CoPE-VideoLM: Leveraging Codec Primitives For Efficient Video Language Modelling

Sayan Deb Sarkar[1,2*] **Rémi Pautrat**[1] **Ondrej Miksik**[1] **Marc Pollefeys**[1,3]
Iro Armeni[2] **Mahdi Rad**[1,†] **Mihai Dusmanu**[1,†]
[1]Microsoft Spatial AI Lab
[2]Stanford University [3]ETH Zurich

Project Page: https://microsoft.github.io/CoPE/

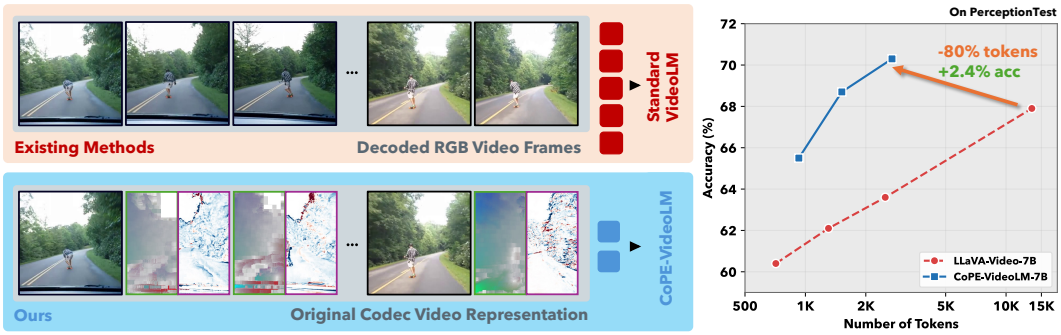


Figure 1: ***CoPE-VideoLM*** is a codec-aware tokenization framework for Video Language Models that replaces dense RGB frame encoding with lightweight structured representations derived from codec primitives. *(Left)* Instead of treating every frame as a full image, the model processes only sparse I-frames through a vision encoder, while P-frames are converted into compact lightweight tokens using their motion vectors and residuals. *(Right)* By avoiding redundant RGB processing, our framework enables a flexible trade-off between maximum efficiency (up to 93% token reduction) and maximum performance (e.g., +2.4% accuracy with 80% fewer tokens).

Abstract

Video Language Models enable AI systems to understand temporal dynamics in videos. To fit within the maximum context window constraint, current methods use keyframe sampling which often misses both macro-level events and micro-level details due to the sparse temporal coverage. Furthermore, processing full images and their tokens for each frame incurs substantial computational overhead. We address these limitations by leveraging video codec primitives (specifically motion vectors and residuals) which natively encode video redundancy and sparsity without requiring expensive full-image encoding for most frames. To this end, we introduce lightweight transformer-based encoders that aggregate codec primitives and align their representations with image encoder embeddings through a pre-training strategy that accelerates convergence during end-to-end fine-tuning. Our approach, *CoPE-VideoLM*, reduces the time-to-first-token by up to 86% and token usage by up to 93% compared to standard VideoLMs. Moreover, by varying keyframe and codec primitive densities, we maintain or exceed performance on 14 diverse video understanding benchmarks, spanning general question answering, temporal and motion reasoning, long-form understanding, and spatial scene understanding.

*Part of work done at Microsoft. † Equal supervision.

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

1 Introduction

Video Language Models (VideoLMs) represent a major advancement in multi-modal AI [41, 45, 119, 113, 101], extending Vision Language Models (VLMs) with temporal reasoning across video sequences. This unlocks a wide range of downstream applications from video question-answering to robotics [35, 55] where agents must understand sequential actions [52, 95]. VideoLMs represent a step toward AI systems that process visual information not as disconnected snapshots but as continuous, temporally coherent experiences. As video dominates digital content, models which can "watch" and understand video at scale [58, 17, 6] become essential for the next generation of AI.

However, VideoLMs are fundamentally bottlenecked by their context windows, dictated by the linear memory and quadratic compute costs of attention [64]. To fit within the budget, existing VideoLMs subsample video keyframes via hand-crafted heuristics (e.g., uniform temporal sampling [23]) or learned methods (e.g., flexible frame selection [9] or token compression [71]). While proprietary models can process hour-long videos at 1 FPS [23], this capability is driven by computational scaling to million-token context windows. Open-source models [113] are limited to sampling a small, fixed number of frames regardless of video length. Such a rigid restriction discards critical information, as information content scales with video duration.

Beyond limited budget, keyframe sampling provides sparse temporal coverage, missing macro-level events for high-level comprehension and micro-level details needed for fine-grained actions [57, 67, 102]. Consecutive frames are also highly redundant (even when downsampled to 1 FPS), so allocating the same token budget to every keyframe is suboptimal. Furthermore, treating each keyframe as a full image significantly increases prefill time, delaying *time-to-first token* (TTFT) [76]. Low TTFT is essential for user experience and robotics applications requiring real-time responsiveness.

A natural source of structure exists in video processing pipelines: video codecs, designed to exploit redundancy and leverage sparsity [38, 86]. Rather than encoding every frame as a full image, codecs select keyframes (I-frames) periodically and encode remaining frames (P-frames) as motion vectors and residuals, preserving temporal structure while minimizing redundancy. Prior works have explored codec-domain signals, but with important limitations: CoViAR [87] and TeamNet [84] trained separate CNNs on motion vectors and residuals, ignoring cross-modal dependencies for action recognition. Video-LaVIT [33] discretizes motion vectors but discards residuals. EMA [116] aggregates I-frames and motion vectors from a group of pictures (GOP) into a fixed-length summary, also discarding residuals and losing temporal ordering. Critically, no existing method preserves both dynamics and appearance in a variable-length, temporally ordered representation.

We propose encoding motion vectors and residuals (i.e., codec primitives) as compact tokens compatible with image features, interleaving them in temporal order alongside keyframe tokens to form a codec-aware sequence during inference. An overview of our approach is shown in Fig. 1. This yields two major advantages: (a) avoiding costly dense RGB encoding for most frames, and (b) exploiting primitive sparsity to significantly reduce token counts and inference latency. Furthermore, motion vectors and residuals of several P-frames can be flexibly grouped to balance temporal granularity and efficiency. Our formulation resolves all three aforementioned gaps: preserve residuals, maintain temporal ordering through interleaved I/P-frame tokens, and allow the token budget to scale with information content. While we focus on VideoLMs, the methodology is in principle broadly applicable to other tasks such as video retrieval or action recognition.

To this end, we use transformer encoders to aggregate motion vectors and residuals per GOP. These are pre-trained to align to the image encoder space, then fine-tuned end-to-end with a VideoLM. Importantly, our codec-aware tokenization approach operates at the representation level, requiring no architectural changes to base video model. To demonstrate this in a fully reproducible setting, we build on top of LLaVA-Video [113] due to its competitive performance and open weights, code, and training data. By condensing temporal information, our method matches or surpasses the baseline's RGB-only accuracy using a fraction of the tokens. Extensive validation across 14 benchmarks demonstrates consistent gains across general video QA, temporal and motion reasoning, long-form reasoning, and spatial scene understanding tasks. Our main contributions are as follows:

1. We propose to encode videos by leveraging codec primitives, preserving motion and appearance in temporal order; codec primitives allow us to skip redundant RGB information, reducing the TTFT by up to 86% and token usage by up to 93%.

2. We introduce a lightweight dual-branch architecture for encoding codec primitives, achieving substantially higher compression rates and lower token counts than image-based encoders.
3. We propose a pre-training framework for codec-primitive encoders that aligns their representation space with that of image encoders, enabling faster training and quicker convergence when integrated with the VideoLM.

2 Related Work

Video Language Models. Recent Multimodal Large Language Models (MLLMs) [72, 80, 48, 47, 49, 39, 108, 106, 107, 12, 15, 122] extend image-based architectures to videos for temporal reasoning. They typically comprise a vision encoder [62, 100], a modality alignment adapter [109, 110, 1], and an LLM backbone [75, 118, 4, 78, 6, 5] for multimodal decoding. Early VideoLMs such as Video-LLaMA [103] and VideoChat2 [41] were limited by short context and redundant tokenization. Subsequent models improve efficiency through extended context windows [101], token pooling or merging [56, 30, 61, 82], timestamp-aware encoding [63], and large-scale instruction tuning [113]. Proprietary models [23, 59, 17, 59, 58, 2] show impressive understanding capabilities but depend on proprietary data and undisclosed architectures. Despite these advances, VideoLMs still process videos as dense RGB frame collections, ignoring the structured redundancy inherent in video codecs. Leveraging primitives like motion vectors and residuals, our approach directly encodes temporal dynamics, supporting efficient long-context understanding while retaining fine-grained details.

Token Pruning. Compression techniques reduce visual redundancy via *heuristic* or *learnable* methods. Heuristic methods use uniform downsampling [46], spatio-temporal pooling [92, 113, 44], or similarity merging [8, 32, 43]. Learnable modules such as Q-Former [40, 4, 111], Perceiver Resampler [97], and memory-based mechanisms [69, 70, 32] generate compact latent representations before passing them to the LLM. Attention-based methods [11, 22, 90, 112] exploit the observation that visual tokens receive diminishing attention in deeper layers [11, 66], pruning or modulating token ratios across layers to remove redundancy. Temporal pooling approaches [92, 56, 93, 85] exploit inter-frame redundancy by downsampling at the frame level, while DyCoke [73] and LLaVA-Scissor [71] further leverage spatio-temporal structure for compression (see App. F for a direct comparative study; codec-native sparsity provides a stronger starting point than post-hoc pruning). Recently, more adaptive approaches such as AdaReTake [81] and FlexSelect [115] dynamically allocate compression budgets across layers or leverage cross-modal attention to filter tokens without retraining. However, these methods rely on dense RGB frame encodings and neglect intrinsic temporal redundancy attempting to remove information post hoc. In contrast, our native codec approach inherently encodes only meaningful temporal changes from the start.

Compressed Video Representation. To bypass costly full-frame processing, prior work has exploited codec primitives from compressed video streams for efficient visual understanding, particularly in action recognition. Early works [87, 84] trained separate CNNs on I- and P-frame signals but ignored inter-modal dependencies and required heavy multi-clip inference. Later methods extended compressed-domain learning to 3D CNNs [18], optical-flow distillation [68, 26, 74], and transformer-based self-attention [10], improving accuracy but still incurring high inference costs or requiring decoded RGB frames during training. CompressedVideoMAE [7] shows that masked autoencoding on motion vectors and residuals alone can match raw-video pretraining at far less compute. Recently, codec-based representations have been explored in VideoLMs. Video-LaVIT [33] discretizes motion vectors into language-like tokens, and EMA [116] aggregates I-frames and motion vectors into a fixed-length summary, similar to Video-VAE [91]. These approaches either discard residuals or collapse temporal ordering; CoPE-VideoLM addresses both by constructing a variable-length, temporally ordered token sequence that preserves fine-grained motion and appearance signals.

3 Method

3.1 Preliminaries

Modern video codecs such as MPEG-4, H.264, and HEVC [25, 37] achieve high compression ratios by exploiting temporal redundancy across frames. Let a video $\mathcal{V}$ be a sequence of frames

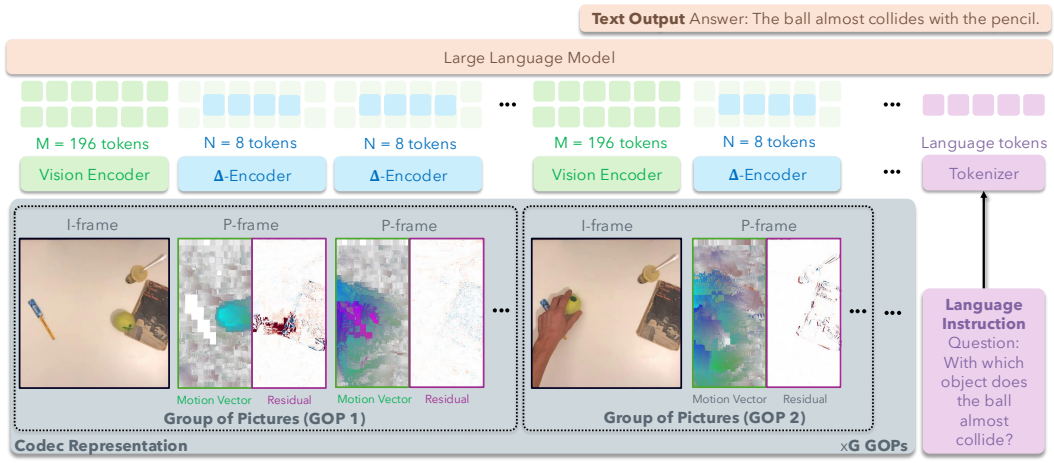


Figure 2: **Overview of our pipeline**. Given a video in its raw codec representation, our framework leverages the GOP structure for efficient, codec-aware tokenization. *I-frames* are processed by a standard frozen vision encoder (ϕ_{RGB}) to produce dense RGB tokens. *P-frames*, however, bypass full RGB decoding. Their raw components, motion vectors and residuals, are instead fed into our lightweight Δ-Encoder (ϕ_{Δ}) to generate a small set of highly compact Δ-tokens. The final token stream, an interleaved sequence of I-frame tokens and Δ-tokens, is consumed by the LLM, enabling dense temporal coverage at a fraction of the standard token count and runtime.

$(F^{(1)}, \ldots, F^{(T)})$. Each of these frames can be an **I-frame** (intra-coded), a **P-frame** (predictive), or optionally a **B-frame** (bi-directional predictive). Consecutive frames are organized in a *Group of Pictures* (GOP) structure as illustrated in Fig. 2.

