CoSeLECT: Adaptive Frame Selection for Video-Language Understanding

Anonymous CVPR submission

Paper ID 31

Abstract

Multimodal Large Language Models (MLLMs) have shown strong performance on image understanding tasks, but video comprehension remains a significant challenge due to the high computational cost of processing long frame sequences and the limited token capacity of underlying Large Language Models (LLMs). Prior approaches to address this often rely on uniform frame sampling, query-agnostic pruning, or require costly training of dedicated compression modules. In this work, we introduce CoSeLECT, a training-free, plug-and-play, query-guided frame selection method that intelligently subsamples video frames for efficient use in MLLMs. CoSeLECT leverages two key signals: temporal redundancy, which identifies similar frame clusters, and query relevance, which selects frames based on their semantic alignment with the input query. By combining these signals through an adaptive frame selection strategy, CoSeLECT selects frames that are both diverse and highly relevant to the query, without requiring any model-specific tuning. Our results on various base MLLMs show that CoSeLECT consistently outperforms trained and training-free state-of-the-art methods, including LongVU by +3.8% on MLVU and AKS by +4.5% on EgoSchema. A reference implementation is provided in Appendix 12.

1. Introduction

Multimodal Large Language Models (MLLMs) [16, 20, 28] have shown strong performance on image comprehension and reasoning tasks [1, 34]. The standard paradigm projects images into embeddings aligned with a Large Language Model (LLM)'s token space, enabling the LLM to leverage its reasoning capabilities. While effective for images, scaling to videos is far more challenging: hundreds or thousands of frames yield prohibitively many visual tokens. For instance, LLaVA-OneVision [12], despite its 32K token context window [33], can only accommodate ∼160 frames per pass, since each frame produces around 200 tokens. Thus, video understanding performance critically depends on which frames are selected.

A large body of work addresses this token bottleneck. The default remains naive *input-blind* strategies such as uniform or fixed fps sampling [3, 12], which often overload the LLM with redundancy while missing informative moments. More sophisticated compression techniques prune or merge tokens based on spatial or temporal redundancy [4, 8, 10, 25, 26], but these remain query-agnostic. Query-aware methods exist, but typically require costly training [17, 27] or rely on intermediate attention maps [11, 29], limiting plug-and-play usability.

Frame selection has emerged as a more interpretable approach to token reduction, targeting redundancy at the level of entire frames. Existing training-free methods such as MIF and MDF [9] provide plug-and-play compatibility, but leave much of the available signal unused: MIF operates indirectly in caption space, MDF is entirely query-agnostic, and overall these approaches lag behind trained methods. KeyVideoLLM [18] introduces query guidance, but does not explicitly model temporal continuity which can be useful for video understanding. In contrast, trainable methods [35, 36, 43, 44] learn to combine signals more effectively and achieve stronger results, but at the cost of added complexity and training overhead. *Crucially, these heavier methods are typically limited to sparsely pre-sampled frame pools in order to remain computationally feasible—risking the permanent loss of "needle-in-a-haystack" moments before the selection algorithm can even evaluate them, a limitation that becomes particularly acute under resource constraints.*

From these trends, we identify three desiderata for an effective frame selection module in MLLMs: (1) adaptivity to both video content and query, (2) training-free, plug-and-play operation for easy deployment, and (3) scalability to large candidate pools without excessive overhead. *Meeting these desiderata motivates a return to first principles: rather than building increasingly complex modules, the challenge is to* ***effectively*** *fuse simple, efficiently computable signals in a way that scales.*

We propose CoSeLECT (**Co**ntinuity-aware **Se**mantic **L**ocalization and **E**xtraction of **C**andidate **T**okens), a training-free, query-guided frame selection algorithm designed around this principle. The novelty of CoSeLECT lies

not in introducing new signals, but in its lightweight, principled fusion of two readily available ones—frame–text similarity for semantic relevance and inter-frame similarity for temporal continuity. By combining them in a single-stage allocation scheme with duration-aware weighting, CoSeLECT achieves stable, interpretable selections that scale to large candidate pools. This simplicity enables both plug-and-play use and improved performance as the pool grows: from narrow queries (*e.g.*, "What color is the car that passes by home?") to broad temporal ones (*e.g.*, "What activity was the child doing in the park?"), CoSeLECT consistently identifies compact yet comprehensive frame sets, and is shown to outperform even strong trained methods such as LongVU.

We evaluate CoSeLECT on six video understanding benchmarks—VideoMME [7], MLVU [49], MVBench [15], EgoSchema [24], LongVideoBench [37], and Next-QA [38]. Across all settings, CoSeLECT outperforms or matches state-of-the-art methods, both training-free and fine-tuned, **demonstrating the power of scaled simplicity**.

Our contributions are summarized as follows:

- We introduce CoSeLECT, a training-free, query-guided frame selection algorithm with a novel fusion strategy that explicitly balances semantic relevance and temporal continuity.
- We show that training-free CoSeLECT consistently outperforms or matches state-of-the-art methods across six video understanding benchmarks, including gains of +3.8% over LongVU [27] (trained) on MLVU and +4.5% over AKS [30] (training-free SOTA) on EgoSchema.
- We demonstrate that CoSeLECT generalizes robustly across architectures (LLaVA-OneVision, Qwen2.5-VL) and model scales (0.5B/7B) without model-specific tuning, and provide comprehensive ablations validating the contribution of each design choice.

2. Related Works

2.1. From Images to Videos: The Challenge of Token Efficiency

Following the success of Multimodal Large Language Models (MLLMs) in image comprehension [6, 13, 20, 51], the field has naturally progressed towards adapting these architectures for video understanding. Initial works extended popular image-based frameworks by, for example, feeding pooled frame features into the LLM [23, 39], incorporating audio signals via a trained Q-Former [46], or pre-aligning image and video encoders [19].

However, a primary challenge in this transition is the immense computational cost associated with processing a large sequence of visual tokens. Models that uniformly sample dozens of frames [12] face a significant inference burden due to the quadratic complexity of self-attention. To mitigate this, various **token reduction** strategies have been proposed. These include methods that prune visual tokens based on low attention scores [4, 11], condense frame information into representative tokens [17], or employ specialized pooling pathways [40]. Among these strategies, a particularly effective and interpretable approach is **keyframe selection**, which focuses on identifying the most salient full frames rather than individual tokens.

2.2. Approaches to Keyframe Selection

Training-free methods offer desirable plug-and-play compatibility. Some, like MIF and MDF [9], are not directly comparable to our setting: MIF operates indirectly in caption space by ranking generated descriptions, while MDF is entirely query-agnostic, prioritizing only visual diversity. KeyVideoLLM [18] adopts a two-stage coarse-to-fine pipeline but does not explicitly model temporal continuity, and its code has not been released, preventing direct comparison. Other recent training-free approaches include SlowFast-LLaVA [40], which pools tokens through slow and fast pathways; Static-or-Dynamic [29], which explores candidate subsequences and selects based on the query; and QuoTA [22], which augments the query with chain-of-thought reasoning before scoring frames. Adaptive Keyframe Sampling (AKS) [30] remains the most comparable baseline, applying a recursive divide-and-conquer procedure over frame–text similarity scores. As AKS is the strongest existing training-free method, we contrast our algorithmic design in Sec. 10 and show consistent gains for CoSeLECT in Tab. 1.

Trainable methods introduce learnable modules for frame selection, such as Gaussian masks under weak supervision [35], cross-modal distillation with a "FramePrompter" [36], lightweight heads trained with pseudo-labels [43], or ranking-based supervision of frame combinations [44]. While effective, they add components and training overhead, limiting flexibility and plug-and-play use with off-the-shelf MLLMs.

By jointly modeling semantic relevance and temporal diversity in a single-stage adaptive procedure, CoSeLECT achieves efficiency, context-awareness, and broad compatibility without additional supervision.

