AdaCodec: A Predictive Visual Code for Video MLLMs

Haowen Hou[1,2,3*] **Zhen Huang**[2] **Zheming Liang**[2] **Qingyi Si**[3] **Chenglin Li**[2]
Shuai Dong[2] **Kele Shao**[2] **Ruilin Li**[2] **Dianyi Wang**[2] **Nan Duan**[3] **Jiaqi Wang**[3,2†]

[1]Shanghai Jiao Tong University [2]Shanghai Innovation Institute [3]JD.com

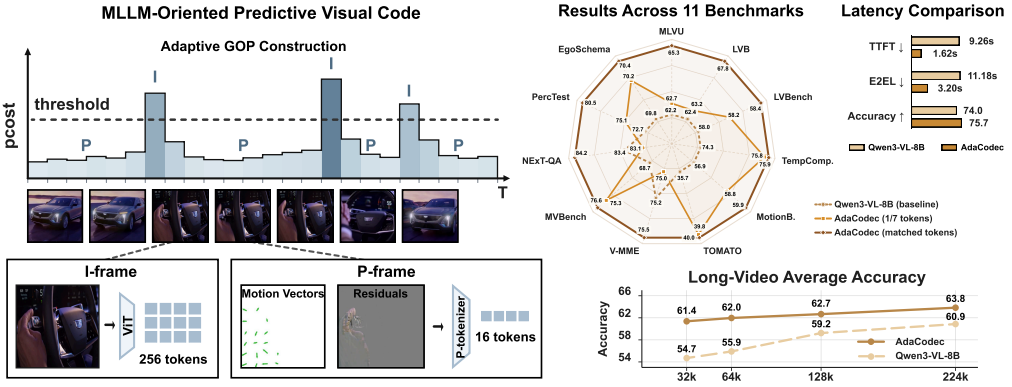

Figure 1. AdaCodec treats the video MLLM visual interface as a predictive code: a frame is encoded into full visual tokens only when it cannot be predicted from prior context, and intermediate frames are sent as compact motion-and-residual P-tokens. **Left:** AdaCodec splits the video into adaptive Groups of Pictures (GOPs), each containing one I-frame (intra-coded frame, encoded independently) followed by a chain of P-frames (predictive frames). AdaCodec places I-frames adaptively via a *pcost* threshold on per-frame predictive cost, encodes each I-frame into full ViT tokens, and encodes each intermediate P-frame into fewer compact motion-and-residual tokens produced by the P-tokenizer. **Right:** AdaCodec matches or surpasses Qwen3-VL-8B on all eleven benchmarks even at $1/7$ the tokens, cuts time-to-first-token (TTFT) and end-to-end latency (E2EL) while raising average accuracy, and leads on long-video accuracy across token budgets from 32k to 224k.

Abstract

Video is temporally redundant: adjacent frames usually share most objects, background, and layout. Yet existing video multimodal large language models (video MLLMs) usually encode each sampled frame as an independent RGB image, causing visual tokens to repeat content already present in earlier frames. This suggests a more direct video interface: send a full reference frame only when the scene cannot be predicted well from prior context, and otherwise transmit a compact description of inter-frame changes. We call this interface a *predictive visual code*, and instantiate it for video MLLMs as **AdaCodec**. AdaCodec spends full visual tokens on a reference frame only when its conditional predictive cost is high; otherwise, it encodes inter-frame changes, including motion and prediction residuals, as compact P-tokens. Across all eleven benchmarks, AdaCodec improves over the Qwen3-VL-8B per-frame RGB baseline at a matched visual-token budget. Even at $1/7$ the budget, AdaCodec with 32k tokens surpasses the 224k baseline on all long-video benchmarks; on five general-video benchmarks, it raises the average score while substantially cutting time-to-first-token from 9.26s to 1.62s.

*Work done during an internship at JD.com.
†Corresponding author.

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

1 Introduction

Video multimodal large language models (video MLLMs) are moving beyond short clips. They are increasingly used for workloads that require long temporal coverage, dense event tracking, and low response latency, including long-video understanding and temporal reasoning [1–10]. Yet their visual interface remains largely unchanged: a video is sampled into RGB frames, each frame is encoded as an image, and the resulting visual tokens are concatenated into the language model context.

However, this per-frame interface is inefficient under temporal redundancy for video. Adjacent frames usually share most objects, background, and layout, so independent per-frame encoding repeatedly sends information that prior context already contains. Token cost then grows roughly linearly with the number of sampled frames. Under a finite context window, this redundancy creates a coverage–detail dilemma: sparse sampling misses short events and fine transitions, while dense sampling consumes context and increases latency.

Most efficiency methods reduce this pressure by selecting frames, compressing frame tokens, or managing frame-derived states through memory, long-context, and multi-rate designs [11–27]. These methods differ in where they save budget, but they share the same basic interface: retained visual evidence is still derived from independent RGB frames. A related line has used codec-aware signals for efficient video understanding [28–32]. These works show that codec structure can carry useful temporal evidence, but they keep the playback-oriented codec output fixed and learn modules that consume its extracted signals. This leaves a sharper representation question: what visual representation should a video MLLM process, so the input removes temporal redundancy while preserving the evidence needed for reasoning?

We draw inspiration from predictive coding, where a system transmits errors from a prediction rather than the raw signal. This principle has biological grounding: the visual system is thought to encode prediction errors, the mismatch between expected and observed input, rather than the input itself [33]. Modern video codecs use the same residual-coding idea in engineering: reference frames carry full content, while predictive frames carry motion and residual signals relative to a reference [34, 35]. These systems have different objectives, but they share the same conditional structure: when nearby samples are redundant, the channel should carry what prediction fails to explain. Standard codecs, however, optimize for bitstreams and human-viewable reconstruction, not for visual tokens consumed by an LLM. We therefore redesign this mechanism as an MLLM interface for video understanding.

We present **AdaCodec**, a **predictive visual code for video MLLMs**. AdaCodec allocates full ViT tokens only to reference frames, and represents predictable intermediate frames with compact P-tokens derived from motion and residuals. Several design choices make the visual code MLLM-oriented, including a redesigned procedure for computing the predictive code, and a *pcost*-driven reset that starts a new reference frame when prediction becomes costly. The MLLM therefore receives an interleaved stream of reference-frame tokens and compact P-tokens, instead of processing each sampled frame as a full RGB image. By eliminating redundant visual tokens before they enter the LLM, AdaCodec also substantially reduces inference latency.

Across eleven benchmarks, AdaCodec improves the performance-cost frontier. Under a matched 224k visual-token budget, it obtains the strongest open-source results in our comparison on all three long-video benchmarks and on two of the three temporal benchmarks. Under tighter budgets, AdaCodec at 32k visual tokens already surpasses the 224k Qwen3-VL-8B baseline on all long-video benchmarks. On the five general video-understanding benchmarks, AdaCodec uses 84.7% fewer visual tokens, cuts time-to-first-token from 9.26s to 1.62s, and improves the average score. Our contributions are three-fold:

(1) We formulate **predictive visual code as the visual interface for video MLLMs**: a full reference frame is used only when conditional predictive cost is high, while predictable frames are compactly encoded through motion and residual cues.

(2) We build **AdaCodec**, including an MLLM-oriented predictive codec, a compact P-frame tokenizer, and a two-stage alignment pipeline that bridges the predictive code with existing MLLM architectures.

(3) Across **eleven benchmarks and controlled efficiency studies**, AdaCodec improves accuracy while substantially reducing both visual-token consumption and inference latency compared with per-frame RGB baselines. Ablations further validate each core design. The project page is available at `https://haowenhou.github.io/AdaCodec-Page/`.

2 Related Work

2.1 Efficient Video Representations for Video MLLMs

Video MLLMs encode each sampled frame into hundreds of visual tokens, so the visual sequence scales with video length and sampling density, therefore quickly dominating context length and compute. Prior work to alleviate this cost falls into three complementary directions. A major line of work reduces cost through frame selection or frame-space subsampling [11–15]. Another line compresses tokens after frame encoding, including aggressive token compression, temporal pooling, dynamic token pruning or merging, and adaptive spatiotemporal compression [16, 21, 17–20, 36]. A third direction focuses on memory or long-context modeling, where sparse memory banks, online memory updates, and long-context adaptation improve scaling to longer videos [22–26]. These methods are effective, but each retained frame is still encoded as an independent RGB image, leaving substantial redundancy among the retained frames. AdaCodec instead replaces the per-frame interface itself, encoding predictable intervals as motions and residuals relative to a reference and spending a full reference frame only where prediction fails.

Two recent compression-oriented works provide useful context, although they do not target the same MLLM interface. OneVision-Encoder explores codec-aligned patch sparsity *inside* the visual encoder via "codec patchification", targeting patch-level sparse computation and encoder efficiency [37]. InfoTok learns adaptive discrete tokens for video reconstruction [38]. AdaCodec is neither encoder-internal sparsity nor a generative reconstruction tokenizer; it is a predictive code at the visual interface that the MLLM consumes.

2.2 Codec-based Video Representations

Using compressed-domain signals for video understanding has a long history. Classical and modern codecs (e.g., AVC/H.264, HEVC/H.265, AV1) represent only sparse keyframes in full and model the remaining frames with predictive side information [34, 35, 39]. Early studies used motion vectors as a low-cost surrogate for optical flow, and later methods such as CoViAR and DMC-Net modeled I-frame, motion, and residual modalities jointly for efficient action recognition [40, 28, 29]. Follow-up work improved multimodal fusion and compressed-domain transformer modeling [41, 42]. Beyond action recognition, compressed representations have been applied to object detection, video object segmentation, pose estimation, video question answering, and video captioning [43–47]. These results support the view that motion and residual signals could encode useful localized temporal changes for higher-level reasoning.

Recently, codec-aware ideas have entered the video MLLM regime. EMA builds GOP-level representations from I-frames and motion vectors [30]. In concurrent work, CoPE-VideoLM uses standardized codec primitives from video streams and learns to align them with MLLM representations, while ReMoRa focuses on refining noisy block-motion representations and leveraging compressed-domain motion signals for long-video understanding [31, 32]. These methods are complementary to AdaCodec but target a different design space: they treat a standards-compliant codec stream as fixed and learn how to consume it. AdaCodec instead redesigns the predictive code itself for MLLM consumption, so coding units, motion estimation, and reference-frame placement are all chosen for the downstream LLM rather than for human playback.

3 Method

Given a video $X = \{x_t\}_{t=0}^{T-1}$ with RGB frames $x_t \in \mathbb{R}^{H\times W\times 3}$, our goal is to construct a predictive visual code that video MLLMs can process effectively. AdaCodec is developed through three design aspects: (1) a predictive visual code that encodes intermediate frames as motion-and-residual updates, (2) a dual-branch visual-token pipeline for video MLLMs, and (3) two-stage training for P-frame representation learning and multimodal alignment of the visual code. Figure 2 illustrates how AdaCodec encodes motion and residual for each P-frame and how the resulting tokens are consumed by the LLM.

