{G}_s$:

$$\mathcal{Z}^{(s)}_{>t} = \left\{ \mathbf{z}_{s,\tau,(h,w)} \in \mathbb{R}^d \;\middle|\; t < \tau \leq T_s,\ 1 \leq h \leq H,\ 1 \leq w \leq W \right\}. \tag{4}$$

CUEMINER approximates the distribution of the visual-token embeddings in $\mathcal{Z}^{(s)}_{>t}$ (Equation 4), defining the resulting distribution as STBMem $\hat{p}^B_{s,>t}$ for target frame t in segment $\mathcal{G}_s$:

$$\hat{p}^B_{s,>t}\left(\mathbf{z}; \mathcal{Z}^{(s)}_{>t}\right), \qquad \mathbf{z} \in \mathbb{R}^d. \tag{5}$$

CUEMINER measures the backward score $S^B_{s,t,(h,w)}$ of the visual token at spatial position (h, w) in target frame t of segment $\mathcal{G}_s$ using STBMem $\hat{p}^B_{s,>t}$. Specifically, CUEMINER evaluates how unique the token is within STBMem $\hat{p}^B_{s,>t}$ and defines this uniqueness as the backward score $S^B_{s,t,(h,w)}$:

$$S^B_{s,t,(h,w)} = -\log \hat{p}^B_{s,>t}\left(\mathbf{z}_{s,t,(h,w)}; \mathcal{Z}^{(s)}_{>t}\right). \tag{6}$$

3.3 SEGMENT-WISE ESTIMATION

How can we accurately estimate the information content of each visual token? Existing methods estimate each visual token's information content at the frame level. However, such frame-wise approximation overlooks temporal redundancy, thereby mistakenly overestimating the information content of a visual token that appears unique within one frame but recurs across consecutive frames (see red in Figure 3(a)). To address this, CUEMINER estimates visual-token information over consecutive frames within each temporal segment. By considering temporal redundancy, this segment-wise approach more accurately measures each token's information content (see blue in Figure 3(a)).

To estimate a local score $S^L_{s,t,(h,w)}$ that represents the information content of each visual token (see the notation in Section 3.2), we propose a **Spatio-Temporal Local Memory (STLMem)**, a segment-specific visual memory that summarizes each segment's visual content. CUEMINER measures how unique each visual token within its segment is relative to the visual content summarized by STLMem.

To construct STLMem, following the notation in Section 3.2, we define $\mathcal{Z}^{(s)}$ as the set of visual-token embeddings generated from all frames within the segment $\mathcal{G}_s$:

$$\mathcal{Z}^{(s)} = \left\{ \mathbf{z}_{s,\tau,(h,w)} \in \mathbb{R}^d \;\middle|\; 1 \leq \tau \leq T_s,\ 1 \leq h \leq H,\ 1 \leq w \leq W \right\}. \tag{7}$$

CUEMINER approximates the distribution of the visual-token embeddings in $\mathcal{Z}^{(s)}$ (Equation 7), defining the resulting distribution as STLMem $\hat{p}^L_s$ for segment $\mathcal{G}_s$:

$$\hat{p}^L_s\left(\mathbf{z}; \mathcal{Z}^{(s)}\right), \qquad \mathbf{z} \in \mathbb{R}^d. \tag{8}$$

CUEMINER measures the local score $S^L_{s,t,(h,w)}$ of the visual token at spatial position (h, w) in frame t of segment $\mathcal{G}_s$ using STLMem $\hat{p}^L_s$. Specifically, CUEMINER evaluates how unique the token is within STLMem $\hat{p}^L_s$ and defines this uniqueness as the local score $S^L_{s,t,(h,w)}$ (Shannon (1948)):

$$S^L_{s,t,(h,w)} = -\log \hat{p}^L_s\left(\mathbf{z}_{s,t,(h,w)}; \mathcal{Z}^{(s)}\right). \tag{9}$$

We estimate three scores for each visual token: the forward score $S^F_{s,t,(h,w)}$ (Equation 3), backward score $S^B_{s,t,(h,w)}$ (Equation 6), and local score $S^L_{s,t,(h,w)}$ (Equation 9). To assign a single importance score to each visual token, CUEMINER fuses these three scores into the importance score $Q_{s,t,(h,w)}$; we provide the process of fusing scores in Appendix A.2. Using $Q_{s,t,(h,w)}$, CUEMINER performs segment-wise visual-token compression; we detail this in Appendix A.3. The compressed tokens from each segment are aggregated and fed to the LLM of the VLLM $\mathcal{M}$.

3.4 DISTRIBUTIONAL SHIFT

How can we accurately partition visually heterogeneous frames into distinct segments? To partition the video frames into segments, previous approaches average the visual-token representations within each frame and place boundaries where the cosine similarity between the averaged representations of adjacent frames drops. However, even when consecutive frames' content changes substantially, yielding markedly different visual-token distributions, cosine similarity fails to partition these frames into distinct segments since it considers only the direction of their averaged representations (see red in Figure 3(b)). Consequently, this makes accurately approximating the distribution over all visual tokens within the segment more difficult. To address this, CUEMINER measures the distributional shift between the frame-wise visual-token distributions of consecutive frames and places boundaries at pronounced shifts (see blue in Figure 3(b)). This enables more accurate approximation of the distribution over all visual tokens within each segment using a simple function.

To measure how much the distributions of visual tokens generated from adjacent frames differ, we propose a **Visual-Token Distributional Shift (VTDS)** score. VTDS is a metric that measures the distributional shift between the frame-wise visual-token distributions of temporally adjacent frames. Using this metric, CUEMINER produces segment boundaries at locations with pronounced distributional shifts.

To formalize VTDS, we use the following notation. Let the T sampled frames be indexed by $r = 1, \ldots, T$. Let $\mathbf{z}_{r,(h,w)} \in \mathbb{R}^d$ denote the d-dimensional embedding of the visual token at spatial position (h, w) in frame r, where each frame contains an $H \times W$ grid of visual tokens. We define the visual-token embeddings in frame r as:

$$\mathcal{Z}_r = \left\{ \mathbf{z}_{r,(h,w)} \in \mathbb{R}^d \;\middle|\; 1 \le h \le H,\ 1 \le w \le W \right\}. \tag{10}$$

For each sampled frame r, CUEMINER approximates the distribution of the visual-token embeddings in $\mathcal{Z}_r$ (Equation 10), defining the resulting frame-wise distribution as $\hat{p}_r(\mathbf{z}; \mathcal{Z}_r)$:

$$\hat{p}_r\left(\mathbf{z}; \mathcal{Z}_r\right), \qquad \mathbf{z} \in \mathbb{R}^d. \tag{11}$$

To determine segment boundaries, CUEMINER measures the shift between consecutive frames via the VTDS score Δ_r. Specifically, CUEMINER defines Δ_r using the symmetric KL divergence:

$$\Delta_r = \frac{1}{2}\left[D_{\mathrm{KL}}\left(\hat{p}_r \,\|\, \hat{p}_{r+1}\right) + D_{\mathrm{KL}}\left(\hat{p}_{r+1} \,\|\, \hat{p}_r\right)\right], \qquad 1 \le r < T. \tag{12}$$

A larger Δ_r indicates a greater shift between consecutive frames' visual-token distributions, whereas a smaller Δ_r indicates greater similarity. CUEMINER then places a temporal segment boundary after frame r when its distributional shift is at least a threshold θ. Accordingly, we define the set of segment boundaries as $\mathcal{B}_\theta = \{r \mid \Delta_r \ge \theta,\ 1 \le r < T\}$, yielding $K = |\mathcal{B}_\theta| + 1$ segments.

4 EXPERIMENTS

Experiments are structured to answer the following research questions:

Q1 Accuracy (Section 4.2). Does CUEMINER preserve accuracy under compression?
Q2 Ablation study (Section 4.3). Does each module of CUEMINER improve accuracy?
Q3 Sensitivity (Section 4.4). How sensitive is CUEMINER's accuracy to its hyperparameters?
Q4 Scalability (Section 4.5). Does CUEMINER retain accuracy with more sampled frames?
Q5 Latency (Section 4.6). Does CUEMINER reduce inference latency?

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate CUEMINER on four benchmarks: MLVU (Zhou et al., 2025), EgoSchema (Mangalam et al., 2023), VideoMME (Fu et al., 2025), and MVBench (Li et al., 2024).

Compared Baselines. We compare CUEMINER with the following baselines: FlashVID (Fan et al., 2026b), FastV (Chen et al., 2024), FastVID (Shen et al., 2025), UniComp (Yuan et al., 2026), VisionZip (Yang et al., 2025), HoliTom (Shao et al., 2025), and SURGE (Tang et al., 2026).

Table 1: Relative accuracy of CUEMINER and competitors, averaged over MLVU, EgoSchema, VideoMME, and MVBench using 32 uniformly sampled video frames at retention ratios of 20%, 10%, and 5%. The vanilla VLLM is set to 100%; bold denotes the best result, and absolute accuracies are reported in Appendix A.6.