I-frames. An I-frame, $I^{(t)}$, is an RGB image encoded independently without the use of preceding or subsequent frames. It is used as a reference point of the GOP and provides a full visual representation.

P-frames. A P-frame $P^{(t)}$ contains only the changes from the previous frame, be it a reference frame $I^{(t-1)}$ or a preceding P-frame $P^{(t-1)}$. The difference is defined by two components: (a) *Motion vectors* $\tau^{(t)}$, which describe block-wise displacements from the reference to the target frame, resembling coarse optical flow, and, (b) *Residuals* $\delta^{(t)}$, which capture block-wise pixel corrections that remain after motion compensation. The recurrence [87, 7] for reconstructing a P-frame is:

$$\hat{I}_i^{(t)} = \hat{I}_{i-\tau_i^{(t)}}^{(t-1)} + \delta_i^{(t)} \quad , \tag{1}$$

where i is the pixel coordinate index. $\hat{I}^{(t-1)}$ is the reference frame at time $t-1$: it equals the raw I-frame $I^{(t-1)}$ when $F^{(t-1)}$ is intra-coded, and the reconstructed frame from earlier P-frames otherwise. Hence, P-frames contain only incremental temporal information, and are therefore smaller in file size than I-frames.

B-frames. A B-frame leverages both preceding and subsequent frames to encode its differences. While this bidirectional prediction achieves the highest compression efficiency, it increases decoding complexity: B-frames must wait for future I/P-frames, creating a mismatch between decode and display order. Consequently, they are less suited for streaming or real-time use. We therefore focus on P-frames, which depend only on past references and align naturally with the causal processing required by VideoLMs.

Group of Pictures (GOP). A GOP is a cycle structure that comprises one I-frame together with any mixture of P- and/or B-frames, e.g., $I\ B\ P\ P\ B\ P$ etc. The GOP structure and length control the trade-offs between the compression efficiency, quality, and the capability for random access. Typical applications use varying configurations: all three frame types are used by the H.264 codec, while MPEG-4 uses the I- and P-frames (see App. H for a detailed explanation).

Implication for VideoLMs. Despite this rich structure, current VideoLMs discard codec information and fully decode and tokenize dense RGB frames, ignoring the inherent sparsity of P- and B-frames. This results in unnecessary computation and inflated token counts, as shown in Sec. 4.1.

3.2 CoPE-VideoLM

Our method explicitly leverages the GOP structure and introduces a *codec-aware tokenization framework* that seamlessly integrates with VideoLMs. Instead of unnecessarily encoding each frame as RGB patches [113] or compressing an entire GOP into a fixed-length summary [116], we retain I-frames as full RGB tokens and encode P-frames into lightweight and compact Δ-*tokens* (delta for difference) obtained from motion vectors and residuals. This preserves temporal ordering and appearance information without exceeding token budgets. An illustration is shown in Fig. 2.

Given a video $\mathcal{V} = (F^{(1)}, \ldots, F^{(T)})$, each frame $F^{(t)}$ is represented as:

$$F^{(t)} = \begin{cases} I^{(t)}, & \text{if } F^{(t)} \text{ is an I-frame,} \\ P^{(t)} = (\tau^{(t)}, \delta^{(t)}), & \text{if } F^{(t)} \text{ is a P-frame,} \end{cases} \tag{2}$$

where $\tau^{(t)}$ is a set of block-wise motion vectors (usually one per up to 16×16 macroblock), and $\delta^{(t)}$ is a set of block-wise residuals. For our downstream processing, we represent these as tensors: $\tau^{(t)} \in \mathbb{Z}^{H \times W \times 2}$ is the sparse tensor constructed from the block-wise motion vectors and $\delta^{(t)} \in \mathbb{R}^{H \times W \times C}$ is the sparse tensor of residuals.

I-frame processing. I-frames $I^{(t)}$ are passed through a frozen vision encoder ϕ_{RGB}, producing M dense tokens:

$$X_I^{(t)} = \phi_{\text{RGB}}(I^{(t)}) \in \mathbb{R}^{M \times d}. \tag{3}$$

P-frame processing. Each P-frame $P^{(t)}$ is mapped into a much more compact representation consisting of $N \ll M$ tokens by the Δ-Encoder ϕ_Δ:

$$X_P^{(t)} = \phi_\Delta(\tau^{(t)}, \delta^{(t)}) \in \mathbb{R}^{N \times d}. \tag{4}$$

P-frame fusion. In the setup described so far, processing all frames at the native frame rate is mandatory as codec primitives are defined relative to previous frames and skipping frames invalidates these dependencies. For example, let us consider a 30 FPS video with a GOP size of 240 frames (8 seconds). Our technique yields $M + 239N$ tokens per GOP, compared to $240M$ if all frames were encoded as RGB images. Fine-grained action recognition may require this exhaustive temporal coverage, but most video understanding tasks can often be done with sparser coverage.

Rather than processing all frames at the native frame rate, we can fuse s consecutive P-frames, encoding their combined changes relative to frame $F^{(t-s)}$ (rather than the immediately preceding frame). The motion vectors are propagated along the trajectory and we accumulate the displacement per frame and sample the next field already shifted by that displacement. Residuals are warping the corresponding grid according to the displacement accumulated per frame and then added together. The maximum number of P-frames that can be fused is bounded by the GOP size. In the running example, using $s = 30$ P-frames (1 FPS) for fusion reduces the per-GOP token count to $M + 7N$, which is much smaller than $M + 239N$ for full P-frame modeling and $8M$ for RGB encoding at 1 FPS. This fusion offers a codec-native way to trade temporal resolution for efficiency and can be tuned to match available compute and task requirements. For clarity, we treat the temporal index (t) as already incorporating the P-frame fusion, so $F^{(t)}$ always depends on $F^{(t-1)}$, though (t) may no longer correspond to the raw frame indices.

Δ-Encoder. The Δ-Encoder ϕ_Δ (Fig. 3) is designed to process the motion vectors $\tau^{(t)}$ and residuals $\delta^{(t)}$ through two specialized branches. The motion vectors $\tau^{(t)}$ are processed via a multi-layer MLP to extract local features over a grid of size $H_G \times W_G$ yielding features in $\mathbb{R}^{(H_G W_G) \times d}$. These features are then compressed via a motion transformer $\theta_{\texttt{motion}}$ with a set of K_τ learnable query tokens that can attend to all $H_G \times W_G$ input tokens and aggregate their information as:

$$\tau_{\texttt{tok}}^{(t)} = \theta_{\texttt{motion}}(\texttt{MLP}(\tau^{(t)})) \ . \tag{5}$$

Only these K_τ compressed motion tokens $\tau^{(t)}_{\texttt{tok}} \in \mathbb{R}^{K_\tau \times d}$ are used in the VideoLM. The residual $\delta^{(t)}$ is embedded by a lightweight ResNet-18 [27] module to extract local features over the same grid size $H_G \times W_G$. Similarly to above, a residual transformer $\theta_{\texttt{residual}}$ aggregates and compresses the raw features to a set of K_δ compressed residual tokens $\delta^{(t)}_{\texttt{tok}} \in \mathbb{R}^{K_\delta \times d}$ as:

$$\delta^{(t)}_{\texttt{tok}} = \theta_{\texttt{residual}}(\texttt{ResNet}(\delta^{(t)})) \ . \tag{6}$$

Interleaved token stream. In practice, we set $K_\tau = K_\delta = 4$ and thus $N = 8$ (see App. G.1; performance plateaus beyond $N = 8$). The final visual sequence $X = [x^{(1)}, \ldots, x^{(T)}]$ is formed by interleaving I-frame tokens $X_I^{(t)}$ and P-frame tokens $X_P^{(t)}$ in their natural temporal order, as shown in Eq. 2. The LLM can consume $X_I^{(t)}$ and $X_P^{(t)}$ alongside textual instructions, without any architectural modifications. This approach reduces the redundant data considerably while preserving accurate temporal coverage and enables smooth scalability to long videos by adjusting the I-frame density, P-frame grouping, and token allocation.

3.3 Training Paradigm

The training is done in two stages. First the Δ-encoder is pre-trained in order to render it compatible with the image encoder. Then, this encoder is integrated into a VideoLM and the whole pipeline is fine-tuned end-to-end.

Δ-encoder pre-training. The primary difficulty lies in ensuring that the Δ-tokens $X_P^{(t)}$ are aligned with the image tokens $X_I^{(t)}$. To achieve this, we pre-train the Δ-Encoder as a modality adapter that enables codec-derived primitives $(\tau^{(t)}, \delta^{(t)})$ to be compatible with the embedding space defined by a vision encoder. By aligning the Δ-tokens with this space, P-frames can then be compactly represented and replace standard RGB tokens within a VideoLM. For pre-training, two additional modules are used on top of the outputs of the Δ-encoder to process consecutive frame pairs. First, a "reference" transformer $\theta_{\texttt{ref}}$ uses the image tokens from the previous frame $I^{(t-1)}$ and the compressed motion vector tokens $\tau^{(t)}_{\texttt{tok}}$, and its aim is to understand how the information moved in the image and how this displacement transforms the reference tokens. This is akin to the warping in Eq. 1. Second, a "warped" transformer $\theta_{\texttt{warped}}$ takes these enriched tokens and the residual tokens $\delta^{(t)}_{\texttt{emb}}$ in order to add the residual information as in Eq. 1. These final features $\hat{X}_P^{(t)}$ should be similar to the image features extracted from the raw $\hat{I}^{(t)}$. The process can be summarized as:

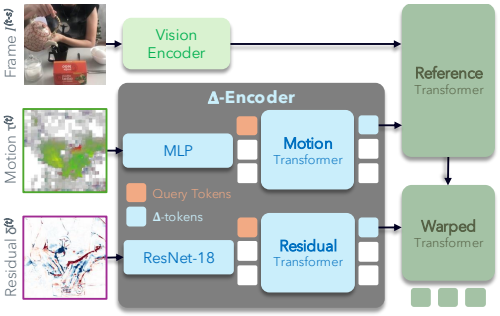


Figure 3: **Δ-encoder** processes motion vectors and residuals through two lightweight branches designed to extract and compress codec information. The resulting motion and residual tokens are concatenated to form the Δ-tokens used for our efficient P-frame representation, which is projected to the RGB token space during pre-training.

$$X_I^{(t-1)} = \phi_{RGB}(I^{(t-1)}) \tag{7}$$

$$\hat{X}^{(t-1)}_{\texttt{warped}} = \theta_{\texttt{ref}}(X_I^{(t-1)}, \tau^{(t)}_{\texttt{tok}}) \tag{8}$$

$$\hat{X}_P^{(t)} = \theta_{\texttt{warped}}(\hat{X}^{(t-1)}_{\texttt{warped}}, \delta^{(t)}_{\texttt{tok}}) \ . \tag{9}$$

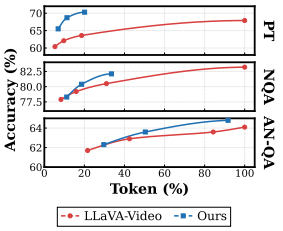
To align Δ-tokens with image tokens, we apply patch-wise regression against the outputs of a frozen vision encoder. Let $X_I^{(t)} = \phi_{\text{RGB}}(\hat{I}^{(t)}) \in \mathbb{R}^{M \times d}$ denote the tokens of the ground-truth target frame. We minimize:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{M} \sum_{i=1}^{M} \left\| X_I^{(t)}(i) - \hat{X}_P^{(t)}(i) \right\|_2^2. \tag{10}$$

Unlike global contrastive losses, this fine-grained objective encourages spatially consistent alignment across patches. As a result, the Δ-Encoder is able to produce representations that are more closely

Table 1: **Token Efficiency vs. Accuracy in Video QA.** We evaluate both CoPE-VideoLM and LLaVA-Video-7B at varying keyframe densities. For each configuration, our method uses the baseline's sampled keyframes as I-frames and adds P-frames per GOP as compact Δ-tokens at a marginal extra token cost. We report accuracy (Acc., %) and the % of tokens used compared to the default setup (64 keyframes). Our method consistently advances the Pareto-optimal frontier, achieving competitive accuracy at substantially reduced token counts.

Model	Sampling	PerceptionTest		NextQA		ActNet-QA	
		Token (%)	Acc. (%)	Token (%)	Acc. (%)	Token (%)	Acc. (%)
LLaVA-Video	1 keyframe / GOP	5.3	60.4 +5.1	8.3	77.9 +0.4	21.6	61.7 +0.6
Ours	+ 7 P-frames / GOP	6.9	65.5	11.2	78.3	29.7	62.3
LLaVA-Video	2 keyframes / GOP	9.7	62.1 +6.6	15.9	79.2 +1.2	42.5	62.9 +0.7
Ours	+ 6 P-frames / GOP	11.3	68.7	18.6	80.4	50.4	63.6
LLaVA-Video	4 keyframes / GOP	18.6	63.6 +6.7	31.0	80.5 +1.6	84.3	63.6 +1.2
Ours	+ 4 P-frames / GOP	20.1	**70.3**	33.5	82.1	91.7	**64.8**
LLaVA-Video	64 keyframes total	100	67.9	100	**83.2**	100	64.1



aligned with the RGB token space, aiming to improve the integration of I- and P-frames during downstream VideoLM training (see App. G.2; skipping pre-training and single stage end-to-end training weakens performance).

