CoRDS: Coreset-based Representative and Diverse Selection for Streaming Video Understanding

Ailar Mahdizadeh[1,2] **Puria Azadi**[1] **Muchen Li**[1,2] **Xiangteng He**[1,2] **Leonid Sigal**[1,2,3]
[1]University of British Columbia [2]Vector Institute for AI [3]CIFAR AI Chair
ailar.mahdizadeh@ubc.ca

Abstract

Streaming video understanding with large vision-language models (VLMs) requires a compact memory that can support future reasoning over an ever-growing visual history. A common solution is to compress the key-value (KV) cache, but existing streaming methods typically rely on local token-wise heuristics, such as recency, temporal redundancy, or saliency, which do not explicitly optimize whether the retained cache is representative of the accumulated history. We propose to view KV-cache compression as a *coreset selection* problem: rather than scoring tokens independently for retention, we select a small subset that covers the geometry of the accumulated visual cache. Our method [1] operates in a joint KV representation and introduces a bicriteria objective that balances coverage in key and value spaces, preserving both retrieval structure and output-relevant information. To encourage a more diverse retained subset, we further introduce an orthogonality-driven diversity criterion that favors candidates contributing new directions beyond the current selection, and connect this criterion to log-determinant subset selection. Across seven open-source VLMs ranging from 3B to 27B parameters and five long-video and streaming-video benchmarks, our method improves over heuristic streaming compression baselines under a fixed cache budget. These results highlight that representative coreset selection offers a more effective principle, than token-wise pruning, for memory-constrained streaming video understanding.

1 Introduction

Large vision-language models (VLMs) have rapidly expanded the scope of visual reasoning, enabling a single multimodal backbone to handle open-ended queries over images, multi-image inputs, and long video sequences [1, 2]. Recent advances in long-context modeling have further shown that extending the effective context window of the language backbone can substantially improve long-video reasoning. For example, LongVA [3] processes up to 2000 frames, or more than 200K visual tokens, without video-specific retraining. These developments make it increasingly plausible to deploy VLMs in always-on, real-time settings such as streaming assistants, wearable companions, AR/VR interfaces, embodied service robots, and other persistent perception agents that must interpret video as it arrives rather than after the full clip has been observed [4].

Streaming video understanding (SVU) differs fundamentally from conventional offline video understanding (OVU). In OVU, the full video is available before inference, enabling frame sampling, token compression, or evidence retrieval with complete temporal context; in some settings, these operations can even be conditioned on the downstream query [5, 6]. In SVU, by contrast, future frames are unavailable, queries may arrive asynchronously, and all pre-query processing must be causal and independent of the query [4, 7, 8]. Thus, memory must remain bounded over unbounded streams, latency must support interactive use, and past evidence must be retained without re-encoding.

[1]Code is available at https://github.com/ailarmhz/CoRDS

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Prior work has sought to mitigate these constraints through *frame sampling* [9, 10], *input-side vision compression* [11, 5, 6], and *KV-cache compression* [12, 4, 13]. These approaches substantially improve the feasibility of long-horizon video reasoning, yet they all face the same design tension: *which frames, tokens, or cached states deserve to be kept when only a tiny fraction of the visual history can remain active?* Existing methods typically answer this question using local token-level heuristics, such as recency, temporal redundancy, norm-based saliency, or attention-derived relevance [12, 4, 13, 10]. For instance, InfiniPot-V [4] scores cached KV entries using key similarity to estimate temporal redundancy and value norms to estimate saliency, respectively. StreamMem [13] instead distils historical context into a small set of memory tokens via cross-attention, effectively summarising the evicted cache rather than selecting from it. While computationally attractive, both families share a fundamental limitation: they do not explicitly optimize whether the retained memory is globally *representative* of the full accumulated history. As a result, cues that are individually weak but collectively important (static visual details, weak complementary signals, temporally distributed structure) may be discarded when they are not locally salient at the moment of compression.

We take a different and more principled approach: instead of scoring cached tokens independently, we formulate KV-cache compression as representative coreset selection. The key question becomes which small subset best covers the accumulated visual KV cache under a fixed memory budget. We instantiate this idea in the cached KV feature space and make three contributions:

(1) A coreset view of streaming KV-cache compression. We introduce a query-agnostic, coverage-oriented formulation for visual KV-cache compression, shifting the objective from local token-wise pruning to selecting a compact memory that represents the accumulated video context or memory.

(2) Joint KV representative coverage. We show, empirically, that representative selection should account for both keys and values: keys determine how future queries retrieve cached entries, while values determine the information read out by attention. Based on this observation, we introduce a joint KV representation and a bicriteria coverage objective that explicitly balances approximation in key space and value space, yielding a more representative retained cache.

(3) Orthogonal diversity for non-redundant memory. A greedy optimization of the representativeness objective, however, may yield a redundant coreset. Further, multiple representative subsets may have equal or similar representative power. To avoid wasting token budget, we introduce an orthogonal anti-redundancy term that favors candidates contributing new directions beyond the span of the current selection. We connect this criterion to log-determinant subset selection, providing a principled motivation for proposed lightweight regularizer on top of the coreset coverage objective.

This results in an inference-time approach that can easily be incorporated into any transformer-based model. Consequently, we illustrate it within various versions of Qwen (2-VL [2], 2.5-VL [14], 3-VL [15], and 3.5 [16]) and LLaVA-Next [17] families of models on five video benchmarks (EgoSchema [18], MLVU [19], Video MME [20], OVO-Bench[21] and StreamingBench [7]) under various memory / token compression budgets. In all cases we illustrate substantial improvement over baseline approaches (*e.g.*, InfiniPot-V [4] and StreamMem [13]), especially in low memory budget regimes, and sometimes even improve on performance of the full model counterparts that have access to uncompressed memory – an effect that accounts for the difficulty of dealing with long and highly imbalanced visual context, even when it fits in memory (a.k.a. needle-in-a-haystack [22]).

2 Related Work

Video Understanding. Video understanding has a long history in computer vision, with work in the area dating back 20+ years. With the advent of deep learning, video understanding has shifted from hand-crafted representations to Transformer-based video foundation models and vision-language models. VLM-based video solutions [14, 23, 24, 1, 25] are highly expressive, as they allow free-form interaction and querying of the videos. However, scaling VLMs to long videos remains a core challenge, as memory and compute scale poorly (linearly and quadratically) with the length of the video in these transformer-based architectures. This is particularly challenging in streaming settings, where the length of the video may not only be long but is unknown and effectively infinite.

Efficient video understanding via visual token pruning. To enable efficient long-video understanding, recent works have been focusing on inference time technique to improve the performance of long video understanding for VLMs without having to retrain/finetune them. A common strategy which lies in this scope is to reduce the number of visual tokens before they are processed by the LLM.

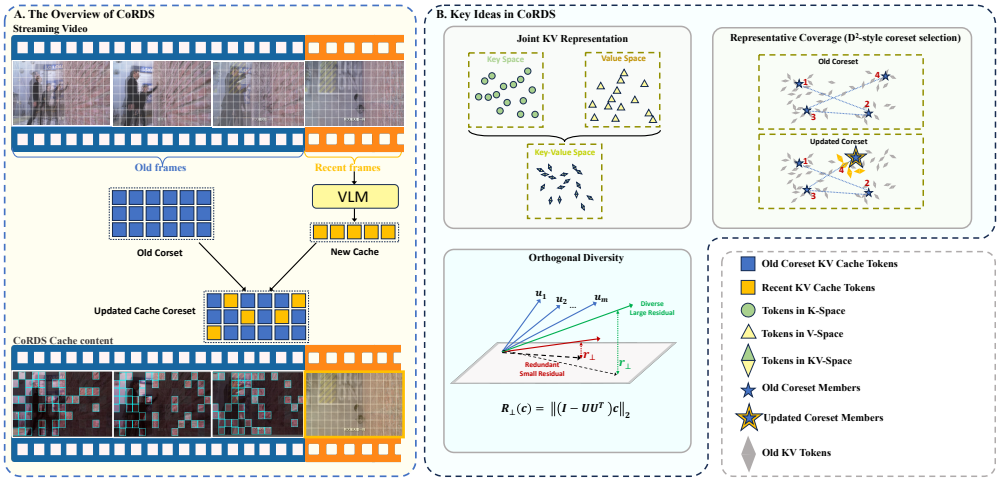

Figure 1: **CoRDS framework.** As streaming frames arrive, the accumulated KV cache is compressed by D^2-style coreset selection in joint K-V space (Eq. 4) with an orthogonal anti-redundancy term (Eq. 9); the persistent cache is then used to answer asynchronous question queries.

This line of work operates at the input-token level and typically exploits redundancy in the visual stream. Existing methods prune or merge tokens based on temporal cues (EVS [26]), feature-similarity clustering (DivPrune [27], Token Merging [28]), or vision-encoder saliency (LLaVA-PruMerge [29]). Despite being effective at reducing number of visual-tokens, these method usally relies solely on the single-frame visual features and relies on simple heruistics [26] to reduce temporal redundancy. Also these line of work does not have a natural mechanism to preserve a pool of memory and therefore hard to adapt to the long streaming video setting. As a result, they are complementary to our setting, where the key challenge is how to update and compress the accumulated KV cache as the video arrives over time.

Streaming video understanding and KV-cache compression. Streaming video understanding focuses on the online setting with unbounded input length. Benchmarks such as StreamingBench [7] and models like StreamingVLM [30] and LongVU [6] address this regime. While some methods operate at the frame level (e.g., AKS [10]), recent work integrates KV-cache compression into streaming pipelines. InfiniPot-V [4], ReKV [12], and StreamMem [13] apply heuristic-based strategies using similarity, norms, or retrieval. Despite their effectiveness, these methods lack formal guarantees on the representativeness of the retained memory. In contrast, our approach formulates KV-cache compression as a coreset selection problem with geometric grounding, offering a principled alternative for maintaining a compact yet representative memory in streaming video understanding.

3 Method

We formulate KV-cache compression as a representative coreset selection problem of the accumulated visual cache. In doing so, we are guided by a few core principles. First, for greatest versatility the approach should be *training-free*, meaning that it needs to rely on capacity of the trained model and its internal representations. Second, the selected coreset must be as *representative* as possible (for performance), *non-redundant* (for memory efficiency) and *query agnostic* (for practical use). Third, given hierarchy of semantic representations at different levels of the transformer, the optimal coresets may differ at these levels arguing for a potentially *local*, as opposed to monolithic, selection policy.

With these design constraints in mind we first establish *where selection should be made*, using the joint KV feature space; then define *how to represent* tokens in that space, through a bicriteria coverage objective optimized by a deterministic D^2-style farthest-first rule; and finally address *how to diversify* the selected subset, using an orthogonal anti-redundancy regularization motivated by a log-determinant approximation guarantee.

Setup, coreset objective, and tractable surrogate. A transformer decoder with L layers maintains, at each layer ℓ, key and value projections for every processed token i:

$$\mathbf{k}_i^{(\ell)} = W_K^{(\ell)} h_i^{(\ell)}, \qquad \mathbf{v}_i^{(\ell)} = W_V^{(\ell)} h_i^{(\ell)}, \tag{1}$$

where $h_i^{(\ell)} \in \mathbb{R}^{d_{\text{model}}}$ is the hidden state. For simplicity, we omit the layer superscript when discussing the selection problem within a fixed layer. At streaming time, the cache is naturally divided into an old cache $\mathcal{O} = \{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^N$ and a recent cache $\mathcal{R}$. The recent cache is always retained to preserve the latest visual state. The old cache, however, must be reduced to a budget-b subset $\mathcal{S} \subset [N]$. The retained cache is therefore $\mathcal{S} \cup \mathcal{R}$, with $|\mathcal{S}| = b$. The ideal subset would maximize downstream performance, which depends on future queries and is intractable at selection time.

We therefore replace it with a *query-agnostic geometric coverage objective*. We view the compressed cache as a coreset: a small subset that approximates the full accumulated cache for future attention queries. Let $\boldsymbol{\phi}_i$ denote a feature representation of cached token i. Ideally, we want a weighted subset $(\mathcal{S}, \mathbf{w})$ that uniformly approximates the full token set over a future attention queries set ($\mathcal{Q}$):

$$\sup_{q \in \mathcal{Q}} \left| \sum_{i \in \mathcal{S}} w_i \, f(q, \boldsymbol{\phi}_i) - \sum_{i=1}^{N} f(q, \boldsymbol{\phi}_i) \right| \leq \varepsilon \sum_{i=1}^{N} f(q, \boldsymbol{\phi}_i), \tag{2}$$

where $f(q, \boldsymbol{\phi}_i)$ denotes the contribution of token i to the attention output, and the supremum ($\sup_{q \in \mathcal{Q}} ||$) is taken over all $q \in \mathcal{Q}$, corresponding to the worst-case approximation error over future queries. Here $w_i \in \mathbb{R}_{\geq 0}$ is the weight assigned to retained token $i \in \mathcal{S}$ (collected into the vector $\mathbf{w} = (w_i)_{i \in \mathcal{S}}$), reflecting how much of the original cache that token represents in the approximation. The approximation error of the compressed cache can be bounded by key-space and value-space quantization errors (Proposition 1, Appendix A).

Since exactly solving Eq. (2) is NP-hard, we instead minimize the tractable *quantization-error surrogate* $\sum_{i \in \mathcal{O}} d_\alpha(i, \mathcal{S})$, with d_α defined in Eq. (4) below, which upper-bounds the coreset error in the joint KV feature space.

When old tokens are organized into video frames, we work at frame level: frame t is represented by the centroid $\bar{\mathbf{k}}_t = \frac{1}{|\mathcal{I}_t|} \sum_{i \in \mathcal{I}_t} \mathbf{k}_i$ (and $\bar{\mathbf{v}}_t$ analogously), and selected frame indices are then expanded to token indices.

Joint KV representation. The first design choice is the space in which coverage should be measured. For each cached token, three natural feature choices are:

$$\boldsymbol{\phi}_i^{(K)} = \mathbf{k}_i, \qquad \boldsymbol{\phi}_i^{(V)} = \mathbf{v}_i, \qquad \boldsymbol{\phi}_i^{(KV)} = \begin{bmatrix} \mathbf{k}_i \\ \mathbf{v}_i \end{bmatrix} \in \mathbb{R}^{d_k + d_v}. \tag{3}$$

Selecting only in key space ($\boldsymbol{\phi}^{(K)}$) preserves retrieval structure but ignores the content read out by attention. Selecting only in value space ($\boldsymbol{\phi}^{(V)}$) preserves output content but ignores which cached tokens future queries are likely to retrieve. We therefore select in the joint KV representation ($\boldsymbol{\phi}^{(KV)}$). Empirically, joint KV selection consistently outperforms K-only or V-only selection (Section 4.7); theoretically, the attention-error bound depends on both key and value approximation errors (Proposition 1, Appendix A). All subsequent construction operates on $\boldsymbol{\phi}_i = \boldsymbol{\phi}_i^{(KV)}$, and the same selected index set is used to retain the corresponding keys and values.

