Linear Scaling Video VLMs for Long Video Understanding

Cristóbal Eyzaguirre[1], Jiajun Wu[1], and Juan Carlos Niebles[1]

Stanford University, Stanford, CA, USA
https://ceyzaguirre4.github.io/StateKV

Abstract. Video vision-language models (VLMs) are increasingly used in long-horizon and streaming settings, yet most video encoders still rely on spatiotemporal self-attention, causing compute and latency to grow quadratically with the number of frames. Existing efficiency methods improve scalability but often lose accuracy relative to full self-attention, for example through aggressive frame/token dropping or coarse attention approximations. We introduce StateKV, an inference-time method that adapts pretrained long-video VLMs to linear-time video prefill by carrying cross-frame context in a fixed-capacity, importance-based recurrent state, paired with a second full per-frame cache used for decoding. Across three long-video benchmarks and seven models spanning three families and multiple scales, StateKV remains close to full self-attention and consistently outperforms dominant sliding-window / recency-based streaming approximations, without fine-tuning or architectural changes. StateKV also reduces video-prefill cost measured FLOPs, enabling stronger accuracy at a fixed compute budget by running larger models. These results suggest a practical step toward scalable long-video understanding.

Keywords: Vision-Language Models · Long-Video Understanding

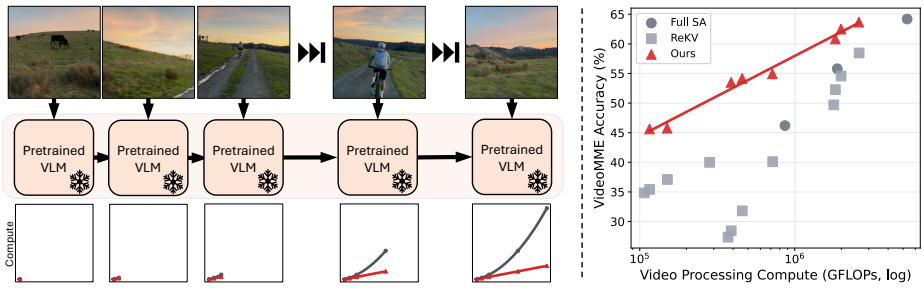

Fig. 1: Overview of StateKV. Left: a frozen pretrained VLM processes a video incrementally, one frame at a time. While StateKV maintains approximately constant marginal video-prefill compute per added frame (red), unmodified full self-attention (gray) incurs in increasing per-frame cost. This yields linear video-prefill scaling in the number of frames for StateKV, in contrast to quadratic scaling for the base model. Right: compute-accuracy frontier at 512 frames (VideoMME), where StateKV surpasses ReKV and full self-attention operating points across practical compute budgets.

1 Introduction

Modern video VLMs are increasingly vital for long-horizon and streaming tasks such as autonomous driving and embodied robotics, where models must integrate evidence over minutes or hours, often in real time. However, a central challenge for long-video VLMs is that their computational cost grows quadratically with the length of the video. Because dominant architectures allow each frame to attend over all previous video tokens, the per-frame computational cost rises as the video grows, and the overall complexity of processing the entire video scales quadratically. This creates a significant bottleneck for practical deployment and rules out real-time streaming applications entirely: a car that has driven for an hour is, in principle, harder to query than one that just started. Linear overall complexity or, equivalently, constant per-frame cost, is therefore central to scalable long-video understanding, and a critical prerequisite for streaming.

Most existing efficiency strategies focus on shrinking the input to the underlying quadratically-scaling transformer, but this often reduces information more than it reduces true complexity. Common approaches (i) subsample frames, (ii) trim visual tokens at the input layer, or (iii) compress the visual context (e.g. by compressing the KV cache). A practical limitation of these methods is that aggressive compression can substantially degrade long-video performance unless a relatively large fraction of the visual information is retained. For instance, prior work found they must keep a large token fraction (around 60%) to avoid severe degradation when using cache-compression [66]. This issue can be even more acute for frame dropping or aggressive patch trimming, which may remove temporal and spatial cues required for long-horizon reasoning. As a result, many frame/patch/token dropping methods improve efficiency mainly by redistributing quadratic cost over a shorter sequence rather than changing the scaling law.

Streaming-prefill methods offer a stronger tradeoff by avoiding compression altogether, instead separating video encoding from text generation and using heuristics to reduce the computational overhead of generating the video-only KV cache. Methods such as ReKV adapt pretrained VLMs, and rely on sliding-window-based mechanisms to process frames sequentially while constructing the video KV cache [20]. Unlike fixed-budget compression methods whose final generation context is $O(1)$ in the number of frames, these approaches usually keep all per-frame visual tokens for decoding, so generation remains $O(N)$ in the number of frames N. Although this means generation is less efficient, this tradeoff is often favorable in long-video settings because the dominant cost by far is the video-prefill stage, and its complexity is reduced from $O(N^2)$ to $O(N)$, yielding end-to-end complexity that is linear in the number of frames. Furthermore, follow-up work has explored how to reduce the latency of the *generation/query* stage on top of this streaming-prefill paradigm, making it suitable for real time applications [15, 42, 80].

While effective in many cases, recency-based heuristics are largely ad-hoc. Our contribution comes from framing streaming video prefill on frozen pretrained backbones as approximating full self-attention with a small set of tokens carried between frames. This formulation makes weaker modeling assumptions than a

strict recency window and leads to a more principled question: which information must be preserved between frames so that streaming prefill remains a good approximation to full attention? Our key observation is that long-video attention in pretrained VLMs is highly structured: most interactions are within-frame, while long-range temporal interactions often concentrate on a small set of "temporal sink" tokens that evolve slowly over time. This structure appears in the tested cases we analyze, is consistent with how VLM backbones are usually trained (image-first, then adapted to videos with variable numbers of frames), and aligns with prior work documenting attention-sink phenomena in language transformers [25, 69], on which VLMs are built. This motivates our novel approach: retain all per-frame tokens for final decoding as in prior work, but reduce the complexity of video-prefill by limiting long-range cross-frame interactions with a fixed-capacity state that identifies and preserves locally relevant tokens and temporal sinks.

We introduce StateKV, an inference-time KV-cache prefill method that adapts a frozen pretrained VLM backbone to this self-attention-approximation view of streaming video prefill. Its design follows a small set of core assumptions and is implemented via two coupled caches per layer: a fixed-capacity temporal state for cross-frame context, and a detailed per-frame cache that preserves intraframe structure. StateKV builds the video KV cache incrementally as frames arrive, then performs standard text decoding conditioned on all video tokens. This yields $O(N)$ video encoding while preserving full per-frame detail for decoding, resulting in end-to-end VideoQA complexity that is linear in the number of frames.

Empirically, StateKV delivers strong long-context results on three long-video benchmarks and consistently outperforms the dominant sliding-window / recency-based streaming approximation family used for linear-time video prefill. Across settings, it more closely matches full ($O(N^2)$) spatiotemporal attention while requiring no fine-tuning or architectural changes. Across three model families and spanning multiple parameter scales, the same trends recur: StateKV remains close to full attention and improves steadily as state capacity increases. Moreover, the reduction in measured FLOPs is dramatic enough to change what is achievable at a given compute budget: StateKV enables running larger, more accurate models for cheaper than full self-attention smaller ones.