Integration into VideoLMs. After pre-training, we integrate the Δ-encoder ϕ_Δ into the VideoLM pipeline for full fine-tuning and interleave the I/ P-frames tokens. Note that the Reference and Warped transformers from the Δ-encoder pre-training stage are ***not*** used at this stage, so no RGB reference frames are processed for the P-frames when training the language model. This yields a substantial compute and memory reduction (refer to Sec. 4.3). The LLM architecture and training objective remain unchanged (standard instruction tuning / next-token prediction loss). The Δ-encoder adds fewer than 15M parameters, representing negligible overhead relative to the 7B LLM.

4 Experiments

Implementation Details. We build upon LLaVA-Video-7B [113] which consists of SigLIP-400M [100] as the vision encoder and Qwen2-7B [94] as the base LLM. Details about the Δ-encoder are in App. B. Training proceeds in two stages: first, we pre-train the Δ-encoder on frame pairs sampled from videos in LLaVA-Video-178K [113]. Second, we fine-tune the full VideoLM on the same dataset, comprising a total of 1.39M QA samples. Following prior work [116, 33, 87], during training, we re-encode videos to a standardized format ensuring uniform processing and consistent temporal resolution. Specifically, we use MPEG-4 encoded videos at 30 FPS with a fusion size of $s = 30$, yielding an effective sampling rate of 1 FPS (see App. G.5 for different fusion sizes). At inference, we use *ffmpeg* to extract I-frames and P-frame primitives via lightweight bitstream parsing to avoid full video decoding while applying the same 1s fusion. Further details are in App. C.

Evaluation. We evaluate our method across 14 video benchmarks spanning four categories: (i) *general video QA*: PerceptionTest [60], NextQA [89], ActNet-QA [99], and VideoMME [20]; (ii) *temporal reasoning*: TempCompass [51], TOMATO [65], CVRR-ES [36] and MVBench [42]; (iii) *long-form and instruction-following*: LongVideoBench [88], LVBench [79], Video-TT [114], and Video-MMMU [29]; and (iv) *spatial scene understanding*: ScanQA [3] and SQA3D [54]. We compare against LLaVA-Video-7B [113], our main baseline, to strictly isolate the impact of codec-aware tokenization, while also benchmarking against comparably scaled VideoLMs to demonstrate overall competitiveness. Comparisons against larger models or differently-trained backbones would conflate these factors with representation choice and are therefore beyond our scope.

Frame Sampling. Most open-sourced VideoLMs with a 32K token window sample 64 frames per video, regardless of length [53, 113]. As a result, unlike proprietary models, they cannot process videos at 1 FPS for videos longer than 64s. Our framework decouples frame rate from token budget: we sample at 1 FPS (counting both I- and P-frames), with the number of keyframes per GOP controlling the token cost rather than the temporal coverage. Concretely, each GOP contributes one I-frame (M tokens) plus up to seven P-frames ($N = 8$ tokens each), so the per-GOP cost ranges from $M + 7N$ (1 keyframe) to $4M + 4N$ (4 keyframes) depending on configuration. This allows the token

Table 2: **General video understanding benchmarks.** The best results among open-sourced methods are highlighted as **first**, second, and third. Our model performs better than open-sourced video language models of comparable parameter and training data scale. Notably, it does so while operating on a fraction of the visual tokens, confirming that codec primitives preserve the semantic and temporal cues needed for diverse understanding tasks.

Model	PerceptionTest	NextQA	ActivityNet-QA	VideoMME	
	val	mc	test	w/o sub	w sub
Proprietary Models					
GPT-5 [59]	-	86.3	-	83.3	86.9
Gemini 3 Pro [24]	-	84.3	-	88.6	87.5
Gemini 2.5 Pro [17]	-	85.3	-	87.8	87.8
Claude Sonnet 4.5 [2]	-	79.2	-	74.2	80.5
Open-sourced VideoLMs					
Video-LaVIT [33]	47.9	-	50.1	-	-
EMA-7B [116]	-	-	52.1	53.4	58.4
VILA-40B [46]	54.0	67.9	58.0	60.1	61.1
LongVA-7B [108]	-	68.3	50.0	52.6	54.3
IXC-2.5-7B [107]	34.4	71.0	52.8	55.8	58.8
LLaVA-OV-7B [39]	57.1	79.4	56.6	58.2	61.5
Apollo-7B [122]	67.3	-	-	61.3	63.3
Oryx-7B [53]	68.6	81.9	-	58.3	62.6
Qwen2-VL-7B [78]	62.3	81.2	-	**63.3**	69.0
LLaVA-Video-7B [113]	67.9	**83.2**	56.5	**63.3**	**69.7**
Ours-7B	**70.3**	82.1	**60.3**	61.9	69.4

budget to scale naturally with video duration. In our experiments, for fair comparison, we sample at 1 FPS up to 64 GOPs. For videos longer than 64 GOPs, we perform uniform GOP sampling and encode only I-/P-frames within each sampled GOP. Due to the compute-intensive nature of VideoLM evaluation, we prioritize author-published results but fill in missing results, as much as possible, with the best previously reported values from technical reports or papers [113]. All the baselines use 64 frames or 1 FPS sampling [105]

4.1 Token Efficiency vs. Video QA Accuracy

We first evaluate whether Δ-tokens from P-frames can match the understanding capabilities of a full 64 frame RGB-only sampling. Our framework decouples temporal coverage from token budget: at fixed I-frame density, additional P-frames extend coverage at marginal token cost. Tab. 1 reports results under different sampling configurations. Our codec-aware representation achieves consistent improvements in accuracy compared to the same-sampling baseline, and closes the gap to the denser sampling configuration while using significantly fewer tokens. The gain is not just a function of seeing more timestamps. On PerceptionTest [60], at 1 keyframe per GOP, adding 7 P-frames increases the token count by only 1.6% and pushes accuracy by 5.1%, exceeding the 4 keyframe per GOP RGB baseline while using $3\times$ fewer tokens. Δ-tokens deliver more accuracy per added token than additional RGB keyframes do, because they encode inter-frame change rather than re-encoding full images. To confirm the model actively relies on this temporal signal, we show that motion vectors and residuals contribute complementary information and removing Δ-tokens at inference causes a sharp drop in performance (App. G.3). Furthermore, the token-efficiency trend holds even when benchmarking against our own architecture restricted to keyframes-only (App. G.4). At higher frame densities, our model performs comparably to 64 keyframe LLaVA-Video-7B [113] while using only a fraction of its tokens. This demonstrates that the Δ-encoder successfully captures motion and appearance cues critical for temporal reasoning which sparse keyframe sampling discards.

4.2 Comparison against RGB-based VideoLMs

We benchmark CoPE-VideoLM against RGB-based VideoLMs across four task categories (see App. I for details on benchmarks): (i) general video QA (Tab. 2), (ii) temporal and motion reasoning (Tab. 3 (a)), (iii) long-form and instruction following (Tab. 3 (b)), and (iv) spatial scene understanding (see App. D; our method performs on par with 2D VLMs using only 25% of tokens).

General Video QA. Tab. 2 reports the performance across 4 diverse video understanding benchmarks; our codec-aware formulation achieves competitive or superior results across the board. By leveraging

Table 3: (a) **Temporal reasoning and motion understanding benchmarks.** CoPE-VideoLM achieves the highest accuracy on TempCompass, TOMATO, and CVRR-ES, confirming that codec primitives provide a strong inductive bias for temporal reasoning. (b) **Long-form and instruction-following benchmarks.** CoPE-VideoLM performs better than other open-sourced models on Video-TT, Video-MMMU, and LVBench, demonstrating compact Δ-tokens effectively scale to longer temporal contexts and complex instruction-following.

Model	TempCompass test MCQ	Tomato test	CVRR-ES test	MVBench test
Proprietary Models				
GPT-5 [59]	80.4	53.0	-	74.1
Gemini 3 Pro [24]	82.8	48.3	-	70.4
Gemini 2.5 Pro [17]	81.9	48.6	-	70.6
Claude Sonnet 4.5 [2]	72.8	39.6	-	62.1
Open-Sourced VideoLMs				
LongVA-7B [108]	56.9	-	-	-
VideoLLaMA2-7B [16]	-	18.5	21.6	54.6
InternVL2-8B [15]	65.3	21.7	-	65.8
VideoChat2-7B [42]	45.5	-	-	51.1
VideoCCAM-9B [19]	-	27.0	-	64.6
IXC-2.5-7B [107]	67.1	-	-	**69.1**
LLaVA-OV-7B [39]	64.8	25.5	42.6	56.7
Apollo-7B [122]	64.9	-	-	-
LLaVA-Video-7B [113]	66.6	24.9	43.6	58.6
Ours-7B	**68.9**	**28.4**	**49.4**	61.9

(a)

Model	Video-TT mc	Video-MMMU test	LVBench test	LongVideoBench val
Proprietary Models				
GPT-5 [59]	-	-	68.8	72.6
Gemini-3-Pro [24]	-	-	78.0	75.9
Gemini-2.5-Pro [17]	-	-	78.4	76.8
Claude-Sonnet-4.5 [2]	-	-	50.5	65.1
Open-Sourced VideoLMs				
EMA-7B [116]	-	-	-	47.0
LongVA-7B [108]	-	23.9	-	-
Kangaroo-8B [50]	-	-	39.4	54.8
mPLUG-Owl3-7B [98]	-	-	43.5	52.1
PLLaVA-34B [92]	-	-	26.1	53.2
InternVL2-8B [15]	-	37.4	-	54.6
TimeMarker [12]	-	-	41.3	56.3
LLaVA-OV-7B [39]	44.0	33.9	38.1	56.5
LLaVA-Video-7B [113]	41.8	36.1	44.2	**58.2**
Ours-7B	**45.5**	**38.2**	**46.4**	56.9

(b)

motion vectors and residuals directly from the compressed stream, CoPE-VideoLM encodes substantially more frames within the same token budget, enhancing temporal coverage without sacrificing spatial fidelity. We observe a performance gap on select benchmarks such as VideoMME compared to LLaVA-Video-7B. However, this stems from training data scale, not our architecture. When comparing models trained on the matched data scales (App. E), our codec formulation achieves superior performance. Notably, among codec-based approaches, CoPE-VideoLM outperforms both Video-LaVIT [33] and EMA [116] across benchmarks, validating that preserving residuals and temporal ordering yields stronger representations than fixed-length summaries or motion-only tokenization.

Temporal And Motion Reasoning. Tab. 3 (a) evaluates benchmarks specifically designed to probe temporal understanding. On TempCompass [51], TOMATO [65] and CVRR-ES [36], CoPE-VideoLM achieves the highest accuracy among all comparable baselines. This confirms that the explicit encoding of motion vectors and residuals provides a stronger temporal signal than dense RGB frame processing. These gains are notable given that temporal reasoning requires precisely the kind of fine-grained inter-frame dynamics that codec primitives natively capture. On MVBench, which emphasizes pictorial semantic understanding over temporal dynamics, our model improves over LLaVA-Video-7B by over 3.0%. The gap to the other models reflects the base model's capacity and training data mixture rather than a codec-specific limitation.

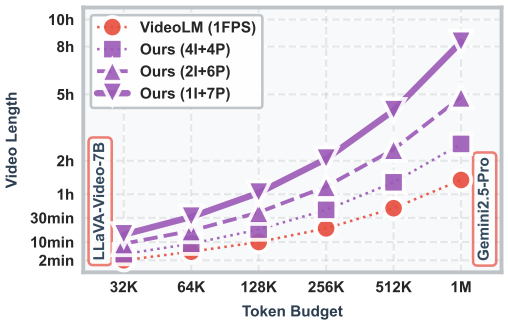
Long-form and Instruction-following. Tab. 3 (b) reports results on benchmarks requiring understanding of extended video sequences and complex instructions. CoPE-VideoLM achieves the best results among open-sourced baselines on Video-TT [114], Video-MMMU [29], and LVBench [79], while remaining competitive on LongVideoBench [88]. The improvement is especially pronounced against EMA [116], even though we train on less data (1.39M video-only vs. 2.4M video+image QA samples) and do not need to increase the GOP count at training time to handle longer videos. These results suggest that preserving residuals and explicit temporal ordering plays an important role for long-form understanding. Overall, compressing P-frames into compact Δ-tokens enables substantially more temporal coverage within the same token budget, supporting both fine-grained temporal reasoning and efficient use of the context window.

4.3 Runtime and Memory

Fig. 4 (a) reports *time-to-first-token* (TTFT) and *end-to-end-latency* (E2EL) to generate 64 text tokens, measured on a single consumer-grade GPU at 1 FPS video input. Compared to 64 keyframe LLaVA-Video-7B, our most compact configuration achieves 86.2% reduction in TTFT and a 56.1% faster E2EL. The improvement comes from skipping the full RGB encoding for P-frames and the shorter overall sequence length processed by the LLM due to the Δ-tokens. App. A shows interactive runtime comparisons across several multi-turn chat sessions on diverse video examples with $2.4 - 8.2\times$ faster TTFT and $76 - 86\%$ fewer tokens. Fig. 4 (b) shows how token budget scales with video duration for our Δ-token formulation. Standard RGB sampling saturates quickly, limiting coverage to short

Model / Sampling	TTFT (s)	E2EL (s)
LLaVA-Video-7B (64 kf)	2.39	3.78
Ours-7B		
1 keyframe per GOP (8 kf + 56 P-frames)	**0.33**	**1.66**
2 keyframes per GOP (16 kf + 48 P-frames)	0.51	1.71
4 keyframes per GOP (32 kf + 32 P-frames)	0.90	2.28

(a)



(b)

Figure 4: (a) **Runtime comparison:** TTFT and E2EL for generating 64 text tokens at 1 FPS on a single GPU. (b) **Video length vs. token budget:** Token budget is shown on a log scale; dashed lines mark evaluated budgets. Δ-token representation enables scaling to significantly longer videos without exceeding context limits.

sequences as memory constraints are encountered. CoPE-VideoLM enables processing up to $8\times$ longer videos within the same context window, demonstrating an order-of-magnitude increase in processing capability. Moreover, our Δ-encoder uses transformer components with well-established scaling properties. Together, these results confirm that codec-aware tokenization can achieve fast inference and long video coverage without architectural modifications or additional hardware.