Method	LOV			LOV 1.5			LOV 2.0			Q3-VL		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	94.9	89.6	85.1	94.8	90.3	83.9	87.7	80.6	75.4	96.3	90.8	80.5
VisionZip (Yang et al., 2025)	97.2	92.3	79.6	96.8	95.1	93.0	90.7	79.3	75.5	95.4	87.9	83.3
HoliTom (Shao et al., 2025)	100.3	**98.6**	96.4	94.9	92.6	91.9	93.5	75.6	73.7	96.1	90.5	83.0
FastVID (Shen et al., 2025)	98.3	96.0	93.9	96.2	92.8	89.7	93.4	88.7	83.3	94.0	89.9	84.6
UniComp (Yuan et al., 2026)	98.6	94.1	91.5	93.8	93.5	91.0	81.6	76.4	69.8	93.8	87.6	80.8
SURGE (Tang et al., 2026)	97.1	93.9	87.5	98.0	94.1	–	95.3	88.5	–	93.7	–	–
FlashVID (Fan et al., 2026b)	98.9	96.0	94.0	97.4	95.6	93.2	95.9	89.1	83.7	97.2	92.9	85.1
CUEMINER (proposed)	**101.0**	98.4	**97.6**	**100.4**	**100.9**	**97.7**	**98.7**	**95.7**	**91.2**	**97.3**	**93.7**	**89.0**

Table 2: Component ablation study on MLVU using the LOV model at 20% retention ratio. The first three rows use cosine similarity for segmentation, while the final row uses VTDS.

Configuration	Acc (%)
STLMem + cosine	47.3
(STL + STF)Mem + cosine	47.7
(STL + STF + STB)Mem + cosine	48.6
(STL + STF + STB)Mem + VTDS	**50.0**

Table 3: Hyperparameter sensitivity on MLVU with LOV 1.5 at 20% retention.

Mixing ratio α		VTDS threshold θ	
α	Acc (%)	θ	Acc (%)
0.2	46.9	0.139	45.8
0.3	**47.6**	0.177	46.4
0.4	47.2	0.214	46.8
0.5	47.1	0.240	**47.7**
0.6	46.3	0.271	47.6

Implementation Details. We evaluate CUEMINER on LLaVA-OneVision (Li et al., 2025a), LLaVA-OneVision 1.5 (An et al., 2025), LLaVA-OneVision 2.0 (An et al., 2026), and Qwen3-VL (Bai et al., 2025) (henceforth LOV, LOV 1.5, LOV 2.0, and Q3-VL). We approximate the distributions in Equation 2, Equation 5, Equation 8, and Equation 11 using gaussian densities (see Appendix A.4). Refer to Appendix A.5 for analysis of CUEMINER's memory reduction while maintaining accuracy.

4.2 ACCURACY (Q1)

We evaluate whether CUEMINER preserves accuracy under aggressive compression ratios. Table 1 shows that CUEMINER achieves higher relative accuracy than the baselines. In particular, at 10% retention ratio on LOV 1.5, CUEMINER outperforms the vanilla LOV 1.5 model. Even on LOV 2.0, the latest VLLM, CUEMINER strongly outperforms all baselines across retention ratios. Task-specific results for temporal, global, and detail video understanding are provided in Appendix A.7.

4.3 ABLATION STUDY (Q2)

We evaluate whether each component module contributes to CUEMINER's accuracy. Table 2 shows that each component contributes to accuracy. Specifically, adding STFMem to STLMem raises accuracy from 47.3% to 47.7%, and adding STBMem further improves it to 48.6%. Replacing cosine-similarity segmentation with VTDS then yields a final increase to 50.0%. Further ablation studies are reported in Appendix A.9.

4.4 SENSITIVITY (Q3)

We assess the sensitivity of accuracy to the mixing ratio α (Appendix A.2) and VTDS threshold θ (Section 3.4). Table 3 shows that accuracy increases from 46.9% at $\alpha = 0.2$ to a peak of 47.6% at $\alpha = 0.3$, then decreases as α increases further. For θ, accuracy rises from 45.8% at $\theta = 0.139$ to a peak of 47.7% at $\theta = 0.240$, before slightly decreasing to 47.6% at $\theta = 0.271$.

Table 4: MLVU accuracy (%) with Q3-VL at 48–96 sampled frames; vanilla achieves 46.5, 51.2, 53.9, and 54.5 at 48, 64, 80, and 96 frames, respectively. Bold denotes the best method.

Method	48 frames			64 frames			80 frames			96 frames		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	43.6	40.2	38.0	47.7	45.4	37.2	48.7	45.7	39.2	49.0	45.4	39.0
VisionZip (Yang et al., 2025)	44.1	38.3	38.0	46.5	43.1	40.3	49.2	43.8	39.1	50.6	41.8	41.1
HoliTom (Shao et al., 2025)	44.4	41.0	37.5	47.6	44.1	38.9	48.5	46.5	38.7	50.9	45.9	37.8
FastVID (Shen et al., 2025)	44.1	40.5	38.6	46.0	45.3	43.0	49.6	45.2	41.6	48.8	44.7	41.8
UniComp (Yuan et al., 2026)	43.3	39.5	37.0	46.8	43.0	42.3	47.1	44.0	37.6	49.2	45.2	40.2
SURGE (Tang et al., 2026)	44.8	39.4	–	45.2	42.2	–	48.3	44.9	–	49.2	44.0	–
FlashVID (Fan et al., 2026b)	44.7	41.8	38.2	47.6	46.9	41.5	49.0	45.8	40.0	49.6	46.6	37.7
CUEMINER (proposed)	**45.6**	**42.4**	**40.5**	**48.9**	**47.5**	**45.9**	**52.0**	**47.5**	**45.1**	**51.3**	**47.3**	**45.4**

Table 5: Inference latency on LOV 1.5 with 32 sampled frames across response lengths under accuracy-matched settings (20% retention for competitors, 5% for CUEMINER; see Table 1). All times are in seconds; bold marks the lowest latency at each response length.

Method	Response length (tokens)							
	8	16	24	32	40	48	56	64
FastV (Chen et al., 2024)	**2.97**	**3.39**	**3.82**	**4.24**	4.67	5.09	5.52	5.94
VisionZip (Yang et al., 2025)	3.00	3.42	3.85	4.27	4.70	5.12	5.55	5.97
HoliTom (Shao et al., 2025)	3.10	3.52	3.95	4.37	4.80	5.22	5.65	6.07
FastVID (Shen et al., 2025)	3.89	4.31	4.74	5.16	5.59	6.01	6.44	6.86
UniComp (Yuan et al., 2026)	3.08	3.50	3.93	4.35	4.78	5.20	5.63	6.05
SURGE (Tang et al., 2026)	3.02	3.44	3.87	4.29	4.72	5.14	5.57	5.99
FlashVID (Fan et al., 2026b)	3.36	3.78	4.21	4.63	5.06	5.48	5.91	6.33
CUEMINER (proposed)	3.01	3.42	3.83	**4.24**	**4.65**	**5.07**	**5.48**	**5.90**

4.5 SCALABILITY (Q4)

We assess whether CUEMINER remains effective as the number of sampled frames increases. Table 4 shows that CUEMINER robustly maintains accuracy as the number of sampled video frames increases. Notably, on Q3-VL with 96 sampled frames at a 5% retention ratio, CUEMINER achieves 45.4% accuracy, outperforming the best available baseline, FastVID (41.8%), by 3.6 percentage points.

4.6 LATENCY (Q5)

We evaluate whether CUEMINER reduces inference latency. Table 5 shows that CUEMINER achieves the lowest latency from a response length of 32 tokens onward. Specifically, at 32 response tokens, CUEMINER ties FastV for the lowest latency at 4.24 seconds and is the sole fastest method thereafter, reaching 5.90 seconds at 64 tokens. Detailed TTFT analysis is provided in Appendix A.8.

5 CONCLUSION

We propose CUEMINER, an efficient video compression approach that preserves temporal reasoning accuracy even at aggressive compression levels, thereby reducing memory consumption during inference. CUEMINER combines three complementary ideas. First, bidirectional scoring mines event-preceding visual evidence, enabling VLLMs to better understand temporal relationships. Second, segment-wise estimation provides more accurate visual-token information estimates, accounting for inherent temporal redundancy in videos. Third, distributional shift accurately separates visually heterogeneous frames into distinct segments, improving the approximation of visual-token distributions. Experiments show that CUEMINER robustly maintains temporal reasoning accuracy across diverse VLLMs, even under aggressive compression, thereby substantially reducing memory consumption.

AI USE STATEMENT

We used generative AI tools to assist with drafting and editing the manuscript, creating and editing software code, and creating and modifying scientific figures. All AI-assisted outputs were reviewed and verified by the authors, who take full responsibility for the final content, code, and figures.

REPRODUCIBILITY STATEMENT

We provide the methodological and implementation details necessary to reproduce our experiments in Sections 3 and 4 and in the Appendix. We also provide anonymous source code at the link given in the main paper to facilitate reproduction of our results.

REFERENCES

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

Xiang An, Yin Xie, Kaicheng Yang, Wenkang Zhang, Xiuwei Zhao, Zheng Cheng, Yirui Wang, Songcen Xu, Changrui Chen, Chunsheng Wu, Huajie Tan, Chunyuan Li, Jing Yang, Jie Yu, Xiyao Wang, Bin Qin, Yumeng Wang, Zizhen Yan, Ziyong Feng, Ziwei Liu, Bo Li, and Jiankang Deng. Llava-onevision-1.5: Fully open framework for democratized multimodal training. *arXiv preprint arXiv:2509.23661*, 2025.

Xiang An, Yin Xie, Feilong Tang, Yunyao Yan, Huajie Tan, Didi Zhu, Changrui Chen, Xiuwei Zhao, Bin Qin, Kaicheng Yang, Yifei Shen, Yuanhan Zhang, Kaichen Zhang, Wenkang Zhang, Zheng Cheng, Nansen Zhang, Chunsheng Wu, Chunjiang Ge, Zimin Ran, Dehua Song, Chunyuan Li, Shikun Feng, Ming Hu, Zhangquan Chen, Junbo Niu, Bo Li, Ziyong Feng, Ziwei Liu, Zongyuan Ge, and Jiankang Deng. Llava-onevision-2: Towards next-generation perceptual intelligence. In *arXiv*, 2026.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Shyamal Buch, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. Flexible frame selection for efficient video reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29071–29082, 2025.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 19–35, 2024. doi: 10.1007/978-3-031-73004-7_2.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1): 1–22, 1977. doi: 10.1111/j.2517-6161.1977.tb01600.x.