2 Related Work

Vision language models and the long-video bottleneck. Modern vision-language models (VLMs) typically pair a strong image encoder (often CLIP-like [49]) with a large language model through a lightweight cross-modal interface, enabling open-ended visual understanding and instruction following [2, 31, 36–38, 58, 65]. Recent open and open-weight systems such as LongVA [84], InternLM-XComposer [82, 83], TimeMarker [14], InternVL2.5/3 [16, 91], Qwen-VL/Qwen2-VL/Qwen2.5-VL/Qwen3-VL [4–6, 64], Molmo [19], Eagle2.5 [11], Apollo [92], and LLaVA-style models including LLaVA-OneVision and LLaVA-Video [30, 87] further push general-purpose multimodal reasoning and support multi-image and video inputs. Extending these models to video requires aggregating information across many

frames; representative video-LLMs include Video-LLaMA [78], VideoChat [32], VideoLLaMA3 [77], Video-ChatGPT [40], LITA [26], Momentor [47], Hawk-Eye [67], and TimeChat [50], which adapt image-first backbones using temporal pooling, per-frame tokenization, and explicit temporal reasoning modules. Closed models such as Gemini, GPT-family systems, and Claude demonstrate increasingly strong fine-grained and long-context video understanding [1,3,17,44,60]. As video duration grows, the video-side prefill cost becomes the dominant constraint, motivating methods that either shrink the number of visual tokens presented to the model or change how long-range video context is processed.

Token reduction without changing asymptotic order. One line of work improves efficiency by reducing the number of visual tokens per frame or the number of frames presented to the model, while leaving the overall sequence-processing pattern unchanged. Early evidence for this family came from ATP, which studied how far single-frame selection can go in VideoQA and highlighted that many benchmarks admit surprisingly strong atemporal baselines [8]; more recent analysis in Codeplexity further argues that current VideoQA models struggle most on questions whose latent programs require integrating evidence across multiple frames [22]. At the frame level, methods based on subsampling, adaptive frame selection, or temporal search retain only a subset of frames for downstream reasoning [8, 9, 75, 76]. At the token level, prior work reduces the number of visual tokens through pooling, similarity-based merging, or resampling into a fixed set of latent queries [27, 33, 34, 71, 72, 85], others instead retain a subset of the original tokens, scored by attention or importance (sometimes query-aware) [13,24,39,70,86], while others combine these operations across both space and time [57, 59]. Recent surveys organize these methods as part of a broader multimodal token-compression literature [53]. Recent codec-aware tokenization approaches such as CoPE-VideoLM [52] also fit naturally in this family: rather than changing the temporal processing order, they reduce the number of tokens contributed by most frames by replacing dense RGB encoding with compact codec-derived representations. These approaches often provide strong practical savings, but they primarily shrink the per-frame representation; they do not directly change the growth of cross-frame attention once many frames remain in context.

Long-video and streaming video inference with changed complexity. In long videos, thousands of frames with hundreds of visual tokens per frame can yield effective sequence lengths in the millions, making both quadratic attention and the linear-in-length KV cache footprint dominant constraints. Hybrid long-video architectures such as VAMBA [51] replace expensive video-token self-attention with linear-time state-space modules to enable long video understanding, but they require architectural changes and costly training. Also requiring training, StreamingVLM [73] induces a fixed attention pattern that relies on sink anchors so real-time inference can enforce the same pattern efficiently.

Query-agnostic fixed-budget schemes instead compress or regulate the video-side KV cache to support longer videos or unbounded streams, e.g., InfiniPot-

V [28], StreamMem [74], and MovieChat [54, 55]. In parallel, a recent line of work reduces long-video cost by decoupling video processing ("prefill") from decoding and constructing a video KV cache incrementally as frames arrive. ReKV [20] is an inference-time adaptation of pretrained VLMs that reduces the complexity of long video encoding via a sliding-window mechanism, then answers questions by decoding conditioned on the accumulated per-frame tokens. Follow-up methods such as HERMES [80], LiveVLM [42], and StreamKV [15] further optimize the language generation stage via hierarchical KV memories, streaming-oriented KV cache construction and retrieval, or segment-level retrieval/compression with summary tokens, respectively. These directions blur the boundary between "long-video" and "streaming" settings: in both cases, the dominant computation is often the video-side prefill, and improving real-time prefill throughput directly benefits online interaction. Related streaming systems such as SDQES [21], streaming dense video captioning [90], VideoLLM-online [12], Flash-VStream [79], and Dispider [48] further emphasize causal processing and online memory management.

Our approach sits between fixed-budget compression and streaming approximations: we compress the running state used to build the prefill memory, while preserving per-frame visual detail for final language generation. Unlike ReKV [20], which imposes a strict sliding-window view of the past, we treat streaming video prefill as approximating full self-attention with a small set of tokens carried between frames. This yields a two-cache structure: a fixed-capacity, importance-based temporal cache used only during streaming video prefill that carries the important tokens, and a full detailed cache retained for final language generation. This design is motivated by empirical structure in long-video attention (dominant within-frame interactions plus a small set of slowly-varying "temporal sink" tokens), and we find that it transfers consistently across multiple pretrained model families and parameter scales.

KV-cache compression for long-context LLMs and multimodal models. A large literature studies reducing memory and bandwidth costs of the KV cache in long-context LLM inference. Training-free eviction and sparsification policies such as H_2O [88] retain a mixture of recent and "heavy hitter" tokens, while learned or lightly-trained approaches compress or sparsify the cache online (e.g., DMC [41] and DMS [29]). Other methods design layer- or task-adaptive cache budgets (e.g., PyramidKV [10]) or provide practical drop-in schemes for decoding acceleration (e.g., SnapKV [35] and RocketKV [7]). For *multimodal* long-context inference, AdaReTaKe [66], Look-M [63] and MEDA [62] study KV allocation/retention across modalities. Compared to these mostly text-centric policies, long-video inference places much of the cost in the *video prefill* phase and exhibits distinct attention structure due to frame-based inputs, motivating video-specific designs.

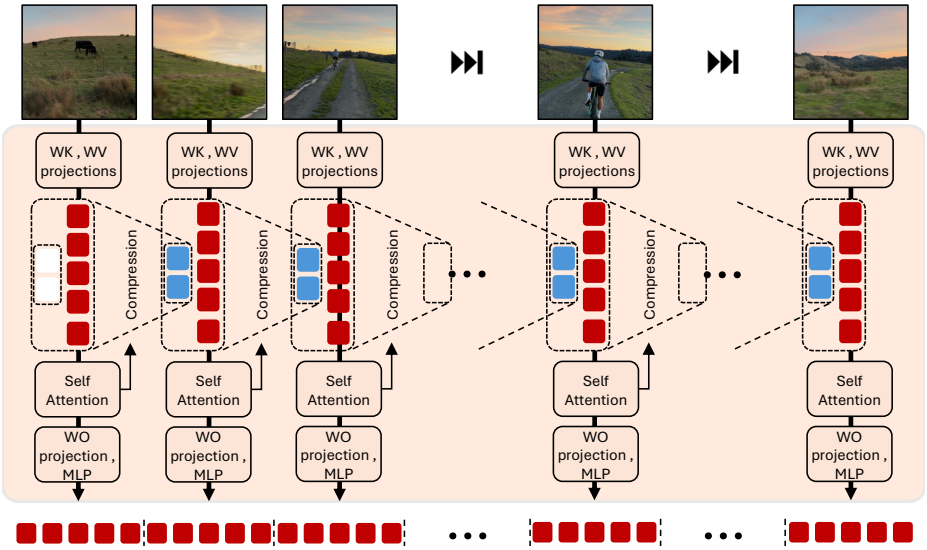

Fig. 2: Single-transformer-layer view of StateKV, showing the required modifications to a transformer block. The video stream is processed frame-by-frame with a frozen backbone. A fixed-capacity compressed state (in blue) allows information from previous frames to flow through a fixed size set of sink tokens during prefill. Separately, we build a full length detailed state for decoding (shown in red).

3 Method

3.1 Core Assumptions

Our design relies on empirical properties of attention in long-video VLMs. Let frame n provide T query tokens and let $\mathcal{H}_{n-1}$ denote the set of keys from frames $< n$. For a fixed layer, let A_n denote the attention weights produced when processing frame n. We define the cross-frame importance of a key $j \in \mathcal{H}_{n-1}$ as

$$s_{n,j} = \sum_{i=1}^{T} A_{n,i,j}. \tag{1}$$

Our first assumption is that there exists a set of temporal sinks $S_n \subseteq \mathcal{H}_{n-1}$ with $|S_n| = K$ and $K \ll |\mathcal{H}_{n-1}|$ such that

$$\sum_{j \in S_n} s_{n,j} \approx \sum_{j \in \mathcal{H}_{n-1}} s_{n,j}. \tag{2}$$

That is, most *inter-frame* attention mass is concentrated on a small set of historical tokens whose size does not grow with the total video length. This is the basis for limiting cross-frame attention to a fixed-capacity temporal memory.