5 Conclusion

Limitations and Future Work. Our current approach focuses on I- and P-frames, lacking support for B-frames and their complex non-causal dependencies. One option to address this would be using the decode order instead of the render order. Furthermore, we currently operate on a tensorized version of the codec primitives. For future work, it would be interesting to stay closer to the raw codec primitives, by operating directly on sets of block-wise motion vectors and quantized DCT coefficients, which could offer even better computational and token efficiency. Finally, we use a fixed P-frame fusion window, which is suboptimal for tasks with varying motion. Exploring sensitivity to codec type, bitrate, and encoding quality remains an open question.

Broader Societal Impact. CoPE-VideoLM cuts the latency and token cost of video understanding by reusing structure that codecs already compute, which makes long-form VideoLMs more practical to run in real-time and on-device settings such as robotics and assistive applications. Since the method reads directly from codec primitives, encoding artifacts or adversarial re-encodings can propagate into model outputs in ways that are harder to inspect than with raw RGB. As with any VideoLM, outputs may also include hallucinated or biased descriptions, so deployments of our model in sensitive settings should be paired with verifications.

Conclusion. Through comprehensive evaluation across 14 video understanding benchmarks, we have demonstrated that codec-aware tokenization offers an alternative to traditional keyframe sampling and outperforms prior codec-based approaches for VideoLMs, providing a compact, temporally rich representation that is complementary to any underlying VideoLM backbone. By leveraging the information natively encoded by video compression algorithms (i.e., motion vectors and residuals), we achieve substantial efficiency gains while maintaining competitive performance. Notably, our approach reduces time-to-first-token by a significant margin (up to 86%) which is essential for real-time applications. As models scale towards larger context windows, our approach becomes increasingly valuable, enabling richer temporal representations with much lower computational overhead than traditional sparse keyframe sampling. Since our approach operates at the representation level and requires no architectural changes to the base model, it can immediately benefit from future advances in VideoLM scale, training data, and context length, hence, positioning codec-aware tokenization as a broadly applicable efficiency primitive for video understanding.

Acknowledgements

The authors would like to thank (in alphabetical order): Isar Meijer and Krzysztof Waraksa from Microsoft for help with training pipeline setup; Kevin Qu, Tao Sun and Jianhao Zheng from Stanford for feedback at different stages of the project.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *ArXiv*, abs/2204.14198, 2022. URL https://api.semanticscholar.org/CorpusID:248476411. 3

[2] Anthropic. Claude sonnet 4.5 system card, 2025. URL https://assets.anthropic.com/m/12f214efcc2f457a/original/Claude-Sonnet-4-5-System-Card.pdf. 3, 8, 9

[3] Daichi Azuma, Taiki Miyanishi, Shuhei Kurita, and Motoaki Kawanabe. Scanqa: 3d question answering for spatial scene understanding. In *CVPR*, pages 19129–19139, 2022. 7, 20, 21

[4] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-VL: A Versatile Vision-Language Model for Understanding, Localization, Text Reading, and Beyond. *arXiv preprint arXiv:2308.12966*, 2023. 3

[5] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025. 3

[6] 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. *arXiv preprint arXiv:2502.13923*, 2025. 2, 3

[7] Shristi Das Biswas, Efstathia Soufleri, Arani Roy, and Kaushik Roy. Towards scalable modeling of compressed videos for efficient action recognition, 2025. URL https://arxiv.org/abs/2503.13724. 3, 4

[8] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *ICLR*, 2023. 3

[9] Shyamal Buch, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. Flexible Frame Selection for Efficient Video Reasoning. In *CVPR*, 2025. 2

[10] Jiawei Chen and Chiu Man Ho. Mm-vit: Multi-modal video transformer for compressed video action recognition. *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 786–797, 2021. URL https://api.semanticscholar.org/CorpusID:237266685. 3

[11] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models, 2024. 3, 21, 22

[12] Shimin Chen, Xiaohan Lan, Yitian Yuan, Zequn Jie, and Lin Ma. Timemarker: A versatile video-llm for long and short video understanding with superior temporal localization ability. *arXiv preprint arXiv:2411.18211*, 2024. 3, 9

[13] Sijin Chen, Xin Chen, Chi Zhang, Mingsheng Li, Gang Yu, Hao Fei, Hongyuan Zhu, Jiayuan Fan, and Tao Chen. Ll3da: Visual interactive instruction tuning for omni-3d understanding reasoning and planning. In *CVPR*, pages 26428–26438, 2024. 21

[14] Yilun Chen, Shuai Yang, Haifeng Huang, Tai Wang, Ruiyuan Lyu, Runsen Xu, Dahua Lin, and Jiangmiao Pang. Grounded 3d-llm with referent tokens. *arXiv preprint arXiv:2405.10370*, 2024. 21

[15] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024. 3, 9, 21

[16] 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 preprint arXiv:2406.07476*, 2024. URL https://arxiv.org/abs/2406.07476. 9

[17] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, 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. 2, 3, 8, 9

[18] Samuel Felipe dos Santos, Nicu Sebe, and Jurandy Almeida. Cv-c3d: Action recognition on compressed videos with convolutional 3d networks. In *2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 24–30, 2019. doi: 10.1109/SIBGRAPI.2019.00012. 3

[19] Jiajun Fei, Dian Li, Zhidong Deng, Zekun Wang, Gang Liu, and Hui Wang. Video-ccam: Enhancing video-language understanding with causal cross-attention masks for short and long videos, 2024. URL https://arxiv.org/abs/2408.14023. 9

[20] Chaoyou Fu, Yuhan Dai, Yondong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*, 2024. 7, 20, 25

[21] Rao Fu, Jingyu Liu, Xilun Chen, Yixin Nie, and Wenhan Xiong. Scene-llm: Extending language model for 3d visual understanding and reasoning. *arXiv preprint arXiv:2403.11401*, 2024. 21

[22] Tianyu Fu, Tengxuan Liu, Qinghao Han, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang. Framefusion: Combining similarity and importance for video token reduction on large visual language models. *arXiv preprint arXiv:2501.01986*, 2024. 3

[23] GeminiTeam. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. 2, 3

[24] Google. Gemini 3 Pro model card, 2025. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf. 8, 9

[25] Dan Grois, Alex Giladi, Kiho Choi, Min Woo Park, Yinji Piao, Min Woo Park, and Kwang Pyo Choi. Performance comparison of emerging evc and vvc video coding standards with hevc and av1. *SMPTE Motion Imaging Journal*, 130:1–12, 2021. URL https://api.semanticscholar.org/CorpusID:235569476. 3

[26] Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew M. Dai, and Dustin Tran. Training independent subnetworks for robust prediction. In *International Conference on Learning Representations*, 2021. 3

[27] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015. URL https://api.semanticscholar.org/CorpusID:206594692. 6, 19

[28] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *NeurIPS*, 36:20482–20494, 2023. 21

[29] Kairui Hu, Penghao Wu, Fanyi Pu, Wang Xiao, Yuanhan Zhang, Xiang Yue, Bo Li, and Ziwei Liu. Video-mmmu: Evaluating knowledge acquisition from multi-discipline professional videos. 2025. URL https://arxiv.org/abs/2501.13826. 7, 9, 26

[30] De-An Huang, Shijia Liao, Subhashree Radhakrishnan, Hongxu Yin, Pavlo Molchanov, Zhiding Yu, and Jan Kautz. Lita: Language instructed temporal-localization assistant. In *ECCV*, 2024. 3

[31] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023. 21

[32] Peng Jin, Ryuichi Takanobu, Caiwan Zhang, Xiaochun Cao, and Li Yuan. Chat-univi: Unified visual representation empowers large language models with image and video understanding. *arXiv preprint arXiv:2311.08046*, 2023. 3

[33] Yang Jin, Zhicheng Sun, Kun Xu, Liwei Chen, Hao Jiang, Quzhe Huang, Chengru Song, Yuliang Liu, Di Zhang, Yang Song, Kun Gai, and Yadong Mu. Video-lavit: Unified video-language pre-training with decoupled visual-motional tokenization. In *International Conference on Machine Learning*, pages 22185–22209, 2024. 2, 3, 7, 8, 9

[34] Zhao Jin, Munawar Hayat, Yuwei Yang, Yulan Guo, and Yinjie Lei. Context-aware alignment and mutual masking for 3d-language pre-training. In *CVPR*, pages 10984–10994, 2023. 21

[35] Kento Kawaharazuka, Jihoon Oh, Jun Yamada, Ingmar Posner, and Yuke Zhu. Vision-language-action models for robotics: A review towards real-world applications. *IEEE Access*, 13:162467–162504, 2025. doi: 10.1109/ACCESS.2025.3609980. 2

[36] Muhammad Uzair khattak, Muhammad Ferjad Naeem, Jameel Hassan, Naseer Muzzamal, Federico Tombari, Fahad Shahbaz Khan, and Salman Khan. How good is my video lmm? complex video reasoning and robustness evaluation suite for video-lmms. *arXiv:2405.03690*, 2024. 7, 9, 26

[37] Thorsten Laude, Yeremia Gunawan Adhisantoso, Jan Voges, Marco Munderloh, and Jörn Ostermann. A comprehensive video codec comparison. *APSIPA Transactions on Signal and Information Processing*, 8: e30, 2019. doi: 10.1017/ATSIP.2019.23. 3

[38] Didier Le Gall. Mpeg: a video compression standard for multimedia applications. *Commun. ACM*, 34 (4):46–58, April 1991. ISSN 0001-0782. doi: 10.1145/103085.103090. URL https://doi.org/10.1145/103085.103090. 2

[39] 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. 3, 8, 9

[40] Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning*, 2023. URL https://api.semanticscholar.org/CorpusID:256390509. 3

[41] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. VideoChat: Chat-Centric Video Understanding. *arXiv preprint arXiv:2305.06355*, 2023. 2, 3

[42] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206, 2024. 7, 9, 24, 26

[43] Xinhao Li, Yi Wang, Jiashuo Yu, Xiangyu Zeng, Yuhan Zhu, Haian Huang, Jianfei Gao, Kunchang Li, Yinan He, Chenting Wang, Yu Qiao, Yali Wang, and Limin Wang. Videochat-flash: Hierarchical compression for long-context video modeling. *arXiv preprint arXiv:2501.00574*, 2024. 3

[44] Yanwei Li, Chengyao Wang, and Jiaya Jia. LLaMA-VID: An Image is Worth 2 Tokens in Large Language Models. In *ECCV*, 2024. 3

[45] Bin Lin, Bin Zhu, Yang Ye, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023. 2

[46] Ji Lin, Hongxu Yin, Wei Ping, Yao Lu, Pavlo Molchanov, Andrew Tao, Huizi Mao, Jan Kautz, Mohammad Shoeybi, and Song Han. VILA: On Pre-training for Visual Language Models, 2023. 3, 8

[47] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved Baselines with Visual Instruction Tuning. *arXiv preprint arXiv:2310.03744*, 2023. 3

[48] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. *NeurIPS*, 2023. 3

[49] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge, 2024. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/. 3

[50] Jiajun Liu, Yibing Wang, Hanghang Ma, Xiaoping Wu, Xiaoqi Ma, xiaoming Wei, Jianbin Jiao, Enhua Wu, and Jie Hu. Kangaroo: A powerful video-language model supporting long-context video input. *arXiv preprint arXiv:2408.15542*, 2024. 9

[51] Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. Tempcompass: Do video llms really understand videos? *arXiv preprint arXiv: 2403.00476*, 2024. 7, 9, 24, 25

[52] Zhenyang Liu, Yongchong Gu, Sixiao Zheng, Yanwei Fu, Xiangyang Xue, and Yu-Gang Jiang. Trivla: A triple-system-based unified vision-language-action model with episodic world modeling for general robot control, 2025. URL https://arxiv.org/abs/2507.01424. 2

[53] Zuyan Liu, Yuhao Dong, Ziwei Liu, Winston Hu, Jiwen Lu, and Yongming Rao. Oryx mllm: On-demand spatial-temporal understanding at arbitrary resolution. *arXiv preprint arXiv:2409.12961*, 2024. 7, 8

[54] Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. In *ICLR*, 2023. 7, 20, 21

[55] Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision-language-action models for embodied ai, 2025. URL https://arxiv.org/abs/2405.14093. 2

[56] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. 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 (ACL 2024)*, 2024. 3

[57] Ming Nie, Dan Ding, Chunwei Wang, Yuanfan Guo, Jianhua Han, Hang Xu, and Li Zhang. Slowfocus: enhancing fine-grained temporal understanding in video llm. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385. 2

[58] OpenAI. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023. 2, 3

[59] OpenAI. GPT-5 system card, 2025. URL https://openai.com/index/gpt-5-system-card/. 3, 8, 9

[60] Viorica Pătrăucean, Lucas Smaira, Ankush Gupta, Adrià Recasens Continente, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Joseph Heyward, Mateusz Malinowski, Yi Yang, Carl Doersch, Tatiana Matejovicova, Yury Sulsky, Antoine Miech, Alex Frechette, Hanna Klimczak, Raphael Koster, Junlin Zhang, Stephanie Winkler, Yusuf Aytar, Simon Osindero, Dima Damen, Andrew Zisserman, and João Carreira. Perception test: A diagnostic benchmark for multimodal video models. In *Advances in Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=HYEGXFnPoq. 7, 8, 22, 24, 25