Bicriteria KV coreset. Directly concatenating keys ($\mathbf{k}$) and values ($\mathbf{v}$) implicitly fixes their relative importance through feature scale. To make this balance explicit, we define a *bicriteria distance* from a candidate token i to a selected set $\mathcal{S}$:

$$d_\alpha(i, \mathcal{S}) = \min_{j \in \mathcal{S}} \left[\alpha \left\| \mathbf{k}_i - \mathbf{k}_j \right\|^2 + (1 - \alpha) \left\| \mathbf{v}_i - \mathbf{v}_j \right\|^2 \right], \qquad \alpha \in [0, 1]. \tag{4}$$

The resulting coreset objective is:

$$\mathcal{S}_b^\star = \arg\min_{|\mathcal{S}| = b} \sum_{i \in \mathcal{O}} d_\alpha(i, \mathcal{S}). \tag{5}$$

The limit $\alpha \to 1$ recovers K-only selection, while $\alpha \to 0$ recovers V-only selection. In practice, we find that value-favoring settings, $\alpha < 0.5$, perform best, which aligns with the attention-output bound: values directly determine output magnitude while keys only modulate which tokens are retrieved.

Greedy D^2-style coreset selection. We minimize Eq. (5) greedily using D^2-style farthest-first selection, the deterministic counterpart of D^2-weighted sampling [31], which seeds cluster centers in proportion to their squared distance from all previously chosen centers.

At each step t, the next token is

$$i_{t+1} = \arg\max_{i \in \mathcal{O} \setminus \mathcal{S}_t} \; d_\alpha(i, \mathcal{S}_t)\,, \tag{6}$$

where $\mathcal{S}_t = \{i_1, \ldots, i_t\}$ is the current selected set. Compared with k-means initialization or greedy variants that do not account for distance weighting, this squared-distance criterion is the one that gives D^2 sampling its $\mathcal{O}(\log k)$-approximation guarantee for the k-means objective. The full greedy pass runs in $\mathcal{O}(b \cdot N)$ per layer-head, making it practical for online streaming inference.

Orthogonal anti-redundancy. Coverage-based selection has a subtle failure mode: two candidates with high d_α scores may lie in nearly the same direction in feature space, so selecting both wastes budget while leaving other directions uncovered (see Figure 2). We address this through an *orthogonal subspace criterion* grounded in log-determinant coverage.

Given the selected key matrix $\mathbf{U}_t = [\mathbf{k}_j]_{j \in \mathcal{S}_t} \in \mathbb{R}^{d_k \times t}$, define the orthogonal projector

$$P_t^K = \mathbf{U}_t \big(\mathbf{U}_t^\top \mathbf{U}_t\big)^\dagger \mathbf{U}_t^\top \tag{7}$$

and the *key-space residual* of candidate i:

$$\mathbf{r}_i = \big(I - P_t^K\big)\mathbf{k}_i, \qquad \mathbf{r}_i^V = \big(I - P_t^V\big)\mathbf{v}_i. \tag{8}$$

The squared norm $\|\mathbf{r}_i\|^2$ measures the component of $\mathbf{k}_i$ that lies *outside* the span already covered by $\mathcal{S}_t$: a large residual means token i contributes a genuinely new direction. Similarly, P_t^V and $\mathbf{r}_i^V$ are defined using the selected value matrix. The *orthogonal novelty score* is

$$\mathrm{Orth}(i \mid \mathcal{S}_t) = \eta \, \|\mathbf{r}_i\|^2 + (1-\eta) \, \left\|\mathbf{r}_i^V\right\|^2, \qquad \eta \in [0,1]. \tag{9}$$

This orthogonal term is motivated by classical log-determinant subset selection. In particular, for the key matrix $\mathbf{U}_\mathcal{S}^K = [\mathbf{k}_j]_{j \in \mathcal{S}}$, define

$$\mathcal{F}(\mathcal{S}) = \log\det\big((\mathbf{U}_\mathcal{S}^K)^\top \mathbf{U}_\mathcal{S}^K + \epsilon I\big)\,.$$

Greedily selecting the candidate with the largest orthogonal residual maximizes the marginal gain of this log-det objective and achieves the $(1 - e^{-1})$ approximation guarantee for monotone submodular maximization (Theorem 2 in Appendix A). We use this result as a motivation for the diversity term. The final algorithm, however, uses orthogonality only as a soft bonus on top of the bicriteria coreset objective; we do not claim the same approximation guarantee for the combined score.

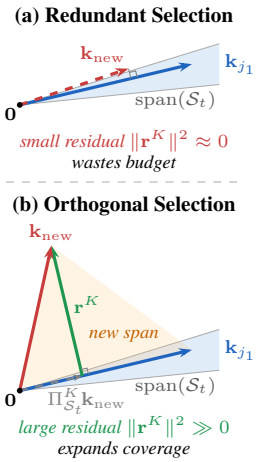

Figure 2: **Orthogonality intuition.** *(a)* A near-span candidate has a small residual $\mathbf{r}^K$ and adds little new coverage, whereas *(b)* a large-residual candidate expands the span into a new region.

Final selection score. Pure orthogonal selection is not sufficient: a geometrically novel token may be irrelevant to the model output, and exact span projection is expensive for large caches. We therefore keep bicriteria coverage as the backbone and use orthogonality as a lightweight anti-redundancy regularizer. At step t, each candidate is scored, and the next representative is selected as follows:

$$\mathrm{score}(i \mid \mathcal{S}_t) = \widetilde{d}_\alpha(i, \mathcal{S}_t) + \lambda \widetilde{\mathrm{Orth}}(i \mid \mathcal{S}_t), \quad i_{t+1} = \arg\max_{i \in \mathcal{O} \setminus \mathcal{S}_t} \mathrm{score}(i \mid \mathcal{S}_t), \tag{10}$$

where $\lambda > 0$ is a small fixed weight and $\widetilde{(\cdot)}$ denotes min-max normalization over the current candidate pool. The min-max normalization is important because the bicriteria distance and orthogonal novelty can have different numerical scales. The orthogonal bonus is implemented with a max-cosine redundancy surrogate (cheaper than exact span projection), making the per-step overhead comparable to a single matrix-vector product. Eq. (10) provides a compact selection criterion that balances coverage of the joint $(\mathbf{k}, \mathbf{v})$ geometry with an orthogonal diversity term that discourages redundant directions, thereby retaining tokens that best preserve the geometry of the accumulated cache.

Streaming update and replacement policy. CoRDS always keeps the most recent tokens, the recent cache $\mathcal{R}$, uncompressed; tokens in $\mathcal{R}$ are never evicted. As decoding proceeds, new tokens enter $\mathcal{R}$ and push its oldest tokens out, at which point those tokens become eligible for compression. At compression step τ, let $\mathcal{E}_\tau$ be the tokens that have exited $\mathcal{R}$ since step $\tau - 1$. The candidate pool is $\mathcal{O}_\tau = \mathcal{S}_{\tau-1} \cup \mathcal{E}_\tau$, where $\mathcal{S}_{\tau-1}$ is the coreset retained at the previous step. CoRDS rescores the entire pool with Eq. (10) and selects exactly b tokens as the new coreset $\mathcal{S}_\tau$. Every token in $\mathcal{O}_\tau \setminus \mathcal{S}_\tau$ is evicted, including tokens retained at earlier steps. The cache after the update is $\mathcal{S}_\tau \cup \mathcal{R}_\tau$, so its size stays fixed at $|M| = b + |\mathcal{R}|$. Old and new tokens therefore compete under the same joint K/V coverage and diversity criterion, and memory remains bounded at every step.

Layer-selective compression and cascade reuse. The selection rule above can be applied independently to each decoder layer. However, not all layers benefit equally from compression. Early layers tend to retain more redundant frame-level visual structure, while later layers contain more task-conditioned features that are more sensitive to pruning. We therefore apply the coreset selector only to selected layers and leave the remaining layers to reuse the indices selected by the most recent active layer. All layers thus hold compressed caches bounded by $|M|$. Therefore, to reduce the cost of running selection on multiple layers, we use a cross-layer cascade. A small set of anchor layers performs the full coreset selection, while nearby follower layers reuse the selected token indices from their anchor. Importantly, follower layers reuse only the indices, not the K/V vectors themselves. If layer ℓ' follows anchor layer ℓ with selected indices $\mathcal{S}^{(\ell)}$, then layer ℓ' retains its own cached values at those positions, $K^{(\ell')} \leftarrow K^{(\ell')}[\mathcal{S}^{(\ell)}]$ and $V^{(\ell')} \leftarrow V^{(\ell')}[\mathcal{S}^{(\ell)}]$. This preserves layer-specific representations while avoiding redundant selection passes.

4 Experiments

4.1 Experimental Setup: Models, Benchmarks, and Budgets

We evaluate our training-free, query-agnostic compressor on seven open VLMs: Qwen2-VL-7B [2], Qwen2.5-VL-3B/7B [14], and LLaVA-NeXT-Video-7B [17], and the newer Qwen3-VL-8B [15], Qwen3.5-VL-9B, and Qwen3.5-VL-27B [16]. Following the OVU/SVU split of recent streaming-oriented work [4, 13], we evaluate on three long-video benchmarks originally designed for offline evaluation (EgoSchema [18], MLVU [19], and VideoMME [20], reported on the Short/Medium/Long splits), which we run under a causal streaming protocol (Sec. 4.2), and two *streaming* benchmarks, OVO-Bench [21] and StreamingBench [7], where queries arrive at arbitrary timestamps. In every experiment, visual tokens are processed causally and the compression rule never accesses the question.

Algorithm 1 CoRDS: Coreset-based KV-Cache Compression

Require: Old cache $\mathcal{O}=\{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^{N}$, recent cache $\mathcal{R}$ (kept), budget b, hyperparams $\alpha, \eta, \lambda, \varepsilon_0$
Ensure: Compressed cache $(\mathcal{S} \cup \mathcal{R})$, $|\mathcal{S}|=b$

$\mathcal{S}_0 \leftarrow \{\arg\max_i \ \|\mathbf{k}_i+\mathbf{v}_i\|\}$ ▷ seed
for $t = 0, 1, \ldots, b-1$ **do**
  for each $i \in \mathcal{O} \setminus \mathcal{S}_t$ **do**
    $d_\alpha(i, \mathcal{S}_t)$ via Eq. (4)
    $\mathrm{Orth}(i \,|\, \mathcal{S}_t)$ via Eq. (9)
  end for
  Min-max normalize both terms
  $i_{t+1} \leftarrow \arg\max_{i \in \mathcal{O} \setminus \mathcal{S}_t} \mathrm{score}(i \,|\, \mathcal{S}_t)$ (Eq. (10))
  $\mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{i_{t+1}\}$
end for
return $\mathcal{S}_b \cup \mathcal{R}$

We use each model's standard sampling: Qwen2(.5)-VL with 768 frames (∼50K tokens) and LLaVA-NeXT-Video with 128 frames (∼25K tokens). Accuracy is reported at two cache budgets, $|M|$=3K and $|M|$=6K visual KV tokens. For methods whose official results are available under matching backbone and budget, we quote numbers from the original papers; otherwise, we rerun them in our evaluation pipeline.

4.2 Long-Video Benchmarks under the Streaming Protocol

For the offline long-video benchmarks, we use EgoSchema [18], MLVU [19], and VideoMME [20] as controlled testbeds for streaming-style compression. Although these benchmarks are commonly evaluated in an offline setting, we process visual tokens causally as the video unfolds, rather than assuming that the entire video is available upfront. The cache is therefore compressed online before the question is accessed, preserving the query-agnostic setting used throughout our evaluation, as in InfiniPot-V and StreamMem. We compare against three families of baselines on the offline

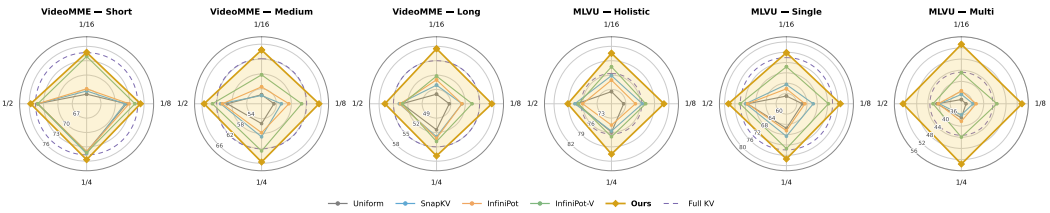

Figure 3: **Per-task accuracy across compression ratios on Qwen2-VL-7B.** Each radar compares methods across compression ratios; the dashed ring denotes Full KV.

benchmarks: the uncompressed Full KV cache as a conceptual upper bound[2]; the query-agnostic vision-token compressors STC [6] and TTC [32]; and the KV-cache compressors InfiniPot-V [4] and StreamMem [13]. We include reference points from GPT-4V/4o and proprietary long-video systems (LLaVA-OV [33], LongVU [6]) where reported. Table 1 reports accuracy on EgoSchema, MLVU, and VideoMME across four backbones at two cache budgets. At $|M|$=6K ($\sim 8\times$ compression), our method matches or exceeds the uncompressed Full KV baseline on nearly all Qwen2(.5)-VL backbones, while at $|M|$=3K ($\sim 17\times$ compression) it still outperforms InfiniPot-V at twice the budget. In addition, on Qwen3-VL-8B, Qwen3.5-VL-9B, and Qwen3.5-VL-27B (Table 1), CoRDS beats InfiniPot-V in all 18 matched comparisons. Overall, our method improves over prior compression baselines under matched budgets across the evaluated backbones, and in many cases matches or exceeds the Full KV reference. This suggests that the performance gains come from retaining a more informative and less redundant subset of visual KV tokens, rather than merely increasing the cache.

4.3 Streaming Video Understanding

Streaming evaluation is the strictest test of a query-agnostic compressor: the question arrives at an arbitrary timestamp, so query-dependent rules cannot affect tokens compressed before the question is known. Table 2 reports OVO-Bench and StreamingBench with Qwen2-VL-7B at $|M|$=6K. The uncompressed Qwen2-VL-7B aggregates are taken from the OVO-Bench [21], while InfiniPot-V [4] is re-run under our setting at matched budget. On OVO-Bench, our method exceeds both baselines on the overall score, with gains distributed across Backward, Real-time, and Forward sub-tracks rather than concentrated in one regime. On StreamingBench, our compressor exceeds InfiniPot-V on the majority of sub-tasks, with the largest gains on categories that require evidence dispersed across long stretches such as Prospective Reasoning, Counting, and Causal Reasoning.