3. Method

We introduce **CoSeLECT** (**Co**ntinuity-aware **Se**mantic **L**ocalization and **E**xtraction of **C**andidate **T**okens), a query-aware frame selection method for understanding long videos with multimodal large language models. A central challenge in video-based LLM reasoning lies in efficiently utilizing a limited token context window when processing temporally extended visual content. Our approach addresses this by selectively identifying frames that are both semantically relevant to the query and representative of the video's visual narrative. By balancing these two complementary signals—textual relevance and visual continuity—we aim to maxi-

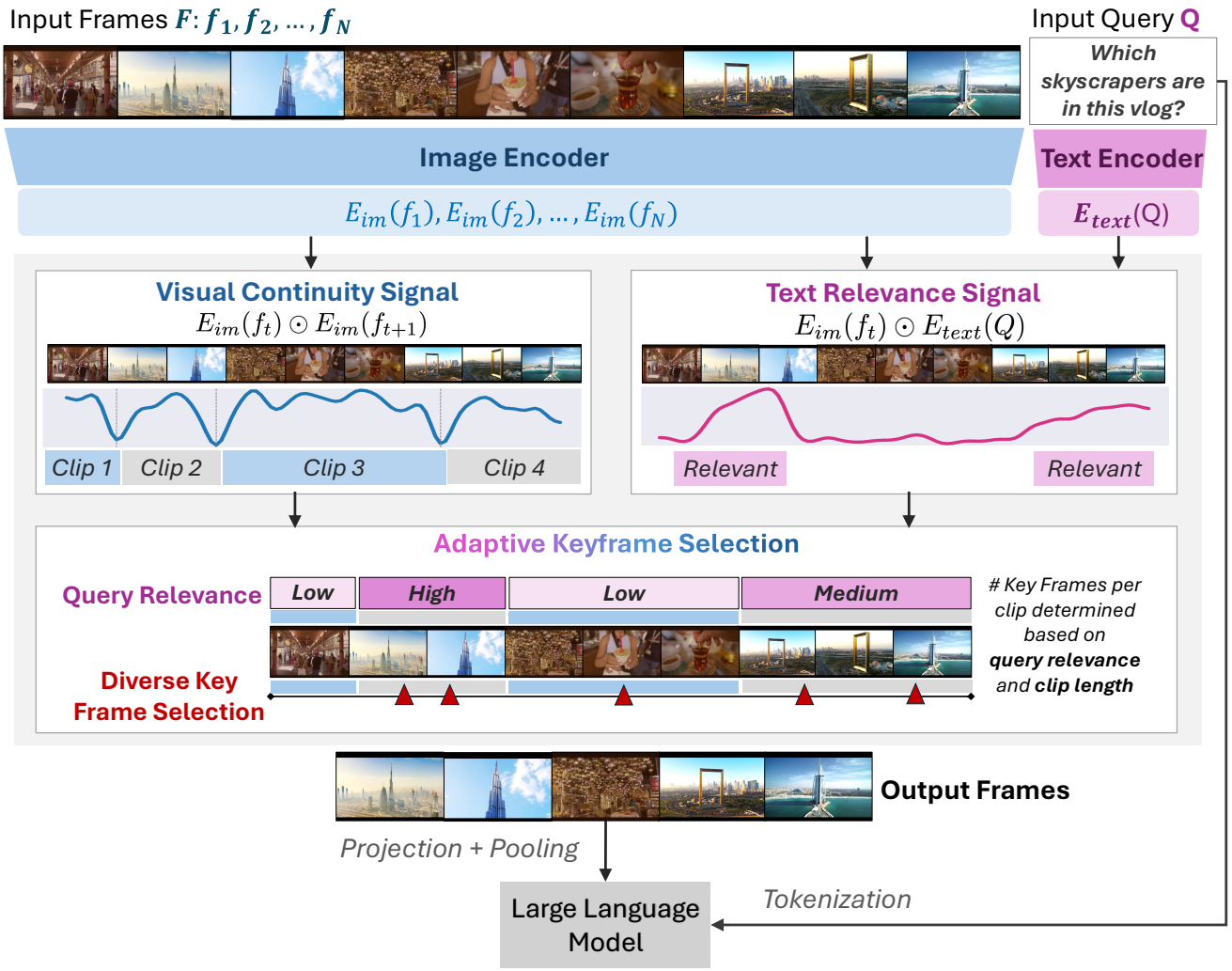


Figure 1. **CoSeLECT**: A query-aware frame selection method that identifies frames which are both **semantically relevant** to the query and **representative** of the video's visual narrative for understanding long videos with token budget constrained multimodal language models.

mize the informational value of each selected frame while remaining within token constraints. Unlike uniform sampling, which implicitly assumes that temporal regularity correlates with information value, our method dynamically adapts frame selection to the content and query, reducing redundancy and improving downstream reasoning.

3.1. Setting the Stage

Multimodal Large Language Models (MLLMs) are subject to strict token limits, which makes aggressive frame sampling essential for processing video inputs. For example, a 3-minute video sampled at 1 frame per second yields 180 frames—yet even advanced models like LLaVA-OneVision [12] can accommodate only about 160 frames within a 32K token context window [33], since each frame typically consumes ∼200 tokens.

The standard MLLM video pipeline typically proceeds as follows: (1) a video and query are provided, (2) frames are uniformly sampled (*e.g.*, at 1 fps), (3) each frame is encoded via a visual encoder, and (4) the resulting embeddings are passed—alongside the query—into the LLM. Crucially, this process is query-agnostic at the sampling stage and assumes uniform temporal coverage is sufficient. In practice, this often leads to the selection of semantically uninformative frames and results in suboptimal use of the model's limited token budget.

We propose a query-aware alternative to this sampling step that adaptively selects frames based on both their semantic relevance to the query and their contribution to the video's overall visual continuity. Our approach comprises three key components: a *Text Relevance Signal* that estimates frame-query alignment, a *Visual Continuity Signal* that de-

tects coherent temporal segments, and an *Adaptive Frame Selection* strategy that integrates these signals to effectively allocate the limited frame budget.

3.2. Text Relevance Signal

Some frames in a video are substantially more relevant to the user's query than others. A query-aware sampling method must therefore estimate semantic alignment between the query and each frame to ensure the most informative content is retained. This signal forms the core of our relevance-driven sampling strategy.

We begin by uniformly sampling the input video into N candidate frames $\{f_1, f_2, \ldots, f_N\}$, from which we aim to select K frames to send to the LLM, where $N > K$. For clarity, in the tables we denote N as pre-$\mathbf{E}_{\text{im}}$ and K as post-$\mathbf{E}_{\text{im}}$, corresponding to the number of image embeddings before and after selection, respectively. Each frame is embedded using SigLIP-So400M-patch14-384 [45], a vision-language model that produces aligned image-text representations. Since the finetuned SigLIP is also used as the vision encoder in LLaVA-OneVision [12], these embeddings can be directly reused, avoiding redundant computation. (Worth noting that this doesn't hold true if you swap out vision encoders.) The image and text encoders, $E_{im}(\cdot)$ and $E_{text}(\cdot)$ respectively, are used to compute a relevance score for each frame.

For a given query Q, we compute cosine similarity between each frame and query embeddings:

$$\mathrm{S}_{\text{text}}(t) = E_{im}(f_t) \odot E_{text}(Q) \tag{1}$$

where $\odot$ denotes cosine similarity between normalized embedding vectors. High values of $\mathrm{S}_{\text{text}}(t)$ indicate that the frame is semantically aligned with the query. To reduce noise, we apply Gaussian smoothing across the temporal dimension, ensuring more stable relevance scores and reducing sensitivity to abrupt local fluctuations.

3.3. Visual Continuity Signal

Focusing solely on query relevance results in redundancy or neglecting important temporally contextual segments. To encourage broad and meaningful coverage of the video, we introduce a second signal that identifies scene transitions and partitions the video into visually coherent segments.

This signal is based on the intuition that sudden visual changes often correspond to new scenes or shifts in content. We compute cosine similarity between consecutive frame embeddings:

$$\mathrm{S}_{\text{frame}}(t) = E_{im}(f_t) \odot E_{im}(f_{t+1}) \tag{2}$$

After applying Gaussian smoothing to incorporate local context, we detect visual discontinuities by identifying significant drops in similarity:

$$\text{SceneBoundaries} = \{k \mid \mathrm{S}_{\text{frame}}(k) < \tau_{\text{sim}}\} \tag{3}$$

with a threshold $\tau_{\text{sim}} = 0.8$, chosen via empirical validation Tab. 8. These boundaries segment the video into subclips $C = \{C_1, \ldots, C_M\}$, each representing a coherent visual scene. This helps ensure our frame selection covers the full narrative arc of the video rather than focusing narrowly on isolated high-relevance moments. As our ablations Sec. 4.3 (see *Frame Selection Strategy*) show, this continuity prior plays a crucial role in maintaining context necessary for strong video understanding.

3.4. Adaptive KeyFrame Selection

While the text relevance and visual continuity signals offer powerful tools individually, leveraging them in isolation risks suboptimal results. Focusing only on relevance may oversample redundant moments (*e.g.*, repeating frames in a key scene), while emphasizing visual coverage alone may dilute attention across low-information segments. The challenge, then, is to **adaptively allocate limited frame tokens to scenes that matter most—those that are both content-rich and semantically aligned with the query—while ensuring diverse temporal coverage**.

We achieve this by designing a selection strategy that fuses these two signals and distributes the frame budget in proportion to the estimated importance of each visual subclip.

For each identified subclip C_i, we compute a composite relevance score that captures both standout moments and overall alignment with the query:

$$R_i = \max(\mathrm{S}_{\text{text}}|_{C_i}) + \text{mean}(\mathrm{S}_{\text{text}}|_{C_i}) \tag{4}$$

The maximum highlights peak relevance within the subclip, while the mean ensures we value consistent alignment across its duration. This helps us select subclips that are either uniformly relevant or contain particularly important frames. We empirically verify the importance of this design choice in Tab. 7.

We then weight this relevance score by the square root of the subclip's duration D_i:

$$W_i = R_i \cdot \sqrt{D_i} \tag{5}$$

This duration-adjusted weighting ensures longer relevant segments receive more frames than shorter ones, but the relationship is sublinear (square root). Without this adjustment, extremely long subclips would dominate the frame budget regardless of content relevance, while using the square root prevents this overrepresentation while still acknowledging that longer segments typically contain more diverse content in comparison to shorter ones. We empirically verify the importance of this design choice in Tab. 7.