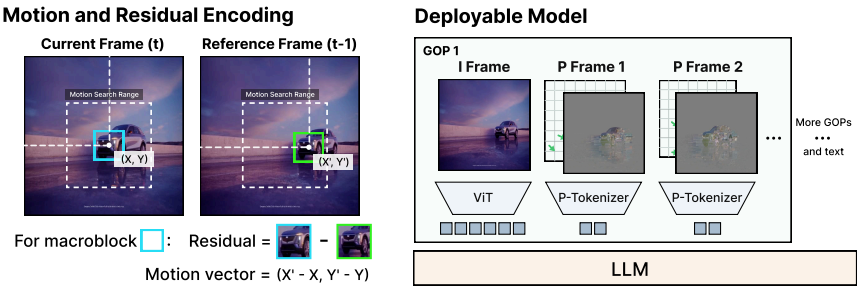

Figure 2: **AdaCodec method overview. Left:** Motion-and-residual encoding for a P-frame. For each macroblock in the target frame, AdaCodec searches a local window in the reference frame for the best-matching block; the displacement gives the motion vector and the per-pixel difference gives the residual. **Right:** Deployable model. Each GOP encodes its I-frame with the ViT and each P-frame with the P-tokenizer.

3.1 MLLM-Oriented Predictive Visual Code

Codec preliminaries. A standard predictive codec stores occasional intra-coded keyframes (I-frames) in full and represents the remaining frames through motion-compensated prediction plus residual correction. The forward-predicted frames are P-frames, while bidirectionally predicted B-frames depend on both past and future references.

AdaCodec redesign for MLLM tokenization. AdaCodec adapts this predictive-coding paradigm to a video-MLLM interface. Standard codecs optimize a standards-compliant bitstream for transmission and human-perceived reconstruction, whereas a video MLLM consumes a visual-token sequence for reasoning. Table 1 lists the core redesign choices that distinguish this code from a playback-oriented codec; the remaining redesign choices are in Appendix A.

Table 1: Core redesigns from a playback-oriented codec to AdaCodec's MLLM-oriented predictive code.

Component	Playback-oriented codec	AdaCodec (MLLM-oriented)
Block partition	Heterogeneous block sizes chosen for bitrate.	Macroblocks aligned to the ViT patch grid, yielding more stable P-frame tokens.
Motion reference	Reference pictures selected under codec syntax.	Each P-frame is estimated from the immediately preceding sampled frame for larger temporal gaps.
Search window	Tuned to high-FPS playback.	Enlarged local window to absorb the larger displacement between low-FPS frames.
GOP scheduling	Separate content-analysis pass.	Reuses the predictive cost from motion search to trigger adaptive I-frame insertion.

Motion and residual encoding. We partition X into groups of pictures (GOPs), denoted by $\{\mathcal{G}_j\}_{j=1}^{J}$. For each GOP $\mathcal{G}_j = \{x_{s_j}, \ldots, x_{e_j}\}$, the first frame x_{s_j} is an I-frame, and each subsequent frame is a P-frame represented by motion vectors and residual signals relative to the preceding sampled frame. Thus, for every $t > s_j$, x_t is predicted from x_{t-1}.

AdaCodec does not select a subset of macroblocks. It partitions every frame into a fixed, non-overlapping grid blocks. For a macroblock location $b \in \mathbb{Z}^2$ on the current frame, let x_t^b be the current block and x_{t-1}^{b+d} be the block at offset d in the reference frame. AdaCodec searches in a local window $\mathcal{D}_b$ to find the best-matching block, as depicted in Figure 2. We define the candidate block prediction cost and select the motion vector by

$$\ell_t^b(d) = \mathrm{SAD}\big(x_t^b, x_{t-1}^{b+d}\big) + \lambda\|d\|_1, \qquad m_t^b = \arg\min_{d \in \mathcal{D}_b} \ell_t^b(d). \tag{1}$$

Here $d \in \mathbb{Z}^2$ is a 2D vector, $\mathrm{SAD}(\cdot,\cdot)$ is the sum of absolute pixel differences between two blocks, and the ℓ_1 term mildly favors smaller displacements when several candidates match similarly well. The 2D motion vector m_t^b points from the current 2D block location b to the matched reference location $b + m_t^b$. The residual of this block is then the pixel difference between the current block and the reference block:

$$r_t^b = x_t^b - x_{t-1}^{b+m_t^b}. \tag{2}$$

In practice, we use hexagonal search with local refinement to approximate the minimizer efficiently.

Each P-frame is represented by the signed residual $r_t \in \mathbb{R}^{H\times W\times 3}$ and block motion vectors $\{m_t^b\}_b$. We broadcast block motion to pixels: for every pixel p in block b, we set $M_t(p) = m_t^b$. Thus, all

pixels in one macroblock share the same 2D displacement, giving $M_t \in \mathbb{R}^{H\times W\times 2}$. The P-frame input is the five-channel concatenation $u_t = [r_t,\, M_t] \in \mathbb{R}^{H\times W\times 5}$. Because the macroblock grid coincides with the patch grid, each ViT patch receives one constant motion vector in its motion channels.

Adaptive GOP construction. The goal is to keep frames with low predictive cost in the same GOP and split when temporal prediction becomes less reliable. Instead of fixed-length GOPs, we use a lightweight content-adaptive splitting rule. The same motion search above yields a frame-level predictive cost by summing the selected block costs:

$$\ell_t = \sum_b \ell_t^b(m_t^b). \tag{3}$$

Here ℓ_t is the aggregate cost of predicting frame x_t from x_{t-1} under the selected block motions. A large ℓ_t indicates that x_t is poorly predicted as a P-frame and therefore contains substantial novel content, making it a strong candidate for a new I-frame. This reuse lets AdaCodec make GOP decisions without a separate GOP-analysis pass, reducing duplicate computation and improving encoding speed.

We always designate frame 0 as an I-frame. For each subsequent frame, we start a new GOP when $\ell_t > \gamma$, where γ controls the GOP-length distribution and thus the token budget. We choose γ on the training split by targeting a median of 8 P-frames per GOP, and reuse the resulting threshold for all training and evaluation runs. To satisfy temporal-length constraints in MLLM training, we cap the number of P-frames per GOP by $K_{\max}$; once the cap is reached, we force an I-frame split.

3.2 Dual-Branch Visual Tokenization Architecture

The deployable AdaCodec architecture (Figure 2) contains a reference-frame encoder E_I and a P-frame tokenizer (P-tokenizer) E_P. The P-tokenizer is initialized from a standard pretrained ViT. AdaCodec widens the pretrained ViT patch embedding from 3 to 5 input channels for $u_t = [r_t,\, M_t]$, copies the RGB kernels, and zero-initializes the two motion-vector channels. It then appends learnable tokens after the patch sequence and uses their output states as $E_P(u_t)$. This architecture adapts to any ViT-style visual encoder. Details of the model architecture are described in Appendix B.

Given one GOP $\mathcal{G} = \{x_0, x_1, \ldots, x_K\}$, we treat I-frame x_0 as the reference frame, and then encode the reference frame and the t-th P-frame as

$$z^I = E_I(x_0) \in \mathbb{R}^{N_I\times d}, \qquad z_t^P = E_P(u_t) + e_t \in \mathbb{R}^{N_P\times d}, \quad t = 1, \ldots, K, \tag{4}$$

where e_t is the temporal position embedding and $N_P \ll N_I$. Therefore, a GOP with K P-frames uses $N_I + KN_P$ tokens instead of $(K+1)N_I$, giving the token ratio analyzed in Section 4.3 and Appendix E. The resulting reference and P-frame tokens are arranged into the LLM visual embedding space, then inserted into the multimodal prompt in temporal order.

3.3 Two-Stage Training

We train AdaCodec in two stages. Stage 1 learns the compact P-tokenizer under frozen visual supervision, whereas Stage 2 aligns the resulting visual code with the language model through multimodal training.

Stage 1: teacher-feature alignment for P-tokenizer. Each training sample contains one reference frame, n intermediate P-frames, and one target frame. A frozen teacher visual encoder extracts

$$z^I = E_I(x_0), \qquad z^T = E_T(x_T), \tag{5}$$

where E_T shares weights with E_I. To train E_P, we attach an auxiliary feature predictor H_ϕ, a lightweight transformer used only in Stage 1. Intermediate P-frames are encoded as $\{z_t^P\}_{t=1}^n$, and the auxiliary predictor outputs

$$\hat{z}^T = H_\phi\left(z^I, \{z_t^P\}_{t=1}^n\right). \tag{6}$$

We optimize

$$\mathcal{L}_{\text{stage1}} = \|\hat{z}^T - z^T\|_1 + \left(1 - \cos(\hat{z}^T, z^T)\right). \tag{7}$$

During this stage, E_I and E_T remain frozen, while E_P and H_ϕ are optimized.

Stage 2: multimodal alignment. After Stage 1, we keep the learned E_P and discard the auxiliary predictor H_ϕ. Under a fixed visual token budget, we uniformly sample multiple adaptive GOPs across the full video timeline and preserve their temporal order to form $\mathcal{V}_{\text{code}}$. We then optimize the standard autoregressive next-token prediction loss:

$$\mathcal{L}_{\text{stage2}} = -\sum_i \log p\left(y_i \mid y_{<i}, \mathcal{V}_{\text{code}}, q\right), \tag{8}$$

where q is the text instruction and y is the target response. We freeze all visual-side modules and update only the language model of the MLLM.

4 Experiments

We evaluate AdaCodec on long-video, temporal, and general video-understanding benchmarks. The experiments proceed in three phases. We first present the main results: benchmark accuracy across eleven benchmarks (§4.2) and system efficiency (§4.3). Two analyses then inspect AdaCodec's behavior, examining how its accuracy advantage scales with the visual-token budget (§4.4) and how its adaptive GOP construction tracks video content (§4.5). Finally, we validate design necessity at two levels, the predictive coding (§4.6) and our codec design choices (§4.7).

4.1 Experimental Setup

Model details and fairness protocol. We use Qwen3-VL-8B [48] as the base MLLM. Because Qwen3-VL-8B ViT uses a temporal Conv3D visual stem, a spatial 2×2 merger, and DeepStack visual injection, AdaCodec must produce P-frame tokens that match these native interfaces; Appendix B describes this adaptation. The details of the data sources and training hyperparameters are described in Appendix C. For a fair comparison, we keep the number of visual tokens produced from each RGB frame the same for AdaCodec and Qwen3-VL-8B. When the full frame sequence exceeds the visual context limit, for the baseline we follow the official uniform temporal sampling strategy, while for AdaCodec we uniformly sample GOPs over time.