4.4 Memory Footprint and Throughput

Table 3 reports peak VRAM, prefill FPS, and decode throughput on Qwen2-VL-7B at video lengths $L \in \{300, 500, 800, 1000\}$ seconds. Full KV runs out of memory beyond $L = 500\,\text{s}$, while both InfiniPot-V and our method maintain a flat memory footprint across all lengths. Our base configuration matches InfiniPot-V on memory and decode throughput. Adding the follower strategy, in which a small set of anchor layers run the full selection rule and the rest reuse their decisions (Appendix, Table 11), preserves the memory footprint and overtakes InfiniPot-V on prefill FPS at every length and on decode throughput from $L \geq 500\,\text{s}$. Prefill FPS includes the per-block reselection overhead.

4.5 Compression Ratios

Figure 3 sweeps the compression ratio $\{1/16, 1/8, 1/4, 1/2\}$ on Qwen2-VL-7B across the three VideoMME splits (Short, Medium, Long) and three MLVU task types (Holistic, Single, Multi), comparing our method against Uniform, SnapKV, InfiniPot, InfiniPot-V, and the Full KV reference. Our method consistently traces the outer edge of every radar regardless of ratio, and on several tasks, lies entirely outside the Full KV ring and holds across task types and length splits rather than being an artifact of any single benchmark. The advantage is largest at aggressive ratios ($1/16$ and $1/8$), where all baselines visibly contract toward the centre while ours remains flat, supporting the claim

[2]Note, that this is not a true upper bound in terms of performance as long context may actually negatively impact accuracy, resulting in compressed cache model performing better in practice.

Method		Size	# Frames (Max)	Budget |M|	EgoSchema	MLVU	VideoMME
GPT4-V		–	1fps	–	55.6	–	60.7
GPT4-o		–	1fps	–	72.2	66.2	77.2
LLaVA-OV [33]		7B	32	8K	60.1	64.7	58.2
LongVU [6]		7B	1fps	8K	67.6	65.4	60.6
LongVU [6]		3B	1fps	8K	59.1	55.9	51.5
Qwen2-VL	Original	7B	768	50K	65.2	65.8	63.9
	+ TTC [32]	7B	768	6K	–	58.4	59.8
	+ STC [6]	7B	768	6K	–	57.9	60.1
	+ InfiniPot-V [4]	7B	768	6K	65.6	65.8	62.8
	+ StreamMem [13]	7B	4.0/0.5 fps	6K	67.2	65.9	62.1
	+ **CoRDS (ours)**	7B	768	6K	**68.4** +1.2	**70.1** +4.2	**64.8** +2.0
	+ TTC [32]	7B	768	3K	–	54.8	55.3
	+ STC [6]	7B	768	3K	–	55.0	56.1
	+ InfiniPot-V [4]	7B	768	3K	–	63.1	61.2
	+ **CoRDS (ours)**	7B	768	3K	**68.1**	**67.4** +4.3	**64.3** +3.1
LLaVA-Next -Video	Original	7B	128	25K	67.6	68.7	62.8
	+ InfiniPot-V [4]	7B	128	6K	65.8	65.2	61.1
	+ **CoRDS (ours)**	7B	128	6K	**66.2** +0.4	**68.1** +2.9	**62.5** +1.4
	+ **CoRDS (ours)**	7B	128	3K	**66.0**	**67.8**	**62.0**
Qwen2.5-VL	Original	3B	768	50K	64.4	63.3	60.3
	+ InfiniPot-V [4]	3B	768	6K	61.8	62.1	59.3
	+ StreamMem [13]	3B	4.0/0.5 fps	6K	62.2	62.3	59.5
	+ **CoRDS (ours)**	3B	768	6K	**64.1** +1.9	**69.6** +7.3	**61.1** +1.6
	+ **CoRDS (ours)**	3B	768	3K	**63.4**	**68.3**	**61.7**
Qwen2.5 -VL	Original	7B	768	50K	62.8	68.5	64.9
	+ **CoRDS (ours)**	7B	768	6K	**66.4**	**71.5**	**65.7**
	+ **CoRDS (ours)**	7B	768	3K	**64.8**	**70.0**	**65.3**
Qwen3-VL	Original	8B	768	Full	67.9	74.8	67.4
	+ InfiniPot-V [4]	8B	768	6K	66.4	74.4	68.0
	+ **CoRDS (ours)**	8B	768	6K	**69.5** +3.1	**76.9** +2.5	**70.1** +2.1
	+ InfiniPot-V [4]	8B	768	3K	63.4	72.8	66.9
	+ **CoRDS (ours)**	8B	768	3K	**66.4** +3.0	**76.6** +3.8	**68.4** +1.5
Qwen3.5-VL	Original	9B	768	Full	68.7	76.5	68.9
	+ InfiniPot-V [4]	9B	768	6K	67.9	75.9	69.2
	+ **CoRDS (ours)**	9B	768	6K	**70.0** +2.1	**77.7** +1.8	**69.5** +0.3
	+ InfiniPot-V [4]	9B	768	3K	65.2	74.2	67.9
	+ **CoRDS (ours)**	9B	768	3K	**67.8** +2.6	**77.4** +3.2	**68.9** +1.0
Qwen3.5-VL	Original	27B	768	Full	73.8	78.4	73.6
	+ InfiniPot-V [4]	27B	768	6K	73.6	77.4	73.1
	+ **CoRDS (ours)**	27B	768	6K	**75.0** +1.4	**79.6** +2.2	**73.9** +0.8
	+ InfiniPot-V [4]	27B	768	3K	73.4	77.2	72.7
	+ **CoRDS (ours)**	27B	768	3K	**73.8** +0.4	**77.9** +0.7	**73.3** +0.6

Table 1: **Offline video understanding benchmarks.** Green numbers indicate the absolute improvement over the closest compressed competitor under the same backbone and cache budget.

that gains come from *which* tokens are kept rather than how many. Per-ratio numerical results across all backbones are reported in Appendix B, Table 6.

4.6 Qualitative Analysis

We further assess whether the retained cache is *representative* of the full accumulated KV cache, beyond downstream accuracy. For each video, we first run the model without compression to obtain a

Table 2: Integrated comparison on OVO-Bench and StreamingBench using `Qwen2-VL-7B` and`Qwen3-VL-8B` under a 6K compression budget. Qwen2-VL-7B aggregates are taken from the OVO-Bench[21].

	Backward				Real-time							Forward				
Method	*ASI*	*EPM*	*HLD*	**Avg**	*ACR*	*ATR*	*FPD*	*OCR*	*OJR*	*STU*	**Avg**	*REC*	*CRR*	*SSR*	**Avg**	**Overall**
Qwen2-VL-7B	–	–	–	46.46	–	–	–	–	–	–	55.98	–	–	–	48.74	50.39
+InfiniPot-V	58.11	51.85	39.78	49.91	46.79	64.66	72.28	63.76	57.61	44.38	60.94	32.38	58.75	65.98	49.18	53.34
+CoRDS (ours)	58.69	53.06	39.64	**50.46**	47.26	64.43	73.00	64.40	55.98	42.55	**61.55**	32.85	55.55	65.67	**49.26**	**53.76**
Qwen3-VL-8B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
+InfiniPot-V	67.57	51.85	11.29	43.57	60.55	71.55	66.34	77.18	59.78	53.37	64.80	43.70	53.75	66.61	54.69	54.35
+CoRDS (ours)	68.56	53.50	12.12	**44.73**	60.19	71.57	65.97	77.05	60.17	57.01	**65.33**	45.08	55.53	67.63	**56.08**	**55.38**

StreamingBench — Real-time Visual Understanding

Method	**CS**	**OR**	**AR**	**PR**	**ActR**	**SU**	**EU**	**Cnt**	**TR**	**CaR**	**Avg**
Qwen2-VL-7B	–	–	–	–	–	–	–	–	–	–	–
InfiniPot-V	84.36	77.78	79.91	74.44	68.40	65.54	77.88	38.82	74.39	77.68	72.75
CoRDS (ours)	85.97	81.02	78.79	81.92	67.82	63.96	77.82	40.93	72.88	80.00	**73.43**
Qwen3-VL-8B	–	–	–	–	–	–	–	–	–	–	–
InfiniPot-V	82.56	79.73	83.06	80.56	69.33	61.51	77.48	48.21	75.32	76.98	73.47
CoRDS (ours)	83.97	81.07	84.22	84.16	69.40	64.24	77.59	52.31	76.71	76.15	**74.98**

StreamingBench codes: CS = Clips Summarize, OR = Object Recog., AR = Attribute Recog., PR = Prospective Reas., ActR = Action Recog., SU = Spatial Underst., EU = Event Underst., Cnt = Counting, TR = Text-Rich Underst., CaR = Causal Reas.

Table 3: Efficiency across varying video durations on Qwen2-VL-7B. V: peak VRAM (GB), F: prefill FPS, T: decode throughput (tok/s). Subscripts denote std over 5 runs; V is deterministic. Full KV OOMs for $L \geq 600$ s.

	$L = 300$ s			$L = 500$ s			$L = 800$ s			$L = 1000$ s		
Method	V↓	F↑	T↑	V↓	F↑	T↑	V↓	F↑	T↑	V↓	F↑	T↑
Full KV	24.4	$32.4_{\pm 0.2}$	$13.8_{\pm 0.6}$	35.9	$27.5_{\pm 0.4}$	$\mathbf{30.5}_{\pm 0.4}$	OOM			OOM		
InfiniPot-V [4]	**10.1**	$42.8_{\pm 0.6}$	$\mathbf{23.2}_{\pm 0.4}$	**10.7**	$43.9_{\pm 0.5}$	$22.0_{\pm 0.4}$	**11.6**	$43.8_{\pm 1.7}$	$20.5_{\pm 3.0}$	**11.8**	$44.4_{\pm 1.8}$	$22.2_{\pm 2.2}$
CoRDS (w/o follower)	10.2	$36.7_{\pm 0.2}$	$\mathbf{23.2}_{\pm 0.4}$	11.6	$33.2_{\pm 0.2}$	$23.5_{\pm 0.2}$	13.8	$28.4_{\pm 0.2}$	$\mathbf{23.4}_{\pm 0.3}$	14.3	$27.1_{\pm 0.2}$	$22.8_{\pm 0.2}$
CoRDS	**10.1**	$\mathbf{43.2}_{\pm 0.4}$	$22.4_{\pm 0.2}$	**10.7**	$\mathbf{45.0}_{\pm 0.3}$	$22.5_{\pm 0.2}$	**11.6**	$\mathbf{45.9}_{\pm 0.5}$	$23.2_{\pm 0.3}$	**11.8**	$\mathbf{45.9}_{\pm 0.8}$	$\mathbf{23.8}_{\pm 0.2}$

reference full KV cache, and then run each compression method under the same budget to record the retained tokens. For every full-cache token, we compute the cosine distance to its nearest retained neighbour and plot the CDF (see Fig. 4); curves further to the left indicate better coverage. To avoid bias toward our own selection objective, we report this CDF under three distance metrics: joint $K\|V$ (after per-component normalisation), K-only, and V-only. Fig. 4 shows the result for the *MLVU anomaly recognition* task with Qwen2-VL-7B at $|M|$=6K; CDFs for the remaining MLVU tasks are in Appendix B.5. Across all three metrics, CoRDS covers the full cache better than InfiniPot-V, with peak gaps of $\Delta = 0.136$ (Joint $K\|V$), 0.156 (K-only), and 0.084 (V-only) at small distances ($d \approx 0.08$–0.12). Since the improvement also appears under K-only and V-only metrics, the gain is unlikely to be an artifact of evaluating with the same metric used for selection. This provides a plausible explanation for the downstream gains in Sec. 4.2: CoRDS retains a more representative and less redundant subset of the visual KV cache.

4.7 Ablations: Selector, Feature Space, and Layer Subset

Table 4 reads left to right as controlled decisions, each locking in the previous winner (★) on a held-out MLVU PlotQA slice. **(i) Selector:** D^2 style farthest-point matches the best accuracy at the lowest wall-clock. **(ii) Feature space:** keys-only and values-only both underperform; KV-joint $[k_i; v_i]$ recovers full accuracy, indicating both subspaces carry complementary information. **(iii) Layer subset:** all-layer orthogonality fails to help (Appendix B.7), so we restrict it to a subset; *bottom 25%* is the accuracy-prioritized variant. Additional ablations on $\alpha, \eta, \lambda, \varepsilon_0$) are in Appendix B.4 (Tables 7, 8, 9, and 10, respectively), and follower configurations are covered in Appendix B.6, Table 11.

Table 4: **Design journey:** Each step locks in one choice (★) which feeds the next.

Step (i). *Selector*			**Step (ii).** *Feature space*			**Step (iii).** *Orth. compression: which layers?*		
Variant	**Acc.↑**	**Time↓**	**Variant**	**Acc.↑**	**Time↓**	**Variant**	**Acc.↑**	**Time↓**
K-means representative	60.0	1.93×	K-only (k_i)	66.0	1.00×	Top 50% layers	62.0	0.94×
Greedy shortlist	66.5	1.19×	V-only (v_i)	66.5	1.07×	Top 25% layers	74.5	0.81×
★ **D^2 style farthest-point**	70.0	1.00×	★ **KV-joint** $[k_i; v_i]$	70.0	1.08×	★ **Bottom-25% layers**	77.0	2.01×

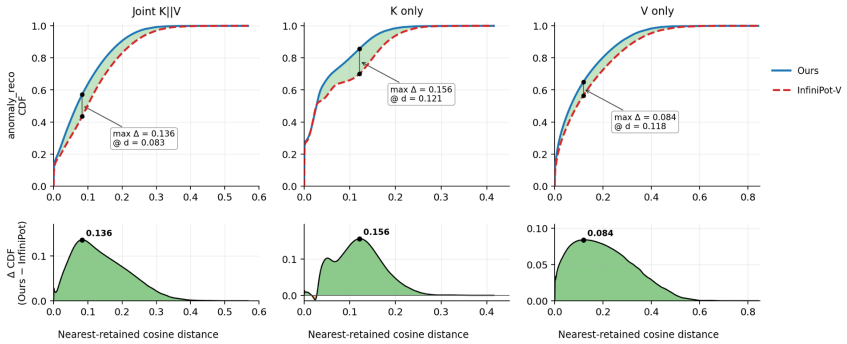

Figure 4: **Coverage CDF on MLVU anomaly recognition** (Qwen2-VL-7B, $|M|$=6K). Curves further left indicate better full-cache coverage under joint $K\|V$, K-only, and V-only metrics.

Table 5: **Ablations beyond the validation slice** (Qwen3-VL-8B; B25/T25 = bottom/top 25% layers).