The frame budget of K total frames is distributed across subclips proportionally to these weights:

$$k_i = \lfloor K \cdot \frac{W_i}{\sum_{j=1}^{M} W_j} \rfloor \tag{6}$$

where k_i is the number of frames allocated to subclip C_i.

Finally, for each subclip allocated multiple frames, we partition its temporal span into k_i equal segments and select the frame with highest S_{text} from each segment. This strategic partitioning prevents temporal clustering of selected frames while still maintaining focus on the most semantically relevant content within each temporal region. Without this partitioning, frame selection would likely cluster around peaks of relevance, potentially missing important context elsewhere in the subclip.

In summary, CoSeLECT provides a principled approach to query-aware frame selection by jointly modeling semantic relevance and visual continuity. This dual-signal strategy allows us to adaptively allocate limited frame tokens to the most informative and contextually important parts of a video. **We formally present our approach in Algorithm 1.**

4. Experiments, Results and Discussion

4.1. Experimental Framework

Implementation We use the image and text encoder from the finetuned version of SigLIP-ViT-SO400M-patch14-384 [45] associated with the LLaVA-OneVision [12] to encode both video frames and the text query. Input images are resized to 384×384 and tokenized using a patch size of 14×14 before encoding. For the language foundation model, we report results using LLaVA-OneVision [12] with Qwen2-7B and Qwen2-0.5B as the decoder LLM. We also report results with CoSeLECT applied to Qwen2.5-VL-7B [2] to demonstrate that our approach's effectiveness is agnostic to the choice of the base MLLM. We run all our evaluations on single 40G A40 nodes with 8 GPUs. We evaluate our method using varying sizes of the pre-embedding frame pool—specifically, 32, 64, 400, and 1600 frames—which we refer to as pre-$\mathbf{E}_{im}$ frames. Regardless of the pre-$\mathbf{E}_{im}$ pool size, we adaptively select 32 frames (unless specified otherwise), which we refer to as post-$\mathbf{E}_{im}$ frames before feeding image tokens into the LLM. This ensures that the sequence length of the tokens provided to the language model remains below 8k, allowing for consistent comparison across different frame selection strategies.

Benchmarks We evaluate our models on well known video evaluation benchmarks: VideoMME [7], MLVU [50]'s `dev` split, MVBench [14], EgoSchema [24]'s `subset` split, LongVideoBench [37]'s `val` set, NextQA [38]'s `test` set.

4.2. Key Results

Quantitative Comparison against Training-Free Methods Table 1 compares our training-free method with the LLaVA-OV baseline and other recent approaches under identical backbones. Since AKS is the current state of the art for training free adaptive frame selection, we not only reproduced their reported results by integrating their released code into our setup, but also extended their method to larger pre-E_{im} budgets for a truly fair comparison. **Our method establishes a new state of the art in training-free frame selection, outperforming all competitors on 3 of 4 benchmarks and ranking second only on VideoMME**. The gains are especially pronounced on long-horizon benchmarks: 64.8 on EgoSchema vs. the next best 62.8 (+2.0, a 3.2% relative gain), and 59.3 on Long-VideoBench vs. 57.9 (+1.4, a 2.4% relative gain).

Quantitative Comparison against Trained Methods Table 2 compares our training-free method, CoSeLECT, against a range of trained frame-selection models. To ensure fairness, we align vision encoder, LLM, and pre-E_{im} sizes, and report the best result in each cluster. Despite requiring no training, CoSeLECT outperforms most trained methods. Notably, many trained approaches only report results on small pre-E_{im} budgets, as training becomes prohibitively expensive on larger frame sets. In contrast, our method remains scalable and competitive. Furthermore, models such as LongVU rely on computationally heavy spatiotemporal token compression, yet our simple similarity-based frame selection surpasses it (67.9 vs. 65.4 on MLVU). The only method ahead of us is ViLA, which leverages a much larger ViT-G vision encoder. *These results underscore that a lightweight, training-free strategy can rival—and in many cases surpass—specialized trained models.*

Qualitative Results We qualitatively evaluate our model across diverse tasks and compare it to the baseline in Sec. 11 (Figures 4–7). Figure 8 illustrates how CoSeLECT adapts frame selection to the text query by showing differences in sampled frames for the same video under different questions.

4.3. Ablations

Efficient Video Understanding Table 3 evaluates performance in a token-constrained scenario—passing only a small number of tokens to the downstream LLM—by benchmarking our method against state-of-the-art, training-free token pruning techniques. While prior methods perform pruning directly over tokens, our approach reduces tokens indirectly by selecting a subset of informative frames. Thus, it can be viewed as a frame-level token reduction strategy. Specifically, a 25% pruning ratio corresponds to selecting 8 of the original 32 frame tokens, and 12.5% pruning uses just 4. We begin with 400 frames per video and adaptively select the top frames to match these targets. Our method consistently matches or exceeds state-of-the-art results under extreme compression. At 25%, we achieve the highest VideoMME score, tying with VisionZip and FastVID. At 12.5%, we outperform all competitors on MVBench and MLVU—even

Algorithm 1 CoSeLECT- Query-Aware Frame Selection

Require: Video $V = \{f_1, f_2, ..., f_N\}$, query Q, image encoder E_{im}, text encoder E_{text}, output frames K

// —- Embedding Extraction —-

Extract embeddings $E_{im}(f_i)$ for all frames and $E_{text}(Q)$ for query

// —- Text Relevance Signal —-

$S_{\text{text}}(t) = \text{smooth}(E_{im}(f_t) \odot E_{text}(Q))$ # Compute query-frame relevance

// —- Visual Continuity Signal —-

$S_{\text{frame}}(t) = \text{smooth}(E_{im}(f_t) \odot E_{im}(f_{t+1}))$ # Compute frame-to-frame similarity

SceneBoundaries $= \{k \mid S_{\text{frame}}(k) < \tau_{\text{sim}}\}$ # $\tau_{\text{sim}} = 0.8$

Partition video into subclips $C = \{C_1, ..., C_M\}$ at scene boundaries

Record duration D_i for each subclip C_i

// —- Compute Composite Relevance —-

for each C_i in C **do**

  $R_i = \max(S_{\text{text}}|_{C_i}) + \text{mean}(S_{\text{text}}|_{C_i})$

end for

// —- Adaptive Frame Selection —-

for each C_i in C **do**

  $W_i = R_i \cdot \sqrt{D_i}$ # Duration-adjusted relevance weight

end for

$k_i = \lfloor K \cdot \frac{W_i}{\sum_{j=1}^{M} W_j} \rfloor$ # Distribute frame budget

for each C_i with allocation $k_i > 0$ **do**

  Partition C_i into k_i equal temporal segments

  Select frame with highest S_{text} from each segment

end for

return K Selected Frames

Table 1. Comparison of **training-free** token reduction/frame selection methods built on the LLaVA-OneVision model (LLaVA-OV) across benchmarks. CoSeLECT outperforms all methods in 3/4 tasks. *(**Bold** denotes the best result per task, underline denotes the second-best result, and † indicates reproduced results.)*

Method	Context Length	Frames pre-E_{im}	VideoMME	MVBench	Long-VideoBench	Ego-Schema
LLaVA-OV (Baseline)	8k	N/A	58.4	57.8	56.8	62.8
SlowFast-LLaVA [41]	3.6k	32	56.1	56.4	55.1	61.4
LLaVA-OV + Static or Dynamic [29]	8k	128	59.9	–	–	60.5
LLaVA-OV + QuoTA [22]	8k	64	60.7	–	57.4	–
LLaVA-OV + BOLT [21]	8k	fps	59.9	–	59.6	64.0
LLaVA-OV + AKS† [31]	8k	64	58.2	57.4	56.4	62.6
LLaVA-OV + AKS† [31]	8k	400	60.2	56.6	57.4	61.4
LLaVA-OV + AKS† [31]	8k	1600	**61.8**	58.1	57.9	62.0
LLaVA-OV + CoSeLECT (ours)	8k	32	57.7	57.8	57.1	63.4
LLaVA-OV + CoSeLECT (ours)	8k	64	58.6	57.6	57.7	**64.8**
LLaVA-OV + CoSeLECT (ours)	8k	400	59.7	58.1	**59.3**	64.0
LLaVA-OV + CoSeLECT (ours)	8k	1600	61.1	**58.2**	58.8	63.0

with fewer input frames. These results suggest that *high-quality token reduction can be achieved by frame selection alone, without operating at token granularity, when leveraging a rich initial candidate pool.*