[61] Long Qian, Juncheng Li, Yu Wu, Yaobo Ye, Hao Fei, Tat-Seng Chua, Yueting Zhuang, and Siliang Tang. Momentor: Advancing video large language model with fine-grained temporal reasoning, 2024. 3

[62] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. URL https://api.semanticscholar.org/CorpusID:231591445. 3

[63] Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. TimeChat: A Time-sensitive Multimodal Large Language Model for Long Video Understanding. *arXiv*, abs/2312.02051, 2023. 3

[64] Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. FlashAttention-3: Fast and Accurate Attention with Asynchrony and Low-Precision. *NeurIPS*, 2024. 2

[65] Ziyao Shangguan, Chuhan Li, Yuxuan Ding, Yanan Zheng, Yilun Zhao, Tesca Fitzgerald, and Arman Cohan. Tomato: Assessing visual temporal reasoning capabilities in multimodal foundation models, 2024. URL https://arxiv.org/abs/2410.23266. 7, 9, 25

[66] Kele Shao, Keda Tao, Kejia Zhang, Sicheng Feng, Mu Cai, Yuzhang Shang, Haoxuan You, Can Qin, Yang Sui, and Huan Wang. When tokens talk too much: A survey of multimodal long-context token compression across images, videos, and audios. *arXiv preprint arXiv:2507.20198*, 2025. 3

[67] 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 Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 54582–54599. PMLR, 13–19 Jul 2025. 2

[68] Zheng Shou, Xudong Lin, Yannis Kalantidis, Laura Sevilla-Lara, Marcus Rohrbach, Shih-Fu Chang, and Zhicheng Yan. Dmc-net: Generating discriminative motion cues for fast compressed video action recognition, 2019. URL https://arxiv.org/abs/1901.03460. 3

[69] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Xun Guo, Tian Ye, Yan Lu, Jenq-Neng Hwang, et al. Moviechat: From dense token to sparse memory for long video understanding. *arXiv preprint arXiv:2307.16449*, 2023. 3

[70] Enxin Song, Wenhao Chai, Tian Ye, Jenq-Neng Hwang, Xi Li, and Gaoang Wang. Moviechat+: Question-aware sparse memory for long video question answering. *arXiv preprint arXiv:2404.17176*, 2024. 3

[71] Boyuan Sun, Jiaxing Zhao, Xihan Wei, and Qibin Hou. Llava-scissor: Token compression with semantic connected components for video llms. *arXiv preprint arXiv:2506.21862*, 2025. 2, 3, 21, 22

[72] Yunlong Tang, Jing Bi, Siting Xu, Luchuan Song, Susan Liang, Teng Wang, Daoan Zhang, Jie An, Jingyang Lin, Rongyi Zhu, Ali Vosoughi, Chao Huang, Zeliang Zhang, Pinxin Liu, Mingqian Feng, Feng Zheng, Jianguo Zhang, Ping Luo, Jiebo Luo, and Chenliang Xu. Video understanding with large language models: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1, 2025. doi: 10.1109/TCSVT.2025.3566695. 3

[73] 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. 3, 21, 22

[74] Hayato Terao, Wataru Noguchi, Hiroyuki Iizuka, and Masahito Yamamoto. Efficient compressed video action recognition via late fusion with a single network. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10096477. 3

[75] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aur'elien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*, 2023. 3

[76] Pavan Kumar Anasosalu Vasu, Fartash Faghri, Chun-Liang Li, Cem Koc, Nate True, Albert Antony, Gokula Santhanam, James Gabriel, Peter Grasch, Oncel Tuzel, et al. Fastvlm: Efficient vision encoding for vision language models. In *CVPR*, 2025. 2

[77] Haochen Wang, Yucheng Zhao, Tiancai Wang, Haoqiang Fan, Xiangyu Zhang, and Zhaoxiang Zhang. Ross3d: Reconstructive visual instruction tuning with 3d-awareness. *arXiv preprint arXiv:2504.01901*, 2025. 21

[78] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution. *arXiv preprint arXiv:2409.12191*, 2024. 3, 8, 21

[79] Weihan Wang, Zehai He, Wenyi Hong, Yean Cheng, Xiaohan Zhang, Ji Qi, Shiyu Huang, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. Lvbench: An extreme long video understanding benchmark, 2024. 7, 9, 26

[80] Xiao Wang, Guangyao Chen, Guangwu Qian, Pengcheng Gao, Xiao-Yong Wei, Yaowei Wang, Yonghong Tian, and Wen Gao. Large-scale multi-modal pre-trained models: A comprehensive survey. *Machine Intelligence Research*, 20(4):447–482, 2023. ISSN 2731-538X. doi: 10.1007/s11633-022-1410-8. URL https://www.mi-research.net/en/article/doi/10.1007/s11633-022-1410-8. 3

[81] Xiao Wang, Qingyi Si, Shiyu Zhu, Jianlong Wu, Li Cao, and Liqiang Nie. AdaReTaKe: Adaptive redundancy reduction to perceive longer for video-language understanding. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5417–5432, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.283. URL https://aclanthology.org/2025.findings-acl.283/. 3

[82] Yueqian Wang, Xiaojun Meng, Jianxin Liang, Yuxuan Wang, Qun Liu, and Dongyan Zhao. Hawkeye: Training video-text llms for grounding text in videos, 2024. 3

[83] Zehan Wang, Haifeng Huang, Yang Zhao, Ziang Zhang, and Zhou Zhao. Chat-3d: Data-efficiently tuning large language model for universal dialogue of 3d scenes. *arXiv preprint arXiv:2308.08769*, 2023. 21

[84] Zhengwei Wang, Qi She, and Aljosa Smolic. Team-net: Multi-modal learning for video action recognition with partial decoding. *ArXiv*, abs/2110.08814, 2021. URL https://api.semanticscholar.org/CorpusID:239015960. 2, 3

[85] Yuetian Weng, Mingfei Han, Haoyu He, Xiaojun Chang, and Bohan Zhuang. Longvlm: Efficient long video understanding via large language models. In *ECCV*, 2024. 3

[86] Thomas Wiegand, Gary J. Sullivan, Gisle Bjøntegaard, and Ajay Luthra. Overview of the h.264/avc video coding standard. *IEEE Trans. Circuits Syst. Video Technol.*, 13:560–576, 2003. URL https://api.semanticscholar.org/CorpusID:3540699. 2

[87] Chao-Yuan Wu, Manzil Zaheer, Hexiang Hu, R Manmatha, Alexander J Smola, and Philipp Krähenbühl. Compressed video action recognition. In *CVPR*, 2018. 2, 3, 4, 7

[88] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding, 2024. URL https://arxiv.org/abs/2407.15754. 7, 9, 26

[89] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9777–9786, 2021. 7, 22, 25

[90] Long Xing, Qidong Huang, Xiao wen Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, and Dahua Lin. Pyramiddrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. *ArXiv*, abs/2410.17247, 2024. URL https://api.semanticscholar.org/CorpusID:273507889. 3

[91] Yazhou Xing, Yang Fei, Yingqing He, Jingye Chen, Jiaxin Xie, Xiaowei Chi, and Qifeng Chen. Video-vae+: Large motion video autoencoding with cross-modal video vae. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. 3

[92] Lin Xu, Yilin Zhao, Daquan Zhou, Zhijie Lin, See Kiong Ng, and Jiashi Feng. Pllava : Parameter-free llava extension from images to videos for video dense captioning, 2024. 3, 9, 21, 22

[93] Mingze Xu, Mingfei Gao, Zhe Gan, Hong-You Chen, Zhengfeng Lai, Haiming Gang, Kai Kang, and Afshin Dehghan. Slowfast-llava: A strong training-free baseline for video large language models. *arXiv*, 2024. 3

[94] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024. URL https://arxiv.org/abs/2407.10671. 7

[95] Ruihan Yang, Qinxi Yu, Yecheng Wu, Rui Yan, Borui Li, An-Chieh Cheng, Xueyan Zou, Yunhao Fang, Hongxu Yin, Sifei Liu, Song Han, Yao Lu, and Xiaolong Wang. Egovla: Learning vision-language-action models from egocentric human videos, 2025. URL https://arxiv.org/abs/2507.12440. 2

[96] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. *arXiv preprint arXiv:2412.04467*, 2024. 21, 22

[97] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024. 3

[98] Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. mplug-owl3: Towards long image-sequence understanding in multi-modal large language models, 2024. URL https://arxiv.org/abs/2408.04840. 9

[99] Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *AAAI*, pages 9127–9134, 2019. 7, 22, 25

[100] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11941–11952, 2023. URL https://api.semanticscholar.org/CorpusID:257767223. 3, 7, 24

[101] Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, Peng Jin, Wenqi Zhang, Fan Wang, Lidong Bing, and Deli Zhao. VideoLLaMA 3: Frontier Multimodal Foundation Models for Image and Video Understanding. *arXiv preprint arXiv:2501.13106*, 2025. 2, 3

[102] Chuhan Zhang, Ankush Gupta, and Andrew Zisserman. Temporal query networks for fine-grained video understanding. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[103] Hang Zhang, Xin Li, and Lidong Bing. Video-LLaMA: An Instruction-tuned Audio-Visual Language Model for Video Understanding. *arXiv preprint arXiv:2306.02858*, 2023. 3

[104] Jiawei Zhang, Chejian Xu, and Bo Li. Chatscene: Knowledge-enabled safety-critical scenario generation for autonomous vehicles. In *CVPR*, pages 15459–15469, 2024. 21

[105] Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, et al. Lmms-eval: Reality check on the evaluation of large multimodal models. *arXiv preprint arXiv:2407.12772*, 2024. 8

[106] Pan Zhang, Xiaoyi Dong, Bin Wang, Yuhang Cao, Chao Xu, Linke Ouyang, Zhiyuan Zhao, Shuangrui Ding, Songyang Zhang, Haodong Duan, Wenwei Zhang, Hang Yan, Xinyue Zhang, Wei Li, Jingwen Li, Kai Chen, Conghui He, Xingcheng Zhang, Yu Qiao, Dahua Lin, and Jiaqi Wang. Internlm-xcomposer: A vision-language large model for advanced text-image comprehension and composition. *arXiv preprint arXiv:2309.15112*, 2023. 3

[107] Pan Zhang, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Rui Qian, Lin Chen, Qipeng Guo, Haodong Duan, Bin Wang, Linke Ouyang, Songyang Zhang, Wenwei Zhang, Yining Li, Yang Gao, Peng Sun, Xinyue Zhang, Wei Li, Jingwen Li, Wenhai Wang, Hang Yan, Conghui He, Xingcheng Zhang, Kai Chen, Jifeng Dai, Yu Qiao, Dahua Lin, and Jiaqi Wang. Internlm-xcomposer-2.5: A versatile large vision language model supporting long-contextual input and output. *arXiv preprint arXiv:2407.03320*, 2024. 3, 8, 9

[108] 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. *arXiv preprint arXiv:2406.16852*, 2024. URL https://arxiv.org/abs/2406.16852. 3, 8, 9

[109] Qiming Zhang, Yufei Xu, Jing Zhang, and Dacheng Tao. Vsa: learning varied-size window attention in vision transformers. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV*, pages 466–483. Springer, 2022. 3

[110] Qiming Zhang, Jing Zhang, Yufei Xu, and Dacheng Tao. Vision transformer with quadrangle attention. *arXiv preprint arXiv:2303.15105*, 2023. 3

[111] Shaolei Zhang, Qingkai Fang, Zhe Yang, and Yang Feng. Llava-mini: Efficient image and video large multimodal models with one vision token, 2025. URL https://arxiv.org/abs/2501.03895. 3

[112] Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. In *International Conference on Machine Learning*, 2025. 3

[113] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data, 2024. URL https://arxiv.org/abs/2410.02713. 2, 3, 5, 7, 8, 9, 19, 20, 21, 25

[114] Yuanhan Zhang, Yunice Chew, Yuhao Dong, Aria Leo, Bo Hu, and Ziwei Liu. Towards video thinking test: A holistic benchmark for advanced video reasoning and understanding, 2025. URL https://arxiv.org/abs/2507.15028. 7, 9, 26

[115] Yunzhu Zhang, Yu Lu, Tianyi Wang, Fengyun Rao, Yi Yang, and Linchao Zhu. Flexselect: Flexible token selection for efficient long video understanding, 2025. URL https://arxiv.org/abs/2506.00993. 3

[116] Zijia Zhao, Yuqi Huo, Tongtian Yue, Longteng Guo, Haoyu Lu, Bingning Wang, Weipeng Chen, and Jing Liu. Efficient motion-aware video mllm. *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24159–24168, 2025. URL https://api.semanticscholar.org/CorpusID:277104804. 2, 3, 5, 7, 8, 9

[117] Duo Zheng, Shijia Huang, and Liwei Wang. Video-3d llm: Learning position-aware video representation for 3d scene understanding. *arXiv preprint arXiv:2412.00493*, 2024. 21

[118] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *arXiv preprint arXiv:2306.05685*, 2023. 3

[119] Bin Zhu, Bin Lin, Munan Ning, Yang Yan, Jiaxi Cui, HongFa Wang, Yatian Pang, Wenhao Jiang, Junwu Zhang, Zongwei Li, et al. Languagebind: Extending video-language pretraining to n-modality by language-based semantic alignment. *arXiv preprint arXiv:2310.01852*, 2023. 2

[120] Chenming Zhu, Tai Wang, Wenwei Zhang, Jiangmiao Pang, and Xihui Liu. Llava-3d: A simple yet effective pathway to empowering lmms with 3d-awareness. *arXiv preprint arXiv:2409.18125*, 2024. 21