Second, we assume that these sink sets evolve slowly over time, in the sense that the next useful sink set can be well approximated from the previous one together with the current frame:

$$S_{n+1} \approx \mathrm{TopK}\big(S_n \cup \{\text{tokens in frame } n+1\}\big). \tag{3}$$

This assumption justifies an incremental update rule that evicts low-importance entries and admits newly salient tokens from the current frame, rather than re-optimizing over the entire video prefix at every step.

Finally, because StateKV retains all video tokens for text decoding, the compressed state only needs to approximate frame-to-frame interactions during video encoding; it does not need to be the final conditioning memory for language generation. We therefore select temporal sinks using video-only attention scores and make no assumptions about how text queries might alter sink identities.

We formalize the concentration and slow-evolution assumptions into testable mechanisms, and probe these claims empirically in the Supplementary Material. That analysis, together with the main results in Section 4, provides evidence that the resulting importance-based approximation works well across tested parameter scales, cache sizes, and model families.

3.2 Method Overview

Preliminaries. Let $A \in \mathbb{R}^{L\times L}$ denote the attention matrix for a given layer and head, where $A_{i,j}$ is the normalized attention from query token i to key token j, and let $V \in \mathbb{R}^{L\times d}$ be the corresponding value vectors. The full self-attention output for token i is

$$y_i^{\text{full}} = \sum_{j=1}^{L} A_{i,j} V_j.$$

Video tokens are grouped into frames; for N video frames with T tokens per frame we have $L = TN$ video tokens.

Streaming cache construction with two memory states. Let a video consist of N frames, each yielding T visual tokens. We build the video representation in a streaming fashion, processing frames in order and constructing a key-value (KV) cache incrementally. The method maintains two per-layer KV states: (i) a *detailed state* (`dstate`) that stores all per-frame video tokens and is used for final text decoding; and (ii) a small *compressed state* (`cstate`) of fixed capacity that is used only as cross-frame context during cache construction.

Concretely, for each transformer layer $\ell \in \{1, \ldots, L_{\text{layers}}\}$ we maintain

$$D_n^\ell = \{(K_{1:n}^\ell, V_{1:n}^\ell)\}, \quad C_n^\ell = \{(\bar{K}_n^\ell, \bar{V}_n^\ell)\}, \quad |\bar{K}_n^\ell| = |\bar{V}_n^\ell| = B. \tag{4}$$

where D_n^ℓ contains *all* visual tokens accumulated up to frame n (and thus grows as $O(nT)$), while C_n^ℓ is a compressed memory with a fixed budget B (e.g., 1024 tokens). Importantly, C_n^ℓ is the *only* information from frames $< n$ that frame n can attend to during cache building. The detailed cache D_n^ℓ is updated by appending the current frame tokens after each step, and is never queried by future frames during cache building. By contrast, the compressed state is refreshed after every frame, so it evolves over time even though its capacity remains fixed.

Key insight: dynamic compressed state. Our method relies on the assumption that the tokens useful for processing frame n are either (i) tokens that were already useful for frame $n-1$ (and therefore retained in C^{ℓ}_{n-1}), or (ii) tokens from the current frame n. This makes the compressed state *dynamic* rather than fixed across the whole video: we are not attempting to summarize the entire past into a single static memory. Instead, we maintain a small working set that is refreshed every frame so that frame n has access to the information it needs.

3.3 Per-frame cache-builder forward pass

Let $X_n \in \mathbb{R}^{T\times d}$ denote the hidden states of the visual tokens for frame n at the input of a given layer. The cache builder computes, at each layer ℓ, queries for the current frame and keys/values for both the current frame and the compressed memory:

$$Q^{\ell}_n = X_n W^{\ell}_Q, \qquad K^{\ell}_n = X_n W^{\ell}_K, \qquad V^{\ell}_n = X_n W^{\ell}_V, \qquad \bar{K}^{\ell}_{n-1}, \bar{V}^{\ell}_{n-1} \in C^{\ell}_{n-1}.$$

We then perform attention for frame n against the concatenation of the compressed memory and the current frame tokens:

$$\mathrm{Attn}\big(Q^{\ell}_n, [\bar{K}^{\ell}_{n-1}; K^{\ell}_n], [\bar{V}^{\ell}_{n-1}; V^{\ell}_n]\big) = \mathrm{softmax}\left(\frac{Q^{\ell}_n [\bar{K}^{\ell}_{n-1}; K^{\ell}_n]^{\top}}{\sqrt{d_h}} + M_n\right) \cdot [\bar{V}^{\ell}_{n-1}; V^{\ell}_n] \quad (5)$$

where d_h is the head dimension and M_n is the appropriate causal and modality mask (for cache building we only require causal structure within the stream order, with all memory tokens preceding the current frame tokens).

This yields updated hidden states $X^{\ell}_{n,\mathrm{out}}$ for the current frame, which are passed to the next layer. After the final layer, we append the per-layer (K^{ℓ}_n, V^{ℓ}_n) to the detailed state:

$$D^{\ell}_n \leftarrow D^{\ell}_{n-1} \cup \{(K^{\ell}_n, V^{\ell}_n)\}.$$

Thus, the memory complexity of the detailed cache grows linearly as the model processes more frames, but we avoid the quadratic compute of cross-frame attention by restricting cross-frame interaction to the fixed-size compressed state.

Updating the compressed state via attention-driven selection Finally, after processing frame n, we update the compressed state C^{ℓ}_n by selecting a fixed number of tokens to carry forward. Let $\mathcal{U}^{\ell}_n$ denote the candidate pool for compression at layer ℓ. In StateKV we use an incremental pool consistent with the "slowly-evolving sinks" assumption:

$$\mathcal{U}^{\ell}_n = \bar{S}^{\ell}_{n-1} \cup \{1, \ldots, T\}_{(\text{frame } n \text{ tokens})},$$

where $\bar{S}^{\ell}_{n-1}$ indexes the tokens currently stored in C^{ℓ}_{n-1}. We assign each candidate token $j \in \mathcal{U}^{\ell}_n$ an importance score based on *video-only attention*. Let A^{ℓ}_n denote the attention weights produced when processing frame n at layer ℓ (aggregated over heads and queries within the frame). One simple instantiation is

$$s^{\ell}_{n,j} = \frac{1}{T}\sum_{i=1}^{T} A^{\ell}_{n,i,j},$$

where $A^{\ell}_{n,i,j}$ is the normalized attention from query token i in frame n to candidate key token j (which may refer either to a memory token or a token in frame n). We then keep the top-B candidates:

$$\begin{aligned} \bar{S}^{\ell}_{n} &= \mathrm{TopK}\big(\{s^{\ell}_{n,j} : j \in \mathcal{U}^{\ell}_{n}\},\, B\big), \\ C^{\ell}_{n} &\leftarrow \{(\bar{K}^{\ell}_{n}, \bar{V}^{\ell}_{n})\} = \{(K^{\ell}_{j}, V^{\ell}_{j}) : j \in \bar{S}^{\ell}_{n}\}. \end{aligned} \tag{6}$$

This procedure realizes a fixed-capacity temporal cache that is refreshed each frame by evicting low-importance memory entries and admitting newly salient tokens from the current frame. In all experiments we compute scores from video-only attention statistics, per the final assumption.