Benchmarks. We organize benchmarks into three groups for presentation clarity. (1) Long-video: MLVU `test` [10], LongVideoBench `val` [49], LVBench `test` [50]; (2) Temporal: TempCompass `test MCQ` [51], MotionBench `val` [52], TOMATO `test` [53]; (3) General video understanding: Video-MME `test` [9], MVBench `test` [54], NExT-QA `test` [55], PerceptionTest `val` [56], EgoSchema `test` [57]. Benchmark descriptions are in Appendix D. We report macro-average accuracy on MLVU following standard practice. We use the `lmms-eval` toolkit for evaluation [58].

4.2 Main Results Across Benchmarks

Table 2 compares AdaCodec with the Qwen3-VL-8B baseline and other open-source models. The baseline uses per-frame RGB input at 2 FPS with a 224k visual-token budget. We report two operating points that answer complementary questions.

(i) 1/7 token budget. This setting asks how performant AdaCodec is while using a significantly lower token budget saved by its predictive code. Both methods consume the same frame sequence at 2 FPS. For long-video benchmarks, we explicitly cap AdaCodec at 32k visual tokens against the 224k-token RGB baseline; for temporal and general benchmarks, AdaCodec naturally uses about 1/7 of the baseline tokens through its compact representation. This setting isolates token efficiency.

(ii) Comparable token budget. This setting asks whether the saved tokens can be converted into denser temporal evidence under a comparable total token budget. For long-video benchmarks, we match the baseline's 224k visual-token budget. For temporal and general benchmarks, we increase AdaCodec's frame rate from 2 FPS to 16 FPS, matching the baseline's total token use by statistics. This setting isolates the gain from richer video coverage at fixed cost.

Compactness preserves accuracy. At 1/7 of the baseline's tokens, AdaCodec maintains accuracy across all three categories. On long-video, results slightly exceed the baseline ($+0.5$, $+0.8$, $+0.2$ on MLVU, LongVideoBench, LVBench). The visual code thus preserves substantially more temporal information per token than per-frame RGB sampling. Temporal gains hold across TempCompass,

Table 2: Main benchmark results across long-video, temporal, and general video-understanding tasks. Higher is better. "LVB" denotes LongVideoBench, "V-MME" denotes Video-MME, and "-" indicates that the benchmark is not reported. **Bold** and underlined numbers indicate the best and second-best results among open-source models, respectively. For external models, we use official reports when available; entries not reported there are taken from Molmo2 [59] or evaluated by us.

Method	Long-video			Temporal			General				
	MLVU	LVB	LVBench	TempComp.	MotionB.	TOMATO	V-MME	MVBench	NExT-QA	PercTest	EgoSchema
Closed-source models											
GPT-5 [60]	77.7	72.6	65.2	80.4	65.4	53.0	86.9	74.1	86.3	79.4	75.6
GPT-5 mini [60]	69.1	69.7	54.7	74.9	59.9	44.1	82.3	66.5	83.2	72.0	70.9
Gemini 3 Pro [61]	75.7	75.9	77.0	82.8	62.6	48.3	87.5	70.4	84.3	77.6	68.9
Gemini 2.5 Pro [62]	81.5	76.8	75.7	81.9	62.0	48.6	87.8	70.6	85.3	78.4	72.2
Gemini 2.5 Flash [62]	75.1	73.1	64.9	80.2	59.3	39.1	84.2	67.0	81.8	74.7	70.2
Claude Sonnet 4.5 [63]	64.0	65.1	50.5	72.8	58.5	39.6	80.5	62.1	79.2	64.3	73.1
Open-source models											
InternVL3.5-8B [64]	53.2	62.1	43.4	70.3	56.6	24.6	68.6	72.1	81.7	72.7	58.6
Keye-VL-1.5-8B [27]	53.8	66.0	42.8	75.5	55.1	33.0	**76.2**	56.9	75.8	64.2	56.3
GLM-4.1V-9B [65]	56.6	65.7	44.0	72.3	59.0	30.0	75.6	68.4	81.3	74.2	62.6
MiniCPM-V-4.5-8B [66]	60.6	63.9	50.4	72.7	59.7	29.8	73.5	60.5	78.8	70.9	49.6
Eagle2.5-8B [67]	60.4	66.4	50.9	74.4	55.7	31.0	75.7	74.8	85.0	81.0	**72.2**
PLM-8B [68]	52.6	56.9	44.5	72.7	61.4	33.2	65.4	**77.1**	84.1	**82.7**	68.8
LLaVA-Video-7B [69]	52.8	58.2	44.2	66.6	54.2	24.9	69.7	58.6	83.2	68.8	57.3
VideoChat-Flash-7B [70]	56.0	64.7	48.2	69.4	60.6	32.5	69.7	74.0	85.5	76.5	51.3
Molmo2-8B [59]	60.2	67.5	52.8	73.4	**62.2**	39.6	75.8	75.9	**86.2**	82.1	62.0
Molmo2-O-7B [59]	55.2	63.7	49.6	73.0	60.6	36.2	69.2	74.8	84.3	79.6	56.8
Codec-aware video MLLMs											
CoPE-VideoLM-7B [31]	-	56.9	46.4	68.9	-	28.3	69.4	61.9	82.1	70.3	-
ReMoRa-7B [32]	-	60.8	-	-	-	-	64.4	-	84.2	67.7	-
Ours (AdaCodec on Qwen3-VL-8B)											
Qwen3-VL-8B [48]	62.2	62.4	58.0	74.3	56.9	35.7	75.2	68.7	83.4	72.7	69.8
AdaCodec (1/7 token budget)	62.7 +0.5	63.2 +0.8	58.2 +0.2	75.8 **+1.5**	58.8 **+1.9**	39.8 **+4.1**	75.0 -0.2	75.3 **+6.6**	83.1 -0.3	75.1 **+2.4**	70.2 +0.4
AdaCodec (comparable token budget)	**65.3** **+3.1**	**67.8** **+5.4**	**58.4** +0.4	**75.9** **+1.6**	59.9 **+3.0**	**40.0** **+4.3**	75.5 +0.3	76.6 **+7.9**	84.2 +0.8	80.5 **+7.8**	70.4 +0.6

MotionBench, and TOMATO ($+1.5$, $+1.9$, $+4.1$), so fine-grained motion is retained rather than traded for long-context coverage. On the five general benchmarks, AdaCodec gains on three (largest $+6.6$ on MVBench) and trails by at most 0.3 on the other two. Reduced visual tokens therefore do not come at the cost of general capability.

Extra coverage converts into accuracy. At a matched token budget, AdaCodec improves over the baseline on every benchmark, with the best open-source results on all three long-video benchmarks and on two of the three temporal benchmarks. On long-video, gains are $+3.1$, $+5.4$, $+0.4$ on MLVU, LongVideoBench, and LVBench. On temporal, gains are $+1.6$, $+3.0$, $+4.3$ across TempCompass, MotionBench, and TOMATO. The largest general-benchmark gains are $+7.9$ on MVBench and $+7.8$ on PerceptionTest. These gains under a matched token budget indicate that AdaCodec's predictive code converts its compactness into accuracy, rather than merely compressing the input.

4.3 Efficiency and Latency

AdaCodec aims to improve both benchmark accuracy and system efficiency. We summarize the operating point below. The token-efficiency derivation, latency measurement protocol, codec-build overhead, and memory footprint are detailed in Appendix E.

Token efficiency. The maximum GOP length in AdaCodec is 17 frames (1 I-frame + 16 P-frames). Under this longest-GOP regime, AdaCodec incurs an 11.8% token cost relative to per-frame RGB encoding. On real evaluation videos, content changes shorten some GOPs, giving an average GOP length of 10.21 frames (Table 3). The measured token cost is still only 15.4% of the baseline, an 84.6% reduction. Section 4.5 further analyzes this content-dependent GOP adaptation.

Table 3: Token efficiency at the theoretical cap and over all evaluation videos. GOP length counts the I-frame and P-frames.

Metric	Theoretical cap	All evaluation videos
GOP length↑	17.00	10.21
P-frames/GOP↑	16.00	9.21
Token ratio↓	11.8%	15.4%
Token reduction↑	88.2%	84.6%

Latency and memory. Table 4 reports the latency and peak-memory comparison on the five general video-understanding benchmarks under matched hardware and decoding settings. Aggregated over 11,347 unique videos, AdaCodec uses 8,550.4 visual tokens per video on average against 55,893.2 for the per-frame RGB baseline (84.7% reduction), cuts time-to-first-token (TTFT) from 9.26s to 1.62s and total end-to-end latency (E2EL) from 11.18s to 3.20s, and raises the five-benchmark average score from 74.0 to 75.7. Codec-build denotes predictive code calculation and *pcost*-based GOP splitting needed to construct the AdaCodec input on a consumer-level 16-core CPU; even when this 0.12s cost is charged to TTFT, AdaCodec remains 5.3× faster than the baseline. The AdaCodec visual code thus delivers a Pareto improvement over per-frame RGB input: fewer visual tokens, faster response, and stronger downstream performance, at a one-time +1.9 GB peak-memory cost (+5.5%).

Table 4: Latency and peak-memory comparison. Score is the five-benchmark mean.

Method	Tokens↓	Build↓	TTFT↓	E2EL↓	Mem.↓	Score↑
RGB base.	55,893.2	–	9.26s	11.18s	**34.6 GB**	74.0
AdaCodec	**8,550.4**	0.12s	**1.62s**	**3.20s**	36.5 GB	**75.7**

4.4 Performance across Visual Token Budgets

On the three long-video benchmarks, we sweep visual-token budgets of 32k, 64k, 128k, and 224k. Temporal and general benchmarks are out of scope here because their videos consume fewer visual tokens overall. AdaCodec dominates the baseline across the full budget range on all three benchmarks (Figure 3), so the gain is not tied to a single operating point. At 32k visual tokens, AdaCodec already surpasses the 224k-token baseline, consistent with the claim that predictive coding makes more efficient use of the visual-token budget than independent RGB-frame encoding.

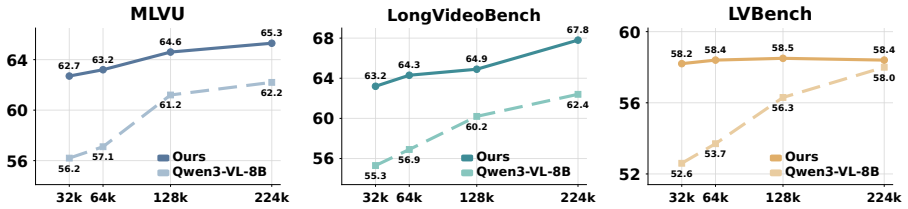

Figure 3: Long-video accuracy under visual-token budget sweeps.