Robustness across Architectural Changes We demonstrate CoSeLECT's robustness across model architectures and scales (Table 4). In our baseline setting, when applied to LLaVA-OV with Qwen2-7B, CoSeLECT improves the average score from 59.6 to 61.6 (3.4% relative gain). These gains persist when scaling down the LLM size: with the smaller Qwen2-0.5B backbone, CoSeLECT boosts performance from 42.0 to 43.2 (2.9% relative gain). The effectiveness of our approach also extends beyond a single MLLM family. On Qwen2.5-VL-7B-Inst, CoSeLECT improves average performance from 61.6 to 63.0 (2.3% relative gain). Finally, we observe consistent gains even on proprietary closed-source models: applied to GPT-4o-mini, CoSeLECT

Table 2. Benchmarking performance of **trained** token reduction/frame selection methods against CoSeLECT, compared with training-based approaches. All methods use the same frame, LLM and vision encoder configurations within each block of the table (separated by midrules). *(Since the backbones aren't consistent, best performance is denoted per block. **Bold** denotes the best result within each block)*

Method	Training Free	LLM Size	Vision Encoder Size	Context Length	Frames pre-E_{im}	NextQA	MLVU
SeViLA [43]	✗	3B	1.1B	2K	32	73.8	–
ViLA [36]	✗	3B	1.4B	2K	32	**74.8**	–
CoSeLECT + Qwen2.5-VL-3B-Instruct [33]	✓	3B	0.4B	2K	32	74.6	–
GCG [35]	✗	7B	0.4B	1K	32	74.6	–
CoSeLECT + LLaVA-OV	✓	7B	0.4B	1K	32	**76.9**	–
Frame-Voyager [44]	✗	7B	0.4B	2K	32	73.9	65.6
CoSeLECT + LLaVA-OV	✓	7B	0.4B	2K	32	**78.0**	**66.1**
LongVA [47]	✗	7B	0.4B	224K	32	69.3	56.3
VideoChat2 [14]	✗	7B	0.3B	8K	16	–	47.9
CoSeLECT + LLaVA-OV	✓	7B	0.4B	8K	32	**80.0**	**63.5**
LongVU [27]	✗	7B	0.4B	8K	1fps	–	65.4
CoSeLECT + LLaVA-OV	✓	7B	0.4B	8K	1600	**80.1**	**67.9**

Table 3. Performance against token reduction techniques. Retained ratio indicates the percentage of original tokens remaining after token reduction. *(**Bold** denotes the best performance within each block.)*

Method	Retained Ratio (%)	VideoMME	MVBench	MLVU	LongVideoBench	Average
LLaVA-OV	100%	58.4	57.8	62.4	56.8	58.9
LLaVA-OV + Uniform frame sampling	25%	52.9	57.6	57.7	54.8	55.8
LLaVA-OV + DyCoke [32]	25%	51.0	49.5	55.8	48.1	51.1
LLaVA-OV + FastV [5]	25%	56.2	54.7	61.5	55.5	57.0
LLaVA-OV + VisionZip [42]	25%	58.0	56.6	64.8	51.2	57.7
LLaVA-OV + PruneVID [22]	25%	57.5	55.0	64.6	55.4	58.1
LLaVA-OV + FastVID [26]	25%	58.0	56.5	64.1	56.3	58.7
LLaVA-OV + CoSeLECT (Ours)	25%	**58.0**	**57.9**	**66.1**	**57.4**	**59.9**
LLaVA-OV + Uniform frame sampling	12.5%*	50.1	55.8	55.6	50.9	53.1
LLaVA-OV + FastV [5]	15%	54.7	53.2	59.8	54.9	55.7
LLaVA-OV + VisionZip [42]	15%	55.5	54.3	54.4	53.9	54.5
LLaVA-OV + PruneVID [22]	15%	56.1	54.6	63.1	53.6	56.9
LLaVA-OV + FastVID [26]	15%	**57.7**	56.0	63.3	**56.2**	**58.3**
LLaVA-OV + CoSeLECT (Ours)	12.5%*	56.6	**56.5**	**64.8**	54.9	58.2

increases performance from 64.9 to 66.1 (1.8% relative gain). *Together, these results highlight the robustness and generality of CoSeLECT across diverse architectures and scales.*

Complementary Signals for Frame Selection Our adaptive strategy selects frames that are *both semantically aligned with the query and temporally diverse*. As shown in Table 5, CoSeLECT outperforms query-relevance-only selection across all benchmarks, with an average gain of 0.9%. Notably, the visual continuity signal alone already improves over uniform sampling (59.6 → 60.5) and performs on par with text similarity (60.7 vs. 60.5). Together, they yield the best overall score of 61.4. *When combined, the two signals complement each other: semantic relevance guides selection toward informative frames, while continuity ensures diverse coverage and provides a reliable fallback that consistently outperforms uniform sampling.*

Impact of encoder choice As shown in Table 6, lightweight encoders such as CLIP-B achieve competitive performance across benchmarks, with only minor degradation relative to larger models like SIGLIP or DINO-ViT-B/16. We believe this serves as a practical datapoint for users selecting configurations under compute, time, or task constraints.

Dissecting CoSeLECT further Beyond the core ablations, Sec. 8 presents a more exhaustive exploration of CoSeLECT's mechanics and hyperparameters. We analyze the composite relevance and duration weighting schemes, the effect of initial frame pool size, and sensitivity to the visual similarity threshold for scene segmentation. These results provide deeper justification for our design choices and further highlight CoSeLECT's advantages over alternatives.

Table 4. Exploration of performanceCoSeLECT across a variety of different LLM backbone varieties and sizes on a slew of tasks. CoSeLECT demonstrates consistent performance improvement across all backbones. pre-E_{im} set to 800 and post-E_{im} set to 32.

Method	LLM	Context Length	VideoMME	MVBench	MLVU	LongVideo Bench	EgoSchema	Average
GPT-4o-mini	GPT-4o-mini	8k	–	–	64.9	–	–	64.9
+ CoSeLECT	GPT-4o-mini	8k	–	–	66.1	–	–	**66.1**
LLaVA-OV	Qwen2-7B	8k	58.4	57.8	62.4	56.8	62.8	59.6
+ CoSeLECT	Qwen2-7B	8k	59.7	58.1	67.3	59.3	64.0	**61.6**
LLaVA-OV	Qwen2-0.5B	8k	43.7	46.3	46.5	46.8	26.6	42.0
+ CoSeLECT	Qwen2-0.5B	8k	45.6	46.5	49.5	47.8	26.6	**43.2**
Qwen2.5-VL-7B-Inst	Qwen2.5-7B	8k	61.3	68.3	59.5	58.9	59.8	61.6
+ CoSeLECT	Qwen2.5-7B	8k	63.3	69.5	63.4	59.2	59.6	**63.0**

Table 5. Exploring optimal frame selection strategy. Demonstrating why our method incorporates **both** text similarity and visual continuity signal—not just one or the other.

Frame selection method	Frames pre-E_{im}	Frames post-E_{im}	VideoMME	MVBench	MLVU	Long-VideoBench	EgoSchema	Average
Uniform sampling	32	32	58.4	57.8	62.4	56.8	62.8	59.6
32 top text similarity frames	400	32	60.7	57.6	65.7	57.7	61.8	60.7
Only Visual Continuity	400	32	59.1	57.9	64.9	57.6	63.0	60.5
CoSeLECT (Ours)	400	32	59.7	58.1	65.9	59.3	64.0	**61.4**

Table 6. Performance comparison across encoder configurations in the similarity computation phase. The backbone is LLaVA-OneVision-7B with pre-E_{im} set to 800 and post-E_{im} set to 32. The first two columns specify the encoders used for Image–Image similarity (Visual Continuity Signal) and Image–Text similarity (Text Relevance Signal). *(**Bold** denotes the best average result.)*

Image–Image Sim	Image–Text Sim	Num Params	ViMME	MVBench	MLVU	LongVideo Bench	EgoSchema	Average
CLIP-B	CLIP-B	151M	59.9	58.3	66.3	58.5	63.6	61.3
DINO-ViT-B/16	SIGLIP	86M + 878M	60.8	58.3	67.3	59.3	63.6	**61.9**
SIGLIP	SIGLIP	878M	60.6	58.3	66.2	59.3	**64.5**	61.8

5. Limitations

CoSeLECT inherits the biases and failure modes of the underlying vision–language encoders (*e.g.*, SigLIP, Qwen2.x). In addition, our method currently operates on RGB frames only and does not exploit audio or motion-specific features, which may be important for certain tasks.

6. Conclusion

In this work, we introduce CoSeLECT, a training-free adaptive frame selection approach addressing the fundamental challenge of efficiently processing long videos within the token constraints of current MLLMs. By effectively balancing temporal redundancy and semantic relevance, CoSeLECT dynamically adapts frame selection to both video content and the input query, overcoming critical limitations of naive sampling and costly fine-tuning methods. Extensive experiments across diverse benchmarks demonstrate that CoSeLECT consistently matches or outperforms state-of-the-art approaches—including fine-tuned and dynamic token pruning baselines—while maintaining strong generalization across multiple MLLM architectures and scales. Given its simplicity and effectiveness, we hope that our approach will be adopted, extended, and leveraged across a broad spectrum of future multimodal models and applications.