Virtual sequence length and cache positions. Because C^{ℓ}_{n} is capacity-limited, the number of keys stored in the compressed state differs from the number of tokens seen so far in the stream. However, positional encodings (RoPE [56]) depend on the logical position in the stream, not on the current cache size. We therefore maintain a *virtual sequence length* L_n that counts all tokens processed up to time n independent of the number of tokens retained in C^{ℓ}_{n}. When constructing attention for frame n, RoPE positions and cache positions are derived from L_n rather than from the physical cache length of C^{ℓ}_{n}.

Consistent RoPE scaling across cache building and generation. When the total sequence length exceeds the base model's trained maximum context, RoPE scaling must be applied *consistently* to all keys/values that will be reused during generation. In particular, the video KV cache must be compatible with subsequently generated tokens. To ensure this, we: (i) determine the maximum expected sequence length for the entire run (video frames plus prompt plus maximum generation length) prior to cache building; (ii) activate the corresponding RoPE scaling configuration before the first frame is processed; and (iii) keep the same scaling active for the full duration of cache building and text decoding.

Formally, let $\phi(\cdot;\alpha)$ denote the RoPE embedding function with scaling parameters α (e.g., YARN [46]). We apply

$$Q \leftarrow \phi(Q;\alpha), \quad K \leftarrow \phi(K;\alpha)$$

with the *same* α for both cache building and generation. Changing α after building the KV cache would make the cached K/V incompatible with newly rotated queries/keys and leads to severe degradation; thus α is fixed end-to-end for each evaluation run.

Decoding using the detailed cache. After the streaming cache builder finishes processing all frames, we run standard autoregressive decoding for the text output. Crucially, decoding uses the detailed state D^{ℓ}_{N} (all video tokens) as the conditioning KV cache. The compressed state C^{ℓ}_{N} is not used during decoding; it exists solely to approximate cross-frame interactions during cache construction. This design matches the assumption that the temporal cache only needs to be an approximation for video encoding, since the final generation stage conditions on the full set of stored video tokens.

4 Results

Experimental Setup. Our setup isolates the effect of the self-attention approximation while keeping the rest of the inference stack fixed. All experiments use Hugging Face [68] model implementations and default prompts from the lmms-eval suite [81]. We evaluate on VideoMME [23] (subtitles-free setting), MLVU [89], and OVOBench [43] (*Real-Time Visual Perception* subset). Following common long-video evaluation protocol [11], we sample video at 1 FPS and cap each example at 512 frames. We first refactor execution into two stages: video prefill and language generation. We then implement a streaming version of the full self-attention baseline so it can process very long sequences. This conversion is mathematically exact and, with linear memory attention kernels (e.g., SDPA/FlashAttention [18]) keeps peak memory usage manageable, although per frame compute still grows linearly with processed context so late frames become very slow. From that baseline, we modify only the attention operation to obtain ReKV and StateKV, so differences in performance can be attributed only to the approximation itself. RoPE scaling, data loading, prompts, and generation hyperparameters are matched across methods. Unless stated otherwise, we report efficiency primarily in terms of measured FLOPs (profiled using PyTorch [45] profiler and verified against theoretical calculations) and asymptotic scaling, which are stable across hardware and kernel implementations. The Supplementary Material also includes a wall-time comparison showing that, even when Full SA uses FlashAttention-2 [18] and StateKV uses an eager attention path during cache building, the constant per-frame cost of StateKV still overtakes the linearly increasing cost of full self-attention at sufficiently long sequence lengths. We additionally provide a custom Triton [61] kernel that replaces eager attention with a FlashAttention-2 [18] based implementation that returns the per-key scores needed for token selection and moves the crossover to shorter sequences.

Pareto frontier at fixed long-video length. Fig. 3 shows performance versus total prefill compute for a 512-frame video, with multiple compression settings for both StateKV and ReKV. Within each color, moving across triangles corresponds to increasing the StateKV cache budget B, while moving across squares corresponds to increasing the ReKV recency window R; this makes the within-model compute-accuracy scaling explicit. As expected, stronger compression reduces FLOPs but also weakens the approximation to full self-attention. Even under this tradeoff, StateKV remains consistently closer to Full SA than ReKV at comparable compute, and traces a stronger frontier across budgets. Notably, the StateKV operating points follow a smooth log-linear relationship between compute and accuracy, which enables predictable test-time scaling.

FLOPs reduction enables larger backbones. The practical implication of this frontier is that compute savings can be reinvested in model scale. Although any approximation introduces some loss relative to exact Full SA for the same backbone, the loss for StateKV is small enough that moving to a larger model typically more than compensates for it. As a result, running larger models with

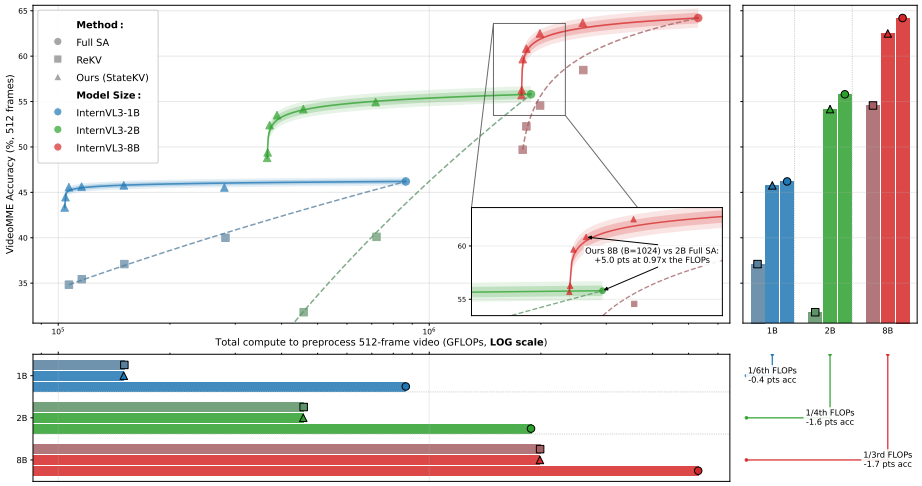

Fig. 3: Total compute to preprocess a 512-frame video (in GFLOPs) versus performance on VideoMME across three model sizes of the same model family (InternVL3 1B, 2B, 8B). Marker shape denotes which self-attention approximation (or Full SA) is used, while color denotes model size: circles are the 512-frame Full Self-Attention, triangles are StateKV operating points at cache budgets $B \in \{16, 64, 256, 1024, 4096, 16384\}$, and squares are ReKV operating points at retained-frame budgets $R \in \{1, 4, 16, 64\}$. Constant-FLOPs-per-frame methods reduce FLOPs compared to full self-attention while preserving accuracy. Of these, StateKV achieves accuracy competitive with full self-attention and enables larger models under the same budget. For instance, StateKV-8B with $B = 4096$ achieves 62.5% accuracy at similar compute cost as Full SA-1B (46.2%). Compared to existing sliding-window based methods like ReKV, StateKV achieves superior accuracy at comparable compression levels. The right and bottom supporting bars use one representative operating point per method: StateKV with $B = 4096$ and ReKV with a 16-frame window (plus Full SA). These are compute-matched, so the bars isolate the quality difference at similar compute.

StateKV yields substantially higher VideoMME accuracy at compute budgets comparable to or below smaller Full SA baselines. In Fig. 3, this appears as overlaps between a larger model's StateKV log-linear compute-accuracy curve, and the operating points for the unmodified Full SA baseline. The marginal-cost variant of Fig. 3, provided in the Supplementary Material, contains a double overlap, indicating that StateKV allows practitioners to run a model two scales larger at a similar per-frame-cost to the Full SA variant (InternVL3-8B with StateKV for a similar per-frame cost as the InternVL3-1B baseline).