4.5 Adaptive GOP Behavior

Figure 4 first illustrates the adaptive I-frame reset mechanism on a dynamic example from NextQA. Several *pcost* spikes cross the threshold, so AdaCodec inserts new I-frames before prediction errors accumulate. Between I-frame resets, intermediate frames remain compact P-token inputs; the cumulative token curve therefore grows much more slowly than per-frame RGB. The policy thus saves tokens in predictable intervals and spends full visual tokens when prediction becomes difficult.

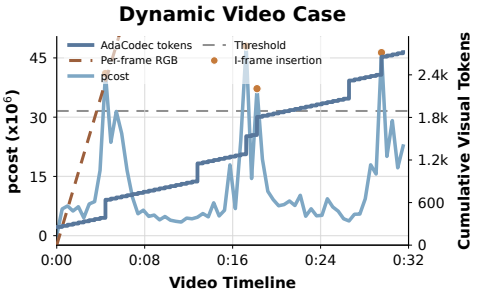

Figure 4: Dynamic-case behavior under adaptive GOP construction. Spikes in *pcost* trigger I-frame insertions, while P-frames keep AdaCodec's cumulative token growth far below per-frame RGB.

The same rule yields different GOP patterns across video regimes. The global average GOP length of 10.21 from Table 3 hides a wide content-dependent spread. On MLVU `test` [10], visually stable categories such as anomaly recognition (16.69) and tutorial-style videos (12.22) sustain long predictive chains, whereas egocentric (9.07) and dynamic content require more frequent I-frame refreshes. The anomaly-recognition gain comes from this behavior: AdaCodec preserves far more of the original sequence within the same context budget, reaching 71.8 against Qwen3-VL-8B's 51.2 (+20.6). Per-category breakdowns and additional case studies are in Appendix F.

4.6 Necessity of Predictive Coding

We test the necessity of predictive coding at the representation level. The GOP structure and full-RGB I-frame representation are fixed; only the P-frame representation changes (Table 5). When excluding the I-only setting, this experiment also serves as a coverage control: Per-frame RGB, Thumbnail P, and AdaCodec use the same 2 FPS frame sequence and GOP boundaries, so the RGB baseline observes exactly the same frames as AdaCodec rather than fewer frames under a fixed token budget.

Table 5: Representation ablation under matched GOP coverage. All settings use full-RGB I-frames; only the P-frame input changes.

Setting	P-frame input	Long↑	Temporal↑	General↑
I-only	Omitted	52.7	48.5	68.2
Per-frame RGB	Full RGB	58.6	55.6	74.0
Thumbnail P (untrained)	RGB thumbnail	58.2	53.8	73.6
Thumbnail P (trained)	RGB thumbnail	61.4	54.4	73.2
AdaCodec	**P-tokens**	**63.8**	**58.2**	**75.7**

Against I-only ($+11.1$ / $+9.7$ / $+7.5$), P-tokens recover the motion and residual signal that a single keyframe discards. *Against per-frame RGB* ($+5.2$ / $+2.6$ / $+1.7$), AdaCodec wins with a shorter visual prefix, so the gain is not from seeing more frames; the shorter prefix also reduces attention dilution from long token sequences. The all-RGB prefix can exceed Qwen3-VL-8B's native context window, so we use YaRN context extrapolation [71] for this setting.

Thumbnail P provides a stricter control; we replace each P-frame with a low-resolution RGB thumbnail that Qwen3-VL-8B encodes into $N_P = 16$ tokens, matching the AdaCodec per-frame budget. AdaCodec exceeds the untrained Thumbnail P row by $+5.6$ / $+4.4$ / $+2.1$, and still exceeds the trained row by $+2.4$ / $+3.8$ / $+2.5$. Low-resolution RGB loses fine visual detail and does not represent temporal changes explicitly. The remaining gap therefore comes from the predictive coding representation of AdaCodec, not only from adaptive token budget allocation.

4.7 Codec Design Ablations

Section 4.6 tests the necessity of the predictive coding interface itself; we now test the necessity of two specific codec-design choices: ViT-aligned macroblocks and *pcost*-guided GOP construction. Results are reported in Table 6; the full ablations are in Appendix G.

Table 6: Core design ablations for AdaCodec. The default uses 16×16 macroblocks and adaptive GOP construction; deltas are relative to default.

Setting	Long↑	Temporal↑	General↑
AdaCodec	**60.3**	**56.2**	**74.1**
Dynamic Macroblocks	58.4 -1.9	54.8 -1.4	72.6 -1.5
Fixed GOP, n_P=8	58.2 -2.1	56.0 -0.2	73.7 -0.4
Fixed GOP, n_P=16	59.7 -0.6	55.4 -0.8	72.3 -1.8

Replacing ViT-aligned 16×16 macroblocks with H.264-style dynamic partitions lowers all three category averages by 1.4–1.9 points, indicating that the MLLM benefits from motion fields aligned to the ViT patch grid. Fixed GOP schedules also trail the adaptive *pcost* policy on every category, with the largest gap of 2.1 on long-video against n_P=8. The remaining axes, N_P, $K_{\max}$, and the threshold target, stay within ±1.1 around the default in Appendix G, so the main sensitivity comes from the MLLM-oriented codec design rather than from narrow hyperparameter tuning.

5 Conclusion

Limitations. Our experiments use a fixed input resolution; we leave dynamic resolution input to future work. AdaCodec also uses a uniform per-P-frame token budget ($N_P = 16$), and adapting it to per-frame motion or residual complexity could further improve the efficiency–accuracy frontier.

Conclusion. We introduced AdaCodec, an MLLM-oriented redesign of predictive visual code. Rather than adapting a fixed playback codec stream, AdaCodec provides a series of codec redesigns for the visual-token interface of video MLLMs. It allocates full ViT tokens to high-cost reference frames and represents predictable intermediate frames with compact motion-and-residual P-tokens. This design removes repeated visual evidence before it enters the LLM context while preserving temporal changes needed for reasoning. Across long-video, temporal, and general video-understanding benchmarks, AdaCodec consistently improves over a per-frame RGB interface under matched or smaller visual-token budgets, with substantially lower response latency. Ablations show that both predictive coding and the MLLM-oriented codec redesign are necessary for these gains.

Acknowledgments and Disclosure of Funding

We appreciate the support from JD.com. This work received no external financial funding. The authors declare no competing interests.

References

[1] 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. doi: 10.48550/arXiv.2305.06355. URL https://arxiv.org/abs/2305.06355.

[2] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad 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 (Volume 1: Long Papers)*, pages 12585–12602, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.679. URL https://aclanthology.org/2024.acl-long.679/.

[3] Hang Zhang, Xin Li, and Lidong Bing. Video-LLaMA: An Instruction-tuned Audio-Visual Language Model for Video Understanding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 543–553, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-demo.49. URL https://aclanthology.org/2023.emnlp-demo.49/.

[4] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-LLaVA: Learning United Visual Representation by Alignment Before Projection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5971–5984, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.342. URL https://aclanthology.org/2024.emnlp-main.342/.

[5] Peng Jin, Ryuichi Takanobu, Wancai Zhang, Xiaochun Cao, and Li Yuan. Chat-UniVi: Unified Visual Representation Empowers Large Language Models with Image and Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13700–13710, June 2024. doi: 10.1109/CVPR52733.2024.01300. URL https://openaccess.thecvf.com/content/CVPR2024/html/Jin_Chat-UniVi_Unified_Visual_Representation_Empowers_Large_Language_Models_with_Image_CVPR_2024_paper.html.

[6] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-OneVision: Easy Visual Task Transfer. *arXiv preprint arXiv:2408.03326*, 2024. doi: 10.48550/arXiv.2408.03326. URL https://arxiv.org/abs/2408.03326.

[7] 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. doi: 10.48550/arXiv.2409.12191. URL https://arxiv.org/abs/2409.12191.

[8] Shuai Dong, Kanghui Tian, Linghan Chen, Wenqi Shao, Hongrui Wu, Haowen Hou, Yongfu Zhu, Xuehui Wang, Ziyue Wang, Wenpu Liu, Weichu Xie, Yuqi Xu, Congcong Wang, Kaiwen Tuo, Caoyuan Ma, Taiqiang Wu, Ji Zhao, Wenzheng Chang, Dianyi Wang, Siyuan Wang, Zhongyu Wei, Yu Qiao, and Jiaqi Wang. Turns Matter: Advancing Streaming Video RL via Turn-Level Credit. *arXiv preprint arXiv:2611.06728*, 2026. URL https://arxiv.org/abs/2611.06728.

[9] Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, Peixian Chen, Yanwei Li, Shaohui Lin, Sirui Zhao, Ke Li, Tong Xu, Xiawu Zheng, Enhong Chen, Caifeng Shan, Ran He, and Xing Sun. Video-MME: The First-Ever Comprehensive Evaluation Benchmark of Multi-modal LLMs in Video Analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24108–24118, June 2025. doi: 10.1109/CVPR52734.2025.02245. URL https://openaccess.thecvf.com/content/CVPR2025/html/Fu_Video-MME_The_First-Ever_Comprehensive_Evaluation_Benchmark_of_Multi-modal_LLMs_in_CVPR_2025_paper.html.

[10] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Zhengyang Liang, Shitao Xiao, Minghao Qin, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. MLVU: Benchmarking Multi-task Long Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13691–13701, June 2025. doi: 10.1109/CVPR52734.2025.01278.

URL https://openaccess.thecvf.com/content/CVPR2025/html/Zhou_MLVU_Benchmarking_Multi-task_Long_Video_Understanding_CVPR_2025_paper.html.

[11] Hao Liang, Jiapeng Li, Tianyi Bai, Xijie Huang, Linzhuang Sun, Zhengren Wang, Conghui He, Bin Cui, Chong Chen, and Wentao Zhang. KeyVideoLLM: Towards Large-scale Video Keyframe Selection. *arXiv preprint arXiv:2407.03104*, 2024. doi: 10.48550/arXiv.2407.03104. URL https://arxiv.org/abs/2407.03104.

[12] Sicheng Yu, Chengkai Jin, Huanyu Wang, Zhenghao Chen, Sheng Jin, Zhongrong Zuo, Xiaolei Xu, Zhenbang Sun, Bingni Zhang, Jiawei Wu, Hao Zhang, and Qianru Sun. Frame-Voyager: Learning to Query Frames for Video Large Language Models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/d18d208fa9c333483e5724ade7beff0f-Abstract-Conference.html.

[13] Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive Keyframe Sampling for Long Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 29118–29128, June 2025. doi: 10.1109/CVPR52734.2025.02711. URL https://openaccess.thecvf.com/content/CVPR2025/html/Tang_Adaptive_Keyframe_Sampling_for_Long_Video_Understanding_CVPR_2025_paper.html.