7. LLM Usage

We used large language models (LLMs) such as ChatGPT to assist with proofreading and polishing the presentation of this paper. Their role was limited to improving readability, grammar, and flow; all technical content, experimental design, and scientific claims were developed, implemented, and validated by the authors. The use of LLMs was therefore restricted to editorial support, without influence on the novelty or substance of the research.

References

[1] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 2425–2433, 2015. 1

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

[3] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. 1

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

[5] 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, 2024. 7

[6] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023. 2

[7] Chao Fu, Peng Chen, Yujie Shen, Yujie Qin, Meng Zhang, Xiang Lin, Zhi Qiu, Weiyao Lin, Jian Yang, and Xinyu Zheng. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*, 2024. 2, 5

[8] Tianyu Fu, Tengxuan Liu, Qinghao Han, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang. Framefusion: Combining similarity and importance for video token reduction on large visual language models, 2024. 1

[9] Wei Han, Hui Chen, Min-Yen Kan, and Soujanya Poria. Self-adaptive sampling for efficient video question-answering on image–text models, 2024. 1, 2

[10] Yefei He, Feng Chen, Jing Liu, Wenqi Shao, Hong Zhou, Kaipeng Zhang, and Bohan Zhuang. Zipvl: Efficient large vision-language models with dynamic token sparsification, 2024. 1

[11] Xiaohu Huang, Hao Zhou, and Kai Han. Prunevid: Visual token pruning for efficient video large language models. In *Arxiv*, 2024. 1, 2

[12] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024. 1, 2, 3, 4, 5

[13] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C.H. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv*, 2023. 2

[14] 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, 2024. 5, 7

[15] Kunchang Li, Yali Wang, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *CVPR*, 2024. 2

[16] Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. Mini-gemini: Mining the potential of multi-modality vision language models. *arXiv:2403.18814*, 2023. 1

[17] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *ECCV*, 2024. 1, 2

[18] Hao Liang, Jiapeng Li, Tianyi Bai, Xijie Huang, Linzhuang Sun, Zhengren Wang, Conghui He, Bin Cui, Chong Chen, and Wentao Zhang. Keyvideollm: Towards large-scale video keyframe selection, 2024. 1, 2

[19] Bin Lin, Bin Zhu, Yang Ye, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *EMNLP*, 2024. 2

[20] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 1, 2

[21] Shuming Liu, Chen Zhao, Tianqi Xu, and Bernard Ghanem. Bolt: Boost large vision-language model without training for long-form video understanding, 2025. 6

[22] Yongdong Luo, Wang Chen, Xiawu Zheng, Weizhong Huang, Shukang Yin, Haojia Lin, Chaoyou Fu, Jinfa Huang, Jiayi Ji, Jiebo Luo, and Rongrong Ji. Quota: Query-oriented token assignment via cot query decouple for long video comprehension, 2025. 2, 6, 7

[23] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *ACL*, 2024. 2

[24] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding, 2023. 2, 5

[25] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *arXiv preprint arXiv:2403.15388*, 2024. 1

[26] 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, 2025. 1, 7

[27] Xiaoqian Shen, Yunyang Xiong, Changsheng Zhao, Lemeng Wu, Jun Chen, Chenchen Zhu, Zechun Liu, Fanyi Xiao, Balakrishnan Varadarajan, Florian Bordes, Zhuang Liu, Hu Xu, Hyunwoo J. Kim, Bilge Soran, Raghuraman Krishnamoorthi, Mohamed Elhoseiny, and Vikas Chandra. Longvu: Spatiotemporal adaptive compression for long video-language understanding, 2024. 1, 2, 7

[28] Min Shi, Fuxiao Liu, Shihao Wang, Shijia Liao, Subhashree Radhakrishnan, De-An Huang, Hongxu Yin, Karan Sapra,

Yaser Yacoob, Humphrey Shi, Bryan Catanzaro, Andrew Tao, Jan Kautz, Zhiding Yu, and Guilin Liu. Eagle: Exploring the design space for multimodal llms with mixture of encoders. In *ICLR*, 2025. 1

[29] Yumeng Shi, Quanyu Long, and Wenya Wang. Static or dynamic: Towards query-adaptive token selection for video question answering, 2025. 1, 2, 6

[30] Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive keyframe sampling for long video understanding. *arXiv preprint arXiv:2502.21271*, 2025. 2, 3

[31] Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive keyframe sampling for long video understanding, 2025. 6

[32] Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. Dycoke: Dynamic compression of tokens for fast video large language models, 2025. 7

[33] Qwen2 Team. Qwen2 technical report. *arXiv*, 2024. 1, 3, 7

[34] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2015. 1

[35] Haibo Wang, Chenghang Lai, Yixuan Sun, and Weifeng Ge. Weakly supervised gaussian contrastive grounding with large multimodal models for video question answering, 2024. 1, 2, 7

[36] Xijun Wang, Junbang Liang, Chun-Kai Wang, Kenan Deng, Yu Lou, Ming Lin, and Shan Yang. Vila: Efficient video-language alignment for video question answering, 2024. 1, 2, 7

[37] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *arXiv*, 2024. 2, 5

[38] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *CVPR*, 2021. 2, 5

[39] Lin Xu, Yilin Zhao, Daquan Zhou, Zhijie Lin, See Kiong Ng, and Jiashi Feng. Pllava : Parameter-free llava extension from images to videos for video dense captioning. *arXiv*, 2024. 2

[40] Mingze Xu, Mingfei Gao, Zhe Gan, Hong-You Chen, Zhengfeng Lai, Haiming Gang, Kai Kang, and Afshin Dehghan. Slowfast-llava: A strong training-free baseline for video large language models. *arXiv:2407.15841*, 2024. 2

[41] Mingze Xu, Mingfei Gao, Zhe Gan, Hong-You Chen, Zhengfeng Lai, Haiming Gang, Kai Kang, and Afshin Dehghan. Slowfast-llava: A strong training-free baseline for video large language models, 2024. 6

[42] 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, 2024. 7

[43] Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. Self-chained image-language model for video localization and question answering, 2023. 1, 2, 7

[44] Sicheng Yu, Chengkai Jin, Huanyu Wang, Zhenghao Chen, Sheng Jin, Zhongrong Zuo, Xiaolei Xu, Zhenbang Sun, Bingni Zhang, Jiawei Wu, Hao Zhang, and Qianru Sun. Frame-voyager: Learning to query frames for video large language models, 2025. 1, 2, 7

[45] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training, 2023. 4, 5, 1

[46] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv*, 2023. 2

[47] Peiyuan Zhang, Kaichen Zhang, Bo Li, Guangtao Zeng, Jingkang Yang, Yuanhan Zhang, Ziyue Wang, Haoran Tan, Chunyuan Li, and Ziwei Liu. Long context transfer from language to vision, 2024. 7

[48] Shaojie Zhang, Jiahui Yang, Jianqin Yin, Zhenbo Luo, and Jian Luan. Q-frame: Query-aware frame selection and multi-resolution adaptation for video-llms, 2025. 3

[49] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Shitao Xiao, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. Mlvu: Benchmarking multi-task long video understanding. *arXiv*, 2024. 2

[50] 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, 2025. 5

[51] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv*, 2023. 2

[52] Orr Zohar, Xiaohan Wang, Yann Dubois, Nikhil Mehta, Tong Xiao, Philippe Hansen-Estruch, Licheng Yu, Xiaofang Wang, Felix Juefei-Xu, Ning Zhang, Serena Yeung-Levy, and Xide Xia. Apollo: An exploration of video understanding in large multimodal models, 2024. 1

CVPR #31

CoSeLECT: Adaptive Frame Selection for Video-Language Understanding

Supplementary Material

8. Appendix: Dissecting the Method - Ablations and Insights

8.1. Effect of Composite Relevance and Duration Weighting

Table 7 highlights the contribution of each component in our design. We find that replacing the duration adjustment term $\sqrt{D_i}$ with a linear factor D_i, or simplifying the composite relevance to rely solely on either the maximum or the average similarity score in $R_i = \max(S_{\text{text}}|_{C_i}) + \text{mean}(S_{\text{text}}|_{C_i})$, consistently lowers performance. In contrast, combining both max and average signals with square-root duration weighting yields the strongest results across benchmarks.

8.2. Choosing Similarity Threshold

From our test set we obtain the best performance at $\tau_{\text{sim}} = 0.8$. This parameter can further be tuned for any new tasks as well.

8.3. Effect of varying the pre-E_{im}

pre-E_{im} refers to the number of encoded frames that the selection algorithm runs on to make its selection. Table 9 highlights how the size of the pre-embedding frame pool impacts performance. Increasing the candidate pool from 32 frames (uniform sampling) to 400 frames (our method) improves average performance from 58.9 to 61.1, with gains particularly notable on tasks like VideoMME (from 58.4 to 59.7, +1.3) and Long-VideoBench (from 56.8 to 59.3, +2.5). However, this trend is not uniform across benchmarks—MVBench shows only marginal improvements (+0.3), suggesting that the optimal pool size is task-dependent. We also observe diminishing returns: for instance, switching from uniform sampling (32 frames) to top text-similarity frames (400 frames) yields a +2.2 average gain, while further increases yield smaller improvements. These findings suggest that tuning the number of candidate frames using a validation set is beneficial. Furthermore, this parameter offers a flexible trade-off between performance and compute, allowing our method to adapt to different resource budgets—a practical strength in real-world settings.