Rong Fan, Kaiyan Xiao, Minghao Zhu, Liuyi Wang, Kai Dai, and Zhao Yang. GroundVTS: Visual token sampling in multimodal large language models for video temporal grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10408–10418, 2026a.

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

Bo Fang, YuXin Song, Haoyuan Sun, Qiangqiang Wu, Wenhao Wu, and Antoni B. Chan. Threading keyframe with narratives: MLLMs as strong long video comprehenders. In *International Conference on Learning Representations (ICLR)*, 2026.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, Peixian Chen, Yanwei Li, Shaohui Lin, Sirui Zhao, Ke Li, Tong Xu, Xiawu Zheng, Enhong Chen, Caifeng Shan, Ran He, and Xing Sun. 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 (CVPR)*, pp. 24108–24118, 2025.

Kai Hu, Feng Gao, Xiaohan Nie, Peng Zhou, Son Tran, Tal Neiman, Lingyun Wang, Mubarak Shah, Raffay Hamid, Bing Yin, and Trishul Chilimbi. M-LLM based video frame selection for efficient video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13702–13712, 2025.

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

Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. LLaVA-interleave: Tackling multi-image, video, and 3d in large multimodal models. In *International Conference on Learning Representations (ICLR)*, 2025b.

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

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-LLaVA: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5971–5984. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.342.

Shuming Liu, Chen Zhao, Tianqi Xu, and Bernard Ghanem. BOLT: Boost large vision-language model without training for long-form video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3318–3327, 2025.

Bingjun Luo, Tony Wang, Chaoqi Chen, and Xinpeng Ding. ST-SimDiff: Balancing spatiotemporal similarity and difference for efficient video understanding with MLLMs. In *International Conference on Learning Representations (ICLR)*, 2026.

Yuanbin Man, Ying Huang, Chengming Zhang, Bingzhe Li, Wei Niu, and Miao Yin. Adacm^2: On understanding extremely long-term video with adaptive cross-modality memory reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8534–8544, 2025.

Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 46212–46244, 2023. doi: 10.52202/075280-2004.

Geoffrey McLachlan and David Peel. *Finite Mixture Models*. Wiley, 2000. ISBN 9780471006268. doi: 10.1002/0471721182.

Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3): 379–423, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x.

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

Leqi Shen, Guoqiang Gong, Tao He, Yifeng Zhang, Pengzhang Liu, Sicheng Zhao, and Guiguang Ding. FastVID: Dynamic density pruning for fast video large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Guangyu Sun, Archit Singhal, Burak Uzkent, Mubarak Shah, Chen Chen, and Garin Kessler. From frames to clips: Efficient key clip selection for long-form video understanding. *arXiv preprint arXiv:2510.02262*, 2025.

Chong Tang, Sannara Ek, Dirk Koch, Robert Mullins, Alex Weddell, and Jagmohan Chauhan. SURGE: Surprise-guided token reduction for efficient video understanding with VLMs. In *International Conference on Learning Representations (ICLR)*, 2026.

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

Yiyu Wang, Xuyang Liu, Xiyan Gui, Xinying Lin, Boxue Yang, Chenfei Liao, Tailai Chen, and Linfeng Zhang. Accelerating streaming video large language models via hierarchical token compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18523–18533, 2026.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19792–19802, 2025.

Yifeng Yao, Yike Yun, Jing Wang, Huishuai Zhang, Dongyan Zhao, Ke Tian, Zhihao Wang, Minghui Qiu, and Tao Wang. K-frames: Scene-driven any-k keyframe selection for long video understanding. *arXiv preprint arXiv:2510.13891*, 2025.

Chao Yuan, Shimin Chen, Minliang Lin, Limeng Qiao, Guanglu Wan, and Lin Ma. Unicomp: Rethinking video compression through informational uniqueness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18609–18618, 2026.

Kairui Zhang, Zhenyu Yang, Bing Wang, Shengsheng Qian, and Changsheng Xu. Querystream: Advancing streaming video understanding with query-aware pruning and proactive response. In *International Conference on Learning Representations (ICLR)*, 2026.

Qizhe Zhang, Aosong Cheng, Ming Lu, Renrui Zhang, Zhiyong Zhuo, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. Beyond text-visual attention: Exploiting visual cues for effective token pruning in VLMs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20857–20867, 2025a.

Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. Beyond attention or similarity: Maximizing conditional diversity for token pruning in mllms. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, 2025b. doi: 10.52202/085713-0855.

Shaolei Zhang, Qingkai Fang, Zhe Yang, and Yang Feng. LLaVA-mini: Efficient image and video large multimodal models with one vision token. In *International Conference on Learning Representations (ICLR)*, 2025c.

Zheyu Zhang, Ziqi Pang, Shixing Chen, Xiang Hao, Vimal Bhat, and Yu-Xiong Wang. One token per highly selective frame: Towards extreme compression for long video understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, 2025d. doi: 10.52202/085713-1922.

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

Zirui Zhu, Hailun Xu, Yang Luo, Yong Liu, Kanchan Sarkar, Zhenheng Yang, and Yang You. FOCUS: Efficient keyframe selection for long video understanding. In *International Conference on Learning Representations (ICLR)*, 2026.

A APPENDIX

A.1 NOTATION

We summarize the frequently used notation in Table 6.

Table 6: Frequently used notation.

Notation	Definition
T	Number of sampled frames.
K	Number of segments.
$\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_K\}$	Collection of segments.
$\mathbf{z}_{s,t,(h,w)} \in \mathbb{R}^d$	visual token at spatial position (h, w).
$\mathcal{Z}^{(s)}_{<t}$	Visual tokens generated from frames preceding target frame t.
$\hat{p}^F_{s,<t}\left(\mathbf{z}; \mathcal{Z}^{(s)}_{<t}\right)$	STFMem.
$S^F_{s,t,(h,w)}$	Forward score of the visual token at (h, w) in target frame t.
$\mathcal{Z}^{(s)}_{>t}$	Visual tokens generated from frames subsequent to target frame t.
$\hat{p}^B_{s,>t}\left(\mathbf{z}; \mathcal{Z}^{(s)}_{>t}\right)$	STBMem .
$S^B_{s,t,(h,w)}$	Backward score of the visual token.
$\mathcal{Z}^{(s)}$	Visual tokens generated from all frames within segment $\mathcal{G}_s$.
$\hat{p}^L_s\left(\mathbf{z}; \mathcal{Z}^{(s)}\right)$	STLMem.
$S^L_{s,t,(h,w)}$	Local score of the visual token at (h, w) in frame t of $\mathcal{G}_s$.
Δ_r	Distributional shift between adjacent frames.
$D_{\mathrm{KL}}(\cdot\|\cdot)$	Kullback–Leibler divergence.

A.2 DETAILS OF FUSING SCORE

The score fusion process consists of two steps: independently normalizing the three scores (Appendix A.2.1) and then linearly combining them into the final importance score (Appendix A.2.2).

A.2.1 NORMALIZATION

Since the three scores are derived from different reference memories and have different numerical scales, we independently apply min–max normalization across all visual tokens within each segment $\mathcal{G}_s$ to the forward score $S^F_{s,t,(h,w)}$, backward score $S^B_{s,t,(h,w)}$, and local score $S^L_{s,t,(h,w)}$, obtaining the normalized forward score $\hat{S}^F_{s,t,(h,w)}$, backward score $\hat{S}^B_{s,t,(h,w)}$, and local score $\hat{S}^L_{s,t,(h,w)}$, respectively, each taking values in $[0, 1]$.

A.2.2 COMBINATION

We first combine the normalized forward and backward saliency scores with equal weights to obtain the bidirectional saliency, $\hat{S}_{s,t,(h,w)} = (\hat{S}^F_{s,t,(h,w)} + \hat{S}^B_{s,t,(h,w)})/2$. We then linearly combine the bidirectional saliency with the normalized local information score to obtain the final importance score $Q_{s,t,(h,w)}$ for each visual token, $Q_{s,t,(h,w)} = (1-\alpha)\hat{S}^L_{s,t,(h,w)} + \alpha\hat{S}_{s,t,(h,w)}$, where $0 \le \alpha \le 1$. Equivalently, the final importance score is expressed as:

$$Q_{s,t,(h,w)} = (1-\alpha)\hat{S}^L_{s,t,(h,w)} + \frac{\alpha}{2}\hat{S}^F_{s,t,(h,w)} + \frac{\alpha}{2}\hat{S}^B_{s,t,(h,w)}. \tag{13}$$

Thus, $Q_{s,t,(h,w)}$ assigns high importance to visual tokens that are informative within their local segment or temporally pivotal with respect to preceding or subsequent visual memory.

For the first frame of segment $\mathcal{G}_s$ with $T_s > 1$, i.e., $t = 1$, no preceding visual observations are available within the segment. We therefore replace the unavailable forward saliency with the local information score while retaining the backward saliency:

$$Q_{s,1,(h,w)} = \left(1 - \frac{\alpha}{2}\right)\hat{S}^L_{s,1,(h,w)} + \frac{\alpha}{2}\hat{S}^B_{s,1,(h,w)}.$$

Conversely, for the last frame of segment $\mathcal{G}_s$ with $T_s > 1$, i.e., $t = T_s$, no subsequent visual observations are available within the segment. We therefore replace the unavailable backward saliency with the local information score while retaining the forward saliency:

$$Q_{s,T_s,(h,w)} = \left(1 - \frac{\alpha}{2}\right) \hat{S}^L_{s,T_s,(h,w)} + \frac{\alpha}{2} \hat{S}^F_{s,T_s,(h,w)}.$$

For a single-frame segment $\mathcal{G}_s$, i.e., $T_s = 1$, neither preceding nor subsequent visual observations are available within the segment. We therefore replace both unavailable directional saliency scores with the local information score, so the final importance score reduces to:

$$Q_{s,1,(h,w)} = \hat{S}^L_{s,1,(h,w)}.$$

A.3 Details of Segment-wise Visual-Token Selection

Beyond estimating the visual-token importance score $Q_{s,t,(h,w)}$, selecting visual tokens requires a budget that determines how many tokens to retain from each temporal segment and a selection strategy that determines which tokens to retain under that budget. In this section, we describe segment-wise token budget allocation and within-segment token selection in Sections A.3.1 and A.3.2, respectively.