[14] Hui Sun, Shiyin Lu, Huanyu Wang, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Ming Li. MDP3: A Training-free Approach for List-wise Frame Selection in Video-LLMs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 24090–24101, October 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Sun_MDP3_A_Training-free_Approach_for_List-wise_Frame_Selection_in_Video-LLMs_ICCV_2025_paper.html.

[15] Shaojie Zhang, Jiahui Yang, Jianqin Yin, Zhenbo Luo, and Jian Luan. Q-Frame: Query-aware Frame Selection and Multi-Resolution Adaptation for Video-LLMs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22056–22065, October 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Zhang_Q-Frame_Query-aware_Frame_Selection_and_Multi-Resolution_Adaptation_for_Video-LLMs_ICCV_2025_paper.html.

[16] Yanwei Li, Chengyao Wang, and Jiaya Jia. LLaMA-VID: An Image is Worth 2 Tokens in Large Language Models. In *Computer Vision – ECCV 2024*, volume 15104 of *Lecture Notes in Computer Science*, pages 323–340. Springer, 2024. doi: 10.1007/978-3-031-72952-2_19. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/html/6290_ECCV_2024_paper.php.

[17] 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 *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 54582–54599. PMLR, 2025. URL https://proceedings.mlr.press/v267/shen25j.html.

[18] Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. DyCoke: Dynamic Compression of Tokens for Fast Video Large Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18992–19001, June 2025. doi: 10.1109/CVPR52734.2025.01769. URL https://openaccess.thecvf.com/content/CVPR2025/html/Tao_DyCoke_Dynamic_Compression_of_Tokens_for_Fast_Video_Large_Language_CVPR_2025_paper.html.

[19] 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 *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5417–5432, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.283. URL https://aclanthology.org/2025.findings-acl.283/.

[20] 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 Vision Language Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22654–22663, October 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Fu_FrameFusion_Combining_Similarity_and_Importance_for_Video_Token_Reduction_on_ICCV_2025_paper.html.

[21] Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. TimeChat: A Time-sensitive Multimodal Large Language Model for Long Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14313–14323, June 2024. doi: 10.1109/CVPR52733.2024.01357. URL https://openaccess.thecvf.com/content/CVPR2024/html/Ren_TimeChat_A_Time-sensitive_Multimodal_Large_Language_Model_for_Long_Video_CVPR_2024_paper.html.

[22] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, Yan Lu, Jenq-Neng Hwang, and Gaoang Wang. MovieChat: From Dense Token to Sparse Memory for Long Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18221–18232, June 2024. doi: 10.1109/CVPR52733.2024.01725. URL https://openaccess.thecvf.com/content/CVPR2024/html/Song_MovieChat_From_Dense_Token_to_Sparse_Memory_for_Long_Video_CVPR_2024_paper.html.

[23] Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. MA-LMM: Memory-Augmented Large Multimodal Model for Long-Term Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13504–13514, June 2024. doi: 10.1109/CVPR52733.2024.01282. URL https://openaccess.thecvf.com/content/CVPR2024/html/He_MA-LMM_Memory-Augmented_Large_Multimodal_Model_for_Long-Term_Video_Understanding_CVPR_2024_paper.html.

[24] Rui Qian, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Shuangrui Ding, Dahua Lin, and Jiaqi Wang. Streaming Long Video Understanding with Large Language Models. In *Advances in Neural Information Processing Systems*, volume 37, pages 119336–119360, 2024. doi: 10.52202/079017-3792. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/d7ce06e9293c3d8e6cb3f80b4157f875-Abstract-Conference.html.

[25] 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. doi: 10.48550/arXiv.2406.16852. URL https://arxiv.org/abs/2406.16852.

[26] Yukang Chen, Fuzhao Xue, Dacheng Li, Qinghao Hu, Ligeng Zhu, Xiuyu Li, Yunhao Fang, Haotian Tang, Shang Yang, Zhijian Liu, Ethan He, Hongxu Yin, Pavlo Molchanov, Jan Kautz, Jim Fan, Yuke Zhu, Yao Lu, and Song Han. LongVILA: Scaling Long-Context Visual Language Models for Long Videos. In *Proceedings of the International Conference on Learning Representations (ICLR)*. OpenReview.net, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/2e163450c1ae3167832971e6da29f38d-Abstract-Conference.html.

[27] Biao Yang, Bin Wen, Boyang Ding, Changyi Liu, Chenglong Chu, Chengru Song, Chongling Rao, Chuan Yi, Da Li, Dunju Zang, Fan Yang, Guorui Zhou, Guowang Zhang, Han Shen, Hao Peng, Haojie Ding, Hao Wang, Haonan Fan, Hengrui Ju, Jiaming Huang, Jiangxia Cao, Jiankang Chen, Jingyun Hua, Kaibing Chen, Kaiyu Jiang, Kaiyu Tang, Kun Gai, Muhao Wei, Qiang Wang, Ruitao Wang, Sen Na, Shengnan Zhang, Siyang Mao, Sui Huang, Tianke Zhang, Tingting Gao, Wei Chen, Wei Yuan, Xiangyu Wu, Xiao Hu, Xingyu Lu, Yi-Fan Zhang, Yiping Yang, Yulong Chen, Zeyi Lu, Zhenhua Wu, Zhixin Ling, Zhuoran Yang, Ziming Li, Di Xu, Haixuan Gao, Hang Li, Jing Wang, Lejian Ren, Qigen Hu, Qianqian Wang, Shiyao Wang, Xinchen Luo, Yan Li, Yuhang Hu, and Zixing Zhang. Kwai Keye-VL 1.5 Technical Report. *arXiv preprint arXiv:2509.01563*, 2025. doi: 10.48550/arXiv.2509.01563. URL https://arxiv.org/abs/2509.01563.

[28] Chao-Yuan Wu, Manzil Zaheer, Hexiang Hu, R. Manmatha, Alexander J. Smola, and Philipp Krähenbühl. Compressed Video Action Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6026–6035, June 2018. doi: 10.1109/CVPR.2018.00631. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Wu_Compressed_Video_Action_CVPR_2018_paper.html.

[29] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1268–1277, June 2019. doi: 10.1109/CVPR.2019.00136. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Shou_DMC-Net_Generating_Discriminative_Motion_Cues_for_Fast_Compressed_Video_Action_CVPR_2019_paper.html.

[30] Zijia Zhao, Yuqi Huo, Tongtian Yue, Longteng Guo, Haoyu Lu, Bingning Wang, Weipeng Chen, and Jing Liu. Efficient Motion-Aware Video MLLM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24159–24168, June 2025. doi: 10.1109/CVPR52734.2025.02250. URL https://openaccess.thecvf.com/content/CVPR2025/html/Zhao_Efficient_Motion-Aware_Video_MLLM_CVPR_2025_paper.html.

[31] Sayan Deb Sarkar, Rémi Pautrat, Ondrej Miksik, Marc Pollefeys, Iro Armeni, Mahdi Rad, and Mihai Dusmanu. CoPE-VideoLM: Leveraging Codec Primitives For Efficient Video Language Modeling. *arXiv preprint arXiv:2602.13191*, 2026. doi: 10.48550/arXiv.2602.13191. URL https://arxiv.org/abs/2602.13191.

[32] Daichi Yashima, Shuhei Kurita, Yusuke Oda, and Komei Sugiura. ReMoRa: Multimodal Large Language Model based on Refined Motion Representation for Long-Video Understanding. *arXiv preprint arXiv:2602.16412*, 2026. doi: 10.48550/arXiv.2602.16412. URL https://arxiv.org/abs/2602.16412. Accepted to CVPR 2026.

[33] Rajesh P. N. Rao and Dana H. Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999. doi: 10.1038/4580.

[34] Thomas Wiegand, Gary J. Sullivan, Gisle Bjøntegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, 2003. doi: 10.1109/TCSVT.2003.815165.

[35] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22 (12):1649–1668, 2012. doi: 10.1109/TCSVT.2012.2221191.

[36] Jinlong Li, Liyuan Jiang, Haonan Zhang, and Nicu Sebe. Token Reduction via Local and Global Contexts Optimization for Efficient Video Large Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10451–10461, June 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Li_Token_Reduction_via_Local_and_Global_Contexts_Optimization_for_Efficient_CVPR_2026_paper.html.

[37] Feilong Tang, Xiang An, Yunyao Yan, Yin Xie, Bin Qin, Kaicheng Yang, Yifei Shen, Yuanhan Zhang, Chunyuan Li, Shikun Feng, Changrui Chen, Huajie Tan, Ming Hu, Manyuan Zhang, Bo Li, Ziyong Feng, Ziwei Liu, Zongyuan Ge, and Jiankang Deng. OneVision-Encoder: Codec-Aligned Sparsity as a Foundational Principle for Multimodal Intelligence. *arXiv preprint arXiv:2602.08683*, 2026. doi: 10.48550/arXiv.2602.08683. URL https://arxiv.org/abs/2602.08683.

[38] Haotian Ye, Qiyuan He, Jiaqi Han, Puheng Li, Jiaojiao Fan, Zekun Hao, Fitsum Reda, Yogesh Balaji, Huayu Chen, Sheng Liu, Angela Yao, James Zou, Stefano Ermon, Haoxiang Wang, and Ming-Yu Liu. InfoTok: Adaptive Discrete Video Tokenizer via Information-Theoretic Compression. *arXiv preprint arXiv:2512.16975*, 2025. doi: 10.48550/arXiv.2512.16975. URL https://arxiv.org/abs/2512.16975.

[39] Jingning Han, Bohan Li, Debargha Mukherjee, Ching-Han Chiang, Adrian Grange, Cheng Chen, Hui Su, Sarah Parker, Sai Deng, Urvang Joshi, Yue Chen, Yunqing Wang, Paul Wilkins, Yaowu Xu, and James Bankoski. A Technical Overview of AV1. *Proceedings of the IEEE*, 109(9):1435–1462, September 2021. doi: 10.1109/JPROC.2021.3058584. URL https://doi.org/10.1109/JPROC.2021.3058584.

[40] Bowen Zhang, Limin Wang, Zhe Wang, Yu Qiao, and Hanli Wang. Real-Time Action Recognition with Enhanced Motion Vector CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2718–2726, June 2016. doi: 10.1109/CVPR.2016.297. URL https://openaccess.thecvf.com/content_cvpr_2016/html/Zhang_Real-Time_Action_Recognition_CVPR_2016_paper.html.

[41] Zhengwei Wang, Qi She, and Aljosa Smolic. TEAM-Net: Multi-modal Learning for Video Action Recognition with Partial Decoding. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2021. doi: 10.5244/C.35.138. URL https://www.bmva-archive.org.uk/bmvc/2021/conference/papers/paper_0483.html.