[121] Ziyu Zhu, Xiaojian Ma, Yixin Chen, Zhidong Deng, Siyuan Huang, and Qing Li. 3d-vista: Pre-trained transformer for 3d vision and text alignment. In *ICCV*, pages 2911–2921, 2023. 21

[122] Orr Zohar, Xiaohan Wang, Yann Dubois, Nikhil Mehta, Tong Xiao, Philippe Hansen-Estruch, Licheng Yu, Xiaofang Wang, Felix Juefei-Xu, Ning Zhang, Serena Yeung-Levy, and Xide Xia. Apollo: An exploration of video understanding in large multimodal models. *arXiv preprint arXiv:2412.10360*, 2024. 3, 8, 9

Appendix

In the appendix, we provide the following:

1. *Runtime Comparison (Sec. A)*
2. *Details about Δ-encoder (Sec. B)*
3. *Training details (Sec. C)*
4. *Spatial video question-answering (Sec. D)*
5. *Scale of training data (Sec. E)*
6. *Comparison with token pruning (Sec. F)*
7. *Ablation study (Sec. G)*
    - *Varying the number of Δ-tokens (Sec. G.1)*
    - *Two-Stage training (Sec. G.2)*
    - *Are Δ-tokens used by the VideoLM? (Sec. G.3)*
    - *Benefits of codec primitives (Sec. G.4)*
    - *Scaling to higher frame rates (Sec. G.5)*
    - *Next-frame retrieval using the Δ-encoder (Sec. G.6)*
8. *Video Decoding Illustration (Sec. H)*
9. *Details on training data and evaluation benchmarks (Sec. I)*
10. *Qualitative results (Sec. J)*

A Runtime Comparison

We provide supplementary videos demonstrating side-by-side interactive comparisons between CoPE-VideoLM and LLaVA-Video [113] across several examples. The videos illustrate the difference in TTFT and E2EL during multi-turn chat sessions at 1 FPS video input. Across all the examples, CoPE-VideoLM achieves $2.4 - 8.2\times$ faster TTFT while using $76 - 86\%$ fewer tokens, demonstrating the real-world user experience benefit of codec-aware tokenization.

Extraction overhead. Including extraction and model processing at 64 frame positions, CoPE-VideoLM takes 0.47s versus 3.50s for the RGB baseline ($7.4\times$ speedup). When localized transcoding of selected GOPs is required, the total increases to 0.63–0.69s, retaining a 5.1–$5.6\times$ speedup.

B Details about Δ-Encoder

The Δ-Encoder converts codec primitives, motion vectors and residuals, into a compact set of $N = K_\tau + K_\delta$ tokens aligned with the vision encoder's embedding space. The entire module is lightweight (< 15M parameters) and operates purely in the compressed domain during VideoLM fine-tuning. Here, we provide additional architectural and implementation details.

Motion Vector Branch. Given motion vectors $\tau^{(t)} \in \mathbb{Z}^{H\times W\times 2}$, we first perform min–max normalization to map all values to $[-1, 1]$. We then "patchify" the motion-field into non-overlapping 16×16 blocks, yielding a grid of size $H_G \times W_G$ where $H_G = H/16$ and $W_G = W/16$. Each patch is flattened into a vector of dimension $16^2 \times 2$, and a lightweight two-layer MLP with shared weights is applied independently to each patch, producing per-block embeddings with the same feature dimension as the vision encoder ($d = 1152$). To aggregate these features, we employ a transformer equipped with K_τ learnable query tokens. These tokens are concatenated with the motion features along the sequence length dimension at the input of the transformer and then processed together using regular multi-headed attention layers. The transformer contains 4 layers with hidden dimension d, uses 9 attention heads, and adopts PreNorm residual blocks throughout.

Residual Branch. Residuals, extracted directly as RGB tensors $\delta^{(t)} \in \mathbb{R}^{H\times W\times 3}$ during bitstream parsing, are processed through a truncated ResNet-18 backbone (all convolutional layers up to the final global pooling) [27]. The resulting spatial features share the same grid resolution as above

$(H_G \times W_G)$. A second transformer, architecturally identical to the motion branch but with its own learned K_δ queries, is used to compress these features.

Pre-training the Δ-Encoder. As mentioned in Sec. 3.3, the Δ-Encoder is augmented with two auxiliary transformer modules, *reference* and *warped* branches, that enable it to reconstruct the token representation of the target frame without decoding RGB pixels. These transformers are architecturally identical to the ones in the motion vector and residual branch.

C Training Details

Data Setup. As mentioned in Sec. 4, during training, we use MPEG-4 encoded videos at 30 FPS with aggregation to fuse P-frame primitives over 1s. To optimally align the Δ-tokens with the RGB latent space, we empirically found that alternating I- and P-frames in the temporal sequence yields the best performance. Specifically, for our main training run on LLaVA-Video-178K [113], we utilize a GOP size of 240, enabling a balanced sampling of 4 keyframes and 4 P-frames per GOP at a 1 FPS effective rate. For spatial video question answering fine-tuning, we adopt a lower GOP and fusion size (details in App. D). Crucially, during inference, we make no assumptions regarding the native GOP size, extracting I- and P-frames with fusion via *ffmpeg* directly from the compressed stream.

Pre-training. We pre-train the Δ-encoder with the *warped* and *reference* branches together for 1.5K GPU-hours using 16×A100-80G, running 113K iterations with a global batch size of 1024 and a LR $1e-3$, optimized with AdamW and a cosine scheduler with 1000 warmup steps.

VideoLM training. In the second phase, we fine-tune the Δ-encoder, adapter, and LLM decoder with the vision encoder frozen. We use a learning rate of 1e−5, an effective total batch size of 128, and train for 21K GPU hours on 64× A100-80G, for 10.9K steps. We keep the same hyperparameter settings as LLaVA-Video-7B [113].

D Spatial Video Question Answering

We evaluate our method on two standard 3D QA benchmarks: (1) SQA3D [54] for situated reasoning, and (2) ScanQA [3], for spatial understanding. Both datasets require associating multi-view observations with 3D spatial structure, making them a natural testbed for assessing whether compressed-domain video cues can support geometry-aware reasoning.

Training. We follow the same training pipeline used in our video-language experiments. However, since these videos are comparatively shorter in duration ($\sim$ 15s total), we re-encode them with a GOP size of 120 at 30 FPS with accumulation size $s = 10$. This leads to around 3-4 GOPs per video. Following standard practice in 3D LMMs, we fine-tune our base model on ScanQA and SQA3D training split, without any dataset-specific heuristics or architectural modifications. We train and evaluate with 6 keyframes + 6 P-frames per GOP to align better with 32 frame setting of the RGB-based models.

Results. We report the results in Tab. D.1. To maintain fairness with VideoLMs, we show results for both our original and fine-tuned model. Despite using a quarter of the number of tokens compared to LLaVA-Video-7B [113], the original CoPE-VideoLM matches state-of-the-art VideoLMs. With fine-tuning, our performance becomes comparable to the leading 3D VLMs, despite our method not having access to camera poses or 3D point-clouds. However, we note that fine-tuning would likely benefit other VideoLMs similarly; the key takeaway is that codec-aware tokens retain sufficient spatial information for 3D reasoning at a fraction of the token cost.

E Scale of Training Data

As mentioned in Sec. 4.2, CoPE-VideoLM underperforms LLaVA-Video on select benchmarks, e.g., VideoMME [20]. This gap reflects training data scale and composition rather than the codec formulation. Tab. E.1 isolates this effect by reporting results from the LLaVA-Video paper [113] across its incremental training stages. LLaVA-Video begins with 0.25M samples (LLaVA-Hound)

Table D.1: **Evaluation of 3D question-answering** on SQA3D [54] and ScanQA [3]. "Expert models" are customized for specific tasks with task-oriented decoders. "EM" stands for top-1 exact match and "EM-R" means the refined exact match following [31]. "–" indicates the number is not available. We show results for our method in zero-shot setup where it performs comparable to state-of-the-art VideoLMs despite using only 25% of tokens. We further show results after fine-tuning where the performance increases significantly, even outperforming a significant number of 3D VLMs which employ additional inputs (e.g., point-clouds, camera poses).

Method	Point Encoder	Vision Encoder	$SQA3D_{test}$		$ScanQA_{val}$				
			EM	EM-R	CIDEr	BLEU-4	METEOR	ROUGE	EM
Expert Models									
SQA3D [54]	✓	✗	46.6	–	–	–	–	–	–
ScanQA [3]	✓	✗	–	–	64.9	10.1	13.1	33.3	21.1
3D-VLP [34]	✓	✗	–	–	–	11.2	13.5	34.5	21.7
3D-VisTA [121]	✓	✗	–	–	–	–	13.9	35.7	22.4
3D VLMs									
Chat-3D [83]	✓	✗	–	–	53.2	6.4	11.9	28.5	–
3D-LLM [28]	✓	✓	–	–	69.4	12.0	14.5	35.7	20.5
Scene-LLM [21]	✓	✓	53.6	–	80.0	11.7	15.8	35.9	27.2
LL3DA [13]	✓	✗	–	–	76.8	–	15.9	37.3	–
LEO [31]	✓	✓	50.0	52.4	80.0	11.5	16.2	39.3	21.5
ChatScene [104]	✓	✓	54.6	57.5	87.7	14.3	18.0	41.6	21.6
Grounded 3D-LLM [14]	✓	✓	–	–	72.7	13.4	–	–	–
LLaVA-3D [120]	✗	✓	55.6	57.6	91.7	14.5	20.7	50.1	27.0
Video-3D-LLM [117]	✗	✓	58.6	–	102.1	16.4	20.0	49.3	30.1
Ross3D [77]	✗	✓	**63.0**	**65.7**	**107.0**	**17.9**	**20.9**	**50.7**	**30.8**
VideoLMs (Zero-shot)									
InternVL2-8B [15]	✗	✓	33.0	45.3	62.5	3.3	14.5	34.3	–
Qwen2-VL-7B [78]	✗	✓	40.7	46.7	53.9	3.0	11.4	29.3	–
LLaVA-Video-7B [113] 100% tokens	✗	✓	48.5	–	88.7	3.1	17.7	44.6	–
Ours-7B *(Zero-shot)* 25.78% tokens	✗	✓	46.5	49.8	70.9	7.1	14.7	38.2	–
Ours-7B *(Fine-tuned)* 25.78% tokens	✗	✓	56.6	59.3	96.9	14.9	19.1	46.4	27.5

and progressively adds LLaVA-Video-178K (1.58M), three QA datasets (1.64M), and finally 1.1M LLaVA-OneVision images (2.74M). Two observations stand out. First, VideoMME improves substantially with the image alignment data (61.9 → 63.4), which our pipeline lacks entirely. Second, adding the QA datasets reduces VideoMME (63.2 → 61.9). Since this configuration closely mirrors our data mixture, and we observe similar VideoMME accuracy, this confirms a data-mixture effect rather than a codec-specific limitation. We additionally include a "sampled" variant as reported in the LLaVA-Video paper [113], where LLaVA-Video is trained on a randomly sampled 1.08M subset of LLaVA-Video-178K; our method outperforms it by a wide margin across all benchmarks.

Table E.1: **Effect of training data scale.** LLaVA-Video rows are reported in [113] across incremental training stages. Against the most comparable configurations (+ 3 QA datasets and "sampled"), our method outperforms or remains on par across benchmarks despite using fewer samples.

Training Data	**Total Data**	**NextQA**	**PerceptionTest**	**VideoMME**
LLaVA-Video [113]				
LLaVA-Hound	0.25M	64.4	51.4	54.1
+ LLaVA-Video-178K	1.58M	80.1	57.1	63.2
+ 3 QA datasets	1.64M	80.1	69.0	61.9
+ LLaVA-OV (Images)	2.74M	**83.2**	67.9	**63.4**
LLaVA-Video-178K (sampled)	1.08M	73.2	55.9	59.6
Ours				
LLaVA-Video-178K + 3 QA datasets	1.39M	82.1	**70.3**	61.9

F Comparison with Token Pruning

As discussed in Sec. 2, token compression methods operate on dense vision tokens *after* full RGB encoding. Although they reduce the LLM's token count, these still pay the full image embedding cost for every frame, which dominates prefill latency. In contrast, CoPE-VideoLM avoids full image encoding for most frames entirely by leveraging codec-level sparsity. Tab. F.1 compares CoPE-VideoLM against FastV [11], DyCoke [73], PLLaVA [92], VisionZip [96], and LLaVA-Scissor [71].

All baselines use 50% token budget of the default frame configuration, while our method enables up to 93% lower token usage, as shown in Sec. 4.1.

Table F.1: **Comparison with token compression methods.** CoPE-VideoLM outperforms post-hoc pruning approaches across all three benchmarks while being faster in TTFT, as P-frames bypass the vision encoder entirely.

Method	ActNet-QA	Next-QA	VideoMME w/o sub
FastV [11]	47.95	81.1	57.5
DyCoke [73]	47.88	81.1	57.4
PLLaVA [92]	47.59	81.0	56.9
VisionZip [96]	45.42	78.5	54.2
LLaVA-Scissor [71]	47.89	81.2	57.4
Ours	**60.28**	**82.1**	**61.9**

G Ablation Study

To reduce carbon footprint and computation cost compared to the main high compute run, for the ablation experiments, we train CoPE-VideoLM on 3 QA datasets: PerceptionTest [60], NextQA [89], and ActNet-QA [99], comprising 60K samples for 2 days on 16×A100. Due to this, the results shown in Sec. G.1, Sec. G.2, and Sec. G.3 are not directly comparable to these reported in Sec. 4.