	(a) Feature space			**(b) Orth. term**		**(c) Layer policy**			
Dataset	Joint K/V	K-only	V-only	λ=0	λ=0.25	CoRDS-B25	CoRDS-T25	InfiniPot-V	InfiniPot-V-B25
EgoSchema	**68.8**	68.3	67.8	71.10	**71.80**	**71.90**	71.45	65.80	64.00
VideoMME	**59.7**	58.5	58.1	63.68	**64.37**	**59.17**	58.43	51.67	51.10
StreamingBench	**73.9**	72.4	72.7	74.21	**75.75**	**75.60**	75.30	69.00	67.20

Table 5 also repeats the key ablations on Qwen3-VL-8B. (a) Joint K/V beats the best single-space variant on all three complete sets (+0.5/+1.2/+1.2). (b) Orthogonality gives a modest but consistent gain (+0.70/+0.69/+1.54), as expected for a secondary regularizer. (c) Bottom-25% beats top-25% selection, while the same policy *lowers* InfiniPot-V's accuracy; under identical layer policies CoRDS leads by 7.9–8.4 points, so its advantage stems from coreset quality and the layer schedule mainly buys efficiency.

5 Conclusion

We presented **CoRDS**, a training-free, query-agnostic KV-cache compressor that reframes streaming KV cache compression as representative coreset selection. CoRDS combines three ingredients: a joint KV representation justified by an attention-output error bound, a bicriteria D^2-style farthest-first objective balancing retrieval-side and content-side coverage, and an orthogonal anti-redundancy regularizer that discourages selecting tokens which lie in directions already spanned by the current subset. A layer-selective schedule with cross-layer cascade keeps overhead on par with existing streaming compressors. Across seven open VLMs (3B-27B) and five long-video and streaming benchmarks, CoRDS consistently outperforms heuristic baselines under matched budgets and, at aggressive compression ratios, often matches or exceeds the uncompressed Full-KV reference, indicating that *which* tokens are kept matters more than *how many*. Coverage-CDF diagnostics confirm that gains arise from a more representative and less redundant retained cache. Promising next steps include lightweight late-stage query conditioning, adaptive per-layer budgets, and extending the coreset view to audio-visual and embodied long-context agents.

Acknowledgments and Disclosure of Funding

This work was funded, in part, by the Vector Institute for AI, Canada CIFAR AI Chair, NSERC Canada Research Chair (CRC), and NSERC Discovery Grant. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, the Digital Research Alliance of Canada[3], companies[4] sponsoring the Vector Institute, and Advanced Research Computing at the University of British Columbia. Additional hardware support was provided by John R. Evans Leaders Fund CFI grant and Digital Research Alliance of Canada under the Resource Allocation Competition award.

References

[1] 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. In *Advances in Neural Information Processing Systems*, 2024. `https://arxiv.org/abs/2408.03326`.

[2] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[3] 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. *Transactions on Machine Learning Research*, 2025.

[4] Minsoo Kim, Kyuhong Shim, Jungwook Choi, and Simyung Chang. Infinipot-v: Memory-constrained kv cache compression for streaming video understanding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[5] Rui Qian, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Jiaqi Wang, and Dahua Lin. Streaming long video understanding with large language models. In *Advances in Neural Information Processing Systems*, 2024. `https://arxiv.org/abs/2405.16009`.

[6] 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. In *Forty-second International Conference on Machine Learning*, 2025.

[7] Junming Lin, Zheng Fang, Chi Chen, Haoxuan Cheng, Zihao Wan, Fuwen Luo, Ziyue Wang, Peng Li, Yang Liu, and Maosong Sun. Streamingbench: Assessing the gap for mllms to achieve streaming video understanding. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12147–12151. IEEE, 2026.

[8] Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, and Xiaojie Jin. Flash-vstream: Efficient real-time understanding for long video streams. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 21059–21069, 2025.

[9] Bhavika Suresh Devnani, Jitesh Jain, Humphrey Shi, and Judy Hoffman. Training-free adaptive frame selection for video-language understanding, 2026.

[10] Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive keyframe sampling for long video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29118–29128, 2025.

[11] Jindong Jiang, Xiuyu Li, Zhijian Liu, Muyang Li, Guo Chen, Zhiqi Li, De-An Huang, Guilin Liu, Zhiding Yu, Kurt Keutzer, et al. Storm: Token-efficient long video understanding for multimodal llms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5830–5841, 2025.

[12] Shangzhe Di, Zhelun Yu, Guanghao Zhang, Haoyuan Li, Hao Cheng, Bolin Li, Wanggui He, Fangxun Shu, Hao Jiang, et al. Streaming video question-answering with in-context video kv-cache retrieval. In *ICLR*, 2025.

[3] `https://alliance.can.ca`

[4] `https://vectorinstitute.ai/#partners`

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

[14] 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.

[15] Qwen Team. Qwen3-VL technical report. *arXiv:2511.21631*, 2025.

[16] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026.

[17] Yuanhan Zhang, Bo Li, Haotian Liu, Yong Jae Lee, Liangke Gui, Di Fu, Jiashi Feng, Ziwei Liu, and Chunyuan Li. LLaVA-NeXT: A strong zero-shot video understanding model. `https://llava-vl.github.io/blog/2024-04-30-llava-next-video/`, April 2024.

[18] 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) Datasets and Benchmarks Track*, 2023.

[19] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Shitao Xiao, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. MLVU: A comprehensive benchmark for multi-task long video understanding. *arXiv preprint arXiv:2406.04264*, 2024.

[20] Chaoyou Fu, Yuhan Dai, Yondong 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, Rongrong Ji, and Xing Sun. Video-MME: The first-ever comprehensive evaluation benchmark of multi-modal LLMs in video analysis. *arXiv preprint arXiv:2405.21075*, 2024.

[21] Junbo Niu, Yifei Li, Ziyang Miao, Chunjiang Ge, Yuanhang Zhou, Qihao He, Xiaoyi Dong, Haodong Duan, Shuangrui Ding, Rui Qian, et al. Ovo-bench: How far is your video-llms from real-world online video understanding? In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18902–18913, 2025.

[22] Zijia Zhao, Haoyu Lu, Yuqi Huo, Yifan Du, Tongtian Yue, Longteng Guo, Bingning Wang, Weipeng Chen, and Jing Liu. Needle in a video haystack: A scalable synthetic evaluator for video mllms. In *International Conference on Learning Representations*, 2025.

[23] Yi Wang, Xinhao Li, Ziang Yan, Yinan He, Jiashuo Yu, Xiangyu Zeng, Chenting Wang, Changlian Ma, Haian Huang, Jianfei Gao, Min Dou, Kai Chen, Wenhai Wang, Yu Qiao, Yali Wang, and Limin Wang. InternVideo2.5: Empowering video MLLMs with long and rich context modeling. *arXiv:2501.12386*, 2025.

[24] Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, and Lidong Bing. VideoLLaMA 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv:2406.07476*, 2024.

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

[26] Natan Bagrov, Eugene Khvedchenia, Borys Tymchenko, Shay Aharon, Lior Kadoch, Tomer Keren, Ofri Masad, Yonatan Geifman, Ran Zilberstein, Tuomas Rintamaki, et al. Efficient video sampling: Pruning temporally redundant tokens for faster vlm inference. *arXiv preprint arXiv:2510.14624*, 2025.

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

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

[29] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22857–22867, 2025.

[30] Ruyi Xu, Guangxuan Xiao, Yukang Chen, Liuning He, Kelly Peng, Yao Lu, and Song Han. StreamingVLM: Real-time understanding for infinite video streams. In *The Fourteenth International Conference on Learning Representations*, 2026.

[31] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

[32] 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 Computer Vision and Pattern Recognition Conference*, pages 18992–19001, 2025.

[33] 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.

[34] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978.

[35] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.

[36] Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Firoj Alam. On the transformation of latent space in fine-tuned NLP models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022.

[37] Zheng Zhao, Yftah Ziser, and Shay B. Cohen. Layer by layer: Uncovering where multi-task learning happens in instruction-tuned large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

[38] Namuk Park, Wonjae Kim, Byeongho Heo, Taekyung Kim, and Sangdoo Yun. What do self-supervised vision transformers learn? In *International Conference on Learning Representations*, 2023.

[39] Zefan Cai, Wen Xiao, Hanshi Sun, Cheng Luo, Yikai Zhang, Ke Wan, Yucheng Li, Yeyang Zhou, Li-Wen Chang, Jiuxiang Gu, Zhen Dong, Anima Anandkumar, Abedelkadir Asi, and Junjie Hu. R-KV: Redundancy-aware KV cache compression for reasoning models. In *Advances in Neural Information Processing Systems*, 2025.

A Theory and Proofs

This appendix provides the formal statements and proofs underlying two ingredients of our method: (i) the bicriteria coverage objective that we minimise as a tractable surrogate for the worst-case attention-output deviation of the compressed cache (Proposition 1), and (ii) the log-determinant guarantee that motivates the orthogonal anti-redundancy term (Theorem 2).

A.1 Notation and Setup

Let $\mathcal{C} = \{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^N$ with $\mathbf{k}_i \in \mathbb{R}^{d_k}$, $\mathbf{v}_i \in \mathbb{R}^{d_v}$. The attention output for query $\mathbf{q} \in \mathbb{R}^{d_k}$ is

$$\mathrm{Attn}(\mathbf{q}, \mathcal{C}) = \sum_{i=1}^{N} a_i(\mathbf{q})\, \mathbf{v}_i, \qquad a_i(\mathbf{q}) = \frac{\exp(\mathbf{q}^\top \mathbf{k}_i / \sqrt{d_k})}{\sum_{j=1}^{N} \exp(\mathbf{q}^\top \mathbf{k}_j / \sqrt{d_k})}, \tag{11}$$

so $a_i(\mathbf{q}) \geq 0$ and $\sum_i a_i(\mathbf{q}) = 1$ for every $\mathbf{q}$. Let $\mathcal{S} \subset [N]$ with $|\mathcal{S}| = b$ be a selected subset, and define the *cluster-weighted* compressed attention: for each $j \in \mathcal{S}$ let $\mathcal{C}_j = \{i : j^\star(i) = j\}$ where $j^\star(i) = \arg\min_{j' \in \mathcal{S}} \|\boldsymbol{\phi}_i - \boldsymbol{\phi}_{j'}\|$ is the nearest representative of token i in the joint feature space $\boldsymbol{\phi}_i = [\mathbf{k}_i; \mathbf{v}_i] \in \mathbb{R}^{d_k + d_v}$. Assign cluster weight $w_j(\mathbf{q}) = \sum_{i \in \mathcal{C}_j} a_i(\mathbf{q})$, so that the compressed attention output is

$$\mathrm{Attn}_w(\mathbf{q}, \mathcal{S}) = \sum_{j \in \mathcal{S}} w_j(\mathbf{q})\, \mathbf{v}_j. \tag{12}$$

This is the natural cluster-centroid approximation: every token i contributes its attention weight to its nearest selected representative.

A.2 Attention Approximation Bound

Proposition 1 (Attention output error is bounded by the bicriteria quantization error)**.** *For any query* $\mathbf{q}$ *and any subset* $\mathcal{S}$ *with cluster assignment* $j^\star$ *defined above,*

$$\left\|\mathrm{Attn}(\mathbf{q}, \mathcal{C}) - \mathrm{Attn}_w(\mathbf{q}, \mathcal{S})\right\|_2 \;\leq\; \sqrt{\sum_{i=1}^{N} a_i(\mathbf{q}) \cdot \min_{j \in \mathcal{S}} \|\mathbf{v}_i - \mathbf{v}_j\|^2}\,. \tag{13}$$

Moreover, since $\min_{j \in \mathcal{S}} \|\mathbf{v}_i - \mathbf{v}_j\|^2 \leq \frac{1}{1-\alpha}\, d_\alpha(i, \mathcal{S})$ *for any* $\alpha \in [0, 1)$*, the bound is controlled by the bicriteria quantization surrogate of Eq.* (4) *in the main text:*

$$\left\|\mathrm{Attn}(\mathbf{q}, \mathcal{C}) - \mathrm{Attn}_w(\mathbf{q}, \mathcal{S})\right\|_2 \;\leq\; \sqrt{\frac{1}{1-\alpha} \sum_{i=1}^{N} a_i(\mathbf{q}) \cdot d_\alpha(i, \mathcal{S})}\,. \tag{14}$$

In particular, the right-hand side depends on both *the key-space and value-space quantization errors of* $\mathcal{S}$*, which is what motivates joint* $(\mathbf{k}, \mathbf{v})$ *selection in the main text.*

Proof. Write the error as a weighted sum over clusters:

$$\begin{aligned}
\mathrm{Attn}(\mathbf{q}, \mathcal{C}) - \mathrm{Attn}_w(\mathbf{q}, \mathcal{S}) &= \sum_{i=1}^{N} a_i(\mathbf{q})\, \mathbf{v}_i - \sum_{j \in \mathcal{S}} \Big(\sum_{i \in \mathcal{C}_j} a_i(\mathbf{q}) \Big) \mathbf{v}_j \\
&= \sum_{i=1}^{N} a_i(\mathbf{q}) \left(\mathbf{v}_i - \mathbf{v}_{j^\star(i)} \right).
\end{aligned} \tag{15}$$

Taking the ℓ_2 norm and applying the triangle inequality gives

$$\left\|\mathrm{Attn}(\mathbf{q}, \mathcal{C}) - \mathrm{Attn}_w(\mathbf{q}, \mathcal{S})\right\|_2 \leq \sum_{i=1}^{N} a_i(\mathbf{q})\, \|\mathbf{v}_i - \mathbf{v}_{j^\star(i)}\|_2. \tag{16}$$

Cauchy–Schwarz with the probability weights $a_i(\mathbf{q})$ (which sum to one) yields

$$\sum_{i=1}^{N} a_i(\mathbf{q}) \, \|\mathbf{v}_i - \mathbf{v}_{j^\star(i)}\|_2 \le \sqrt{\sum_{i=1}^{N} a_i(\mathbf{q})} \cdot \sqrt{\sum_{i=1}^{N} a_i(\mathbf{q}) \, \|\mathbf{v}_i - \mathbf{v}_{j^\star(i)}\|_2^2} = \sqrt{\sum_{i=1}^{N} a_i(\mathbf{q}) \, \|\mathbf{v}_i - \mathbf{v}_{j^\star(i)}\|_2^2}, \tag{17}$$

where the last equality uses $\sum_i a_i(\mathbf{q}) = 1$. Since $\|\mathbf{v}_i - \mathbf{v}_{j^\star(i)}\|^2 = \min_{j \in \mathcal{S}} \|\mathbf{v}_i - \mathbf{v}_j\|^2$ by definition of $j^\star$, this establishes Eq. (13).