Cross-backbone ablation across families and scales. Table 1 compares full self-attention, ReKV, and StateKV across three video benchmarks and multiple backbones spanning different families and parameter scales. Averaged across these settings, StateKV stays close to full self-attention (within roughly a point on average) while consistently improving over ReKV by about 10 points on

Model	Method	VideoMME	MLVU	OVOBench (Real-Time)
InternVL3-1B	Full SA	46.19%	47.05%	55.79%
	ReKV (R=16)	37.11%	33.44%	37.75%
	StateKV (B=4096)	45.8%	46.35%	55.44%
InternVL3-2B	Full SA	55.81%	56.61%	60.22%
	ReKV (R=16)	31.78%	5.49%	33.33%
	StateKV (B=4096)	54.15%	57.31%	61.05%
QWen3-VL-2B	Full SA	58.67%	58.83%	60.45%
	ReKV (R=16)	49.44%	45.6%	46.71%
	StateKV (B=4096)	58%	57.72%	60.93%
QWen3-VL-4B	Full SA	66.59%	68.98%	64.76%
	ReKV (R=16)	52.63%	49.91%	49.22%
	StateKV (B=4096)	65.89%	67.8%	64.87%
InternVL3-8B	Full SA	64.19%	61.14%	71.45%
	ReKV (R=16)	54.56%	31.11%	56.03%
	StateKV (B=4096)	62.52%	62.85%	70.25%
Eagle2.5-8B	Full SA	69.81%	73.85%	69.3%
	ReKV (R=16)	58.7%	55.01%	55.44%
	StateKV (B=4096)	67.96%	70.52%	68.7%
QWen3-VL-8B	Full SA	70.52%	75.82%	67.86%
	ReKV (R=16)	55.52%	50.91%	53.88%
	StateKV (B=4096)	68.11%	71.38%	64.99%

Table 1: Cross-backbone comparison across three long-video benchmarks. For each backbone, we compare exact full self-attention, the recency-based streaming baseline ReKV with a 16-frame retrieval window, and StateKV with cache budget $B = 4096$; these ReKV/StateKV settings are chosen to be compute-matched. Across model families and scales, StateKV stays close to Full SA while consistently outperforming ReKV on VideoMME, MLVU, and OVOBench, indicating that importance-based memory is a stronger approximation to full long-range attention than a strict sliding-window prior.

average. We observe the same trend on MLVU and OVOBench (Real-Time subset), indicating that the gain is not tied to a single backbone design or model size. These cross-family results show that the same importance-based carried-memory intervention transfers across backbones, parameter counts, and cache budgets. For a fair efficiency comparison, ReKV runs retrieving 16 frames and StateKV with cache budget $B = 4096$ are compute-matched.

Scaling behavior. Fig. 4 studies accuracy as we increase the StateKV cache budget B (or, for the baseline, the sliding-window retrieved frames). For all InternVL3 variants analyzed, StateKV improves monotonically across all video lengths and approaches, or in some settings matches, the full self-attention reference at the highest budgets (e.g., InternVL3-8B StateKV reaches 75.0% on short videos, matching Full SA), demonstrating stable scaling with compute. Notably, the bigger models exhibit stronger scaling with cache size, with performance saturating at increasingly large budgets as model parameter counts increase. In contrast, sliding-window attention scales more slowly and remains below full attention at comparable budgets, with 5-10 point gaps across settings, suggesting that expanding only a local window does not recover the global-context benefits

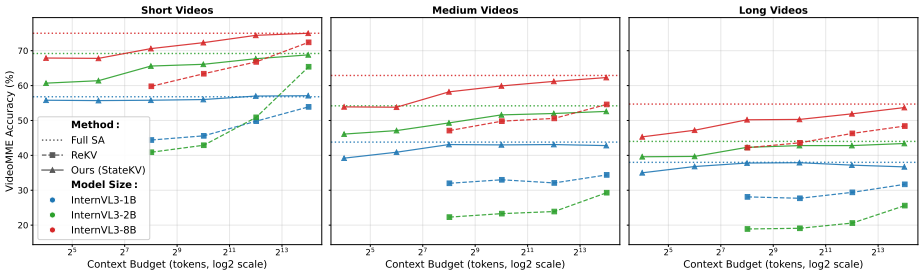

Fig. 4: Comparison of VideoMME accuracy across context budgets for InternVL3-1B/2B/8B. The dotted lines show Full SA (target behavior), while ReKV and StateKV trace budgeted approximations. Across short, medium, and long videos, StateKV stays consistently closer to the Full SA accuracy frontier than ReKV at comparable budgets, indicating a stronger approximation of full attention under constrained compute.

captured by StateKV. StateKV achieves these gains while maintaining linear-in-frames video prefill computational cost, whereas Full SA remains costly on long videos.

Sliding-window instability across settings. Across our experiments, ReKV exhibits unstable behavior in multiple settings: InternVL3-2B shows systematic degradation across operating points in both Fig. 3 and Fig. 4, occasionally underperforming even the 1B variant; InternVL3-8B shows similar instabilities on MLVU (Table 1); and performance gaps relative to full attention remain substantial even at the highest sliding-window budgets tested. These observations suggest that strict recency-based approximations can be a poor match to the attention patterns learned by certain backbones. We verified that this degradation is reproducible under the same shared implementation used across model scales and datasets. The Supplementary Material discusses how StateKV's importance-based selection better approximates full self-attention patterns across these varied settings.

Compute break-even behavior. Fig. 5 highlights intersection points between dotted (Full SA) and solid (StateKV) curves, which mark compute break-even regimes across model sizes. In the left plot (marginal FLOPs per frame), each intersection gives the frame index beyond which adding one more frame is cheaper with a larger StateKV model than with a smaller Full SA baseline. In the right plot (cumulative FLOPs), intersections indicate the video length beyond which total compute is lower for the larger StateKV model over the full processed video. This distinction matters operationally: marginal cost is the relevant quantity for streaming-oriented settings, while cumulative cost is the relevant quantity for long-video processing. The key implication is that, because unmodified Full SA has quadratic video-prefill cost while StateKV is linear in the number of frames, a break-even intersection must exist at sufficiently long durations for each

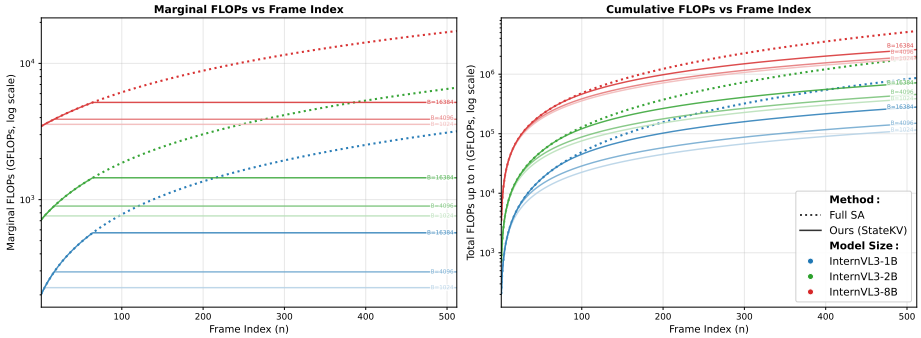

Fig. 5: Compute cost versus frame index. Left: marginal FLOPs per frame. Right: cumulative FLOPs. Dotted curves denote full self-attention and solid curves denote StateKV.

compared pair. In other words, beyond a model- and setup-dependent horizon, it is compute-favorable to run a larger StateKV model rather than a smaller quadratic baseline. We further illustrate how this gap would widen at longer horizons by extrapolating compute curves to 3600 frames (1 FPS over 1 hour) in the Supplementary Material.

5 Conclusion

We presented StateKV, a linear-time approximation to long-video self-attention for video VLMs. Building on recent streaming-prefill methods such as ReKV, our method separates video prefill into a streaming cache-construction stage and final text-decoding over the retained detailed video state. The key improvement is to frame streaming video prefill as approximating full self-attention using a small set of tokens carried between frames, rather than imposing a strict sliding-window prior. This design is motivated by mechanistic evidence on tested cases suggesting that useful inter-frame attention is often concentrated on a relatively small set of tokens and that this set evolves gradually enough to be updated from the previous state together with the current frame. Across multiple backbones, model scales, and long-video benchmarks, StateKV stays closer to full self-attention than sliding-window style baselines while reducing asymptotic video-prefill cost from quadratic to linear in the number of processed frames.