8.4. Effect of varying the post-E_{im}

It may seem intuitive that passing more frames into the downstream LLM should improve performance. However, as shown in Table 10, increasing the number of post-E_{im} frames beyond 32 does not consistently lead to gains. We attribute this to two factors: (1) **Redundancy** — longer frame sequences often contain many visually similar or irrelevant frames, reducing the signal-to-noise ratio. (2) **Model limitations** — current MLLMs have bounded attention capacity, so adding more frames can dilute the contribution of the most informative ones.

Notably, CoSeLECT maintains relatively higher performance across all frame settings, suggesting that adaptive frame selection mitigates both redundancy and attention bottlenecks.

9. Appendix: Runtime Efficiency of CoSeLECT

In our algorithm, we encode N (where $32 \leq N \leq 1600$) frames per data sample using the So400M-patch14-384 SigLIP Encoder [45]. Dense frame sampling is becoming increasingly common in video-centric multimodal LLMs—*e.g.*, Apollo [52] and LongVU [27]—which both advocate for and benefit from fps sampling. *Our approach thus aligns with this emerging trend, rather than representing an atypical design choice.*

Moreover, the independent nature of each forward pass renders the process embarrassingly parallel and well-suited for batching or distributed inference. As demonstrated in Figure 2, FLOPs scale linearly with N as expected, but practical latency remains significantly lower than what the FLOP count would suggest. For example, on an 8-GPU A40 node with 40 GB memory, encoding 1600 frames requires $32\times$ the FLOPs of encoding 50 frames, yet incurs only a $4.1\times$ latency increase. *This gap highlights that, under parallel execution conditions, the wall-time cost of our algorithm is markedly lower than its theoretical complexity implies.*

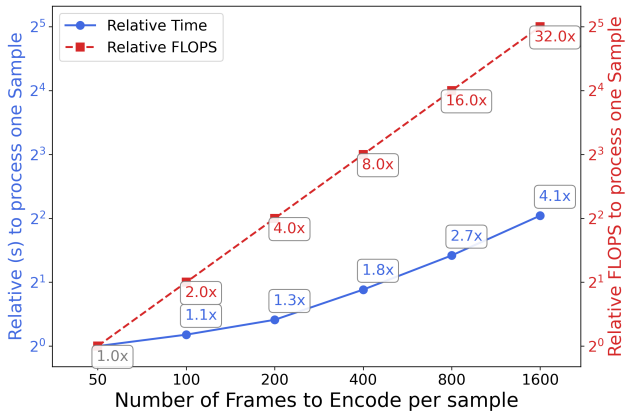


Figure 2. Relative FLOPs and wall-clock latency as we increase frame counts

Figure 3 visualizes how increasing the size of the pre-embedding frame pool impacts wall-time when running our

Table 7. Ablation of key design choices in CoSeLECT, corresponding to Algorithm 1. We test variants that remove or simplify components of the composite relevance and weighting scheme.

Variant	VideoMME	MVBench	MLVU	Long-VideoBench	EgoSchema	Average
D_i instead of $\sqrt{D_i}$	60.1	58.1	66.2	58.6	63.6	61.3
$R_i = \max(S_{\text{text}}\|_{C_i})$ only	60.1	57.7	66.0	58.0	63.6	61.1
$R_i = \text{mean}(S_{\text{text}}\|_{C_i})$ only	60.5	58.1	66.0	58.8	63.8	61.4
CoSeLECT	**60.6**	**58.3**	**66.2**	**59.3**	**64.5**	**61.8**

Table 8. Impact of frame-to-frame similarity threshold τ_{sim} on CoSeLECT performance across benchmarks. This threshold corresponds to the parameter in Algorithm 1, where scene boundaries are defined as $\{k \mid S_{\text{frame}}(k) < \tau_{\text{sim}}\}$ (e.g., $\tau_{\text{sim}} = 0.8$).

Threshold	VideoMME	MVBench	MLVU	Long-VideoBench	EgoSchema	Average
0.2	60.0	58.2	67.1	58.4	64.2	61.6
0.4	60.0	58.0	66.8	57.3	63.8	61.2
0.6	60.0	57.9	66.8	58.0	63.8	61.3
0.8	60.1	58.1	67.3	59.3	64.0	**61.8**

Table 9. Effect of varying the pre-E_{im} frame pool size on CoSeLECT performance across benchmarks. Here, pre-E_{im} denotes the number of candidate frames encoded before applying CoSeLECT, while post-E_{im} indicates the final number of selected frames passed to the LLM.

Method	Frames pre-E_{im}	Frames post-E_{im}	VideoMME	MVBench	MLVU	LongVideoBench	Average
LLaVA-OV + Uniform Sampling	32	32	58.4	57.8	62.4	56.8	58.9
LLaVA-OV + CoSeLECT	32	32	57.7	57.8	63.5	57.1	59.0
LLaVA-OV + CoSeLECT	64	32	58.6	57.6	65.1	57.7	59.8
LLaVA-OV + CoSeLECT	100	32	59.0	57.9	64.5	56.5	59.5
LLaVA-OV + CoSeLECT	200	32	58.9	57.8	65.2	58.3	60.1
LLaVA-OV + CoSeLECT	400	32	59.7	58.1	67.3	**59.3**	61.1
LLaVA-OV + CoSeLECT	800	32	60.6	**58.3**	66.2	**59.3**	61.1
LLaVA-OV + CoSeLECT	1600	32	**61.1**	58.1	**67.9**	58.8	**61.5**

Table 10. Effect of increasing the number of post-frames on baseline uniform sampling and CoSeLECT. Pre-embedding pool size is fixed at 1600.

Method	Post-Frames	VideoMME	MVBench	MLVU	Long-VideoBench	EgoSchema	Average
Baseline	32	58.4	57.8	62.4	56.8	62.8	59.6
CoSeLECT	32	**61.1**	**58.1**	**67.9**	**58.8**	**63.0**	**61.8**
Baseline	64	58.7	56.9	64.3	57.3	63.0	60.0
CoSeLECT	64	**59.7**	**57.1**	**66.8**	**57.8**	62.4	**60.8**
Baseline	128	58.3	56.6	65.9	56.9	63.4	60.2
CoSeLECT	128	**58.9**	**56.8**	**66.3**	**57.7**	**63.6**	**60.7**

method on a 40 GB A40 node with 8 GPUs. It is worth noting that the majority of the time is spent in the vision encoding and LLM forward pass, with only 0.08% of the time spent on inter-frame and text–frame embedding comparisons.

10. Appendix: Comparison with Adaptive Keyframe Sampling (AKS)

Adaptive Keyframe Sampling (AKS) [30] is a recent training-free method that, like CoSeLECT, leverages frame–text similarity for plug-and-play frame selection. While the two share a high-level motivation, they differ fundamentally in how they achieve coverage and allocate frames. Below, we outline the AKS algorithms explicitly and then provide a side-by-side comparison.

Comparison. Despite their different designs, both AKS and CoSeLECT share the same underlying goal: to select a subset of frames S that are both highly relevant to the query and sufficiently representative of the video as a whole. This trade-off can be abstracted as optimizing

$$\arg\max_{S \subset \text{frames}} \left[\alpha \cdot \text{Relevance}(S, \text{query}) + \beta \cdot \text{Coverage}(S, \text{video})\right]$$

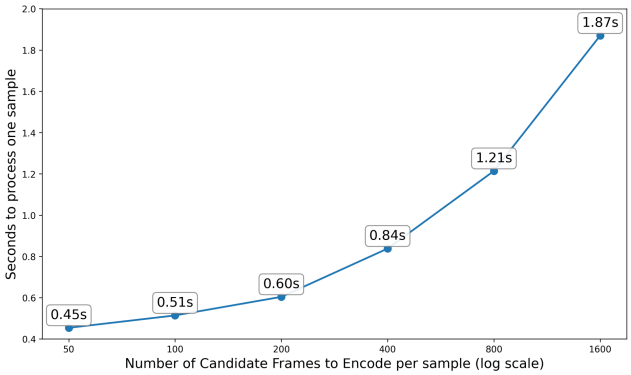


Figure 3. Empirically measured inference latency as the size of the pre-embedding frame pool is varied.

Algorithm 2 AKS Algorithm

Segmentation: Recursively split the video into equal halves.
if (max text-sim – avg text-sim) > threshold **or** recursion depth reached **then**
    stop splitting
else
    recurse on both halves
end if
Budget Allocation: Assign frame budget inversely proportional to recursion depth.
Frame Selection: Select top-k most text-relevant frames in each segment. Every segment receives at least one frame.

The two methods adopt a similar approach for computing **relevance** by leveraging frame–text similarity, but they diverge fundamentally in how they define and enforce **coverage**. AKS relies on statistical variance and recursive partitioning, while CoSeLECT incorporates semantic continuity through inter-frame similarity and adaptive budgeting. Table 11 summarizes these distinctions in detail.