[42] Jiawei Chen and Chiu Man Ho. MM-ViT: Multi-Modal Video Transformer for Compressed Video Action Recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 786–797, January 2022. doi: 10.1109/WACV51458.2022.00086. URL https://doi.org/10.1109/WACV51458.2022.00086.

[43] Shiyao Wang, Hongchao Lu, and Zhidong Deng. Fast Object Detection in Compressed Video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7103–7112, October 2019. doi: 10.1109/ICCV.2019.00720. URL https://doi.org/10.1109/ICCV.2019.00720.

[44] Kai Xu and Angela Yao. Accelerating Video Object Segmentation with Compressed Video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1332–1341, June 2022. doi: 10.1109/CVPR52688.2022.00140. URL https://doi.org/10.1109/CVPR52688.2022.00140.

[45] Zhipeng Fan, Jun Liu, and Yao Wang. Motion Adaptive Pose Estimation from Compressed Videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11699–11708, October 2021. doi: 10.1109/ICCV48922.2021.01151. URL https://doi.org/10.1109/ICCV48922.2021.01151.

[46] Nayoung Kim, Seong Jong Ha, and Je-Won Kang. Video Question Answering Using Language-Guided Deep Compressed-Domain Video Feature. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1688–1697, October 2021. doi: 10.1109/ICCV48922.2021.00173. URL https://doi.org/10.1109/ICCV48922.2021.00173.

[47] Yaojie Shen, Xin Gu, Kai Xu, Heng Fan, Longyin Wen, and Libo Zhang. Accurate and Fast Compressed Video Captioning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15558–15567, October 2023. doi: 10.1109/ICCV51070.2023.01426. URL https://openaccess.thecvf.com/content/ICCV2023/html/Shen_Accurate_and_Fast_Compressed_Video_Captioning_ICCV_2023_paper.html.

[48] 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. doi: 10.48550/arXiv.2511.21631. URL https://arxiv.org/abs/2511.21631.

[49] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. LongVideoBench: A Benchmark for Long-context Interleaved Video-Language Understanding. In *Advances in Neural Information Processing Systems*, volume 37, pages 28828–28857, 2024. doi: 10.52202/079017-0907. URL https://papers.nips.cc/paper_files/paper/2024/hash/329ad516cf7a6ac306f29882e9c77558-Abstract-Datasets_and_Benchmarks_Track.html. Datasets and Benchmarks Track.

[50] Weihan Wang, Zehai He, Wenyi Hong, Yean Cheng, Xiaohan Zhang, Ji Qi, Ming Ding, Xiaotao Gu, Shiyu Huang, Bin Xu, Yuxiao Dong, and Jie Tang. LVBench: An Extreme Long Video Understanding Benchmark. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22958–22967, October 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Wang_LVBench_An_Extreme_Long_Video_Understanding_Benchmark_ICCV_2025_paper.html.

[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? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8731–8772, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.517. URL https://aclanthology.org/2024.findings-acl.517/.

[52] Wenyi Hong, Yean Cheng, Zhuoyi Yang, Weihan Wang, Lefan Wang, Xiaotao Gu, Shiyu Huang, Yuxiao Dong, and Jie Tang. MotionBench: Benchmarking and Improving Fine-grained Video Motion Understanding for Vision Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8450–8460, June 2025. doi: 10.1109/CVPR52734.2025.00791. URL https://openaccess.thecvf.com/content/CVPR2025/html/Hong_MotionBench_Benchmarking_and_Improving_Fine-grained_Video_Motion_Understanding_for_Vision_CVPR_2025_paper.html.

[53] 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. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/16ba99f25a235f1100a4014d71d34ad8-Abstract-Conference.html.

[54] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, Limin Wang, and Yu Qiao. MVBench: A Comprehensive Multi-modal Video Understanding Benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22195–22206, June 2024. doi: 10.1109/CVPR52733.2024.02095. URL https://openaccess.thecvf.com/content/CVPR2024/html/Li_MVBench_A_Comprehensive_Multi-modal_Video_Understanding_Benchmark_CVPR_2024_paper.html.

[55] 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 (CVPR)*, pages 9777–9786, June 2021. doi: 10.1109/CVPR46437.2021.00965. URL https://openaccess.thecvf.com/content/CVPR2021/html/Xiao_NExT-QA_Next_Phase_of_Question-Answering_to_Explaining_Temporal_Actions_CVPR_2021_paper.html.

[56] Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Joseph Heyward, Mateusz Malinowski, Yi Yang, Carl Doersch, Tatiana Matejovicova, Yury Sulsky, Antoine Miech, Alexandre Fréchette, Hanna Klimczak, Raphael Koster, Junlin Zhang, Stephanie Winkler, Yusuf Aytar, Simon Osindero, Dima Damen, Andrew Zisserman, and Joao Carreira. Perception Test: A diagnostic benchmark for multimodal video models. In *Advances in Neural Information Processing Systems*, volume 36, pages 42748–42761, 2023. doi: 10.52202/075280-1852. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8540fba4abdc7f9f7a7b1cc6cd60e409-Abstract-Datasets_and_Benchmarks.html. Datasets and Benchmarks Track.

[57] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. EgoSchema: A Diagnostic Benchmark for Very Long-form Video Language Understanding. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.nips.cc/paper_files/paper/2023/hash/90ce332aff156b910b002ce4e6880dec-Abstract-Datasets_and_Benchmarks.html. Datasets and Benchmarks Track.

[58] Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. LMMs-eval: Reality check on the evaluation of large multimodal models. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 881–916, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.51. URL https://aclanthology.org/2025.findings-naacl.51/.

[59] Christopher Clark, Jieyu Zhang, Zixian Ma, Jae Sung Park, Mohammadreza Salehi, Rohun Tripathi, Sangho Lee, Zhongzheng Ren, Chris Dongjoo Kim, Yinuo Yang, Vincent Shao, Yue Yang, Weikai Huang, Ziqi Gao, Taira Anderson, Jianrui Zhang, Jitesh Jain, George Stoica, Winson Han, Ali Farhadi, and Ranjay Krishna. Molmo2: Open Weights and Data for Vision-Language Models with Video Understanding and Grounding. *arXiv preprint arXiv:2601.10611*, 2026. doi: 10.48550/arXiv.2601.10611. URL https://arxiv.org/abs/2601.10611.

[60] OpenAI. GPT-5 System Card, August 2025. URL https://openai.com/index/gpt-5-system-card/.

[61] Google DeepMind. Gemini 3 Pro - Model Card, December 2025. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf. Model card update: December 2025.

[62] Google DeepMind. Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities, 2025. URL https://storage.googleapis.com/deepmind-media/gemini/gemini_v2_5_report.pdf.

[63] Anthropic. Claude Sonnet 4.5 System Card, September 2025. URL https://assets.anthropic.com/m/12f214efcc2f457a/original/Claude-Sonnet-4-5-System-Card.pdf.

[64] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. InternVL3.5: Advancing Open-Source Multimodal Models in Versatility, Reasoning, and Efficiency. *arXiv preprint arXiv:2508.18265*, 2025. doi: 10.48550/arXiv.2508.18265. URL https://arxiv.org/abs/2508.18265.

[65] GLM-V Team, Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, Shuaiqi Duan, Weihan Wang, Yan Wang, Yean Cheng, Zehai He, Zhe Su, Zhen Yang, Ziyang Pan, Aohan Zeng, Baoxu Wang, Bin Chen, Boyan Shi, Changyu Pang, Chenhui Zhang, Da Yin, Fan Yang, Guoqing Chen, Haochen Li, Jiale Zhu, Jiali Chen, Jiaxing Xu, Jiazheng Xu, Jing Chen, Jinghao Lin, Jinhao Chen, Jinjiang Wang, Junjie Chen, Leqi Lei, Letian Gong, Leyi Pan, Mingdao Liu, Mingde Xu, Mingzhi Zhang, Qinkai Zheng, Ruiliang Lyu, Shangqin Tu, Sheng Yang, Shengbiao Meng, Shi Zhong, Shiyu Huang, Shuyuan Zhao, Siyan Xue, Tianshu Zhang, Tianwei Luo, Tianxiang Hao, Tianyu Tong, Wei Jia, Wenkai Li, Xiao Liu, Xiaohan Zhang, Xin Lyu, Xinyu Zhang, Xinyue Fan, Xuancheng Huang, Yadong Xue, Yanfeng Wang, Yanling Wang, Yanzi Wang, Yifan An, Yifan Du, Yiheng Huang, Yilin Niu, Yiming Shi, Yu Wang, Yuan Wang, Yuanchang Yue, Yuchen Li, Yusen Liu, Yutao Zhang, Yuting

Wang, Yuxuan Zhang, Zhao Xue, Zhengxiao Du, Zhenyu Hou, Zihan Wang, Peng Zhang, Debing Liu, Bin Xu, Juanzi Li, Minlie Huang, Yuxiao Dong, and Jie Tang. GLM-4.5V and GLM-4.1V-Thinking: Towards Versatile Multimodal Reasoning with Scalable Reinforcement Learning. *arXiv preprint arXiv:2507.01006*, 2025. doi: 10.48550/arXiv.2507.01006. URL https://arxiv.org/abs/2507.01006.

[66] Tianyu Yu, Zefan Wang, Chongyi Wang, Fuwei Huang, Wenshuo Ma, Zhihui He, Tianchi Cai, Weize Chen, Yuxiang Huang, Yuanqian Zhao, Bokai Xu, Junbo Cui, Yingjing Xu, Liqing Ruan, Luoyuan Zhang, Hanyu Liu, Jingkun Tang, Hongyuan Liu, Qining Guo, Wenhao Hu, Bingxiang He, Jie Zhou, Jie Cai, Ji Qi, Zonghao Guo, Chi Chen, Guoyang Zeng, Yuxuan Li, Ganqu Cui, Ning Ding, Xu Han, Yuan Yao, Zhiyuan Liu, and Maosong Sun. MiniCPM-V 4.5: Cooking Efficient MLLMs via Architecture, Data, and Training Recipe. *arXiv preprint arXiv:2509.18154*, 2025. doi: 10.48550/arXiv.2509.18154. URL https://arxiv.org/abs/2509.18154.

[67] Guo Chen, Zhiqi Li, Shihao Wang, Jindong Jiang, Yicheng Liu, Lidong Lu, De-An Huang, Wonmin Byeon, Matthieu Le, Tuomas Rintamaki, Tyler Poon, Max Ehrlich, Tong Lu, Limin Wang, Bryan Catanzaro, Jan Kautz, Andrew Tao, Zhiding Yu, and Guilin Liu. Eagle 2.5: Boosting Long-Context Post-Training for Frontier Vision-Language Models. *arXiv preprint arXiv:2504.15271*, 2025. doi: 10.48550/arXiv.2504.15271. URL https://arxiv.org/abs/2504.15271.