The main consequence in our measured FLOPs-based analysis is a better compute-accuracy tradeoff. Because StateKV preserves accuracy more effectively under compression, its FLOPs savings can be reinvested into larger backbones, yielding operating points where a larger StateKV model is both cheaper than and more accurate than a smaller full-attention baseline. Across model families, parameter scales, and cache sizes, we observe the same qualitative trend: carrying a small, importance-based set of tokens between frames is a stronger approximation

than a strict recency bias. Our supplementary analyses further indicate that the analyzed attention patterns are not well described by a pure sliding-window view of the past, which helps explain why strict sliding-window approximations can be weak in our setting.

This work also has limitations. The mechanistic validation of the assumptions can only be performed on existing models and tested inputs, so it is neither general nor fundamental: although the assumptions are reasonable and supported on the models we analyze, untested backbones or future models may not exhibit the same behavior. Natural next steps include broader analysis across more backbones and video regimes.

Acknowledgments

This work is in part supported by ONR N00014-23-1-2355, ONR MURI MURI N00014-24-1-2748, and the Stanford Institute for Human-Centered Artificial Intelligence (HAI).

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023) 4
2. Alayrac, J.B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al.: Flamingo: a visual language model for few-shot learning. Advances in neural information processing systems **35**, 23716–23736 (2022) 3
3. Anthropic: Claude sonnet 4.5 system card (2025), https://assets.anthropic.com/m/12f214efcc2f457a/original/Claude-Sonnet-4-5-System-Card.pdf 4
4. Bai, J., Bai, S., Yang, S., Wang, S., Tan, S., Wang, P., Lin, J., Zhou, C., Zhou, J.: Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. arXiv preprint arXiv:2308.12966 (2023) 3
5. Bai, S., Cai, Y., Chen, R., Chen, K., Chen, X., Cheng, Z., Deng, L., Ding, W., Gao, C., Ge, C., et al.: Qwen3-vl technical report. arXiv preprint arXiv:2511.21631 (2025) 3
6. Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., et al.: Qwen2. 5-vl technical report. arXiv preprint arXiv:2502.13923 (2025) 3
7. Behnam, P., Fu, Y., Zhao, R., Tsai, P.A., Yu, Z., Tumanov, A.: RocketKV: Accelerating long-context LLM inference via two-stage KV cache compression. In: Forty-second International Conference on Machine Learning (2025), https://openreview.net/forum?id=RyOpooIxDF 5
8. Buch, S., Eyzaguirre, C., Gaidon, A., Wu, J., Fei-Fei, L., Niebles, J.C.: Revisiting the "Video" in Video-Language Understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022), arXiv:2206.01720 4
9. Buch, S., Nagrani, A., Arnab, A., Schmid, C.: Flexible frame selection for efficient video reasoning. In: CVPR (2025) 4

10. Cai, Z., Zhang, Y., Gao, B., Liu, Y., Liu, T., Lu, K., Xiong, W., Dong, Y., Chang, B., Hu, J., Wen, X.: Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. arXiv preprint arXiv:2406.02069 (2024) 5
11. Chen, G., Li, Z., Wang, S., Jiang, J., Liu, Y., Lu, L., Huang, D.A., Byeon, W., Le, M., Ehrlich, M., et al.: Eagle 2.5: Boosting long-context post-training for frontier vision-language models. Advances in Neural Information Processing Systems **38**, 91077–91100 (2026) 3, 10
12. Chen, J., Lv, Z., Wu, S., Lin, K.Q., Song, C., Gao, D., Liu, J.W., Gao, Z., Mao, D., Shou, M.Z.: Videollm-online: Online video large language model for streaming video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18407–18418 (2024) 5
13. Chen, L., Zhao, H., Liu, T., Bai, S., Lin, J., Zhou, C., Chang, B.: An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In: European Conference on Computer Vision. pp. 19–35. Springer (2024) 4
14. Chen, S., Lan, X., Yuan, Y., Jie, Z., Ma, L.: Timemarker: A versatile video-llm for long and short video understanding with superior temporal localization ability. arXiv preprint arXiv:2411.18211 (2024) 3
15. Chen, Y., Bai, X., Wang, Z., Bai, C., Dai, Y., Lu, M.: Streamkv: Streaming video question-answering with segment-based kv cache retrieval and compression. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 40, pp. 3120–3128 (2026) 2, 5
16. Chen, Z., Wang, W., Cao, Y., Liu, Y., Gao, Z., Cui, E., Zhu, J., Ye, S., Tian, H., Liu, Z., et al.: Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. arXiv preprint arXiv:2412.05271 (2024) 3
17. Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., et al.: Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. arXiv preprint arXiv:2507.06261 (2025) 4
18. Dao, T.: Flashattention-2: Faster attention with better parallelism and work partitioning. In: International Conference on Learning Representations. vol. 2024, pp. 35549–35562 (2024) 10
19. Deitke, M., Clark, C., Lee, S., Tripathi, R., Yang, Y., Park, J.S., Salehi, M., Muennighoff, N., Lo, K., Soldaini, L., et al.: Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 91–104 (2025) 3
20. Di, S., Yu, Z., Zhang, G., Li, H., Cheng, H., Li, B., He, W., Shu, F., Jiang, H.: Streaming video question-answering with in-context video kv-cache retrieval. In: International Conference on Learning Representations. vol. 2025, pp. 42115–42127 (2025) 2, 5
21. Eyzaguirre, C., Tang, E., Buch, S., Gaidon, A., Wu, J., Niebles, J.C.: Streaming detection of queried event start. Advances in Neural Information Processing Systems **37**, 100698–100733 (2024) 5
22. Eyzaguirre, C., Vasiljevic, I., Dave, A., Wu, J., Ambrus, R.A., Kollar, T., Niebles, J.C., Tokmakov, P.: Understanding complexity in videoQA via visual program generation. In: Forty-second International Conference on Machine Learning (2025), https://openreview.net/forum?id=6GFPnVHEKB 4
23. Fu, C., Dai, Y., Luo, Y., Li, L., Ren, S., Zhang, R., Wang, Z., Zhou, C., Shen, Y., Zhang, M., et al.: Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In: CVPR (2025) 10