G.1 Varying the Number of Δ-Tokens

The number of Δ-tokens emitted per P-frame controls the expressive capacity of the codec branch. Fewer tokens encourage a more aggressively compressed representation, whereas larger token budgets allow the Δ-Encoder to retain finer motion and appearance cues from the codec primitives. Tab. G.1 reports the effect of varying this value while keeping all other model and sampling configurations fixed. We observe a consistent trend across both benchmarks. Moving from 2 to 4 Δ-tokens yields a noticeable improvement, as the model benefits from an expanded latent space that can more faithfully capture motion and residual structure. Increasing to 8 Δ-tokens further improves performance, reaching the best overall results. Beyond this point, however, allocating more capacity produces diminishing returns: both NextQA and PerceptionTest remain nearly unchanged when increasing to 16 tokens. This indicates that the codec primitives contain a limited amount of signal per fused P-frame, and that 8 tokens are sufficient to encode the relevant temporal variations. For this reason, we adopt 8 Δ-tokens per P-frame as the default configuration throughout, balancing accuracy and token efficiency.

Table G.1: **Number of Δ-tokens per P-frame.** We train several versions of our model with a different number of Δ-tokens per P-frame. Performance significantly increases when going from aggressive compression (2 or 4 tokens) up to 8, which is the value we used for the results in the main paper. Going to 16 further improves performance, but with diminishing returns that do not justify the increased token cost.

Δ-tokens per P-Frame	PerceptionTest	NextQA
1 Keyframe per GOP		
2	63.26 +2.42	75.04 +1.04
4	65.68 +1.65	76.08 +1.29
8	67.33 +0.36	77.37 +0.52
16	67.69	77.89

G.2 Two-Stage Training

We ablate the contribution of each component in our two-stage training pipeline: (i) pre-training the Δ-Encoder to align codec primitives with the RGB embedding space, and (ii) end-to-end fine-tuning of the full VideoLM. The results are shown in Tab. G.2. Fine-tuning the VideoLM without any Δ-encoder pre-training yields reasonable performance, though noticeably worse than the full two-stage approach. In this one-stage setting, the model must simultaneously learn motion–residual interpretation, feature alignment, and multimodal reasoning, which slows convergence and leads to

Table G.2: **Two-stage training.** We attempt to directly train the Δ-encoder together with the LLM without our initial pre-training scheme. This yields significantly lower performance than the two stage setup proposed in the main paper: Stage 1 – Pre-train the Δ-encoder and Stage 2 – Fine-tune pre-trained Δ-encoder & pre-trained LLM together.

Pre-train Δ-Encoder	Fine-tune LLM	PerceptionTest	NextQA
1 Keyframe per GOP			
✗	✓	63.45	74.56
✓	✓	**67.33**	**77.37**

weaker temporal understanding, particularly on PerceptionTest. The best results are achieved when both stages are used. Pre-training ensures that Δ-tokens inhabit a well-structured embedding space, while fine-tuning teaches the LLM how to fuse the I- and P-frame tokens together. This combination consistently provides the strongest performance across both datasets. While two-stage training is clearly advantageous in our data regime, we note that the benefit may diminish when training on significantly larger datasets. In such high-data settings, one-stage training may gradually compensate for the missing pre-training at the cost of more compute.

G.3 Are Δ-tokens used by the VideoLM?

A natural question is whether the VideoLM actually *uses* the Δ-tokens during inference, rather than ignoring them and relying solely on sparse keyframes. To probe this, we evaluate with 1 keyframe + 7 P-frames per GOP but *zero out* all P-frame Δ-tokens at inference time, preserving temporal structure while removing all codec information. We additionally compare motion-only and residual-only variants, and shuffle the full model's Δ-tokens at inference while preserving their content, isolating the contribution of temporal order. As shown in Tab. G.3, performance drops substantially when tokens are zeroed, confirming that the model actively leverages Δ-tokens for temporal reasoning rather than merely interpolating between sparse I-frames.

Table G.3: **Ablation on the use of Δ-tokens.** We evaluate zeroed tokens, individual codec primitives, and shuffled token order at 1 keyframe + 7 P-frames per GOP. Both codec primitives and their temporal ordering contribute to performance.

Configuration	PerceptionTest	NextQA
1 Keyframe per GOP		
+ Δ-tokens = 0	64.4	74.2
+ Only motion vectors	65.1	75.3
+ Only residuals	65.9	76.1
+ Δ-tokens shuffled (inference only)	64.8	74.8
+ Δ-tokens	**67.3**	**77.4**

G.4 Benefits of Codec Primitives

In Sec. 4.1, we compare our method against LLaVA-Video-7B at matched keyframe counts. To disentangle the contribution of Δ-tokens at inference from the effect of training with codec primitives, we evaluate our model using only I-frames (Tab. G.4). Two findings emerge. First, our model with I-frames only already outperforms LLaVA-Video-7B at the same keyframe density, indicating that fine-tuning with interleaved codec primitives teaches the model stronger temporal representations that transfer even when Δ-tokens are absent at inference. Second, adding Δ-tokens on top consistently improves accuracy across all benchmarks, confirming that codec primitives provide complementary temporal signals beyond what the model internalizes during training. Together, these results show that codec-aware training and Δ-token inference are both independently beneficial and additive. Keeping 4 keyframes per GOP fixed, we additionally vary the number of P-frame groups from 0 to 4. Accuracy increases consistently across all five benchmarks, showing the benefit of additional P-frame information at fixed RGB coverage.

Table G.4: **Controlled ablation: codec-aware training with and without Δ-tokens.** We evaluate our model using only I-frames at each keyframe density, removing Δ-tokens at inference to isolate their contribution. Even without Δ-tokens, our model outperforms LLaVA-Video at matched keyframe counts, showing that fine-tuning with codec primitives alone strengthens temporal reasoning. Adding Δ-tokens consistently improves accuracy further, confirming they provide complementary temporal signals at inference.

Sampling	PerceptionTest	NextQA	ActNet-QA	TempCompass	TOMATO
	val	mc	test	test-mc	test
1 keyframe / GOP	63.1	78.0	61.8	60.3	22.4
+ 7 P-frames / GOP	65.5	78.3	62.3	62.4	26.2
2 keyframes / GOP	66.2	79.4	62.5	64.5	26.5
+ 6 P-frames / GOP	68.7	80.4	63.6	65.6	27.1
4 keyframes / GOP	68.1	80.8	63.7	66.2	26.9
+ 1 P-frame / GOP	68.8	81.3	63.9	66.9	27.2
+ 2 P-frames / GOP	69.4	81.6	64.5	67.5	27.8
+ 4 P-frames / GOP	**70.3**	**82.1**	**64.8**	**68.9**	**28.4**

G.5 Scaling to Higher Frame Rates

A key advantage of our tokenization is that P-frames are represented with only $N{=}8$ Δ-tokens, compared to $M{=}196$ tokens for each I-frame, enabling higher effective frame rates under a constrained token budget. To study this trade-off, we increase the effective FPS from 1 to 3 by reducing the P-frame fusion window s from 30 to 10 frames, thereby encoding finer-grained temporal changes at increasing token cost. We evaluate on TempCompass [51] and MVBench [42], as both benchmarks are sensitive to temporal coverage. As shown in Tab. G.5, both benchmarks improve from 1 to 2 FPS, with MVBench gaining over 1.8% and TempCompass nearly 2%. However, performance slightly decreases at 3 FPS on both benchmarks. We note that our model is trained exclusively with $s{=}30$ (1 FPS); the performance drop at higher frame rates likely reflects a train-test mismatch rather than a fundamental limitation, and could be addressed by training with varied fusion windows. Importantly, these improvements show that our framework enables a practical and flexible trade-off between temporal resolution and token efficiency that would be prohibitive with RGB representations.

Table G.5: **Effect of higher frame rates.** We increase the effective FPS by reducing the P-frame fusion window s, encoding finer temporal changes at modest token cost. All configurations use 1 keyframe per GOP. Each I-frame contributes 196+14 tokens (vision + newline), and each P-frame group contributes 8+2 tokens (Δ-tokens + newline), giving $(196{+}14) \times K + (8{+}2) \times P$ tokens per GOP, where K and P are the number of I-frames and P-frame groups respectively.

FPS	Fusion s	P-frames / GOP	Num. Tokens / GOP	TempCompass	MVBench
1	30	4	880	66.89	59.98
2	15	8	1760	**68.86**	**61.87**
3	10	12	2640	68.73	61.67

G.6 Next-Frame Retrieval using Δ-Encoder

To assess the representational quality of our compressed-domain features, we evaluate next-frame retrieval at 1 FPS on PerceptionTest [60]. Formally, the task can be defined as: given a query frame $I^{(t-1)}$ at time $t-1$, the task is to identify its true successor frame $I^{(t)}$ at time t from a database containing all frames from the same video (except itself $I^{(t-1)}$). As a baseline, SigLIP [100] processes the raw RGB frame $I^{(t-1)}$, whereas our model uses $I^{(t-1)}$ together with the current motion vectors $\tau^{(t)}$ and residuals $\delta^{(t)}$ to produce $N{=}8$ Δ-tokens per P-frame via the Δ-Encoder and lightweight transformer branches used during pre-training. For this experiment, we use the intermediate checkpoint after pre-training, as it is compatible with the "reference" and "warped" transformers that are used to transform the features of the previous frame. The results are shown in Tab. G.6. Our method outperforms the baseline by a large margin across the board. When retrieving a single frame, the relatively low absolute performance is expected, as subsequent frames are heavily similar even at 1 FPS, but our method successfully leverages the codec primitives to significantly improve the performance. Furthermore, our method achieves almost perfect performance at 5 frames,

with its very high recall of 94.86%. These strong improvements over SigLIP confirm that the Δ-Encoder preserves semantically meaningful motion and appearance cues that are critical for retrieval. This experiment also highlights that codec primitives contain rich temporal information that is often lost when sampling sparse keyframes.

Table G.6: **Retrieval performance at 1 FPS.** We evaluate the task of next-frame retrieval. The baseline uses the SigLIP embedding of the current frame, while we use the current frame along with the motion vectors and residuals to next frame, as processed by our Δ-encoder and pre-training transformers. The significantly better performance of our method shows that the information of the codec primitives is correctly compressed in the Δ-tokens.

@1 FPS	**Recall**		
	@1	**@2**	**@5**
SigLIP	11.12	47.11	78.65
Ours - Δ-Encoder	**30.09**	**77.15**	**94.86**

H Video Decoding Illustration

As mentioned in Sec. 3.1, the size of each Group of Pictures (GOP) is usually decided adaptively by the codec and video encoder depending on the motion or change. At encoding time, the developer can provide an upper bound for the GOP size, or even fix it. For example, if the GOP size is fixed at 240, the first frame of each GOP is an I-frame and the remaining 239 frames are P-frames, each consisting of motion vectors and residuals. During video decoding, the I-frame or reconstructed frame at the previous timestamp $t-1$ is first moved using the motion vectors and then the residuals are added to get the RGB frame at the current timestep t, as described in Eq. 1. We visualize an example of the decoding process in Fig. 5.

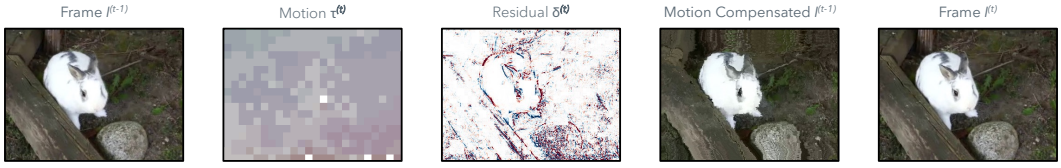


Figure 5: **Codec primer.** We visualize from left to right: the previous frame, the motion vectors and residuals between previous and current frame, the intermediate reconstruction after motion compensation, and the final result after adding the residuals.

I Training Data and Evaluation Benchmarks

Training Data. We train on LLaVA-Video-178K [113], a video instruction-tuning dataset organized by duration (0–30s, 30–60s, 1–2min, 2–3min) with sources from academic benchmarks and YouTube. It covers captioning, open-ended QA, and multiple-choice QA. ActNet-QA [99], NextQA [89], and PerceptionTest [60] are included as QA subsets; following prior work, we treat these together with the rest of LLaVA-Video-178K as a single training corpus totalling 1.39M samples.

General Video QA. PerceptionTest [60] tests fine-grained perceptual reasoning (memory, abstraction, physics) through multiple-choice questions over short diagnostic videos. NextQA [89] targets causal, temporal, and descriptive reasoning on short clips via multiple-choice QA. ActNet-QA [99] poses open-ended questions about complex web videos spanning diverse activities. ActNet-QA [99] uses GPT-based evaluation; we report scores using *gpt-4o-2024-11-20* in Tab. 1 and Azure deployed version of *gpt-3.5-turbo-0613* in Tab. 2 for fair comparison with prior work. VideoMME [20] covers short to long videos with questions on spatial, temporal, and semantic understanding, reported with and without subtitles.