For Eq. (14), observe that

$$d_\alpha(i, \mathcal{S}) = \alpha \min_{j \in \mathcal{S}} \|\mathbf{k}_i - \mathbf{k}_j\|^2 + (1-\alpha) \min_{j \in \mathcal{S}} \|\mathbf{v}_i - \mathbf{v}_j\|^2 \;\ge\; (1-\alpha) \min_{j \in \mathcal{S}} \|\mathbf{v}_i - \mathbf{v}_j\|^2, \tag{18}$$

so $\min_{j \in \mathcal{S}} \|\mathbf{v}_i - \mathbf{v}_j\|^2 \le d_\alpha(i, \mathcal{S})/(1-\alpha)$. Substituting into Eq. (13) gives Eq. (14), in which the right-hand side mixes both $\|\mathbf{k}_i - \mathbf{k}_j\|^2$ and $\|\mathbf{v}_i - \mathbf{v}_j\|^2$ contributions through d_α. □

A.3 Log-Determinant Guarantee for Greedy Orthogonalization

We now justify the orthogonal anti-redundancy term used in Eq. (9) of the main text. The greedy rule that selects the candidate with the largest residual norm is not an ad-hoc heuristic: it is exactly equivalent to maximising the marginal gain of a log-determinant coverage objective, which is monotone submodular and therefore admits a $(1 - e^{-1})$ approximation guarantee.

Theorem 2 (Greedy orthogonalization achieves a $(1 - e^{-1})$ log-det approximation)**.** *Define the log-determinant coverage of selected keys as $\mathcal{F}(\mathcal{S}) = \log\det\big(\mathbf{U}_\mathcal{S}^\top \mathbf{U}_\mathcal{S} + \epsilon I\big)$ where $\mathbf{U}_\mathcal{S} = [\mathbf{k}_j]_{j \in \mathcal{S}}$ and $\epsilon > 0$ is a small regularizer. The greedy rule $i_{t+1} = \arg\max_{i \notin \mathcal{S}_t} \|\mathbf{r}_i\|^2$ with $\mathbf{r}_i$ defined as in Eq.* (8) *of the main text produces a set $\hat{\mathcal{S}}_b$ satisfying*

$$\mathcal{F}(\hat{\mathcal{S}}_b) \;\ge\; \big(1 - e^{-1}\big) \, \mathcal{F}(\mathcal{S}_b^\star), \tag{19}$$

where $\mathcal{S}_b^\star = \arg\max_{|\mathcal{S}|=b} \mathcal{F}(\mathcal{S})$.

Proof. We show (i) the marginal gain of adding token i equals $\log(1 + \|\mathbf{r}_i\|^2 / c_t)$ for a positive scalar c_t, so the greedy rule maximizing $\|\mathbf{r}_i\|^2$ also maximizes marginal $\mathcal{F}$-gain; and (ii) $\mathcal{F}$ is monotone submodular, so the greedy maximizer achieves the $(1 - e^{-1})$ bound of [34].

(i) Marginal gain equals residual norm. Partition the Gram matrix after adding token i to $\mathcal{S}_t$ as

$$\mathbf{U}_{t+1}^\top \mathbf{U}_{t+1} = \begin{bmatrix} \mathbf{U}_t^\top \mathbf{U}_t & \mathbf{U}_t^\top \mathbf{k}_i \\ \mathbf{k}_i^\top \mathbf{U}_t & \|\mathbf{k}_i\|^2 \end{bmatrix}.$$

By the matrix determinant lemma (Schur complement of the (t, t) block):

$$\begin{aligned} &\log\det\big(\mathbf{U}_{t+1}^\top \mathbf{U}_{t+1} + \epsilon I\big) - \log\det\big(\mathbf{U}_t^\top \mathbf{U}_t + \epsilon I\big) \\ &\quad = \log\Big(\|\mathbf{k}_i\|^2 + \epsilon - \mathbf{k}_i^\top \mathbf{U}_t \big(\mathbf{U}_t^\top \mathbf{U}_t + \epsilon I\big)^{-1} \mathbf{U}_t^\top \mathbf{k}_i\Big) - \log \epsilon \\ &\quad = \log\Big(1 + \tfrac{1}{\epsilon} \, \mathbf{k}_i^\top \big(I - P_t^{K,\epsilon}\big) \mathbf{k}_i\Big), \end{aligned} \tag{20}$$

where $P_t^{K,\epsilon} = \mathbf{U}_t(\mathbf{U}_t^\top \mathbf{U}_t + \epsilon I)^{-1} \mathbf{U}_t^\top$ is the regularized projector that converges to P_t^K as $\epsilon \to 0$. The quadratic form $\mathbf{k}_i^\top (I - P_t^{K,\epsilon}) \mathbf{k}_i = \Big\|(I - P_t^{K,\epsilon}) \mathbf{k}_i\Big\|^2$ is the squared regularized residual, which equals $\|\mathbf{r}_i\|^2$ up to an $O(\epsilon)$ term. Since $\log(1 + x/\epsilon)$ is strictly monotone increasing in $x > 0$, the greedy rule maximizing $\|\mathbf{r}_i\|^2$ at each step also maximizes the exact marginal $\mathcal{F}$-gain.

(ii) Submodularity of $\mathcal{F}$. For any $A \subseteq B$ and token $i \notin B$, the marginal gain $\mathcal{F}(A \cup \{i\}) - \mathcal{F}(A) \ge \mathcal{F}(B \cup \{i\}) - \mathcal{F}(B)$ holds because the residual of $\mathbf{k}_i$ with respect to a larger selected set can only decrease (the span of B contains the span of A, so more of $\mathbf{k}_i$ is already projected out). Formally, if

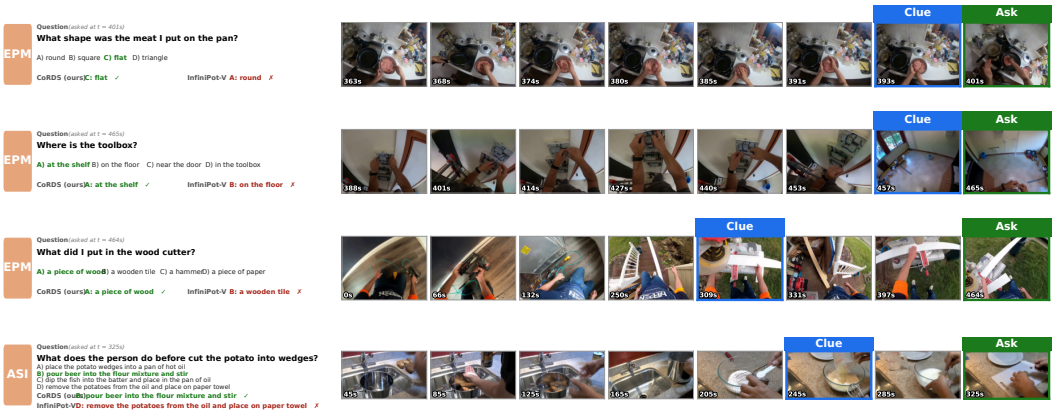

Figure 5: Backward qualitative examples on EPM/ASI: CoRDS (ours) recovers the clue laid down hundreds of seconds before the question, while InfiniPot-V drops it.

$A \subseteq B$ then $P_A \preceq P_B$ in the Loewner order, hence $(I - P_B) \preceq (I - P_A)$ and the Schur complement in Eq. (20) is non-increasing as the conditioned set grows. Monotonicity $\mathcal{F}(A) \leq \mathcal{F}(B)$ for $A \subseteq B$ follows directly from the positive semi-definiteness of the Gram matrix increment. By the greedy submodularity theorem of [34], any monotone submodular function subject to a cardinality constraint is $(1 - e^{-1})$-approximated by the greedy maximizer, yielding Eq. (19). □

B Extended Experimental Results

This appendix expands the experimental section of the main paper. Appendix B.1 gives the full implementation and reproducibility details; Appendix B.2 gives per-task breakdowns of the streaming results; Appendix B.3 reports the per-backbone, per-budget numbers; Appendix B.4 ablates the four scalar hyperparameters $(\alpha, \eta, \lambda, \varepsilon_0)$; Appendix B.7 extends the design ablations of Section 4.7 to the full sweep; Appendix B.5 provides per-task coverage CDFs; Appendix B.6 extends the memory–throughput sweep; Appendix B.8 covers the cross-layer cascade analysis; Appendix C.1 gives per-task streaming results with uncertainties on Qwen3-VL-8B; Appendix C.2 quantifies query-time compression pressure per benchmark; and Appendix C.3 analyzes why compression can outperform Full KV.

B.1 Implementation Details

All experiments are run on a single NVIDIA L40S 48 GB GPU with FlashAttention-2 [35], PyTorch 2.3, `transformers` 4.45, and `bfloat16` weights and activations. Wall-clock results are reported as mean per-sample times over the corresponding evaluation slice. We evaluate four public VLMs without fine-tuning: Qwen2-VL-7B [2], Qwen2.5-VL-3B/7B [14], and LLaVA-NeXT-Video-7B [17]. We use each model's standard sampling setting, with 768 frames and approximately 50K tokens for Qwen2/2.5-VL, and 128 frames and approximately 25K tokens for LLaVA-NeXT-Video. Visual tokens are fed causally, and the question is appended only after compression.

The compressor uses compressed frames to block size ratio of 0.75, and a recent tail of $|\mathcal{R}| = \lfloor |M|/4 \rfloor$. The selection rule in Eq. 10 is fixed at $(\alpha, \eta, \lambda, \varepsilon_0) = (0.25,\ 0.25,\ 0.25,\ 10^{-6})$ and $\beta = 0.15$, with min-max normalization. Compression is active on the bottom 25% of decoder layers, while layers outside the active set reuse the most recent active-layer selection; hence every layer holds a compressed cache bounded by $|M|$, and the full selector is recomputed only in the bottom 25% of layers. Reselection is performed once per streaming block over the union of the previous coreset and the tokens leaving the recent tail (Sec. 3). The same fixed configuration is used, without model- or dataset-specific tuning, for Qwen3-VL-8B [15], Qwen3.5-VL-9B, and Qwen3.5-VL-27B [16], which are evaluated under the identical causal, query-agnostic protocol. Decoding uses greedy generation with each model's default chat template and a 256-token cap for multiple-choice benchmarks. The held-out 200-sample MLVU PlotQA slice used for Table 4 and per-step ablations is fixed across runs, as are all random seeds.

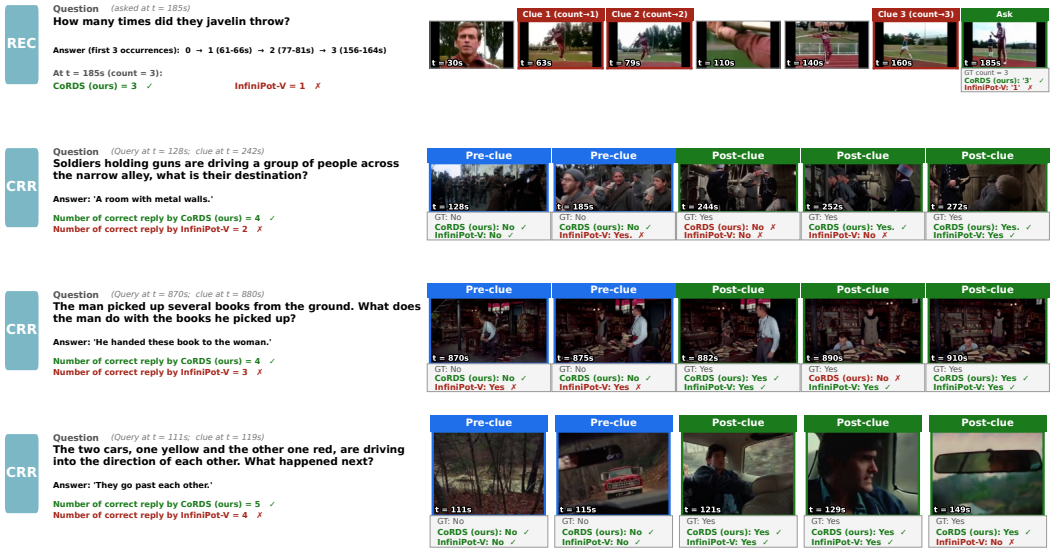

Figure 6: Forward qualitative examples on REC/CRR: CoRDS (ours) accumulates the three javelin-throw occurrences and tracks pre-/post-clue answers across time, while InfiniPot-V undercounts and answers affirmatively before the clue.

B.2 Qualitative Streaming Examples

Figures 5, 7, and 6 show qualitative examples from the OVO-Bench Backward (BW), Real-time (RT), and Forward (FW) tracks. We compare CoRDS against InfiniPot-V on Qwen2-VL-7B at a 6K compression budget.

Backward (Fig. 5). The EPM and ASI examples test long-range episodic recall. The question is asked hundreds of seconds after the relevant clue frame, so the cache must preserve evidence far beyond the recent-tail window. For example, in "What shape was the meat I put on the pan?" asked at $t{=}401$ s, with the clue at $t{=}393$ s, CoRDS retains the flat-meat clue and answers *flat*, while InfiniPot-V drops it and answers *round*. The same pattern appears for "Where is the toolbox?" at $t{=}465$ s, "What did I put in the wood cutter?" at $t{=}464$ s with the clue at $t{=}309$ s, and the ASI cooking-procedure question at $t{=}325$ s. In the latter case, InfiniPot-V selects a later distractor step, *remove the potatoes from the oil*, whereas CoRDS preserves the true antecedent step, *pour beer into the flour mixture and stir*. In all cases, the decisive clue lies outside a short temporal window, and the coverage-driven selection in CoRDS preserves the moment needed to disambiguate the answer.

Real-time (Fig. 7). RT chunks end at the ask time, so the relevant clue often coincides with the most recent visual evidence. The examples span the STU, ATR, ACR, and OCR sub-tracks. CoRDS correctly reads fine-grained on-screen content, including the player number 14, the animal label *Porcupine*, the final odds 5, the spoon count 7, and the character styling *red hue*. It also tracks short-horizon actions, such as *puts the bottle on the table*, *picked up the clothes*, and *Back*. By contrast, InfiniPot-V selects visually plausible but incorrect distractors in each case, such as *Domestic Goat* for the 60–90 cm height range, *32* for the free-throw shooter, and *misty overlay* for the red-hued character. These examples suggest that, even when the decisive evidence is recent, CoRDS preserves a cleaner trace of the relevant frames instead of over-pruning them in favor of generic context.