24. Fu, T., Liu, T., Han, Q., Dai, G., Yan, S., Yang, H., Ning, X., Wang, Y.: Framefusion: Combining similarity and importance for video token reduction on large vision language models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 22654–22663 (2025) 4
25. Gu, X., Pang, T., Du, C., Liu, Q., Zhang, F., Du, C., Wang, Y., Lin, M.: When attention sink emerges in language models: An empirical view. In: International Conference on Learning Representations. vol. 2025, pp. 97114–97144 (2025) 3
26. Huang, D.A., Liao, S., Radhakrishnan, S., Yin, H., Molchanov, P., Yu, Z., Kautz, J.: Lita: Language instructed temporal-localization assistant. In: European Conference on Computer Vision. pp. 202–218. Springer (2024) 4
27. Jin, P., Takanobu, R., Zhang, W., Cao, X., Yuan, L.: Chat-univi: Unified visual representation empowers large language models with image and video understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13700–13710 (2024) 4
28. Kim, M., Shim, K., Choi, J., Chang, S.: Infinipot-v: Memory-constrained kv cache compression for streaming video understanding. In: Belgrave, D., Zhang, C., Lin, H., Pascanu, R., Koniusz, P., Ghassemi, M., Chen, N. (eds.) Advances in Neural Information Processing Systems. vol. 38, pp. 138983–139013. Curran Associates, Inc. (2025), https://proceedings.neurips.cc/paper_files/paper/2025/file/caef5f5e658aa1f7565f063a2cd99726-Paper-Conference.pdf 5
29. Łańcucki, A., Staniszewski, K., Nawrot, P., Ponti, E.M.: Inference-time hyper-scaling with kv cache compression. Advances in Neural Information Processing Systems **38**, 9365–9397 (2026) 5
30. Li, B., Zhang, Y., Guo, D., Zhang, R., Li, F., Zhang, H., Zhang, K., Zhang, P., Li, Y., Liu, Z., et al.: Llava-onevision: Easy visual task transfer. Transactions on Machine Learning Research (2024) 3
31. Li, J., Li, D., Savarese, S., Hoi, S.: Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In: International conference on machine learning. pp. 19730–19742. PMLR (2023) 3
32. Li, K., He, Y., Wang, Y., Li, Y., Wang, W., Luo, P., Wang, Y., Wang, L., Qiao, Y.: Videochat: Chat-centric video understanding. Science China Information Sciences **68**(10), 200102 (2025) 4
33. Li, X., Wang, Y., Yu, J., Zeng, X., Zhu, Y., Huang, H., Gao, J., Li, K., He, Y., Wang, C., Qiao, Y., Wang, Y., Wang, L.: Videochat-flash: Hierarchical compression for long-context video modeling. arXiv preprint arXiv:2501.00574 (2024) 4
34. Li, Y., Wang, C., Jia, J.: Llama-vid: An image is worth 2 tokens in large language models (2024) 4
35. Li, Y., Huang, Y., Yang, B., Venkitesh, B., Locatelli, A., Ye, H., Cai, T., Lewis, P., Chen, D.: Snapkv: Llm knows what you are looking for before generation. Advances in Neural Information Processing Systems **37**, 22947–22970 (2024) 5
36. Liu, H., Li, C., Li, Y., Lee, Y.J.: Improved baselines with visual instruction tuning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 26296–26306 (2024) 3
37. Liu, H., Li, C., Li, Y., Li, B., Zhang, Y., Shen, S., Lee, Y.J.: Llava-next: Improved reasoning, ocr, and world knowledge (January 2024), https://llava-vl.github.io/blog/2024-01-30-llava-next/ 3
38. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. Advances in neural information processing systems **36**, 34892–34916 (2023) 3
39. Lu, Y., Wang, T., Rao, F., Yang, Y., Zhu, L., et al.: Flexselect: Flexible token selection for efficient long video understanding. Advances in Neural Information Processing Systems **38**, 102751–102777 (2026) 4

40. Maaz, M., Rasheed, H., Khan, S., Khan, F.: Video-chatgpt: Towards detailed video understanding via large vision and language models. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 12585–12602 (2024) 4
41. Nawrot, P., Łańcucki, A., Chochowski, M., Tarjan, D., Ponti, E.: Dynamic memory compression: Retrofitting LLMs for accelerated inference. In: Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., Berkenkamp, F. (eds.) Proceedings of the 41st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 235, pp. 37396–37412. PMLR (21–27 Jul 2024), https://proceedings.mlr.press/v235/nawrot24a.html 5
42. Ning, Z., Liu, G., Jin, Q., Li, C., Ding, W., Guo, M., Zhao, J.: Livevlm: Efficient online video understanding via streaming-oriented kv cache and retrieval. arXiv preprint arXiv:2505.15269 (2025) 2, 5
43. Niu, J., Li, Y., Miao, Z., Ge, C., Zhou, Y., He, Q., Dong, X., Duan, H., Ding, S., Qian, R., 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. pp. 18902–18913 (2025) 10
44. OpenAI: Gpt-5 system card (2025), https://openai.com/index/gpt-5-system-card/ 4
45. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019) 10
46. Peng, B., Quesnelle, J., Fan, H., Shippole, E.: Yarn: Efficient context window extension of large language models. In: International Conference on Learning Representations. vol. 2024, pp. 31932–31951 (2024) 9
47. Qian, L., Li, J., Wu, Y., Ye, Y., Fei, H., Chua, T.S., Zhuang, Y., Tang, S.: Momentor: Advancing video large language model with fine-grained temporal reasoning. In: Forty-first International Conference on Machine Learning (2024), https://openreview.net/forum?id=e3geukCBw6 4
48. Qian, R., Ding, S., Dong, X., Zhang, P., Zang, Y., Cao, Y., Lin, D., Wang, J.: Dispider: Enabling video llms with active real-time interaction via disentangled perception, decision, and reaction. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 24045–24055 (2025) 5
49. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PmLR (2021) 3
50. Ren, S., Yao, L., Li, S., Sun, X., Hou, L.: Timechat: A time-sensitive multimodal large language model for long video understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14313–14323 (2024) 4
51. Ren, W., Ma, W., Yang, H., Wei, C., Zhang, G., Chen, W.: Vamba: Understanding hour-long videos with hybrid mamba-transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 21197–21208 (2025) 4
52. Sarkar, S.D., Pautrat, R., Miksik, O., Pollefeys, M., Armeni, I., Rad, M., Dusmanu, M.: Cope-videolm: Leveraging codec primitives for efficient video language modeling (2026), https://arxiv.org/abs/2602.13191 4

53. Shao, K., Tao, K., Zhang, K., Feng, S., Cai, M., Shang, Y., You, H., Qin, C., Sui, Y., Wang, H.: When tokens talk too much: A survey of multimodal long-context token compression across images, videos, and audios. arXiv preprint arXiv:2507.20198 (2025) 4
54. Song, E., Chai, W., Wang, G., Zhang, Y., Zhou, H., Wu, F., Chi, H., Guo, X., Ye, T., Zhang, Y., et al.: Moviechat: From dense token to sparse memory for long video understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18221–18232 (2024) 5
55. Song, E., Chai, W., Ye, T., Hwang, J.N., Li, X., Wang, G.: Moviechat+: Question-aware sparse memory for long video question answering. IEEE Transactions on Pattern Analysis and Machine Intelligence (2025) 5
56. Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., Liu, Y.: Roformer: Enhanced transformer with rotary position embedding. Neurocomputing **568**, 127063 (2024) 9
57. Sun, B., Zhao, J., Wei, X., Hou, Q.: Llava-scissor: Token compression with semantic connected components for video llms. arXiv preprint arXiv:2506.21862 (2025) 4
58. Tang, Y., Bi, J., Xu, S., Song, L., Liang, S., Wang, T., Zhang, D., An, J., Lin, J., Zhu, R., et al.: Video understanding with large language models: A survey. IEEE Transactions on Circuits and Systems for Video Technology (2025) 3
59. Tao, K., Qin, C., You, H., Sui, Y., Wang, H.: Dycoke: Dynamic compression of tokens for fast video large language models. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 18992–19001 (2025) 4
60. Team, G., Georgiev, P., Lei, V.I., Burnell, R., Bai, L., Gulati, A., Tanzer, G., Vincent, D., Pan, Z., Wang, S., et al.: Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. arXiv preprint arXiv:2403.05530 (2024) 4
61. Tillet, P., Kung, H.T., Cox, D.: Triton: an intermediate language and compiler for tiled neural network computations. In: Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages. pp. 10–19 (2019) 10
62. Wan, Z., Shen, H., Wang, X., Liu, C., Mai, Z., Zhang, M.: Meda: Dynamic kv cache allocation for efficient multimodal long-context inference. In: Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). pp. 2485–2497 (2025) 5
63. Wan, Z., Wu, Z., Liu, C., Huang, J., Zhu, Z., Jin, P., Wang, L., Yuan, L.: Look-m: Look-once optimization in kv cache for efficient multimodal long-context inference. In: Findings of the Association for Computational Linguistics: EMNLP 2024. pp. 4065–4078 (2024) 5
64. Wang, P., Bai, S., Tan, S., Wang, S., Fan, Z., Bai, J., Chen, K., Liu, X., Wang, J., Ge, W., et al.: Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. arXiv preprint arXiv:2409.12191 (2024) 3
65. Wang, X., Chen, G., Qian, G., Gao, P., Wei, X.Y., Wang, Y., Tian, Y., Gao, W.: Large-scale multi-modal pre-trained models: A comprehensive survey. Machine Intelligence Research **20**(4), 447–482 (2023) 3
66. Wang, X., Si, Q., Zhu, S., Wu, J., Cao, L., Nie, L.: Adaretake: Adaptive redundancy reduction to perceive longer for video-language understanding. In: Findings of the Association for Computational Linguistics: ACL 2025. pp. 5417–5432 (2025) 2, 5
67. Wang, Y., Meng, X., Liang, J., Wang, Y., Liu, Q., Zhao, D.: Hawkeye: Training video-text llms for grounding text in videos (2024) 4
68. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al.: Transformers: State-of-the-art natural