This makes clear that CoSeLECT differs from AKS in two central respects: (1) **Semantic vs. Statistical Segmentation:** CoSeLECT explicitly respects inter-frame continuity, while AKS partitions by statistical thresholds over similarity scores. (2) **Dual-signal vs. Single-signal Coverage:** CoSeLECT combines frame–frame and frame–text signals for coverage and relevance, whereas AKS relies solely on variance in frame–text similarity.

10.1. Comparison across frame selection methods with a different backbone

Table 12. Performance on three benchmarks with Qwen2.5-VL-7B as the backbone. All methods use pre-$E_{\text{im}} = 800$ and post-$E_{\text{im}} = 32$.

Model	VideoMME	MLVU	LVBench
Qwen2.5-VL-7B + AKS	63.2	61.4	58.9
Qwen2.5-VL-7B + Q-Frame	58.3	**65.4**	58.4
Qwen2.5-VL-7B + CoSeLECT	**63.3**	63.4	**59.2**

Here we add an additional ablation in Tab. 12 that shows performance across AKS [30] and Q-Frame [48] with a different vision backbone.

10.2. Comparison across larger downstream frame counts

To isolate the effect of frame selection under a fixed context budget, we compare uniform sampling with CoSeLECT when selecting 256 frames. As shown in Table 13, CoSeLECT consistently improves over the uniform baseline on both MLVU and VideoMME, indicating that the gains arise from *which* frames are selected rather than from changes in the token-per-frame setting.

Table 13. Control experiment with Qwen2.5-VL under a fixed token-per-frame budget (max_pixels = 90k, min_pixels = 80k). The baseline uses 256 uniformly sampled frames; CoSeLECT selects 256 frames from a 1600-frame pool under the same context constraints.

Setting	VideoMME w/o subs	MLVU
Reported baseline	65.1	71.6
Reproduced baseline (uniform frames)	64.8	71.1
+ CoSeLECT (1600→256 frames)	**66.2**	**71.9**

CVPR
#31

Table 11. Algorithmic comparison of AKS and CoSeLECT.

	Adaptive Keyframe Sampling (AKS)	**CoSeLECT (Ours)**
Segmentation	**Statistical, top-down:** recursively splits the video into equal halves based on text-similarity variance. Segmentation is agnostic to scene boundaries.	**Semantic, bottom-up:** segments the video into variable-length, coherent subclips using inter-frame similarity, aligning with scene changes.
Budget Allocation	**Indirect proxy:** frame budget is determined by recursion depth; deeper segments receive fewer frames.	**Direct signals:** frame budget is allocated using subclip length and average text–frame similarity.
Frame Selection	**Forced allocation:** selects top-k text-relevant frames per segment; every segment must contribute at least one frame.	**Flexible allocation:** selects top-k text-relevant frames; irrelevant subclips can be skipped entirely (zero budget).

CVPR
#31

11. Appendix: Qualitative Results

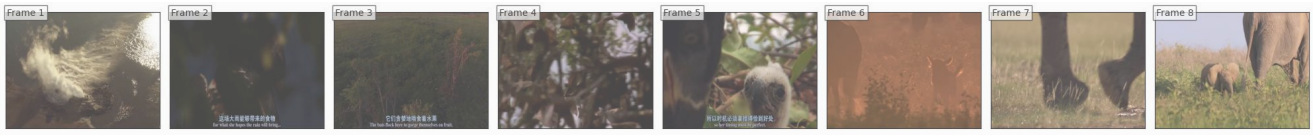
We now present qualitative comparisons between CoSeLECT and the baseline LLaVA-OneVision [12] across a diverse set of task types. These examples highlight how our frame selection strategy leads to more informative context for downstream video understanding.

In Figure 4, we consider a "needle in a haystack" setting, where the correct answer depends on identifying a single rare frame; CoSeLECT successfully surfaces the key frame while uniform sampling fails. Figure 5 examines plot-level reasoning, where CoSeLECT selects a broader variety of frames documenting different activities, including the critical one needed to answer the query. In Figure 6, we show an egocentric scenario where a subtle visual cue (a red clothing artifact) is crucial—CoSeLECT captures it, enabling correct reasoning. Figure 7 demonstrates topic-level reasoning, with our method surfacing semantically diverse frames (*e.g.*, beach scenes, palm trees) that enrich context. Finally, Figure 8 illustrates the query-aware nature of our approach: when posed with different queries on the same long video, CoSeLECT adapts its selections to highlight distinct but relevant evidence, minimizing redundancy while maintaining diversity. Together, these case studies demonstrate that CoSeLECT consistently selects frames that are both query-relevant and temporally diverse, thereby providing richer context than uniform or text-only sampling.

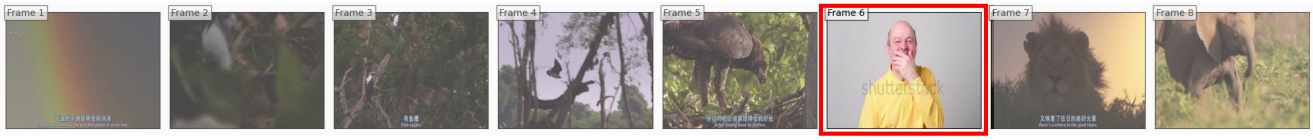
Needle Question Answering

MLLM responds to an inquiry concerning a particular clip (the "needle") embedded within a long video sequence

What is the background color when the man is laughing and covering his mouth?
(A) Black **(B) White** (C) Green (D) Blue



LLaVA-OneVision: (A)



LLaVA-OneVision + CoSeLECT **(ours)**: (B)

Figure 4. **Needle Question Answering:** CoSeLECT successfully localizes the "needle" to provide an accurate answer, whereas uniform sampling fails to do so.

Plot Question Answering

Understanding a plot in a narrative video

How does the cartoon sponge walk after entering the room?
(A) Swaggering **(B) Tip-Toeing** (C) Walking with pause (D) Hopping

LLaVA-OneVision: (A)

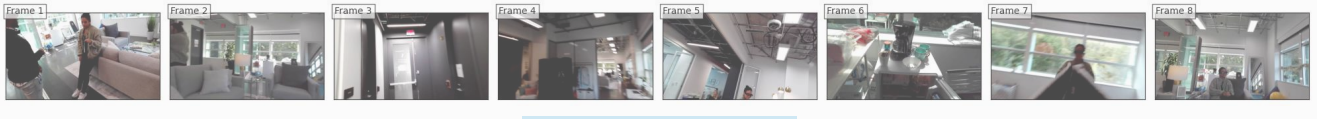
LLaVA-OneVision + CoSeLECT **(ours)**: (B)

Figure 5. **Plot Question Answering:** CoSeLECT samples diverse frames showing different types of walking, including the one containing the correct answer.

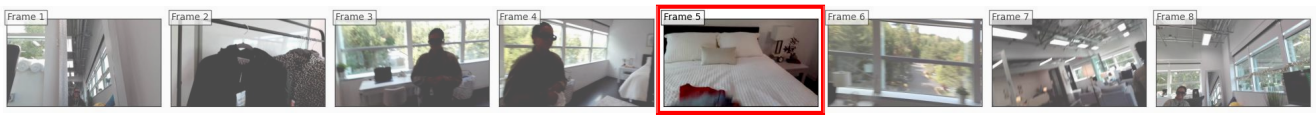
Ego Question Answering

Egocentric perspective long video dataset.

What color was the blouse I picked up?
(A) Green **(B) Red** (C) Yellow (D) Blue



LLaVA-OneVision: (A)



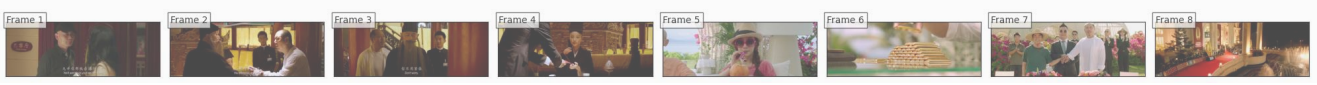
LLaVA-OneVision + CoSeLECT **(ours)**: (B)

Figure 6. **Ego Question Answering:** CoSeLECT captures a subtle artifact of red clothing in the frame—providing just enough context for the LLM to answer correctly.

Topic Reasoning

Answer questions about a principal subject in a long video

What is the main setting shown in the movie?
(A) Desert **(B) Ocean** (C) City (D) Forest



LLaVA-OneVision: (C)

LLaVA-OneVision + CoSeLECT **(ours)**: (B)

Figure 7. **Topic Reasoning:** CoSeLECT captures a broader diversity of concepts related to the query (*e.g.*, multiple scenes by the beach, palm trees), providing richer context for the LLM.