Forward (Fig. 6). FW examples test accumulation and clue-conditioned recall. In the REC "javelin throw" case, asked at $t{=}185$ s, the model must count three occurrences at 61–66 s, 77–81 s, and 156–164 s. CoRDS reports the correct count of 3, while InfiniPot-V collapses to 1 after pruning the two earlier occurrences. The three CRR examples evaluate the same question at successive time points around a clue, where pre-clue probes should answer *No* and post-clue probes should answer *Yes*. Across the three queries, CoRDS scores 4/5, 4/5, and 5/5, compared with 2/5, 3/5, and 4/5 for InfiniPot-V. InfiniPot-V's errors are concentrated in pre-clue frames, where it hallucinates affirmative

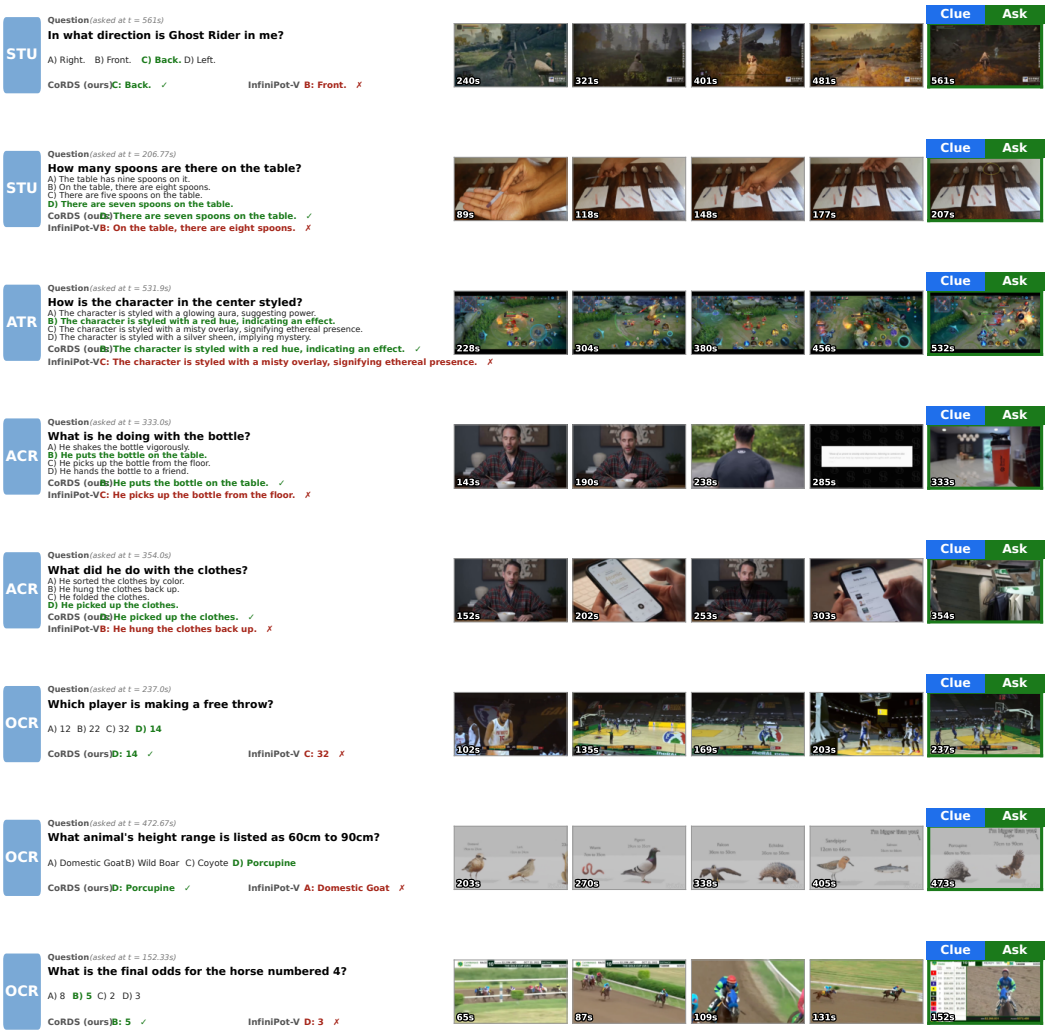

Figure 7: Real-time qualitative examples on STU/ATR/ACR/OCR: with the clue at the ask frame, CoRDS (ours) reads fine-grained on-screen content and short-horizon actions where InfiniPot-V picks plausible distractors.

answers, such as asserting the soldiers' destination or the books' fate before the clue is observed. This behavior is consistent with weaker temporal grounding in the retained cache.

B.3 Per-Backbone, Per-Budget Results

Table 6 reports the full budget sweep $|M| \in \{3\text{K}, 6\text{K}, 12\text{K}, 24\text{K}\}$ for Qwen2-VL-7B against five baselines, organized along the streaming versus non-streaming and query-aware versus query-agnostic axes. CoRDS already reaches the Full KV reference at $|M|$=3K, with an average score of 64.31 compared with 62.85 for Full KV, and remains nearly flat from 3K onward. In contrast, each baseline continues to gain approximately 1–2 points from 3K to 24K, suggesting that they retain redundant tokens that our selector filters out. Moving from Case 2, non-streaming and query-agnostic, to Case 3, streaming and query-agnostic, costs baselines approximately 2–4 points, for example Uniform drops from 60.73 to 56.98. The same shift costs CoRDS only approximately 0.6 points, since query-agnostic coreset selection is naturally compatible with streaming and never accesses the question. The qualitative ordering CoRDS $\geq$ Full KV $>$ InfiniPot-V $>$ StreamMem $>$ STC $\approx$ TTC observed on Qwen2-VL-7B is consistent across Qwen2.5-VL-3B/7B, and LLaVA-NeXT-Video-7B in Table 1. The smallest gap appears on LLaVA-NeXT-Video-7B at $|M|$=3K, corresponding to approximately

Table 6: Offline long video understanding evaluation results under memory-constrained scenario (case 3), with MC (Memory-Constrained) and QA (Query-Agnostic) conditions marked. Results are reported on (1) Video-MME – Short: -3min, Medium: 3-30min, Long: 30min-2h, and (2) MLVU – Holistic, Single-Detail, Multi-Detail LVU.

Case	Streaming		Compression Method	Prefill Budget	Decoding Budget	VideoMME				MLVU			
	MC	QA				Short	Medium	Long	Avg.	Holistic	Single	Multi	Avg.
Qwen2-VL-7B													
–	–	–	Full KV	50K	50K	74.68	62.11	55.00	63.93	76.34	73.91	43.29	65.85
Case 1	✗	✗	FastV [4] (L=2)	48/3K (R=2.8)		54.11	50.11	48.67	50.96	69.59	59.40	33.84	55.01
				48/6K (R=5.8)		59.67	54.55	50.78	55.00	72.00	64.08	33.47	57.60
			SnapKV [24]	50K	3K	74.00	61.00	54.22	63.07	77.08	67.49	39.07	61.21
				50K	6K	74.22	60.55	54.33	63.03	77.59	73.91	42.90	66.10
Case 2	✗	✓	Uniform	50K	3K	70.33	54.67	49.55	58.18	72.29	59.06	33.51	55.54
				50K	6K	72.00	58.78	52.11	60.96	77.08	67.49	39.07	62.11
			SnapKV†	50K	3K	69.00	54.00	50.67	57.89	75.88	63.48	35.35	58.99
				50K	6K	72.11	57.56	52.22	60.63	76.46	66.43	36.22	60.66
Case 3 (CKV)	✓	✓	Uniform	3K	3K	66.00	52.44	48.00	55.48	72.54	59.00	33.51	55.59
				6K	6K	72.33	53.33	48.67	58.11	72.55	62.19	33.67	57.00
				12K	12K	74.00	55.33	51.44	60.26	75.94	65.53	37.01	60.36
				24K	24K	74.22	59.22	53.22	62.22	77.22	71.10	40.78	64.18
			SnapKV‡	3K	3K	66.67	52.22	49.89	56.26	75.88	63.48	35.35	58.99
				6K	6K	72.00	55.33	51.33	59.55	76.46	66.43	36.22	60.66
				12K	12K	74.44	58.89	52.89	62.07	75.71	68.61	35.98	61.31
				24K	24K	74.22	61.00	53.78	63.00	77.66	71.82	39.90	64.37
			InfiniPot [21]	3K	3K	67.11	54.55	51.00	57.55	74.94	61.80	36.60	58.36
				6K	6K	72.89	57.33	51.33	60.52	75.02	63.18	37.30	58.50
				12K	12K	74.00	57.78	53.22	61.67	74.46	66.46	38.30	60.70
				24K	24K	74.22	60.55	53.56	62.78	76.03	71.11	40.29	63.71
			InfiniPot-V	3K	3K	73.89	57.78	51.78	61.11	77.73	70.38	43.15	64.70
				6K	6K	74.11	60.78	53.44	62.78	77.16	72.31	44.75	65.82
				12K	12K	74.22	62.68	53.89	63.59	76.90	73.41	43.97	65.99
				24K	24K	74.22	63.22	53.11	63.52	76.91	73.97	42.18	65.73
			CoRDS	3K	3K	74.73	64.46	57.51	64.31	80.55	75.82	53.29	70.23
				6K	6K	75.26	65.44	57.56	64.84	80.97	76.95	53.72	70.81
				12K	12K	75.69	65.66	56.90	64.84	80.52	77.44	53.60	71.25
				24K	24K	75.69	65.44	56.79	64.73	79.12	78.51	52.91	71.22

$8\times$ compression of its 25K-token prefill, where memory pressure is mildest. This trend supports our claim that selection quality matters most under tighter memory budgets.

B.4 Hyperparameter Ablations

The selection rule uses four scalar hyperparameters: $\alpha \in [0, 1]$ for K/V weighting in the bicriteria distance in Eq. 4, $\eta \in [0, 1]$ for K/V weighting in the orthogonal residual norm in Eq. 10, $\lambda \geq 0$ for the orthogonal-bonus weight in Eq. 10, and $\varepsilon_0 > 0$ for min-max normalization. We sweep one parameter at a time while fixing the others to $(\alpha, \eta, \lambda, \varepsilon_0)$=$(0.25, 0.25, 0.25, 10^{-6})$. All sweeps use the same held-out 200-sample MLVU PlotQA slice with Qwen2-VL-7B at $|M|$=3K.

Key/value weighting α in the bicriteria distance. Table 7 sweeps α over $\{0,\ 0.1,\ 0.25,\ 0.5,\ 0.75,\ 0.9,\ 1\}$. The endpoints $\alpha \to 1$ and $\alpha \to 0$ correspond to the K-only and V-only coreset variants in Table 4, both of which trail the joint variant by 3–5 points. Accuracy peaks on a small plateau around $\alpha \in [0.10, 0.25]$. We use α=0.25 because it consistently improves over balanced (α=0.5) and key-favoring (α=0.75) settings by 2–3 points, in line with the attention-output bound in Proposition 1, Appendix A, where value reconstruction receives stronger weight.

Table 7: **Sensitivity to α (bicriteria K/V weighting).**

α	0.00	0.10	**0.25**	0.50	0.75	0.90	1.00
Accuracy ↑	66.5	71.0	**71.5**	67.5	66.0	65.0	66.0

Key/value weighting η in the orthogonal residual. The orthogonal bonus uses the residual norm $\eta \|\mathbf{r}_i^K\|^2 + (1-\eta) \|\mathbf{r}_i^V\|^2$ in Eq. 10. Table 8 shows much lower sensitivity to η than to α: all values

in $\eta \in [0.10, 0.50]$ stay within ± 0.5 points of the default. This is expected because the bicriteria term already enforces global K/V coverage, while the orthogonal term mainly discourages redundant choices near selection boundaries. We set η=0.25 to match α.

Table 8: **Sensitivity to η (orthogonal residual K/V weighting).**

η	0.00	0.10	**0.25**	0.50	1.00
Accuracy ↑	70.0	71.5	**71.5**	71.0	69.0

Orthogonal-bonus weight λ. λ balances bicriteria coverage with orthogonal anti-redundancy. Table 9 shows that λ=0 recovers the bicriteria-only setting, consistent with Step (iii) in Table 4 and the "Bicriteria only (no orth-bonus)" row in Table 15. Increasing λ is useful up to a point, but λ=1 reduces accuracy by about 2 points because the orthogonal term begins to favor geometrically novel but task-irrelevant tokens. The best region is broad, with similar accuracy for $\lambda \in [0.1, 0.5]$ and a slight peak at λ=0.25.

Table 9: **Sensitivity to λ (orthogonal-bonus weight).**

λ	0.00	0.10	**0.25**	0.50	0.75	1.00
Accuracy ↑	71.5	71.5	**72.0**	71.5	70.5	69.5

Min-max normalization regularizer ε_0. ε_0 prevents instability when $\max_j s_j - \min_j s_j$ is close to zero. Table 10 shows identical accuracy for values in $[10^{-8}, 10^{-4}]$, indicating that the rule is insensitive to this parameter under `bfloat16`. We keep ε_0=10^{-6} as a conservative default for degenerate candidate pools.

Table 10: **Sensitivity to ε_0 (min-max regularizer).**

ε_0	10^{-8}	$\mathbf{10^{-6}}$	10^{-4}	10^{-2}	10^{-1}
Accuracy ↑	71.5	**71.5**	71.5	71.0	68.5

Among the four parameters, α is the main one that needs tuning. The choices of η and λ are stable across broad ranges, and ε_0 mainly serves numerical stability.

B.5 Coverage CDF: Per-Task Analysis

Figure 8 extends the single-task CDF analysis of Figure 4 to all eight MLVU tasks under joint $K\|V$, K-only, and V-only metrics, with insets reporting $\Delta\mathrm{CDF}_{\max}$ and the distance d^* at which it occurs. The eight tasks fall into three regimes: *strong-coverage* (PlotQA, Anomaly, Topic, Order), where our CDF strictly dominates InfiniPot-V across all three metrics with $\Delta\mathrm{CDF}_{\max} \in [0.10, 0.16]$ localized at small d ($d^* \approx 0.08$–0.12, the regime where reconstruction fidelity drives accuracy); *comparable* (Needle, Count), where ours dominates at small d but the curves merge at large d; and *mixed* (Ego, Single), where ours is slightly worse on K-only at large d but still dominates on V-only and joint $K\|V$. The key observation is that our CDF dominates even under K-only and V-only metrics, neither of which the selector directly optimizes (it uses bicriteria d_α on the joint space) — the gain is structural, not metric-aligned, which would only show up in the joint metric. The three regimes correlate with per-task accuracy gains in Table 6: regime (a) tasks show +5–10 points over InfiniPot-V, regime (b) shows +1–3, and regime (c) shows small or near-zero gains, connecting the geometric coverage diagnostic with downstream accuracy.