language processing. In: Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations. pp. 38–45 (2020) 10
69. Xiao, G., Tian, Y., Chen, B., Han, S., Lewis, M.: Efficient streaming language models with attention sinks. In: International Conference on Learning Representations. vol. 2024, pp. 21875–21895 (2024) 3
70. Xing, L., Huang, Q., Dong, X., Lu, J., Zhang, P., Zang, Y., Cao, Y., He, C., Wang, J., Wu, F., Lin, D.: Conical visual concentration for efficient large vision-language models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 14593–14603 (June 2025) 4
71. Xu, L., Zhao, Y., Zhou, D., Lin, Z., Ng, S.K., Feng, J.: Pllava : Parameter-free llava extension from images to videos for video dense captioning (2024) 4
72. Xu, M., Gao, M., Gan, Z., Chen, H.Y., Lai, Z., Gang, H., Kang, K., Dehghan, A.: Slowfast-llava: A strong training-free baseline for video large language models. arXiv preprint arXiv:2407.15841 (2024) 4
73. Xu, R., Xiao, G., Chen, Y., He, L., Peng, K., Lu, Y., Han, S.: Streamingvlm: Real-time understanding for infinite video streams (2025), https://arxiv.org/abs/2510.09608 4
74. Yang, Y., Zhao, Z., Shukla, S.N., Singh, A., Mishra, S.K., Zhang, L., Ren, M.: Streammem: Query-agnostic kv cache memory for streaming video understanding. arXiv preprint arXiv:2508.15717 (2025) 5
75. Ye, J., Wang, Z., Sun, H., Chandrasegaran, K., Durante, Z., Eyzaguirre, C., Bisk, Y., Niebles, J.C., Adeli, E., Fei-Fei, L., et al.: Re-thinking temporal search for long-form video understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8579–8591 (2025), arXiv:2504.02259 4
76. Yu, S., Cho, J., Yadav, P., Bansal, M.: Self-chained image-language model for video localization and question answering. In: NeurIPS (2023) 4
77. Zhang, B., Li, K., Cheng, Z., Hu, Z., Yuan, Y., Chen, G., Leng, S., Jiang, Y., Zhang, H., Li, X., et al.: Videollama 3: Frontier multimodal foundation models for image and video understanding. arXiv preprint arXiv:2501.13106 (2025) 4
78. Zhang, H., Li, X., Bing, L.: Video-llama: An instruction-tuned audio-visual language model for video understanding. In: Proceedings of the 2023 conference on empirical methods in natural language processing: system demonstrations. pp. 543–553 (2023) 4
79. Zhang, H., Wang, Y., Tang, Y., Liu, Y., Feng, J., Dai, J., Jin, X.: Flash-vstream: Memory-based real-time understanding for long video streams. arXiv preprint arXiv:2406.08085 (2024) 5
80. Zhang, H., Yang, S., Fu, J., Ng, S.K., Qiu, X.: Hermes: Kv cache as hierarchical memory for efficient streaming video understanding. arXiv preprint arXiv:2601.14724 (2026) 2, 5
81. Zhang, K., Li, B., Zhang, P., Pu, F., Cahyono, J.A., Hu, K., Liu, S., Zhang, Y., Yang, J., Li, C., et al.: Lmms-eval: Reality check on the evaluation of large multimodal models. In: Findings of the Association for Computational Linguistics: NAACL 2025. pp. 881–916 (2025) 10
82. Zhang, P., Dong, X., Wang, B., Cao, Y., Xu, C., Ouyang, L., Zhao, Z., Ding, S., Zhang, S., Duan, H., Zhang, W., Yan, H., Zhang, X., Li, W., Li, J., Chen, K., He, C., Zhang, X., Qiao, Y., Lin, D., Wang, J.: Internlm-xcomposer: A vision-language large model for advanced text-image comprehension and composition. arXiv preprint arXiv:2309.15112 (2023) 3

83. Zhang, P., Dong, X., Zang, Y., Cao, Y., Qian, R., Chen, L., Guo, Q., Duan, H., Wang, B., Ouyang, L., Zhang, S., Zhang, W., Li, Y., Gao, Y., Sun, P., Zhang, X., Li, W., Li, J., Wang, W., Yan, H., He, C., Zhang, X., Chen, K., Dai, J., Qiao, Y., Lin, D., Wang, J.: Internlm-xcomposer-2.5: A versatile large vision language model supporting long-contextual input and output. arXiv preprint arXiv:2407.03320 (2024) 3
84. Zhang, P., Zhang, K., Li, B., Zeng, G., Yang, J., Zhang, Y., Wang, Z., Tan, H., Li, C., Liu, Z.: Long context transfer from language to vision. arXiv preprint arXiv:2406.16852 (2024), https://arxiv.org/abs/2406.16852 3
85. Zhang, S., Fang, Q., Yang, Y., Feng, Y.: Llava-mini: Efficient image and video large multimodal models with one vision token. In: Yue, Y., Garg, A., Peng, N., Sha, F., Yu, R. (eds.) International Conference on Learning Representations. vol. 2025, pp. 53285–53310 (2025), https://proceedings.iclr.cc/paper_files/paper/2025/file/8400a2ec9ba85bfdb41eb2a584c0876d-Paper-Conference.pdf 4
86. Zhang, Y., Fan, C.K., Ma, J., Zheng, W., Huang, T., Cheng, K., Gudovskiy, D., Okuno, T., Nakata, Y., Keutzer, K., et al.: Sparsevlm: Visual token sparsification for efficient vision-language model inference. In: Proceedings of the 42nd International Conference on Machine Learning (ICML) (2025) 4
87. Zhang, Y., Wu, J., Li, W., Li, B., MA, Z., Liu, Z., Li, C.: Llava-video: Video instruction tuning with synthetic data. Transactions on Machine Learning Research (2024) 3
88. Zhang, Z., Sheng, Y., Zhou, T., Chen, T., Zheng, L., Cai, R., Song, Z., Tian, Y., Ré, C., Barrett, C., Wang, Z.A., Chen, B.: H2o: Heavy-hitter oracle for efficient generative inference of large language models. In: Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) Advances in Neural Information Processing Systems. vol. 36, pp. 34661–34710. Curran Associates, Inc. (2023), https://proceedings.neurips.cc/paper_files/paper/2023/file/6ceefa7b15572587b78ecfcebb2827f8-Paper-Conference.pdf 5
89. Zhou, J., Shu, Y., Zhao, B., Wu, B., Liang, Z., Xiao, S., Qin, M., Yang, X., Xiong, Y., Zhang, B., et al.: Mlvu: Benchmarking multi-task long video understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13691–13701 (2025) 10
90. Zhou, X., Arnab, A., Buch, S., Yan, S., Myers, A., Xiong, X., Nagrani, A., Schmid, C.: Streaming dense video captioning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18243–18252 (2024) 5
91. Zhu, J., Wang, W., Chen, Z., Liu, Z., Ye, S., Gu, L., Tian, H., Duan, Y., Su, W., Shao, J., et al.: Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. arXiv preprint arXiv:2504.10479 (2025) 3
92. Zohar, O., Wang, X., Dubois, Y., Mehta, N., Xiao, T., Hansen-Estruch, P., Yu, L., Wang, X., Juefei-Xu, F., Zhang, N., et al.: Apollo: An exploration of video understanding in large multimodal models. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 18891–18901 (2025) 3