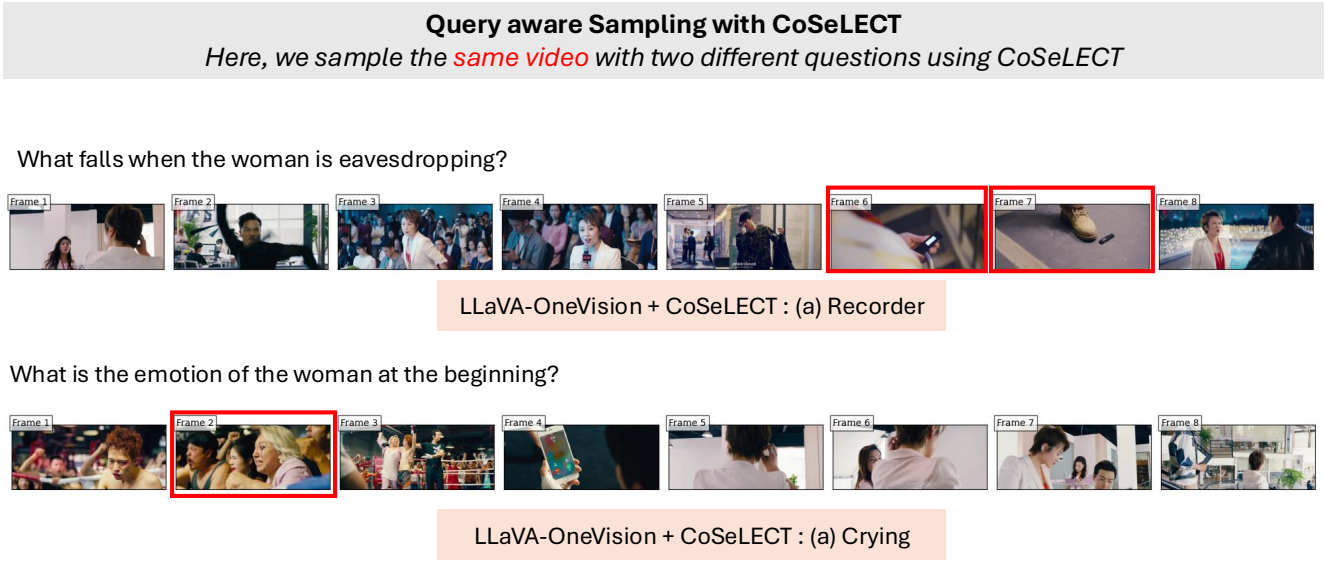


Figure 8. **Query-aware sampling in action:** CoSeLECT selects different frames for different queries from the same long source video. Two observations stand out: (1) In both cases, it surfaces relevant frames and produces the correct answer; (2) The selected frames exhibit minimal redundancy, showing that CoSeLECT generates relevant yet diverse frames.

12. Appendix: Code

```python
from typing import List
import numpy as np
from scipy.ndimage import gaussian_filter1d
from scipy.signal import savgol_filter


def select_frames(
    frame_similarities: np.ndarray,
    text_similarities: np.ndarray,
    num_frames: int = 32,
    min_subclip_duration: int = 15,
    scene_change_threshold: float = 0.8,
) -> List[int]:
    """Select the most informative frame indices from precomputed similarity signals.

    Args:
        frame_similarities: Per-frame cosine similarity between consecutive
            frames. Shape (N,). Low values indicate scene changes.
        text_similarities: Per-frame cosine similarity between each frame and
            a text query. Shape (N,). High values indicate relevant frames.
        num_frames: Number of frames to select.
        min_subclip_duration: Minimum frames for a valid subclip.
        scene_change_threshold: Frame similarity below this triggers a scene
            boundary. Lower = fewer, more dramatic cuts detected.

    Returns:
        Sorted list of selected frame indices in [0, N-1].
    """
    frame_similarities = np.asarray(frame_similarities, dtype=np.float32)
    text_similarities  = np.asarray(text_similarities,  dtype=np.float32)
    assert len(frame_similarities) == len(text_similarities)

    n = len(frame_similarities)
    if n <= num_frames:
        return list(range(n))

    scene_changes   = _detect_scene_changes(frame_similarities,
                                            scene_change_threshold)
    composite_score = _composite_score(frame_similarities, text_similarities)
    subclips        = _build_subclips(scene_changes, text_similarities, n,
                                      min_subclip_duration)
    _allocate_frames(subclips, num_frames)
    keyframes = _extract_keyframes(subclips, frame_similarities,
                                   text_similarities)
    keyframes = sorted(set(keyframes))
    keyframes = _adjust_count(keyframes, num_frames, composite_score)
    return keyframes


def _detect_scene_changes(
    frame_sim: np.ndarray, threshold: float, window: int = 15
) -> np.ndarray:
    smoothed  = savgol_filter(frame_sim, window, 3)
    mask      = smoothed < threshold
    changes   = np.where(np.diff(mask.astype(int)) > 0)[0]
    endpoints = []
    if len(changes) == 0 or changes[0] != 0:
        endpoints.append(0)
    endpoints.extend(changes.tolist())
    if endpoints[-1] != len(frame_sim) - 1:
        endpoints.append(len(frame_sim) - 1)
    return np.array(sorted(set(endpoints)))


def _composite_score(
```

```python
    frame_sim: np.ndarray, text_sim: np.ndarray, window: int = 15
) -> np.ndarray:
    fs    = savgol_filter(frame_sim, window, 3)
    ts    = savgol_filter(text_sim,  window, 3)
    score = ts * (1 - np.abs(np.gradient(fs)))
    return gaussian_filter1d(score, sigma=window / 3)


def _build_subclips(
    scene_changes: np.ndarray,
    text_sim: np.ndarray,
    n: int,
    min_duration: int,
) -> list:
    subclips = []
    for i in range(len(scene_changes) - 1):
        start, end = int(scene_changes[i]), int(scene_changes[i + 1])
        if end - start < min_duration:
            continue
        seg_text = text_sim[start:end]
        subclips.append(dict(
            start=start, end=end,
            duration=end - start,
            relevance=float(np.max(seg_text)),
        ))
    if not subclips:
        subclips = [dict(start=0, end=n - 1, duration=n - 1,
                         relevance=float(np.mean(text_sim)))]
    subclips.sort(key=lambda c: c["relevance"], reverse=True)
    return subclips


def _allocate_frames(subclips: list, num_frames: int) -> None:
    total = sum(c["relevance"] * np.sqrt(c["duration"]) for c in subclips)
    if total == 0:
        total = 1.0
    for c in subclips:
        proportion  = (c["relevance"] * np.sqrt(c["duration"])) / total
        c["n_alloc"] = max(1, int(proportion * num_frames))
    allocated = sum(c["n_alloc"] for c in subclips)
    if allocated < num_frames:
        for i in range(min(num_frames - allocated, len(subclips))):
            subclips[i]["n_alloc"] += 1
    elif allocated > num_frames:
        excess = allocated - num_frames
        for i in range(1, min(excess + 1, len(subclips))):
            if subclips[-i]["n_alloc"] > 1:
                subclips[-i]["n_alloc"] -= 1


def _extract_keyframes(
    subclips: list,
    frame_sim: np.ndarray,
    text_sim: np.ndarray,
) -> List[int]:
    keyframes = []
    for clip in subclips:
        k, start, end = clip["n_alloc"], clip["start"], clip["end"]
        if k <= 0:
            continue
        if k == 1:
            keyframes.append(int(start + np.argmax(text_sim[start:end])))
            continue
        scores = _subclip_scores(frame_sim, text_sim, start, end)
        for i in range(k):
            seg_s = int(i       * (end - start) / k)
            seg_e = int((i + 1) * (end - start) / k)
```

```
            if seg_e <= seg_s:
                seg_e = seg_s + 1
            best = seg_s + np.argmax(scores[seg_s:seg_e])
            keyframes.append(int(start + best))
    return keyframes


def _subclip_scores(
    frame_sim: np.ndarray, text_sim: np.ndarray,
    start: int, end: int,
) -> np.ndarray:
    length = end - start
    win    = min(15, length // 2 * 2 + 1)
    if win < 3:
        win = 3
    smoothed = savgol_filter(text_sim[start:end], win, 3)
    rng = np.max(smoothed) - np.min(smoothed)
    if rng > 1e-10:
        return (smoothed - np.min(smoothed)) / rng
    return np.zeros(length)


def _adjust_count(
    keyframes: List[int], target: int, composite: np.ndarray
) -> List[int]:
    if len(keyframes) < target:
        remaining  = target - len(keyframes)
        candidates = np.argsort(composite)[::-1]
        extras     = [int(i) for i in candidates
                      if i not in keyframes][:remaining]
        keyframes  = sorted(set(keyframes) | set(extras))
    elif len(keyframes) > target:
        scores    = [composite[i] for i in keyframes]
        top       = np.argsort(scores)[::-1][:target]
        keyframes = sorted(keyframes[i] for i in top)
    return keyframes
```

Listing 1. A self-contained reference implementation of the adaptive frame selection algorithm described in Section 3. The snippet below requires only `numpy` and `scipy`, and is agnostic to the choice of vision-language encoder (CLIP, SigLIP, etc.). As a frame-level selection method, it can be dropped into any existing video-MLLM pipeline at the point where frames are sampled, with no other modifications required.