A.3.1 Entropy-Based Token Budget Allocation

After determining the importance score of each visual token, one of the remaining questions is how many visual tokens should be retained from each segment under a given total token budget. A straightforward approach is to allocate the token budget in proportion to the number of frames in each segment, assuming that a segment containing more frames carries more visual information. However, videos inherently exhibit temporal redundancy across frames. Consequently, the naive frame-count-based approach risks assigning a high token budget to a segment composed of highly redundant consecutive frames, although it contains relatively little information. To address this limitation, CueMiner predicts the expected information content of the tokens within each temporal segment and allocates the token budget in proportion to this expectation, thereby assigning more tokens to more informative segments.

To estimate this expectation, CueMiner computes the entropy of the segment's visual-token distribution, since entropy represents the expected information content of the visual tokens within that segment. To compute this entropy, CueMiner uses the approximated segment-specific visual-token distribution $\hat{p}^L_s$ for segment $\mathcal{G}_s$ (Equation 8):

$$\mathcal{H}_s = \mathbb{E}_{\mathbf{z} \sim \hat{p}^L_s(\cdot; \mathcal{Z}^{(s)})} \left[-\log \hat{p}^L_s(\mathbf{z}; \mathcal{Z}^{(s)}) \right]. \tag{14}$$

We allocate an integer token budget B_s to each segment $\mathcal{G}_s$, assigning larger budgets to segments with higher estimated entropy $\mathcal{H}_s$ (Equation 14), subject to

$$B_s \in \mathbb{Z}_{\geq 0}, \qquad B_s \leq N_s, \qquad \sum_{s=1}^{K} B_s = B, \tag{15}$$

where the segment token count $N_s = T_s HW$ is the number of visual tokens in segment $\mathcal{G}_s$, the segment count K is the number of temporal segments, and the total visual-token count $N_{\mathcal{V}} = THW$ is the number of visual tokens generated from the T sampled frames before compression. Given a target retention ratio $\rho \in (0, 1]$, the total token budget $B = \lfloor \rho N_{\mathcal{V}} \rfloor$ is the number of visual tokens retained across all segments, where $\lfloor x \rfloor$ denotes rounding down.

A.3.2 Visual-Token Selection

After determining the segment-dependent token budget B_s (Equation 15), the remaining task is to select visual tokens within each segment under this budget. A naive strategy selects the top-B_s visual tokens according to their estimated importance scores $Q_{s,t,(h,w)}$. However, this strategy selects visual tokens with substantial semantic overlap. Relying solely on $Q_{s,t,(h,w)}$ therefore produces a compressed subset containing mutually redundant tokens, wasting the limited budget on tokens that convey similar information. To address this issue, CueMiner jointly considers token importance

and semantic redundancy during selection. This produces a subset that retains important visual information while reducing redundancy among the selected tokens.

To jointly evaluate the importance and redundancy of a candidate subset, we formulate selection using a determinantal point process (DPP). The intuition behind DPP selection is to maximize the volume spanned by the selected visual-token embeddings after unit normalization and weighting by their importance scores $Q_{s,t,(h,w)}$. Higher importance scores increase the lengths of these weighted vectors, while less redundant, more nearly orthogonal directions allow them to span a larger volume. Maximizing this volume therefore favors a subset whose tokens are both important and mutually non-redundant.

For clarity of explanation, throughout the remainder of this section we fix a temporal segment $\mathcal{G}_s$ and enumerate its $N_s = T_s HW$ visual tokens by $i = 1, \ldots, N_s$. Each token index i corresponds to a unique position (t, h, w) within the segment; we abbreviate the scalar importance score as $Q_i = Q_{s,t,(h,w)} \in [0, 1]$ and the visual-token embedding as $\mathbf{z}_i = \mathbf{z}_{s,t,(h,w)} \in \mathbb{R}^d$. The same formulation applies independently to every segment with its allocated budget B_s.

Before formalizing the volume, we define the matrices that represent the weighted embeddings. To separate embedding direction from importance, we first define the unit-normalized embedding $\hat{\mathbf{z}}_i = \mathbf{z}_i / \|\mathbf{z}_i\|_2 \in \mathbb{R}^d$. This normalization prevents the original embedding magnitude from acting as an additional importance weight. We collect these vectors into the normalized embedding matrix $\hat{\mathbf{Z}} = [\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_{N_s}] \in \mathbb{R}^{d \times N_s}$ and define the importance diagonal matrix $\mathbf{D} = \mathrm{diag}(Q_1, \ldots, Q_{N_s}) \in \mathbb{R}^{N_s \times N_s}$. To weight each embedding by its importance, we construct the weighted embedding matrix $\mathbf{E} \in \mathbb{R}^{d \times N_s}$:

$$\mathbf{E} = \hat{\mathbf{Z}}\mathbf{D} = [Q_1 \hat{\mathbf{z}}_1, \ldots, Q_{N_s} \hat{\mathbf{z}}_{N_s}].$$

The corresponding Gram matrix $\mathbf{L} \in \mathbb{R}^{N_s \times N_s}$, which serves as the DPP kernel, records the inner products between these weighted vectors:

$$\mathbf{L} = \mathbf{E}^\top \mathbf{E} = \mathbf{D}\hat{\mathbf{Z}}^\top \hat{\mathbf{Z}}\mathbf{D}, \qquad \mathbf{L}_{ij} = Q_i (\hat{\mathbf{z}}_i^\top \hat{\mathbf{z}}_j) Q_j.$$

Then, we are able to express the volume spanned by a selected subset of weighted, normalized visual-token embeddings. Let the selected index set $\mathcal{A} \subseteq \{1, \ldots, N_s\}$ contain B_s token indices, and let the selected embedding matrix $\mathbf{E}_\mathcal{A} \in \mathbb{R}^{d \times B_s}$ contain the corresponding columns of $\mathbf{E}$. Its selected Gram matrix $\mathbf{L}_\mathcal{A} = \mathbf{E}_\mathcal{A}^\top \mathbf{E}_\mathcal{A} \in \mathbb{R}^{B_s \times B_s}$ is the principal submatrix of $\mathbf{L}$ indexed by $\mathcal{A}$. Since $\mathbf{E}_\mathcal{A}$ need not be square, we compute the spanned volume using its Gram determinant:

$$\mathrm{Vol}(\mathbf{E}_\mathcal{A}) = \sqrt{\det(\mathbf{E}_\mathcal{A}^\top \mathbf{E}_\mathcal{A})} = \sqrt{\det(\mathbf{L}_\mathcal{A})}.$$

Therefore, to maximize this volume under the segment budget B_s, it suffices to maximize $\det(\mathbf{L}_\mathcal{A})$, since the square root is increasing on nonnegative values. For DPP selection, our objective is thus to find the optimal index set $\mathcal{A}^\star$:

$$\mathcal{A}^\star = \underset{\mathcal{A} \subseteq \{1, \ldots, N_s\},\, |\mathcal{A}| = B_s}{\arg\max} \det(\mathbf{L}_\mathcal{A}).$$

A.4 Approximation of Distributions

We detail how CueMiner approximates the four visual-token distributions: the STFMem distribution $\hat{p}^F_{s,<t}(\mathbf{z}; \mathcal{Z}^{(s)}_{<t})$ (Equation 2), the STBMem distribution $\hat{p}^B_{s,>t}(\mathbf{z}; \mathcal{Z}^{(s)}_{>t})$ (Equation 5), the STLMem distribution $\hat{p}^L_s(\mathbf{z}; \mathcal{Z}^{(s)})$ (Equation 8), and the frame-wise distribution $\hat{p}_r(\mathbf{z}; \mathcal{Z}_r)$ (Equation 11). At a high level, all four distributions are approximated in the same way: each is modeled by a single diagonal gaussian whose parameters are estimated from the visual-token embeddings in its corresponding set: the preceding-token set $\mathcal{Z}^{(s)}_{<t}$ (Equation 1) for STFMem $\hat{p}^F_{s,<t}$, the subsequent-token set $\mathcal{Z}^{(s)}_{>t}$ (Equation 4) for STBMem $\hat{p}^B_{s,>t}$, the segment-token set $\mathcal{Z}^{(s)}$ (Equation 7) for STLMem $\hat{p}^L_s$, and the frame-token set $\mathcal{Z}_r$ (Equation 10) for the frame-wise distribution $\hat{p}_r$. Since all four distributions are approximated using an identical procedure, we describe how to approximate a generic density $\hat{p}(\mathbf{z}; \mathcal{Z})$ from a generic visual-token set $\mathcal{Z} = \{\mathbf{z}_i\}_{i=1}^N$, where $\hat{p}(\mathbf{z}; \mathcal{Z})$ denotes the approximated density for $\mathcal{Z}$, $\mathcal{Z}$ denotes one of the four corresponding sets above, and $\mathbf{z}_i$ denotes an embedding in that set.