B.6 Memory Footprint and Throughput

Table 11 reports the full memory, throughput, and energy breakdown across video lengths $L \in \{200, 500, 800, 1000\}$ s and across the bottom-25, bottom-50, and all-layer activation regimes, with and without the follower module. The sweep complements the headline results in Table 3 and

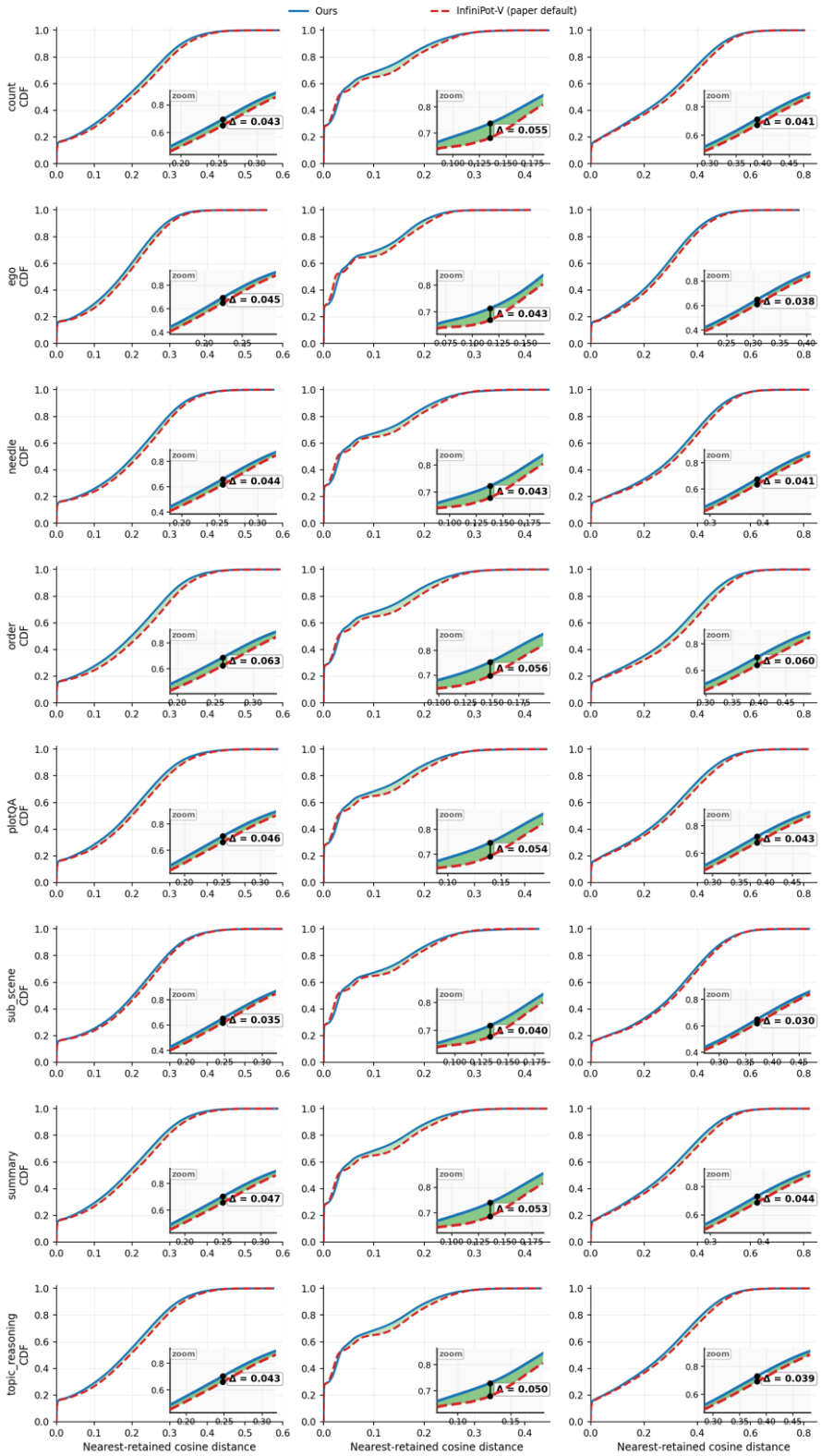

Figure 8: Coverage CDF of nearest-retained cosine distance for eight MLVU tasks under three KV-compression settings (Joint K‖V, K only, V only). Each main panel shows the full CDF; the inset in the lower-right magnifies the region around the maximum gap, with the marker line and annotation reporting $\max \Delta\text{CDF} = \text{CDF}_{\text{ours}} - \text{CDF}_{\text{InfiniPot}}$ and the distance d at which it occurs. Higher CDF at smaller d indicates better coverage of the original token set by the retained tokens.

Table 11: Effect of the follower module across all Ours variants. **Bold** marks the better of {base, + follower} within each variant pair. The all-layer variant (no follower) is shown for reference. The follower module yields a larger relative gain on the lighter bottom-25 backbone than on bottom-50, and its memory benefit grows monotonically with L. At $L = 1000$ s the source video yields ≈ 864 effective frames.

L (s)	Variant	VRAM (GB)↓	FPS↑	tok/s↑	Energy (J)↓
200	bottom-25	9.9	37.5	**23.1**	1577
	+ follower	**9.8**	**41.4**	22.1	**1363**
	bottom-50	9.9	36.4	23.1	1605
	+ follower	**9.8**	**39.2**	**23.6**	**1438**
	all-layer (ref.)	9.9	35.1	22.1	1445
500	bottom-25	11.6	33.2	**23.5**	4825
	+ follower	**10.7**	**45.0**	22.5	**3307**
	bottom-50	10.9	34.1	**23.2**	4365
	+ follower	**10.7**	**42.0**	22.4	**3394**
	all-layer (ref.)	10.8	37.0	22.5	3620
800	bottom-25	13.8	28.4	**23.4**	9133
	+ follower	**11.6**	**45.9**	23.2	**5345**
	bottom-50	12.7	30.7	**23.8**	8143
	+ follower	**11.6**	**42.8**	23.5	**5399**
	all-layer (ref.)	11.7	37.0	20.8	5802
1000	bottom-25	14.3	27.1	22.8	10376
	+ follower	**11.8**	**45.9**	**23.8**	**5806**
	bottom-50	13.1	29.6	**22.7**	9110
	+ follower	**11.8**	**42.6**	22.3	**5837**
	all-layer (ref.)	11.8	37.5	23.7	6328

shows that the follower benefit increases with video length. VRAM savings grow from 0.05 GB at L=200 s to 2.48 GB at L=1000 s, while energy savings range from 14% to 44%, with the largest gains at $L \geq 800$ s. For example, in the bottom-25 regime at L=1000 s, energy drops from 10376 J to 5806 J. The follower module is therefore most useful for long videos, where memory pressure is most severe. Prefill FPS with the follower module also increases with L, reaching 45.0 at 500 s and 45.9 at both 800 s and 1000 s, while the no-follower variants slow down from 33.2 to 28.4 and then 27.1. This pattern follows from selection amortization: per-layer selection cost grows with cache size, so applying the selector at fewer layers yields larger gains as L increases.

B.7 Selector, Feature Space, and Layer Subset: Full Sweep

Tables 12, 13, and 14 expand the three fixed choices in Table 4: selector, feature space, and orthogonal compression layer scope. Beyond the rows already reported in the main paper, Table 12 adds two further selectors: attention shortlist, which selects the top k tokens by mean attention from a probe layer before applying D^2 on the shortlist, and value norm top k. Both are widely used heuristics in the prior literature, but neither matches plain D^2 farthest point selection in accuracy: attention based shortlists collapse to a recency proxy on streaming inputs because the question is unavailable, while value norm top k ignores redundancy. Table 13 adds attention augmented $[k_i;\, v_i;\, a_i]$ and mean centered $[k_i;\, v_i - \bar{v}]$ feature spaces; neither beats plain joint KV, and the attention augmented variant additionally leaks model specific tuning into the selector through the probe layer choice.

Table 14 expands Step (iii) of Table 4 with the all layer variant referenced in Section 4.7 ("all layer orthogonality fails to help"), three parity based partitions, and a single mid layer variant. The all layer variant is the most expensive ($5.18\times$ over the single layer setting) and has the lowest accuracy among the tested configurations, for two structural reasons: late layers in decoder only VLMs encode task conditioned features, such as object identity, action, and scene information, that are sensitive to pruning, and compression in early layers cascades because every downstream layer must operate on the compressed cache, causing per layer noise to accumulate. The bottom 25% scope sidesteps both issues by leaving late layers uncompressed and amortizing early layer selection across follower layers.

Table 12: **Full selector sweep.** Feature space fixed to KV-joint. MLVU PlotQA, 200-sample slice, Qwen2-VL-7B, $|M|$=3K. Top three rows reproduce Table 4; the remaining rows are added in this appendix.

Selector (KV-joint feature space)	Accuracy ↑	Time (×) ↓
K-means representative	60.0	1.93×
Attention-shortlist	68.5	1.10×
Value-norm top-k	64.0	0.85×
Greedy shortlist	66.5	1.19×
D^2 **style farthest-point**	**70.0**	**1.00**×

Table 13: **Full feature-space sweep.** Selector fixed to D^2.

Feature space (D^2 selector)	Accuracy ↑	Time (×) ↓
K-only (k_i)	66.0	1.00×
V-only (v_i)	66.5	1.07×
Attention-augmented $[k_i;\, v_i;\, a_i]$	69.5	1.18×
Mean-centered $[k_i;\, v_i - \bar{v}]$	70.0	1.10×
KV-joint $[k_i;\, v_i]$	**70.0**	**1.08**×

In all variants, every layer is compressed to the same budget $|M|$; the scope only determines in which layers the full selector is *recomputed*, while the remaining layers reuse the indices of the most recent active layer. The all layer variant is the most expensive ($5.18\times$ over the single layer setting) and has the lowest accuracy among the tested configurations. A plausible explanation is that late layers in decoder-only VLMs encode more task-conditioned features, such as object identity, action, and scene information, so their cache geometry is a less suitable basis for query-agnostic selection, while lower layers retain more generic representations [36, 37, 38]. The bottom 25% scope therefore selects tokens from the lower-layer cache geometry, propagates the selected indices to all later layers (which are still pruned to the same budget), and amortizes selection across follower layers. Table 5(c) confirms that bottom-25% selection also beats top-25% selection on Qwen3-VL-8B, and that this policy does not by itself improve InfiniPot-V.

Table 14: **Full layer-scope sweep.** Time is normalized to the fastest setting.

Active orthogonal-compression layers	#Layers	Accuracy ↑	Time (×) ↓
All 28 layers	28	68.5	5.18×
Top 50% layers	14	62.0	1.45×
Even layers	14	70.0	5.18×
Odd layers	14	72.0	1.43×
Top 25% layers	7	74.5	1.25×
Layer 14 only (single mid)	1	73.0	**1.00**×
Bottom-25% layers (L0–L6)	7	**77.0**	3.09×

B.8 Cross-layer cascade

The bottom-25% regime is the strongest accuracy point but the slowest layer-selective regime, for an intuitive reason: when the compression decision is made in the very first layers, every subsequent layer must run on the resulting compressed cache, and any per-layer overhead from the orthogonal bonus accumulates over the rest of the network. The *cross-layer cascade* resolves this: a small number of anchor layers in $\{L_0, \ldots, L_6\}$ run the full orthogonality driven D^2 style selector, and the remaining follower layers in the same group reuse the anchor's selected indices at near-zero cost.

Four observations consolidate from the full cascade analysis (Table 15): (i) *more compression is not better* : compressing all 28 layers hurts accuracy because late layers encode task-conditioned features; (ii) *group structure matters more than pass count*. the alternating {L0, L2, L4} partition with balanced anchor→follower groups beats both denser and sparser partitions; (iii) *anchor position*

Configuration	Video-MME (n=500) Short	Med.	Long	Avg	Time ↓	MLVU$_{\text{dev}}$ (n=200) Plot	Needle	Ego	Hol.	Sing.	Avg	Time ↓
Reference (no cascade): every active layer is an anchor												
All layers (L0–L27) [REF]	64.7	53.6	52.1	56.8	1.000×	64.4	82.6	48.1	64.4	72.9	68.5	1.000×
Bottom-25% (L0–L6)	65.3	63.3	53.9	60.8	0.335×	**76.9**	82.6	48.1	**76.9**	72.9	**75.0**	0.327×
★ Last layer (L27)	**68.3**	**63.3**	**53.9**	**61.8**	**0.118**×	72.1	**84.1**	**55.6**	72.1	**76.0**	74.0	**0.112**×
Ablation 1a: Anchor placement among bottom-25% ***(active = L0–L6; pre-anchor followers run bicriteria-only, post-anchor followers do hard reuse)***												
No cascade (all 7 are anchors)	65.3	63.3	53.9	60.8	0.333×	76.9	82.6	48.1	76.9	72.9	75.0	0.330×
Anchor L0	65.3	63.9	53.3	60.8	**0.140**×	76.9	82.6	48.1	76.9	72.9	75.0	**0.134**×
★ Anchor L1	**65.9**	**65.1**	53.3	61.4	0.151×	76.9	82.6	48.1	76.9	72.9	75.0	0.143×
Anchor L3	**65.9**	**65.1**	**53.9**	**61.6**	0.174×	76.9	82.6	48.1	76.9	72.9	75.0	0.168×
Anchor L6	**65.9**	**65.1**	53.3	61.4	0.207×	76.0	82.6	**51.9**	76.0	**74.0**	75.0	0.204×
Ablation 1b: Late-layer cascade (active layers vary; followers do hard reuse)												
Active L13, L27; anchor L13	67.1	63.3	52.1	60.8	0.120×	73.1	**84.1**	51.9	73.1	75.0	74.0	**0.114**×
★ Active L26, L27; anchor L26	67.7	63.3	**53.9**	61.6	**0.115**×	**75.0**	**84.1**	**55.6**	**75.0**	**76.0**	**75.5**	**0.114**×
Active L27; anchor L27	**68.3**	63.3	**53.9**	**61.8**	0.116×	72.1	**84.1**	**55.6**	72.1	**76.0**	74.0	0.116×
Ablation 2: Cascade depth (anchor = L0; followers do hard reuse on L1 .. Lk)												
$k = 0$ (anchor only)	68.3	63.3	53.3	61.6	0.148×	75.0	**84.1**	55.6	75.0	**76.0**	75.5	0.138×
★ $k = 1$	**68.9**	63.9	**54.5**	**62.4**	0.141×	75.0	**84.1**	55.6	75.0	**76.0**	75.5	0.138×
$k = 2$	67.7	64.5	51.5	61.2	0.142×	74.0	82.6	55.6	74.0	75.0	74.5	0.135×
$k = 3$	66.5	63.3	52.1	60.6	0.141×	74.0	82.6	**59.3**	74.0	**76.0**	75.0	0.140×
$k = 4$	66.5	63.9	52.1	60.8	0.142×	76.0	82.6	**59.3**	76.0	**76.0**	**76.0**	0.137×
$k = 5$	65.9	**65.1**	53.3	61.4	0.143×	76.0	**84.1**	51.9	76.0	75.0	75.5	0.137×
$k = 6$	65.3	63.9	53.3	60.8	**0.138**×	**76.9**	82.6	48.1	**76.9**	72.9	75.0	**0.134**×
Ablation 3: Number of anchors within L0–L6 (others do hard reuse)												
1: {L0}	65.3	63.9	53.3	60.8	**0.140**×	76.9	82.6	48.1	76.9	72.9	75.0	**0.133**×
2: {L0, L3}	**65.9**	64.5	53.3	61.2	0.172×	76.0	82.6	51.9	76.0	74.0	75.0	0.168×
2: {L0, L4}	65.3	63.9	53.3	60.8	0.171×	**77.9**	82.6	48.1	**77.9**	74.0	76.0	0.170×
★ 3: {L0, L2, L4}	65.3	64.5	**53.9**	61.2	0.203×	**77.9**	**84.1**	**55.6**	**77.9**	**76.0**	**77.0**	0.201×
4: {L0, L1, L3, L5}	65.3	**65.1**	53.3	61.2	0.242×	76.9	82.6	51.9	76.9	74.0	75.5	0.232×
7: {L0–L6} (= no cascade)	65.3	63.3	**53.9**	60.8	0.337×	76.9	82.6	48.1	76.9	72.9	75.0	0.331×
Ablation 4: Follower strategy (anchor = L0; followers = L1–L6)												
Full cache: (no follower comp.; upper bound)	**68.3**	63.3	53.3	61.6	0.144×	75.0	**84.1**	**55.6**	75.0	**76.0**	**75.5**	0.138×
Bicriteria only: (no orth-bonus)	65.9	**65.7**	**53.9**	**61.8**	0.212×	**76.9**	81.2	48.1	**76.9**	71.9	74.5	0.203×
Heuristic	65.9	**65.7**	53.3	61.6	0.140×	76.0	81.2	**55.6**	76.0	74.0	75.0	0.138×
Cheap rerank	65.9	**65.7**	53.3	61.6	0.142×	76.0	82.6	51.9	76.0	74.0	75.0	**0.134**×
★ Hard reuse	65.3	63.9	53.3	60.8	**0.139**×	**76.9**	82.6	48.1	**76.9**	72.9	75.0	0.137×
Ablation 5: Decay coefficient γ (anchor = L0; followers = L1–L6)												
$\gamma = 0.00$	**65.9**	**65.7**	**53.9**	**61.8**	**0.205**×	**76.9**	81.2	48.1	**76.9**	71.9	74.5	0.202×
$\gamma = 0.10$	**65.9**	**65.7**	**53.9**	**61.8**	0.213×	**76.9**	81.2	48.1	**76.9**	71.9	74.5	**0.202**×
$\gamma = 0.25$	**65.9**	**65.7**	**53.9**	**61.8**	0.208×	**76.9**	81.2	48.1	**76.9**	71.9	74.5	**0.202**×
★ $\gamma = 0.50$	**65.9**	**65.7**	**53.9**	**61.8**	0.208×	**76.9**	81.2	48.1	**76.9**	71.9	74.5	0.204×
$\gamma = 0.75$	**65.9**	**65.7**	**53.9**	**61.8**	0.209×	**76.9**	81.2	48.1	**76.9**	71.9	74.5	0.207×
$\gamma = 0.90$	**65.9**	**65.7**	**53.9**	**61.8**	0.209×	**76.9**	81.2	48.1	**76.9**	71.9	74.5	**0.202**×
$\gamma = 1.00$	65.3	63.9	53.3	60.8	**0.139**×	**76.9**	**82.6**	48.1	**76.9**	**72.9**	**75.0**	**0.135**×