[68] Jang Hyun Cho, Andrea Madotto, Effrosyni Mavroudi, Triantafyllos Afouras, Tushar Nagarajan, Muhammad Maaz, Yale Song, Tengyu Ma, Shuming Hu, Suyog Jain, Miguel Martin, Huiyu Wang, Hanoona Rasheed, Peize Sun, Po-Yao Huang, Daniel Bolya, Nikhila Ravi, Shashank Jain, Tammy Stark, Shane Moon, Babak Damavandi, Vivian Lee, Andrew Westbury, Salman Khan, Philipp Krähenbühl, Piotr Dollár, Lorenzo Torresani, Kristen Grauman, and Christoph Feichtenhofer. PerceptionLM: Open-Access Data and Models for Detailed Visual Understanding. *arXiv preprint arXiv:2504.13180*, 2025. doi: 10.48550/arXiv.2504.13180. URL https://arxiv.org/abs/2504.13180.

[69] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. LLaVA-Video: Video Instruction Tuning With Synthetic Data. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=EElFGvt39K.

[70] 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*, 2025. doi: 10.48550/arXiv.2501.00574. URL https://arxiv.org/abs/2501.00574.

[71] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient Context Window Extension of Large Language Models. *arXiv preprint arXiv:2309.00071*, 2023. doi: 10.48550/arXiv.2309.00071. URL https://arxiv.org/abs/2309.00071. Revised February 2026.

[72] 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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):9127–9134, 2019. doi: 10.1609/aaai.v33i01.33019127. URL https://doi.org/10.1609/aaai.v33i01.33019127.

[73] Weijia Wu, Yuzhong Zhao, Zhuang Li, Jiahong Li, Hong Zhou, Mike Zheng Shou, and Xiang Bai. A Large Cross-Modal Video Retrieval Dataset with Reading Comprehension. *arXiv preprint arXiv:2305.03347*, 2023. doi: 10.48550/arXiv.2305.03347. URL https://arxiv.org/abs/2305.03347.

[74] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding. In *Computer Vision – ECCV 2016*, volume 9905 of *Lecture Notes in Computer Science*, pages 510–526. Springer, 2016. doi: 10.1007/978-3-319-46448-0_31. URL https://doi.org/10.1007/978-3-319-46448-0_31.

[75] Anna Rohrbach, Atousa Torabi, Marcus Rohrbach, Niket Tandon, Christopher Joseph Pal, Hugo Larochelle, Aaron C. Courville, and Bernt Schiele. Movie Description. *International Journal of Computer Vision*, 123 (1):94–120, 2017. doi: 10.1007/s11263-016-0987-1. URL https://link.springer.com/article/10.1007/s11263-016-0987-1.

[76] Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. TGIF-QA: Toward Spatio-Temporal Reasoning in Visual Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1359–1367, July 2017. doi: 10.1109/CVPR.2017.149. URL https://doi.org/10.1109/CVPR.2017.149.

[77] Gunnar A. Sigurdsson, Abhinav Gupta, Cordelia Schmid, Ali Farhadi, and Karteek Alahari. Actor and Observer: Joint Modeling of First and Third-Person Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7396–7404, June 2018. doi: 10.1109/CVPR.2018.00772. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Sigurdsson_Actor_and_Observer_CVPR_2018_paper.html.

[78] Xiyao Wang, Yuhang Zhou, Xiaoyu Liu, Hongjin Lu, Yuancheng Xu, Feihong He, Jaehong Yoon, Taixi Lu, Fuxiao Liu, Gedas Bertasius, Mohit Bansal, Huaxiu Yao, and Furong Huang. Mementos: A Comprehensive Benchmark for Multimodal Large Language Model Reasoning over Image Sequences. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 416–442, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.25. URL https://aclanthology.org/2024.acl-long.25/.

[79] Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L. Berg. TVQA: Localized, Compositional Video Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1369–1379, Brussels, Belgium, October–November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1167. URL https://aclanthology.org/D18-1167/.

[80] Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B. Tenenbaum. CLEVRER: Collision Events for Video Representation and Reasoning. In *Proceedings of the International Conference on Learning Representations (ICLR)*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HkxYzANYDB.

[81] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan C. Russell. Localizing Moments in Video With Natural Language. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5804–5813, October 2017. doi: 10.1109/ICCV.2017.618. URL https://doi.org/10.1109/ICCV.2017.618.

[82] Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Junfei Wu, Xiaoying Zhang, Benyou Wang, and Xiangyu Yue. Video-R1: Reinforcing Video Reasoning in MLLMs. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=a2JTVVvcEl. NeurIPS 2025 poster.

[83] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina González, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jáchym Kolár, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran K. Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbeláez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard A. Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4D: Around the World in 3,000 Hours of Egocentric Video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18973–18990, June 2022. doi: 10.1109/CVPR52688.2022.01842. URL https://doi.org/10.1109/CVPR52688.2022.01842.

[84] Kaituo Feng, Manyuan Zhang, Hongyu Li, Kaixuan Fan, Shuang Chen, Yilei Jiang, Dian Zheng, Peiwen Sun, Yiyuan Zhang, Haoze Sun, Yan Feng, Peng Pei, Xunliang Cai, and Xiangyu Yue. OneThinker: All-in-one Reasoning Model for Image and Video. *arXiv preprint arXiv:2512.03043*, 2025. doi: 10.48550/arXiv.2512.03043. URL https://arxiv.org/abs/2512.03043.

[85] Kevin Qinghong Lin, Jinpeng Wang, Mattia Soldan, Michael Wray, Rui Yan, Eric Z. Xu, Difei Gao, Rong-Cheng Tu, Wenzhe Zhao, Weijie Kong, Chengfei Cai, Hongfa Wang, Dima Damen, Bernard Ghanem, Wei Liu, and Mike Zheng Shou. Egocentric Video-Language Pretraining. In *Advances in Neural Information Processing Systems*, volume 35, pages 7575–7586. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/31fb284a0aaaad837d2930a610cd5e50-Abstract-Conference.html.

[86] Songhao Han, Wei Huang, Hairong Shi, Le Zhuo, Xiu Su, Shifeng Zhang, Xu Zhou, Xiaojuan Qi, Yue Liao, and Si Liu. VideoEspresso: A Large-Scale Chain-of-Thought Dataset for Fine-Grained Video Reasoning via Core Frame Selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26181–26191, June 2025. doi: 10.1109/CVPR52734.2025.02438. URL https://openaccess.thecvf.com/content/CVPR2025/html/Han_VideoEspresso_A_Large-Scale_Chain-of-Thought_Dataset_for_Fine-Grained_Video_Reasoning_via_CVPR_2025_paper.html.

[87] Siddhant Bansal, Chetan Arora, and C. V. Jawahar. My View is the Best View: Procedure Learning from Egocentric Videos. In *Computer Vision – ECCV 2022*, volume 13673 of *Lecture Notes in Computer*

Science, pages 657–675. Springer, 2022. doi: 10.1007/978-3-031-19778-9_38. URL https://doi.org/10.1007/978-3-031-19778-9_38.

[88] Andreea-Maria Oncescu, João F. Henriques, Yang Liu, Andrew Zisserman, and Samuel Albanie. QUERYD: A Video Dataset with High-Quality Text and Audio Narrations. In *ICASSP 2021 – 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2265–2269. IEEE, 2021. doi: 10.1109/ICASSP39728.2021.9414640. URL https://doi.org/10.1109/ICASSP39728.2021.9414640.

[89] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. VideoGPT+: Integrating Image and Video Encoders for Enhanced Video Understanding. *arXiv preprint arXiv:2406.09418*, 2024. doi: 10.48550/arXiv.2406.09418. URL https://arxiv.org/abs/2406.09418.

[90] Thong Thanh Nguyen, Zhiyuan Hu, Xiaobao Wu, Cong-Duy T. Nguyen, See-Kiong Ng, and Anh Tuan Luu. Encoding and Controlling Global Semantics for Long-form Video Question Answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7049–7066, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.400. URL https://aclanthology.org/2024.emnlp-main.400/.

[91] Lin Chen, Xilin Wei, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Bin Lin, Zhenyu Tang, Li Yuan, Yu Qiao, Dahua Lin, Feng Zhao, and Jiaqi Wang. ShareGPT4Video: Improving Video Understanding and Generation with Better Captions. In *Advances in Neural Information Processing Systems*, volume 37, pages 19472–19495. Curran Associates, Inc., 2024. doi: 10.52202/079017-0614. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/22a7476e4fd36818777c47e666f61a41-Abstract-Datasets_and_Benchmarks_Track.html. Datasets and Benchmarks Track.

[92] Dongjie Yang, Suyuan Huang, Chengqiang Lu, Xiaodong Han, Haoxin Zhang, Yan Gao, Yao Hu, and Hai Zhao. Vript: A Video Is Worth Thousands of Words. In *Advances in Neural Information Processing Systems*, volume 37, pages 57240–57261. Curran Associates, Inc., 2024. doi: 10.52202/079017-1824. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/6903a5aaece71b76623245fc6e32f01b-Abstract-Datasets_and_Benchmarks_Track.html. Datasets and Benchmarks Track.

[93] Abhay Zala, Jaemin Cho, Satwik Kottur, Xilun Chen, Barlas Oguz, Yashar Mehdad, and Mohit Bansal. Hierarchical Video-Moment Retrieval and Step-Captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23056–23065, June 2023. doi: 10.1109/CVPR52729.2023.02208. URL https://openaccess.thecvf.com/content/CVPR2023/html/Zala_Hierarchical_Video-Moment_Retrieval_and_Step-Captioning_CVPR_2023_paper.html.

[94] Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. SQA3D: Situated Question Answering in 3D Scenes. In *International Conference on Learning Representations*. OpenReview.net, 2023. URL https://openreview.net/forum?id=IDJx97BC38.

[95] Yongxin Guo, Jingyu Liu, Mingda Li, Dingxin Cheng, Xiaoying Tang, Dianbo Sui, Qingbin Liu, Xi Chen, and Kevin Zhao. VTG-LLM: Integrating Timestamp Knowledge into Video LLMs for Enhanced Video Temporal Grounding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(3):3302–3310, 2025. doi: 10.1609/aaai.v39i3.32341. URL https://ojs.aaai.org/index.php/AAAI/article/view/32341.

[96] Bo Wu, Shoubin Yu, Zhenfang Chen, Josh Tenenbaum, and Chuang Gan. STAR: A Benchmark for Situated Reasoning in Real-World Videos. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/5ef059938ba799aaa845e1c2e8a762bd-Abstract-round2.html.