Table 7: Incremental peak GPU memory (GiB) during LLM prefill on LOV 1.5 as the number of sampled frames increases from 32 to 192, excluding the 15.89 GiB occupied by model weights. The vanilla model retains all visual tokens, whereas CUEMINER uses DPP selection at an estimated 10% retention ratio across all frame counts and achieves accuracy exceeding that of the vanilla model (see Table 1); bold denotes the lowest memory usage.

Method	32 frames	64 frames	96 frames	128 frames	160 frames	192 frames
Vanilla	1.141	2.421	3.701	4.932	6.157	7.389
CUEMINER (proposed)	**0.152**	**0.304**	**0.457**	**0.590**	**0.734**	**0.886**

Table 8: Absolute accuracy (%) on MLVU with 32 sampled frames, where vanilla LOV, LOV 1.5, LOV 2.0, and Q3-VL achieve 47.7, 46.0, 52.7, and 46.6, respectively. Bold denotes the best at each retention ratio.

Method	**LOV**			**LOV 1.5**			**LOV 2.0**			**Q3-VL**		
	20	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**
FastV (Chen et al., 2024)	43.8	40.6	38.1	41.5	38.2	34.2	44.0	39.6	36.1	45.2	41.9	36.6
VisionZip (Yang et al., 2025)	45.3	42.8	36.1	44.7	41.8	40.1	47.7	40.6	38.1	43.5	40.1	37.7
HoliTom (Shao et al., 2025)	48.8	47.2	46.3	42.6	39.7	39.8	46.8	39.9	37.7	45.1	40.8	36.9
FastVID (Shen et al., 2025)	44.2	43.6	42.9	44.4	40.7	36.8	46.2	43.4	40.5	43.1	41.0	39.9
UniComp (Yuan et al., 2026)	46.8	42.8	43.2	41.8	42.4	40.2	39.5	37.8	37.0	43.4	38.5	36.6
SURGE (Tang et al., 2026)	46.3	44.5	41.3	44.4	42.0	38.9	50.5	44.3	37.5	43.3	39.9	–
FlashVID (Fan et al., 2026b)	45.9	43.1	42.3	44.5	42.3	40.8	51.4	47.5	42.2	45.1	42.3	38.5
CUEMINER (proposed)	**50.0**	**47.3**	**47.4**	**47.6**	**47.4**	**45.6**	**53.5**	**51.5**	**48.7**	**47.6**	**45.9**	**43.2**

Specifically, using the embeddings in the generic visual-token set $\mathcal{Z}$, we compute the empirical coordinate-wise mean and variance as

$$\hat{\boldsymbol{\mu}}_{\mathcal{Z}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{z}_i, \qquad \hat{\mathbf{v}}_{\mathcal{Z}} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{z}_i - \hat{\boldsymbol{\mu}}_{\mathcal{Z}}) \odot (\mathbf{z}_i - \hat{\boldsymbol{\mu}}_{\mathcal{Z}}),$$

where $\odot$ denotes element-wise multiplication. Then, we construct a single diagonal gaussian:

$$\hat{p}(\mathbf{z}; \mathcal{Z}) = \mathcal{N}(\mathbf{z}; \hat{\boldsymbol{\mu}}_{\mathcal{Z}}, \boldsymbol{\Sigma}_{\mathcal{Z}}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_{\mathcal{Z}}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{z} - \hat{\boldsymbol{\mu}}_{\mathcal{Z}})^{\top}\boldsymbol{\Sigma}_{\mathcal{Z}}^{-1}(\mathbf{z} - \hat{\boldsymbol{\mu}}_{\mathcal{Z}})\right),$$

where $\boldsymbol{\Sigma}_{\mathcal{Z}} = \mathrm{diag}(\hat{\mathbf{v}}_{\mathcal{Z}})$ is the diagonal covariance matrix whose diagonal entries are given by $\hat{\mathbf{v}}_{\mathcal{Z}}$.

A.5 DOES CUEMINER REDUCE MEMORY WHILE MAINTAINING ACCURACY?

We assess whether CUEMINER reduces LLM-prefill memory while retaining accuracy. Table 7 shows that CUEMINER reduces memory consumption while maintaining accuracy. Specifically, with 192 sampled frames, the vanilla model consumes 8.3× more incremental memory than CUEMINER, despite CUEMINER achieving higher accuracy at a 10% retention ratio (see Tables 7 and 1).

A.6 PER-BENCHMARK ACCURACY RESULTS

We provide the full accuracy results for each benchmark. Tables 8, 9, 10, and 11 provide the full per-benchmark accuracy results for MLVU, EgoSchema, VideoMME, and MVBench, respectively. Overall, CUEMINER generally maintains robust accuracy across benchmarks, even at low retention ratios. In particular, CUEMINER achieves the best accuracy across all VLLMs and retention ratios on MLVU. Given that MLVU evaluates long video understanding through holistic, single-detail, and multi-detail LVU tasks, this suggests that CUEMINER effectively preserves both global and local visual information required for long-video understanding, even at low retention ratios.

Table 9: Absolute accuracy (%) on EgoSchema with 32 sampled frames, where vanilla LOV, LOV 1.5, LOV 2.0, and Q3-VL achieve 62.8, 49.2, 65.0, and 65.8, respectively. Bold denotes the best at each retention ratio.

Method	LOV			LOV 1.5			LOV 2.0			Q3-VL		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	60.2	57.4	55.6	48.0	46.2	42.6	55.6	51.8	48.0	**65.0**	61.6	52.2
VisionZip (Yang et al., 2025)	61.0	60.6	52.4	48.4	49.6	**49.0**	60.6	52.6	51.6	63.8	58.2	56.4
HoliTom (Shao et al., 2025)	62.2	**62.4**	60.4	47.0	47.6	47.6	**64.2**	42.8	42.0	62.8	59.8	54.8
FastVID (Shen et al., 2025)	62.2	61.0	62.8	47.6	47.0	46.4	62.4	58.0	54.8	61.4	59.8	56.4
UniComp (Yuan et al., 2026)	61.4	60.4	58.2	48.6	47.6	46.6	44.0	40.2	38.0	62.2	59.2	54.6
SURGE (Tang et al., 2026)	62.2	60.4	54.2	49.0	48.4	44.2	61.0	56.8	49.2	62.0	58.6	–
FlashVID (Fan et al., 2026b)	62.0	61.6	**63.6**	48.2	48.2	46.6	62.8	58.2	56.8	64.8	**63.0**	**57.8**
CUEMINER (proposed)	**62.6**	62.0	63.4	**49.2**	**50.8**	48.8	63.8	**60.8**	**59.0**	62.6	59.8	57.4

Table 10: Absolute accuracy (%) on VideoMME with 32 sampled frames, where vanilla LOV, LOV 1.5, LOV 2.0, and Q3-VL achieve 58.5, 60.1, 65.0, and 63.7, respectively. Bold denotes the best at each retention ratio.

Method	LOV			LOV 1.5			LOV 2.0			Q3-VL		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	55.7	52.7	49.6	56.9	53.3	50.0	58.9	55.1	51.6	59.5	56.1	52.8
VisionZip (Yang et al., 2025)	57.7	52.9	48.8	57.0	55.8	54.1	57.0	52.3	49.8	60.2	57.2	53.3
HoliTom (Shao et al., 2025)	**59.3**	57.3	**56.4**	57.0	55.5	54.7	61.6	54.1	53.2	60.2	57.2	53.7
FastVID (Shen et al., 2025)	59.2	57.4	53.9	56.5	54.4	53.4	59.5	57.3	54.7	60.2	56.6	53.1
UniComp (Yuan et al., 2026)	58.7	55.7	52.7	55.4	55.1	53.8	59.3	56.8	55.3	58.5	55.5	51.6
SURGE (Tang et al., 2026)	55.9	53.2	50.9	58.1	55.3	52.6	61.1	58.1	53.1	59.0	54.9	–
FlashVID (Fan et al., 2026b)	58.4	57.3	53.9	57.4	56.8	55.6	61.8	56.7	54.3	**61.1**	58.6	54.7
CUEMINER (proposed)	58.9	**57.5**	55.9	**59.3**	**58.9**	**56.3**	**63.1**	**60.9**	**57.7**	60.6	**59.0**	**56.1**

A.7 TASK-SPECIFIC ACCURACY RESULTS

We evaluate whether CUEMINER preserves temporal, global, and local video understanding under compression using task-specific results. Tables 12, 13, and 14 assess these abilities through State Change, which determines changes in object states throughout a video; Anomaly Recognition, which identifies anomalous behavior in long surveillance footage; and NeedleQA, which locates a specific segment within a long background video, respectively. Table 12 shows strong performance on State Change, indicating that CUEMINER retains temporal information needed to reason about changing object states. Table 13 shows that CUEMINER performs strongly on Anomaly Recognition, indicating that it preserves globally relevant events. Table 14 shows strong performance on NeedleQA, indicating that it preserves sparse local details. Together, these results suggest that CUEMINER not only retains temporal information required to reason about state changes, but also preserves globally relevant events and sparse local details under aggressive compression.

A.8 TTFT ANALYSIS

We assess the TTFT of CUEMINER and competing methods on LOV 1.5. Table 15 shows that the TTFT of CUEMINER is comparable to that of the competing methods. This is because CUEMINER has lower LLM prefill plus first-token generation latency than the competing methods, despite its longer method-specific compression stage than most competing methods. For this TTFT experiment, we adopt accuracy-matched settings: competing methods use a 20% retention ratio, whereas CUEMINER uses 5%. By maintaining robust accuracy at a lower retention ratio, CUEMINER processes fewer visual tokens and thereby achieves lower LLM prefill plus first-token generation latency.