Table 15: Cascade-layer ablation on Video-MME (n=500) and MLVU$_{\text{dev}}$ (n=200) using Qwen2-VL-7B-Instruct. Anchor layers run the full selector, follower layers reuse anchor selections, and highlighted rows mark the recommended accuracy/latency trade-off.

within the group is irrelevant to accuracy (all four tested positions yield 0.750), but L0 minimizes follower overhead; and (iv) *follower strategy is nearly irrelevant*: hard index reuse, value-norm rerank, and attention-based reranking all tie on accuracy, confirming that the representative token set is locked in at the anchor step.

C Scope and Future Directions

C.1 Streaming Results on Qwen3-VL-8B: Per-Task Uncertainty and Significance

Table 16 lists the per-task Qwen3-VL-8B results summarized in Table 2, with the marginal per-method error ($\pm$) of each score. Because CoRDS and InfiniPot-V answer identical questions, the paired task-level differences (Δ) are more informative than the marginal errors; a two-sided exact Wilcoxon signed-rank test on these differences gives p=0.016 on OVO-Bench (9/12 positive), p=0.0098 on StreamingBench (9/10 positive), and p=3.0×10^{-4} when all 22 subtasks are pooled. On OVO-Bench, the Backward, Real-time, and Forward averages improve from 43.57 to 44.73, 64.80 to 65.33, and 54.69 to 56.08, respectively; all track-level differences are positive and negative subtask differences are at most 0.37 points.

Table 16: Per-task Qwen3-VL-8B results under a 6K budget ($\pm$: marginal per-method error; Δ: CoRDS $-$ InfiniPot-V).

OVO-Bench	CoRDS	InfiniPot-V	Δ
ASI	68.56 ± 3.89	67.57 ± 3.79	+0.99
EPM	53.50 ± 3.96	51.85 ± 3.60	+1.65
HLD	12.12 ± 2.18	11.29 ± 2.20	+0.83
ACR	60.19 ± 4.74	60.55 ± 4.92	-0.36
ATR	71.57 ± 5.19	71.55 ± 4.97	+0.02
FPD	65.97 ± 5.10	66.34 ± 5.31	-0.37
OCR	77.05 ± 4.00	77.18 ± 3.94	-0.13
OJR	60.17 ± 4.42	59.78 ± 3.97	+0.39
STU	57.01 ± 3.87	53.37 ± 3.68	+3.64
REC	45.08 ± 3.83	43.70 ± 3.33	+1.38
CRR	55.53 ± 3.32	53.75 ± 2.95	+1.78
SSR	67.63 ± 2.16	66.61 ± 1.96	+1.02
Overall	**55.38**	54.35	+1.03

StreamingBench	CoRDS	InfiniPot-V	Δ
Prospective Reasoning	84.16 ± 4.30	80.56 ± 4.66	+3.60
Counting	52.31 ± 6.67	48.21 ± 6.68	+4.10
Spatial Understanding	64.24 ± 3.10	61.51 ± 3.15	+2.73
Clips Summarize	83.97 ± 2.80	82.56 ± 2.89	+1.41
Text-Rich Understanding	76.71 ± 2.38	75.32 ± 2.43	+1.39
Object Recognition	81.07 ± 2.05	79.73 ± 2.10	+1.34
Attribute Recognition	84.22 ± 2.10	83.06 ± 2.16	+1.16
Event Understanding	77.59 ± 3.39	77.48 ± 3.40	+0.11
Action Recognition	69.40 ± 2.55	69.33 ± 2.55	+0.07
Causal Reasoning	76.15 ± 3.80	76.98 ± 3.75	-0.83
Macro mean	**74.98**	73.47	+1.51

C.2 Query-Time Compression Pressure

The gap between the margins in Tables 1 and 2 is primarily a context-length effect rather than a streaming-versus-offline effect, since both tables use the same causal streaming protocol. Table 17 reports the mean number of visual KV tokens held by Qwen2-VL-7B when the question arrives, and the resulting compression ratio under $|M|$=6K. OVO-Bench and StreamingBench have the lowest pressure ($2.3\times$) because their median questions arrive after about two minutes, whereas MLVU reaches $7.0\times$. Longer contexts contain more redundancy and noise, leaving more room for representative and diverse selection.

Table 17: Query-time visual KV-cache size (Qwen2-VL-7B) and effective compression ratio under $|M|$=6K.

Benchmark	KV tokens at query	Compression ratio
OVO-Bench (median 116 s)	13,673	$2.3\times$
StreamingBench (median 119 s)	14,010	$2.3\times$
EgoSchema (3-minute clips)	21,240	$3.5\times$
VideoMME (2–60 min)	32,995	$5.5\times$
MLVU (3 min–2 h)	42,055	$7.0\times$

C.3 Why Can Compression Outperform Full KV?

Two factors may contribute: removing redundant or distracting entries, and mitigating long-context degradation of the backbone. Table 6 suggests the first is primary. Full KV obtains average scores of

63.93/65.85 on VideoMME/MLVU with Qwen2-VL-7B; uniform compression to 3K reduces these to 55.48/55.59, whereas CoRDS at the same 3K budget reaches 64.31/70.23. Using fewer tokens alone therefore does not improve accuracy; *which* tokens are retained is critical. Long-context degradation may amplify the effect: CoRDS improves over Full KV on VideoMME by only $+0.05$ on Short videos (74.73 vs. 74.68), but by $+2.35$ on Medium (64.46 vs. 62.11) and $+2.51$ on Long videos (57.51 vs. 55.00). This pattern is consistent with CoRDS acting as a structured context-denoising mechanism that retains representative K/V information while removing redundancy, with larger benefits as the context grows. Similar compressed-over-full effects have been observed in prior KV-cache compression work [39]. The effect is not universal: on Qwen3.5-VL-27B at $|M|$=3K, CoRDS is on par with or slightly below Full KV (Table 1).

D Limitations and Future Directions

Limitations. The advantage of CoRDS over Full KV diminishes with model scale and is no longer positive for Qwen3.5-VL-27B at $|M|$=3K (Table 1); our central claim therefore concerns matched-budget comparisons with other compressors. In addition, hyperparameters were selected on Qwen2-VL-7B and transferred without tuning (Table 5), although accuracy remains sensitive to α, and evaluation is limited to decoder-only VLMs of up to 27B parameters. Finally, selection cost grows with video length without follower reuse (Table 3), and persistent video memory may retain privacy-sensitive content (Appendix E).

Future directions. CoRDS makes several deliberate design choices, each pointing to a natural extension. The orthogonal diversity term is grounded in a $(1 - e^{-1})$ guarantee for log-determinant subset selection (Theorem 2), which motivates the regularizer; tightening the analysis to cover the combined bicriteria-plus-orthogonal objective is an interesting theoretical direction. The layer-selective schedule and cross-layer cascade are configured on a held-out slice, which we found effective and lightweight in practice, and learned per-input or per-backbone schedules could yield further gains. For efficiency, the orthogonality term uses a max-cosine surrogate that preserves the qualitative behavior of exact span projection at the cost of a single matrix-vector product per step. CoRDS is query-agnostic by design, which is the setting required for streaming compression where queries arrive after compression; lightweight late-stage query conditioning is a complementary direction. Finally, our evaluation centers on standard multiple-choice long-video and streaming benchmarks, and extending the coreset view to open-ended generation, audio-visual reasoning, and embodied agents is a natural next step.

E Broader Impact

Streaming KV-cache compression reduces the memory and energy required to run long-video VLMs without retraining, which lowers the barrier to deploying multimodal models on consumer hardware, edge devices, and assistive technologies (e.g., wearable companions for users with visual or memory impairments) and reduces compute and carbon costs for long-video research more broadly. The same efficiency gains, however, also reduce the cost of always-on visual perception, which has dual-use implications for surveillance, persistent monitoring, and privacy in public and workplace settings. We release inference-time compression code rather than new pretrained generative models, which limits direct misuse pathways, but practitioners deploying CoRDS in always-on systems should weigh privacy, consent, and data-retention considerations alongside the efficiency benefits, and consider audit mechanisms when compressed video memory is used in safety- or rights-relevant decisions.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 state the three contributions (coreset view of KV-cache compression, joint KV bicriteria coverage, orthogonal anti-redundancy) and the empirical scope (four VLMs, five benchmarks), matching the results reported in Sections 4.2–4.7 and Appendix B.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Appendix D: the combined bicriteria + orthogonal score does not inherit the $(1 - e^{-1})$ guarantee of pure log-determinant selection; the layer-selective schedule, anchor placement, and follower count are determined empirically; the orthogonal term uses a max-cosine surrogate rather than exact span projection; the method is query-agnostic by design and is evaluated on multiple-choice video QA rather than open-ended generation.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Proposition 1 (attention-output error bound motivating joint KV selection) and Theorem 2 ($(1-e^{-1})$ guarantee for greedy log-determinant selection) are stated in Section 3 and fully proved in Appendix A, with assumptions and references to Nemhauser et al. [34] given in the proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experiment statistical significance**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Tables 1, 2, and 5 report mean accuracy with standard deviation over 3 random seeds (sampling order, frame jitter where applicable). Error bars denote 1σ across seeds, computed as the unbiased sample standard deviation. Variation across compression budgets is additionally characterized by the per-task and per-ratio sweeps in Figure 3 and Appendix B.7.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All benchmarks (EgoSchema, MLVU, VideoMME, OVO-Bench, StreamingBench) and backbone weights used are publicly available; code, configuration files, and evaluation scripts will be released at the GitHub repository upon acceptance, with anonymized access for review.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 gives high-level evaluation setup; Appendix B.1 specifies hardware, software stack (PyTorch 2.3, transformers 4.45, FlashAttention-2, bfloat16), per-model frame budgets, recent-tail size $|\mathcal{R}| = \lfloor |M|/4 \rfloor$, decoding settings, and the held-out 200-sample MLVU PlotQA slice used for ablations.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The efficiency experiments for CoRDS and the baseline methods are repeated over five independent runs at each video-context length, and we report the mean and standard deviation in Table 3. For accuracy experiments, variability across compression budgets and task settings is characterized through per-task and per-ratio sweeps in Figure 3, Table 5, and Appendix B, following common practice in this evaluation pipeline.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiments use a single NVIDIA L40S 48 GB GPU with PyTorch 2.3, transformers 4.45, FlashAttention-2, and bfloat16 (Appendix B.1). Per-length VRAM, prefill FPS, decode throughput, and energy are reported in Tables 3 and 10. The full project, including preliminary sweeps and ablations not in the final paper, consumed approximately 4624 L40S-hours.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research conforms with the NeurIPS Code of Ethics; it uses only public benchmarks and publicly released model checkpoints, involves no human subjects or new data collection, and presents an inference-time efficiency method.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Appendix E discusses positive impacts (lower memory and energy footprint enabling on-device and assistive streaming-video applications, and reduced compute barriers for long-video research) and potential negative impacts (the same efficiency gains can lower the cost of always-on visual surveillance and persistent perception in privacy-sensitive settings). We release only inference-time compression code, not new pretrained weights.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper releases an inference-time KV-cache compression algorithm and evaluation code, not new pretrained models, datasets, or generative systems with elevated misuse risk; underlying VLMs are governed by their own published licenses.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All backbones and benchmarks are cited in the references. Qwen2-VL / Qwen2.5-VL are released under the Tongyi Qianwen / Apache-2.0 license; LLaVA-NeXT-Video is released under Apache-2.0; EgoSchema, MLVU, VideoMME, OVO-Bench, and StreamingBench are used under their respective research-use licenses as specified by the original authors. License details and version IDs are listed in our released repository.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release the CoRDS implementation and evaluation scripts (footnote 1). The repository will include a README with installation instructions, an example notebook reproducing Table 1 numbers, the configuration files used in all reported experiments, and a license file.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects; all evaluation is conducted on existing public benchmarks.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human-subjects research; no IRB review was required.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Any LLM use was limited to writing assistance and does not affect the scientific content.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.