Temporal and Motion Reasoning. TempCompass [51] probes temporal understanding through questions about speed, direction, order, and attribute change. TOMATO [65] tests action ordering,

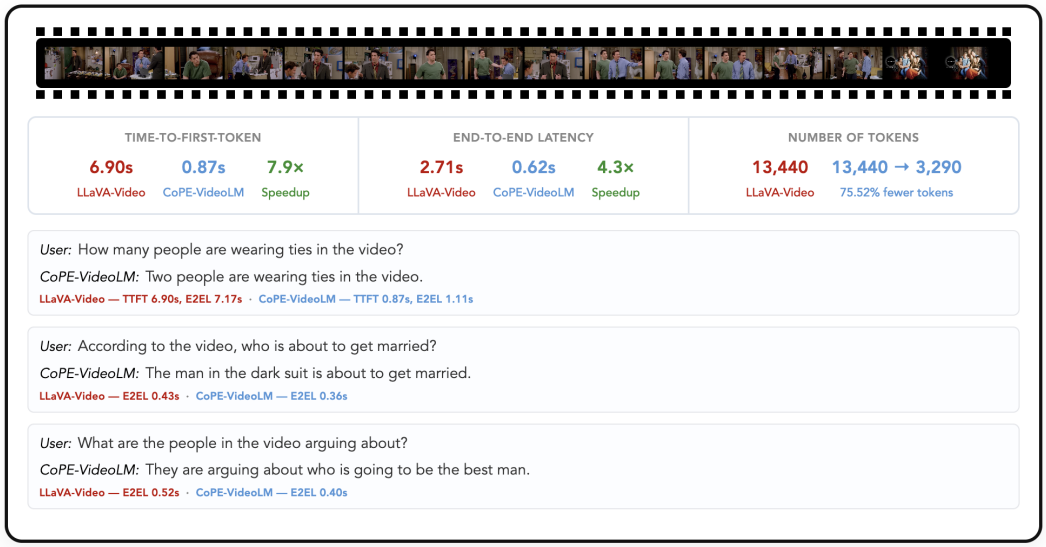


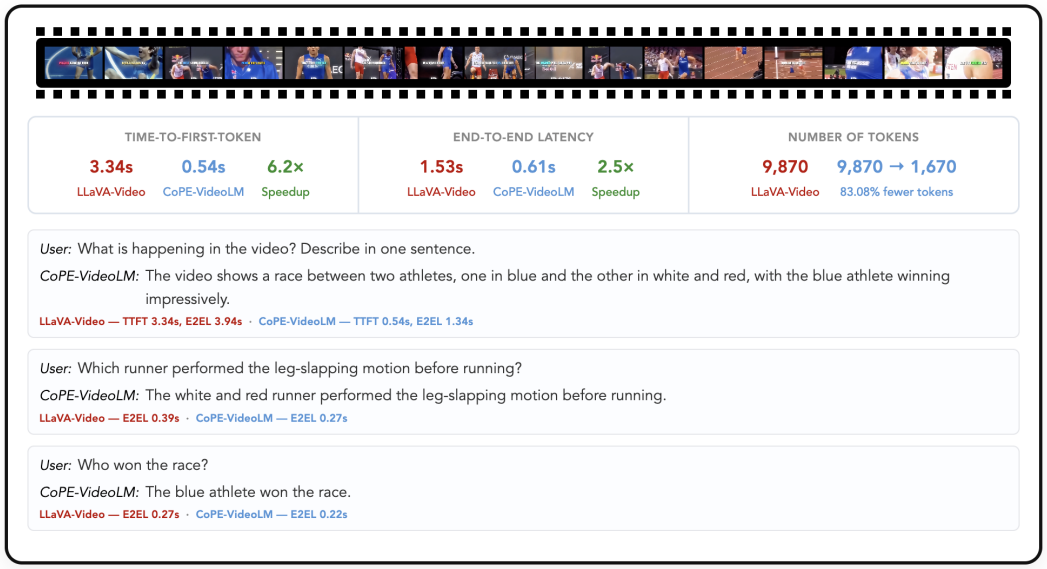
Figure 6: **Qualitative Results.** Multi-turn QA on a TV show clip requiring understanding of character relationships and narrative events. Our method achieves $8.2\times$ faster TTFT and 76% fewer tokens.

duration, and state transition reasoning via multiple-choice QA. CVRR-ES [36] uses compositionally varied questions to expose systematic failure modes in video reasoning. MVBench [42] spans 20 temporal tasks including action sequence, scene transition, and object interaction recognition.

Long-form and Instruction Following. LongVideoBench [88] tests understanding of long, interleaved video-language sequences with multi-step reasoning. LVBench [79] targets long-form video comprehension with videos exceeding several minutes. Video-TT [114] tests complex multi-step video reasoning and holistic understanding. Video-MMMU [29] tests knowledge acquisition from multi-discipline professional videos.

J Qualitative Results

We present qualitative comparisons between CoPE-VideoLM and LLaVA-Video-7B across several videos spanning diverse scenarios: sports with fast motion (hockey, athletics), indoor scenes (TV show, cooking), and outdoor activities (wood chopping). For each example, we report TTFT, E2EL, and token counts for both methods. Note that from the second turn onward, both methods benefit from KV-cache reuse, which narrows the absolute latency gap. However, our method still maintains a consistent relative speedup across all turns due to the shorter cached sequence length from fewer tokens. As shown in Figs. 6–8, CoPE-VideoLM produces correct, fine-grained answers while achieving consistent speedups at substantially reduced token counts.



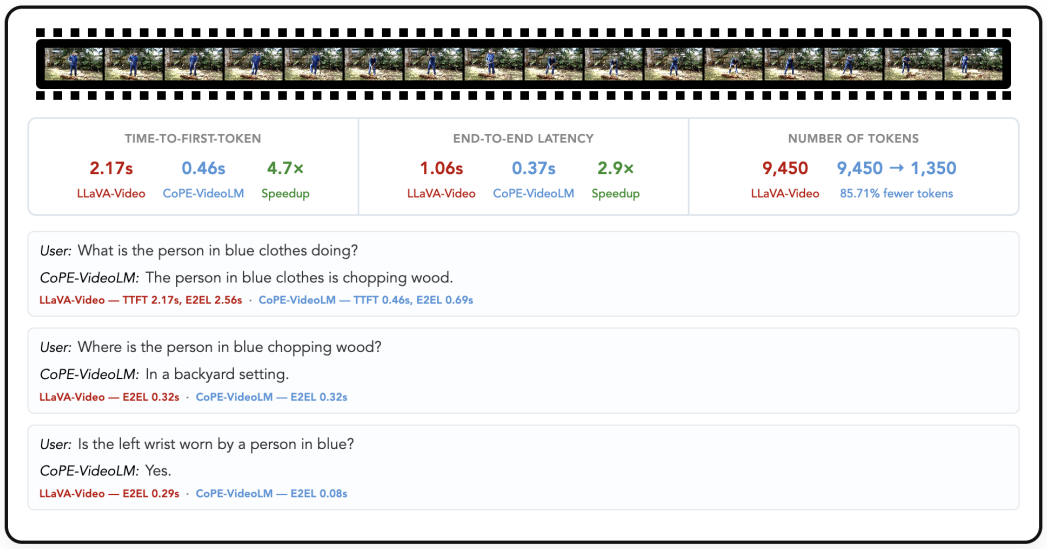
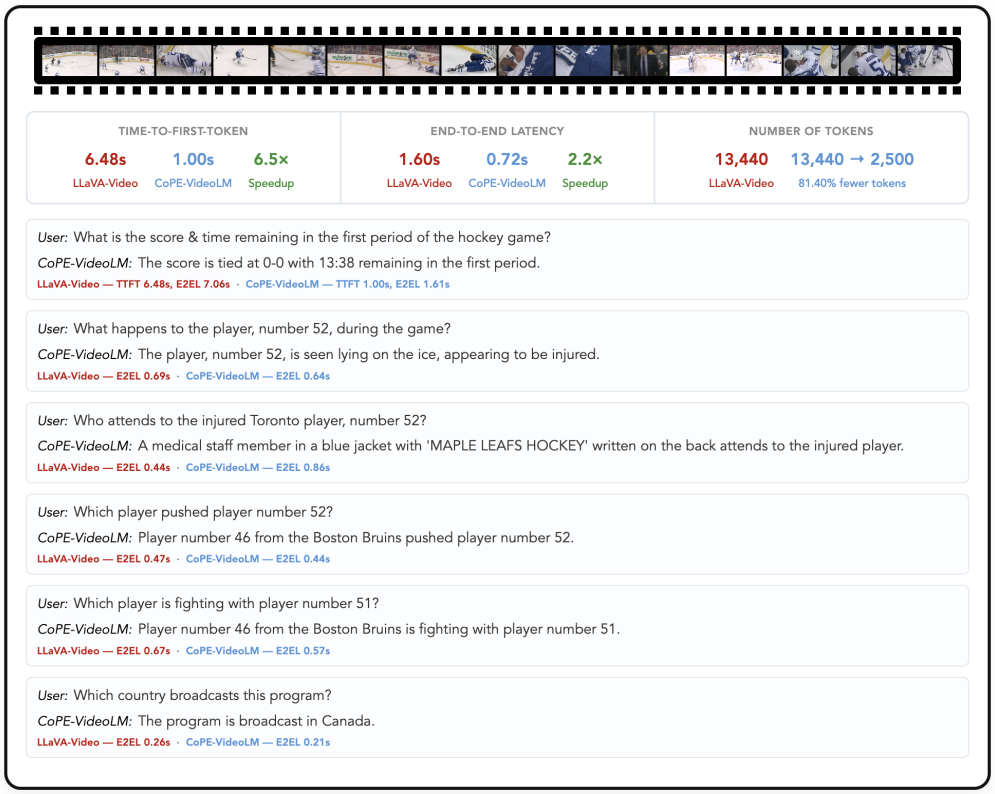


Figure 7: **Qualitative Results.** An athletics race requiring fine-grained action understanding and an outdoor activity video involving spatial and action reasoning. Our method achieves $4.2-6.2\times$ faster TTFT and $84-86\%$ fewer tokens while producing correct, fine-grained answers.



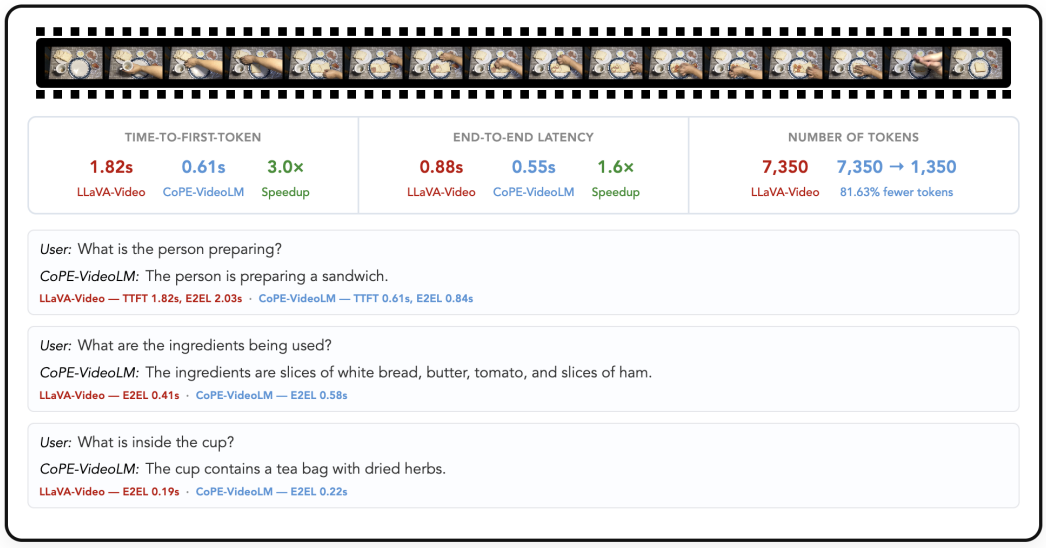


Figure 8: **Qualitative Results.** A hockey game involving multi-step temporal reasoning (injury, player identification, causal events) and, a cooking video with questions about actions, ingredients, and fine-grained object details. Our method provides accurate responses at $2.4-3.9\times$ faster TTFT and 82% fewer tokens.

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: We state our contributions in the abstract and particularly in the introduction (Sec. 1) including a summary of them. The rest of the sections alongside experiments support the claims.

   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: We discuss different limitation aspects of our method in Sec. 5.

   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: [N/A]

Justification: Our paper does not include theoretical results.

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. **Experimental result reproducibility**

   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: We discuss the implementation and training details in Sec. 4 and the appendix.

   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: [No]

Justification: We use publicly available datasets for training and provide sufficient training and implementation details in Sec. 4 and the Appendix. We intend to release the code and model weights publicly.

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: In Sec. 4, we describe experimental settings while providing full details in the Appendix. We also provide all dataset information in Appendix.

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: [No]

Justification: We do not report error bars but extensively evaluate our method on 14 benchmarks. The number of samples and reported metrics are statistically significant.

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: We provide all details regarding compute resources in Sec. 4 and Appendix.

   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: We followed the NeurIPS Code of Ethics in this work. We took care to handle data responsibly, consider possible impacts, and follow all relevant ethical guidelines.

   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: We mention broader impacts of our work in Sec. 5.

    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: This work focuses on improving the computational efficiency of existing video language models through codec-aware video representations. It does not introduce or release any new datasets or data collection pipelines. The experiments use existing publicly available datasets and benchmarks. The work does not pose additional risks that would require specific safeguards for responsible release. We discuss potential misuse and other societal risks in Sec. 5.

    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: We properly cite the creators and owners of all data, code, and models used in our work. All usage has adhered to the corresponding terms.

    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: [N/A]

    Justification: We show interactive results on multiple videos available from public benchmarks in the supplementary material and appendix. However, these examples are not newly released assets.

    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 any crowdsourcing or research with human subjects.

    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 any crowdsourcing or research with human subjects.

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: We do not use LLMs for the development of our method, however, our work is based on efficient tokenization for Video LLMs.

    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.