Table 11: Absolute accuracy (%) on MVBench with 32 sampled frames, where vanilla LOV, LOV 1.5, LOV 2.0, and Q3-VL achieve 58.2, 52.4, 65.9, and 67.8, respectively. Bold denotes the best at each retention ratio.

Method	LOV			LOV 1.5			LOV 2.0			Q3-VL		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	56.3	53.5	50.9	50.7	50.1	47.9	60.1	54.4	52.5	65.0	62.2	55.0
VisionZip (Yang et al., 2025)	57.2	53.9	44.1	50.7	50.2	49.9	60.2	51.9	48.6	65.5	59.2	56.3
HoliTom (Shao et al., 2025)	57.5	**57.1**	**55.9**	50.7	49.8	48.8	60.3	51.0	50.7	**66.1**	**63.5**	57.8
FastVID (Shen et al., 2025)	58.5	56.8	54.4	51.0	50.6	50.0	**64.8**	62.6	57.8	64.8	62.3	56.6
UniComp (Yuan et al., 2026)	57.0	55.6	53.9	48.9	48.8	48.5	60.9	55.6	43.0	64.8	61.5	54.8
SURGE (Tang et al., 2026)	56.4	55.6	52.4	52.0	49.6	–	64.0	61.1	–	64.4	–	–
FlashVID (Fan et al., 2026b)	**58.7**	56.9	54.6	52.1	51.3	50.7	62.1	58.9	55.2	**66.1**	63.3	56.9
CUEMINER (proposed)	57.5	56.7	55.0	**52.2**	**52.1**	**51.8**	64.7	**64.5**	**61.3**	65.5	62.9	**59.6**

Table 12: Absolute accuracy (%) on the State Change subtask of MVBench with 32 sampled frames, where vanilla LOV, LOV 1.5, LOV 2.0, and Q3-VL achieve 52.0, 53.5, 61.0, and 72.5, respectively. Bold denotes the best among the reported compression methods; dashes indicate unavailable results.

Method	LOV			LOV 1.5			LOV 2.0			Q3-VL		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	49.0	50.0	45.0	47.0	44.0	37.0	51.5	45.0	43.5	66.0	63.0	50.5
VisionZip (Yang et al., 2025)	48.0	44.5	37.5	50.0	50.0	47.5	56.5	40.5	40.5	70.0	63.0	56.0
HoliTom (Shao et al., 2025)	51.0	51.0	47.5	46.0	45.5	48.0	47.0	36.5	36.0	65.0	64.0	57.0
FastVID (Shen et al., 2025)	51.5	50.5	48.0	50.5	50.5	47.0	59.0	56.0	50.5	69.0	66.5	**62.5**
UniComp (Yuan et al., 2026)	51.5	51.0	**49.0**	44.0	43.5	44.0	52.5	39.0	31.0	65.5	65.0	54.0
SURGE (Tang et al., 2026)	**53.5**	51.5	47.0	48.5	49.0	–	**61.5**	59.5	–	65.5	–	–
FlashVID (Fan et al., 2026b)	52.0	48.5	48.0	50.5	52.5	47.0	60.0	54.5	48.0	**72.5**	65.5	56.0
CUEMINER (proposed)	51.0	**52.0**	**49.0**	**51.0**	**54.0**	**52.0**	**61.5**	**60.5**	**58.5**	72.0	**69.0**	**62.5**

A.9 ABLATION STUDY

We conduct additional quantitative and qualitative ablation studies to assess the contributions of individual components of CUEMINER.

A.9.1 DOES BACKWARD SCORING PRESERVE PRE-EVENT VISUAL EVIDENCE?

We investigate whether the gain from backward scoring $S^B_{s,t,(h,w)}$ (Equation 6) arises from preserving temporally antecedent visual evidence, rather than merely introducing an additional saliency signal. As shown in Table 16, removing the backward score $S^B_{s,t,(h,w)}$ consistently reduces accuracy across all model–retention pairs. This result indicates that backward scoring provides additional temporal evidence that improves state-transition reasoning beyond forward scoring alone.

We further qualitatively examine what visual evidence is discovered by backward scoring. To illustrate this point, we use an example from the MVBench State Change task: Figure 6(a) shows eight sampled frames, with no apple on the table in the first three frames and a hand placing an apple on it in later frames, leaving the apple on the table in the final frame. In this example, existing surprisal-based compression methods, which typically identify newly emerging visual content using temporal surprisal in the forward direction, do not excavate the preceding evidence of the apple's absence since the first three frames contain similar content and are therefore not detected as temporally distinctive, whereas the apple-placement frame is retained since it differs from the preceding observations without the apple. However, if only the apple-placement frame is retained, the model observes the event itself but cannot interpret the state transition because it lacks the preceding evidence that the apple was initially absent. This motivates preserving the pre-event state without the apple to understand the state transition correctly.

Table 13: Absolute accuracy (%) on the Anomaly Recognition subtask of MLVU with 32 sampled frames, where vanilla LOV, LOV 1.5, LOV 2.0, and Q3-VL achieve 48.7, 56.4, 64.1, and 53.8, respectively. Bold denotes the best compression method; dashes indicate unavailable results.

Method	LOV			LOV 1.5			LOV 2.0			Q3-VL		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	46.2	43.6	35.9	43.6	35.9	20.5	43.6	38.5	28.2	**59.0**	35.9	28.2
VisionZip (Yang et al., 2025)	53.8	41.0	20.5	56.4	51.3	46.2	46.2	41.0	38.5	51.3	30.8	28.2
HoliTom (Shao et al., 2025)	56.4	**53.8**	46.2	46.2	38.5	43.6	33.3	23.1	23.1	41.0	41.0	30.8
FastVID (Shen et al., 2025)	46.2	46.2	38.5	56.4	56.4	43.6	51.3	43.6	38.5	48.7	38.5	35.9
UniComp (Yuan et al., 2026)	**59.0**	**53.8**	53.8	41.0	46.2	46.2	33.3	25.6	20.5	35.9	20.5	15.4
SURGE (Tang et al., 2026)	56.4	46.2	38.5	**61.5**	53.8	46.2	61.5	48.7	33.3	51.3	41.0	–
FlashVID (Fan et al., 2026b)	46.2	46.2	43.6	56.4	53.8	46.2	41.0	43.6	35.9	51.3	41.0	28.2
CUEMINER (proposed)	53.8	48.7	**56.4**	**61.5**	**64.1**	**56.4**	**64.1**	**56.4**	**48.7**	53.8	**46.2**	**38.5**

Table 14: Absolute accuracy (%) on the NeedleQA subtask of MLVU with 32 sampled frames, where vanilla LOV, LOV 1.5, LOV 2.0, and Q3-VL achieve 60.0, 60.0, 55.0, and 50.0, respectively. Bold denotes the best among the reported compression methods; dashes indicate unavailable results.

Method	LOV			LOV 1.5			LOV 2.0			Q3-VL		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	50.0	43.3	38.3	51.7	46.7	36.7	45.0	41.7	41.7	50.0	43.3	35.0
VisionZip (Yang et al., 2025)	50.0	36.7	35.0	45.0	46.7	43.3	48.3	43.3	38.3	**51.7**	41.7	35.0
HoliTom (Shao et al., 2025)	**61.7**	**60.0**	**51.7**	51.7	35.0	40.0	55.0	46.7	43.3	50.0	40.0	35.0
FastVID (Shen et al., 2025)	55.0	50.0	45.0	50.0	40.0	31.7	51.7	45.0	41.7	46.7	36.7	36.7
UniComp (Yuan et al., 2026)	60.0	51.7	48.3	41.7	41.7	40.0	41.7	41.7	41.7	48.3	41.7	40.0
SURGE (Tang et al., 2026)	46.7	43.3	30.0	51.7	41.7	43.3	51.7	43.3	38.3	46.7	33.3	–
FlashVID (Fan et al., 2026b)	51.7	51.7	46.7	51.7	45.0	40.0	**61.7**	**55.0**	43.3	48.3	38.3	33.3
CUEMINER (proposed)	**61.7**	**60.0**	43.3	**58.3**	**50.0**	**46.7**	**61.7**	**55.0**	**55.0**	50.0	**50.0**	**53.3**

Consistent with this motivation, Figure 6(b) shows that SURGE assigns high scores to the salient placement event in green but low scores to the preceding evidence of the apple's absence. By contrast, Figure 6(c) visualizes the forward and backward scores as heatmaps before token selection, with regions receiving high forward scores shown in red and regions receiving high backward scores shown in blue. The red regions highlight the apple-placement event, whereas the blue regions highlight the preceding state without the apple. Thus, the two scores highlight complementary temporal evidence corresponding to the post-transition event and its antecedent context, respectively. Together with the quantitative ablation in Table 16, this example supports the role of backward scoring in preserving pre-event visual evidence required for state-transition reasoning.

A.9.2 Is VTDS-Based Segmentation Effective?

We assess the effectiveness of segmentation using the VTDS score Δ_r (Equation 12) in terms of benchmark accuracy.

Table 17 shows that VTDS achieves the best accuracy across all four VLLMs and retention ratios, outperforming both fixed-length intervals and cosine similarity. Moreover, cosine-similarity-based segmentation yields lower accuracy than fixed-length intervals in some settings. This suggests that cosine-similarity-based segmentation groups frames with heterogeneous visual content into a single segment, making the visual-token distributions represented by STFMem $\hat{p}^F_{s,<t}$ (Equation 2), STBMem $\hat{p}^B_{s,>t}$ (Equation 5), and STLMem $\hat{p}^L_s$ (Equation 8) harder to approximate accurately with a gaussian density (see the Implementation Details in Section 4.1 and Figure 3(b)), thereby distorting the corresponding forward scores $S^F_{s,t,(h,w)}$ (Equation 3), backward scores $S^B_{s,t,(h,w)}$ (Equation 6), and local scores $S^L_{s,t,(h,w)}$ (Equation 9) and reducing compression accuracy.

Table 15: Time to First Token (TTFT) breakdown on LOV 1.5 with 32 sampled frames under accuracy-matched settings, using a 20% retention ratio for competing methods and 5% for CUEMINER with DPP selection; relative accuracies are from Table 1. TTFT comprises vision encoding, method-specific compression, and LLM prefill plus first-token generation; all times are measured in seconds.

Method	Relative acc. (%)	TTFT	Vision encoding	Compression	LLM prefill + first-token generation
FastV (Chen et al., 2024)	94.8	2.58	2.26	0.01	0.31
VisionZip (Yang et al., 2025)	96.8	2.61	2.26	0.04	0.31
HoliTom (Shao et al., 2025)	94.9	2.71	2.26	0.14	0.31
FastVID (Shen et al., 2025)	96.2	3.50	2.26	0.93	0.31
UniComp (Yuan et al., 2026)	93.8	2.69	2.26	0.12	0.31
SURGE (Tang et al., 2026)	98.0	2.62	2.26	0.05	0.31
FlashVID (Fan et al., 2026b)	97.4	2.96	2.26	0.39	0.31
CUEMINER (proposed)	97.7	2.64	2.26	0.25	0.13

Table 16: Backward-score ablation on the MVBench State Change task with 32 sampled frames, comparing CUEMINER with a variant obtained by removing the backward score. Bold denotes the better accuracy (%) within each model–retention pair.

Variant	**LOV**			**LOV 1.5**			**LOV 2.0**			**Q3-VL**		
	20	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**
w/o Backward	50.0	50.0	48.0	50.5	53.0	51.5	60.0	59.5	57.5	70.5	68.0	62.0
CUEMINER (proposed)	**51.0**	**52.0**	**49.0**	**51.0**	**54.0**	**52.0**	**61.5**	**60.5**	**58.5**	**72.0**	**69.0**	**62.5**

We qualitatively assess how VTDS separates visually heterogeneous frames. Figure 7 shows that VTDS-based segmentation separates visually heterogeneous frames into distinct segments, whereas cosine similarity fails to split them. This result shows that cosine similarity risks grouping visually heterogeneous content into a single segment, suggesting that a simple probabilistic density function is insufficient to accurately approximate the distribution of visual-token embeddings within that segment.

A.9.3 DO THE SCORING TERMS CONTRIBUTE TO ACCURACY?

We assess whether the forward score $S^F_{s,t,(h,w)}$ (Equation 3), backward score $S^B_{s,t,(h,w)}$ (Equation 6), and local score $S^L_{s,t,(h,w)}$ (Equation 9) contribute to CUEMINER's accuracy gains. Table 18 shows that incorporating additional scoring terms generally improves accuracy, with the full combination achieving the best result in every completed setting. In particular, fusing all three scores improves accuracy by 14.2% relative to the uniform-score variant on LOV 1.5 at a 10% retention ratio.

A.9.4 DOES ENTROPY-BASED BUDGETING IMPROVE ACCURACY?

We assess the effectiveness of entropy-based segment-wise token budgeting, which uses the segment entropy $\mathcal{H}_s$ (Equation 14), in an ablation study. Table 19 shows that CUEMINER's entropy-based adaptive budgeting consistently achieves the highest accuracy across model–retention settings. Uniform allocation generally outperforms frame-proportional allocation, suggesting that frame count alone can over-allocate tokens to segments containing highly redundant consecutive frames.

A.9.5 DOES A SINGLE-GAUSSIAN SUFFICE FOR CUEMINER?

We assess whether a single-gaussian surrogate suffices for CUEMINER by comparing it with gaussian mixture models (GMMs) (McLachlan & Peel, 2000) for the visual-token distributions in Equation 2, Equation 5, and Equation 8. Table 20 shows that GMM variants fitted using expectation-maximization (EM) (Dempster et al., 1977) reduce downstream accuracy. This indicates that the greater expressiveness of GMMs does not translate into better compression decisions, supporting the single-gaussian surrogate as an effective choice for CUEMINER.

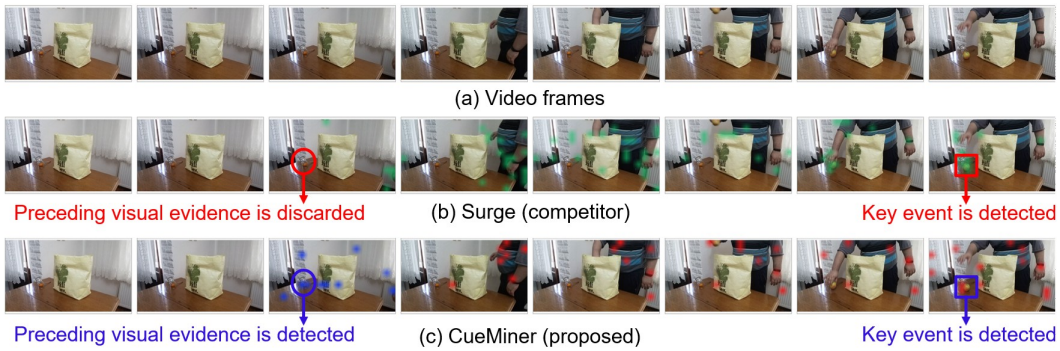

Figure 6: (a) Sampled video frames; (b) SURGE assigns high scores to the salient placement event but low scores to the preceding evidence of the apple's absence; (c) in CUEMINER, the forward score $S^{F}_{s,t,(h,w)}$ highlights the apple-placement event, whereas the backward score $S^{B}_{s,t,(h,w)}$ highlights the preceding state without the apple, with high forward-score regions shown in red and high backward-score regions in blue; heatmaps show scores before token selection.

Table 17: MLVU accuracy results, where all methods use the same number of segment boundaries and fixed-length segmentation places them at fixed frame intervals. Bold denotes the best accuracy.

Segmentation method	**LOV**			**LOV 1.5**			**LOV 2.0**			**Q3-VL**		
	20	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**
Fixed-length intervals	44.1	44.0	43.1	46.4	43.7	42.1	52.4	48.0	46.0	44.4	44.6	41.9
Cosine similarity	48.6	45.7	45.5	46.2	45.4	44.0	51.0	48.4	46.3	45.4	44.5	41.4
CUEMINER (proposed)	**50.0**	**47.3**	**47.4**	**47.6**	**47.4**	**45.6**	**53.5**	**51.5**	**48.7**	**47.6**	**45.9**	**43.2**

A.9.6 IS CUEMINER EFFECTIVE WITHOUT DPP SELECTION?

We evaluate whether CUEMINER remains effective without DPP selection. Table 21 shows that CUEMINER with Top-k selection generally outperforms the baselines while achieving accuracy comparable to CUEMINER with DPP selection. Specifically, with 32 sampled frames, CUEMINER w/ Top-k achieves the second-best accuracy in 11 of 12 model–retention settings; the only exception is LOV at 10% retention, where HoliTom ranks second. These results show that the proposed ideas remain effective even without DPP selection.

We additionally assess whether CUEMINER remains effective without DPP selection as the number of sampled frames increases. Interestingly, as the number of sampled frames increases, CUEMINER with Top-k selection becomes increasingly effective, surpassing CUEMINER with DPP selection at most retention ratios from 64 frames onward and at all retention ratios with 80 or 96 frames (Table 22). Specifically, at 48 frames, CUEMINER with DPP selection achieves the highest accuracy at all retention ratios, whereas CUEMINER with Top-k selection ranks second at 20% and 5% retention (45.2% and 39.7%) but falls below FlashVID at 10% retention (41.1% vs. 41.8%). However, at 64 frames, CUEMINER with Top-k selection surpasses CUEMINER with DPP selection at 10% and 5% retention (47.9% vs. 47.5%; 46.7% vs. 45.9%) and remains ahead at all retention ratios at 80 and 96 frames. These results suggest that, when processing a large number of sampled frames, CUEMINER achieves higher accuracy with Top-k selection than with DPP selection.

Motivated by these results, we evaluate the efficiency of CUEMINER with Top-k selection in terms of time to first token (TTFT) as the number of sampled frames increases. Table 23 shows that CUEMINER with Top-k selection achieves the lowest TTFT across all evaluated frame–retention settings. Specifically, with 96 sampled frames at 5% retention, CUEMINER with Top-k selection achieves a TTFT of 1.18 seconds, compared with 1.25 seconds for the fastest competing baseline, VisionZip. The TTFT of CUEMINER with DPP selection is up to approximately 1.52 times that of CUEMINER with Top-k selection, reaching 2.04 versus 1.34 seconds with 96 sampled frames at 10% retention. These results suggest that CUEMINER with Top-k selection is efficient in reducing TTFT.

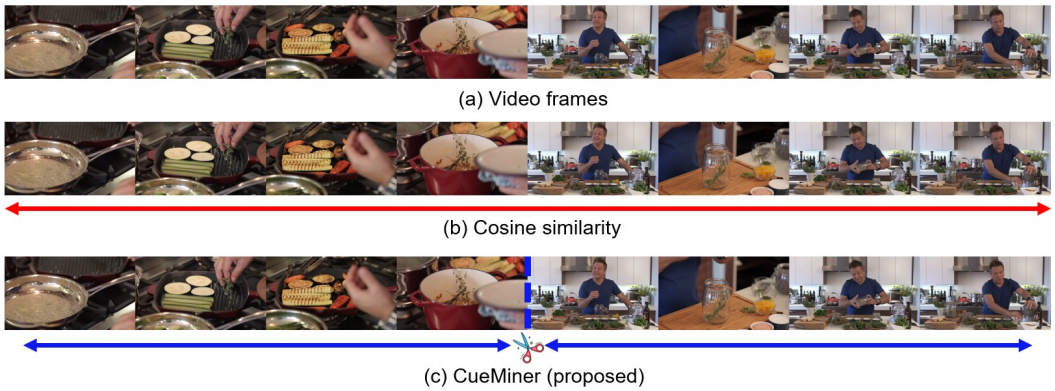

Figure 7: (a) Video frames, (b) segmentation based on cosine similarity, and (c) segmentation based on the proposed VTDS. VTDS-based segmentation separates visually heterogeneous frames into distinct segments, whereas cosine-similarity-based segmentation does not.

Table 18: Absolute accuracy (%) of a scoring-term ablation for CUEMINER on MLVU with 32 sampled frames. All CUEMINER components other than the three scoring terms, including DPP selection, are fixed across variants; the uniform-score variant assigns uniform values to all three scores, and bold denotes the best completed variant.

Quality	**LOV**			**LOV 1.5**			**LOV 2.0**			**Q3-VL**		
	20	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**
Uniform score	46.9	43.9	44.2	43.6	41.5	42.8	50.1	48.2	45.4	44.4	41.9	38.7
$S^L_{s,t,(h,w)}$	48.2	45.1	46.5	47.2	45.4	44.2	51.8	50.2	46.9	46.2	45.0	41.7
$S^L_{s,t,(h,w)}$ and $S^F_{s,t,(h,w)}$	48.7	45.6	47.2	47.1	46.1	44.8	51.9	50.8	47.8	47.1	45.6	42.2
$S^L_{s,t,(h,w)}$ and $S^B_{s,t,(h,w)}$	49.2	45.4	47.3	47.5	46.7	44.8	52.1	51.0	47.4	47.4	45.7	42.5
CUEMINER (proposed)	**50.0**	**47.3**	**47.4**	**47.6**	**47.4**	**45.6**	**53.5**	**51.5**	**48.7**	**47.6**	**45.9**	**43.2**

Taken together, these results show that, when processing a large number of sampled frames, CUEMINER with Top-k selection rather than DPP selection offers both higher accuracy and lower inference latency, as measured by TTFT. These results suggest that, when processing inputs with many sampled frames, Top-k selection is preferable to DPP selection for CUEMINER.

Table 19: Budgeting ablation on MLVU with 32 sampled frames, with all other CUEMINER components fixed across variants. We compare uniform allocation, which assigns an equal budget to every segment, frame-proportional allocation, which allocates tokens according to the number of frames in each segment, and CUEMINER, which assigns each segment a budget according to its entropy $\mathcal{H}_s$. Bold denotes the best accuracy.

Budgeting	**LOV**			**LOV 1.5**			**LOV 2.0**			**Q3-VL**		
	20	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**
Uniform	48.1	45.2	46.2	46.1	46.3	44.8	50.2	48.5	41.5	46.5	45.5	41.5
Frame-proportional	46.3	43.3	42.0	46.5	43.9	44.5	48.2	45.8	42.2	44.0	44.4	42.6
CUEMINER (proposed)	**50.0**	**47.3**	**47.4**	**47.6**	**47.4**	**45.6**	**53.5**	**51.5**	**48.7**	**47.6**	**45.9**	**43.2**

Table 20: MLVU accuracy (%) on LOV 1.5 for a single gaussian and a GMM, with DPP selection fixed across variants. Bold denotes the best.

Density estimator	Mixture components	EM iterations	20% retention	10% retention	5% retention
Single-gaussian	1	–	**47.6**	**47.4**	**45.6**
GMM	2	5	47.1	44.8	43.7
GMM	2	10	47.1	44.3	44.0
GMM	4	5	46.2	45.6	44.3
GMM	4	10	45.8	43.7	44.1

Table 21: Absolute accuracy (%) on MLVU with 32 sampled frames. CUEMINER (proposed) uses DPP selection, whereas CUEMINER w/ Top-k selects the top-B_s visual tokens within each segment $\mathcal{G}_s$ according to the computed quality-term importance score $Q_{s,t,(h,w)}$ (Equation 13), keeping the segment budgets and all other settings fixed; bold denotes the best and underline the second-best method.

Method	**LOV**			**LOV 1.5**			**LOV 2.0**			**Q3-VL**		
	20	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**	**20**	**10**	**5**
FastV (Chen et al., 2024)	43.8	40.6	38.1	41.5	38.2	34.2	44.0	39.6	36.1	45.2	41.9	36.6
VisionZip (Yang et al., 2025)	45.3	42.8	36.1	44.7	41.8	40.1	47.7	40.6	38.1	43.5	40.1	37.7
HoliTom (Shao et al., 2025)	48.8	<u>47.2</u>	46.3	42.6	39.7	39.8	46.8	39.9	37.7	45.1	40.8	36.9
FastVID (Shen et al., 2025)	44.2	43.6	42.9	44.4	40.7	36.8	46.2	43.4	40.5	43.1	41.0	39.9
UniComp (Yuan et al., 2026)	46.8	42.8	43.2	41.8	42.4	40.2	39.5	37.8	37.0	43.4	38.5	36.6
SURGE (Tang et al., 2026)	46.3	44.5	41.3	44.4	42.0	38.9	50.5	44.3	37.5	43.3	39.9	–
FlashVID (Fan et al., 2026b)	45.9	43.1	42.3	44.5	42.3	40.8	51.4	47.5	42.2	45.1	42.3	38.5
CUEMINER w/ Top-k	<u>49.4</u>	46.6	<u>46.5</u>	<u>46.5</u>	<u>45.6</u>	<u>44.3</u>	<u>51.7</u>	<u>49.2</u>	<u>45.8</u>	<u>46.7</u>	<u>44.4</u>	<u>42.1</u>
CUEMINER (proposed)	**50.0**	**47.3**	**47.4**	**47.6**	**47.4**	**45.6**	**53.5**	**51.5**	**48.7**	**47.6**	**45.9**	**43.2**

Table 22: MLVU accuracy (%) with Q3-VL at 48–96 sampled frames; CUEMINER (proposed) uses DPP selection, whereas CUEMINER w/ Top-k selects the top-B_s visual tokens within each segment $\mathcal{G}_s$ according to the computed quality-term importance score $Q_{s,t,(h,w)}$ (Equation 13); bold denotes the best and underline the second-best method.

Method	48 frames			64 frames			80 frames			96 frames		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	43.6	40.2	38.0	47.7	45.4	37.2	48.7	45.7	39.2	49.0	45.4	39.0
VisionZip (Yang et al., 2025)	44.1	38.3	38.0	46.5	43.1	40.3	49.2	43.8	39.1	50.6	41.8	41.1
HoliTom (Shao et al., 2025)	44.4	41.0	37.5	47.6	44.1	38.9	48.5	46.5	38.7	50.9	45.9	37.8
FastVID (Shen et al., 2025)	44.1	40.5	38.6	46.0	45.3	43.0	49.6	45.2	41.6	48.8	44.7	41.8
UniComp (Yuan et al., 2026)	43.3	39.5	37.0	46.8	43.0	42.3	47.1	44.0	37.6	49.2	45.2	40.2
SURGE (Tang et al., 2026)	44.8	39.4	–	45.2	42.2	–	48.3	44.9	–	49.2	44.0	–
FlashVID (Fan et al., 2026b)	44.7	<u>41.8</u>	38.2	47.6	46.9	41.5	49.0	45.8	40.0	49.6	46.6	37.7
CUEMINER w/ Top-k	<u>45.2</u>	41.1	<u>39.7</u>	<u>48.8</u>	**47.9**	**46.7**	**52.9**	**47.9**	**46.6**	**52.5**	**48.5**	**46.9**
CUEMINER (proposed)	**45.6**	**42.4**	**40.5**	**48.9**	<u>47.5</u>	<u>45.9</u>	<u>52.0</u>	<u>47.5</u>	<u>45.1</u>	<u>51.3</u>	<u>47.3</u>	<u>45.4</u>

Table 23: Time to first token (TTFT, seconds) on MLVU with Q3-VL at 48–96 sampled frames and retention ratios of 20%, 10%, and 5%. Bold denotes the lowest latency.

Method	48 frames			64 frames			80 frames			96 frames		
	20	10	5	20	10	5	20	10	5	20	10	5
FastV (Chen et al., 2024)	0.78	0.76	0.73	1.02	0.99	0.98	1.34	1.15	1.02	1.93	1.84	1.82
VisionZip (Yang et al., 2025)	0.59	0.50	0.48	0.78	0.65	0.60	1.00	0.82	0.73	1.68	1.39	1.25
HoliTom (Shao et al., 2025)	0.87	0.77	0.76	1.03	0.97	0.93	1.36	1.29	1.26	2.21	2.16	1.95
FastVID (Shen et al., 2025)	1.26	0.76	0.64	1.54	1.19	0.94	2.40	1.66	1.22	4.06	2.89	1.79
UniComp (Yuan et al., 2026)	0.62	0.53	0.52	0.85	0.71	0.69	1.12	0.94	0.81	1.76	1.52	1.36
SURGE (Tang et al., 2026)	0.58	0.48	–	0.78	0.64	–	1.01	0.82	–	1.71	1.39	–
FlashVID (Fan et al., 2026b)	1.09	0.98	0.94	1.32	1.24	1.23	1.77	1.64	1.61	2.95	2.85	2.79
CUEMINER w/ Top-k	**0.56**	**0.46**	**0.45**	**0.77**	**0.62**	**0.57**	**0.97**	**0.79**	**0.70**	**1.65**	**1.34**	**1.18**
CUEMINER (proposed)	0.76	0.56	0.53	1.01	0.74	0.65	1.33	0.96	0.79	2.51	2.04	1.54