[97] Luowei Zhou, Chenliang Xu, and Jason J. Corso. Towards Automatic Learning of Procedures From Web Instructional Videos. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1):7590–7598, 2018. doi: 10.1609/aaai.v32i1.12342. URL https://ojs.aaai.org/index.php/AAAI/article/view/12342.

Appendix

A Codec Redesign Details

Table 7 reports every component in which AdaCodec departs from a standards-compliant playback codec. The first four rows reproduce the core redesigns highlighted in the main text (Section 3); the last three (color space, frame types, entropy coding) are the choices that follow from targeting an MLLM-token sequence rather than a playback bitstream.

Table 7: Full component-wise comparison between a playback-oriented codec and AdaCodec. The upper block reproduces the core redesigns of Section 3; the lower block lists the additional configuration choices specific to the MLLM-token target.

Component	**Playback-oriented codec**	**AdaCodec (MLLM-oriented)**
Block partition	Heterogeneous block sizes chosen for bitrate.	Macroblocks aligned to the ViT patch grid, yielding more stable P-frame tokens.
Motion reference	Reference pictures selected under codec syntax.	Each P-frame is estimated from the immediately preceding sampled frame to handle high motions in larger temporal gaps.
Search window	Tuned to high-FPS playback.	Enlarged local window to absorb the larger displacement between low-FPS frames.
GOP scheduling	Separate content-analysis pass.	Reuses the predictive cost from motion search to trigger adaptive I-frame insertion for efficiency.
Color space	YCbCr with chroma subsampling.	RGB, matching vision-backbone inputs.
Frame types	I, P, and bidirectional B.	I/P only; each predictive frame uses past context, compatible with streaming.
Entropy coding	Required for bitstreams.	Omitted; the output is a token sequence, not a bitstream.

B Model Implementation Details

P-token construction. For each P-frame, AdaCodec first forms the five-channel tensor $u_t = [r_t,\, M_t] \in \mathbb{R}^{H\times W\times 5}$ from the residual and motion vectors defined in Section 3. The P-tokenizer uses an architecture-matched ViT backbone initialized from a pretrained visual encoder, with the patch-embedding input widened from three to five channels. The RGB kernels are copied from the pretrained stem, and the two motion-vector channels are initialized to zero before training. A standard ViT admits extra learnable tokens appended after the patch sequence without any change to the backbone, so we attach N_L learnable latent tokens to the patch tokens before the visual transformer:

$$\tilde{z}_t^P = F_P([B_P(u_t), q_1, \ldots, q_{N_L}])_{q_1:q_{N_L}}\,, \tag{9}$$

where B_P is the patch embedding, F_P is the residual visual backbone, and the subscript selects the output states of the appended latent tokens. A block attention mask prevents patch tokens from attending to the latent tokens, while the latent tokens attend to all patch tokens. This mask preserves the pretrained patch-token computation while letting the latent tokens aggregate information from the full predictive representation. The resulting P-tokens are learned summary tokens conditioned on the residual-and-motion representation, rather than sampled image patches.

Stage 1 reconstruction module. Stage 1 trains the P-frame tokenizer E_P through an auxiliary feature predictor H_ϕ that maps the I-frame embedding and the P-token states back to the teacher feature at the target frame (Section 3). The reconstruction head is active only in Stage 1 and is removed before Stage 2, leaving only E_P for downstream multimodal alignment.

Qwen3-VL ViT interface. Qwen3-VL introduces three changes to the standard ViT visual encoder that the P-tokenizer must match: (a) the patch embedding is a Conv3D with temporal size two, which lets the stem take two frames jointly rather than a single image; (b) a 2×2 spatial merger at the

Table 8: Shared public source families for Stage 1 and Stage 2 training. Molmo2 contributes its AskModelAnything, VideoCapQA, and VideoSubtitleQA subsets. Split handling is described in the text.

Dataset		
ActivityNet-QA [72]	LongVILA [26]	TextVR [73]
Charades [74]	LSMDC [75]	TGIF-QA [76]
CharadesEgo [77]	Mementos [78]	TVQA [79]
CLEVRER [80]	Molmo2 [59]	Video-ChatGPT [2]
DiDeMo [81]	NExT-QA [55]	Video-R1 [82]
Ego4D [83]	OneThinker [84]	VideoChat2 [54]
EgoClip [85]	Perception Test [56]	VideoEspresso [86]
EgoProceL [87]	QuerYD [88]	VideoGPT+ [89]
EgoQA [90]	ShareGPT4Video [91]	Vript [92]
HiREST [93]	SQA3D [94]	VTG-IT [95]
LLaVA-Video-178K [69]	STAR [96]	YouCook2 [97]

output reduces every four adjacent tokens into one merged visual token; and (c) three intermediate layers are exported through a DeepStack visual-injection path into the first three language-model layers. AdaCodec adapts to each change without modifying the pretrained backbone. For (a), we encode a single I-frame or P-frame tensor by duplicating it along the temporal dimension and feeding the pair to the original Conv3D stem. For (b), we apply the same 2×2 merger to both streams: at 512×512 inputs with patch size 16, an I-frame yields 32×32 patch tokens and $N_I = 256$ merged visual tokens, and the N_L latent P-token states are arranged as a square grid and passed through the same merger, giving $N_P = N_L/4$ merged P-tokens per P-frame. For (c), the P-tokenizer exposes the matching intermediate layers alongside the final output, and the Qwen3-VL forward pass feeds them through the native DeepStack injection path.

C Training Details

Both training stages use the same public video-instruction data source. Stage 1 samples tuples containing a reference frame, a sequence of intermediate P-frames, and a target frame from the training videos for teacher-feature alignment; Stage 2 uses the paired instruction-response examples for next-token training. The Stage 2 instruction-tuning mixture contains 3,904,313 training examples. Table 8 lists the shared public source families. When a dataset has official train/validation/test partitions, both stages use only the training partition.

Training hyperparameters. Table 9 reports the training hyperparameters for the two training stages. During stage 2 training, we train with 64k visual token budget for 40,000 steps, and then train on 224k for 5,000 steps.

Table 9: Training hyperparameters for AdaCodec.

Setting	Stage 1	Stage 2
Optimizer	AdamW, $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.01	AdamW, $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$, weight decay 0
Learning rate	1×10^{-4} peak; cosine decay to 1×10^{-5}; 4,000 warmup steps	1×10^{-5} peak; cosine decay; 0.1 warmup ratio
Global batch size	128	128
Optimization steps	130,000	45,000

Compute resources. The training runs on 64 NVIDIA H800 GPUs and span approximately 12 days of wall-clock time. Latency measurements in Section 4.3 use a single H800 for prefill and decoding, and the codec-build step is timed on a 16-core consumer CPU. The full research effort uses more resources, including preliminary runs and ablation experiments.

D Benchmark Descriptions

We evaluate on eleven public video benchmarks described below.

Long-video benchmarks. MLVU samples long-form videos from heterogeneous genres such as movies, surveillance, egocentric clips, cartoons, and gameplay, and pairs each clip with multi-task evaluation across varying durations and task types [10]. LongVideoBench is a multiple-choice QA benchmark for video-language interleaved inputs up to one hour long; its referring-reasoning questions ask the model to retrieve the relevant video context and reason over detailed multimodal evidence [49]. LVBench targets extreme long-video understanding on public videos spanning hours, with tasks designed around long-term memory, extended comprehension, and information extraction [50].

Temporal benchmarks. TempCompass evaluates temporal perception over attributes such as speed and direction, and requests answers in multiple formats. To force genuine temporal reasoning, it pairs videos that hold their static content fixed while diverging in time-varying attributes, so single-frame cues and language priors cannot suffice [51]. MotionBench assesses how well video models comprehend fine-grained motion. The benchmark draws clips from heterogeneous sources and partitions evaluation into six motion-oriented question categories, each targeting a specific aspect of motion-level perception [52]. TOMATO targets visual temporal reasoning. It defines six task types: action count, direction, rotation, shape and trend, velocity and frequency, and visual cues. TOMATO is designed so that the answer requires more than a single frame, the original frame order, and evidence drawn from across the clip [53].

General video-understanding benchmarks. Video-MME evaluates MLLMs on short, medium, and long videos drawn from six visual domains and 30 subfields, testing whether models can answer video-centered questions across diverse content and temporal scales [9]. MVBench converts public video annotations into 20 multiple-choice tasks that require temporal understanding beyond a single frame [54]. NExT-QA defines three video-QA question types (causal action reasoning, temporal action reasoning, and common-scene comprehension), framed to move beyond surface scene description toward explanation of actions [55]. PerceptionTest probes the transfer of pre-trained video models. The benchmark pairs four skill areas (memory, abstraction, physics, and semantics) with four reasoning types (descriptive, explanatory, predictive, and counterfactual), administered jointly over video, audio, and text inputs [56]. EgoSchema is built from Ego4D three-minute egocentric clips and asks five-option questions whose evidence spans long temporal certificate sets, making it a test of first-person video reasoning [57].

E Efficiency and Latency Details

Token-efficiency derivation. For a GOP with one I-frame and K P-frames, the visual-token ratio against a per-frame RGB input is

$$\rho(K) = \frac{N_I + KN_P}{(K+1)N_I}, \tag{10}$$

where N_I and N_P are tokens per I-frame and per P-frame. With AdaCodec's default setup, $N_I = 256$ and $N_P = 16$. Under the maximum predictive chain length used in our implementation (16 P-frames per GOP), the architectural minimum ratio is $\rho(16) = 0.118$, an 88.2% reduction. In practice, GOP length is content-dependent. Aggregated over all evaluation videos, the realized average GOP length is 10.21 frames, i.e., 9.21 P-frames per GOP. Substituting the empirical average into Eq. (10) yields a 15.4% visual-token ratio, an 84.6% reduction relative to per-frame RGB input.

Latency measurement protocol. We evaluate runtime on the five general video-understanding benchmarks listed in Section 4.1 using time-to-first-token (TTFT) and total end-to-end latency (E2EL). All methods run on the same hardware with batch size 1, the same prompt template, identical decoding hyperparameters, the same 64 generated tokens, and the same input resolution. AdaCodec uses its I/P-frame visual code, whereas the per-frame RGB baseline feeds every frame as an RGB image. Aggregated over 11,347 unique videos, AdaCodec uses 8,550.4 visual tokens per video on average, whereas the per-frame RGB baseline uses 55,893.2 (84.7% reduction). Reducing visual prefilling lowers TTFT by design; AdaCodec also reduces total generation latency while improving downstream score, so the gain cannot be explained by discarding visual information for speed.

Codec-build overhead. Constructing the AdaCodec visual code online incurs a one-time codec-build step per video: predictive coding calculation and *pcost*-driven GOP splitting. On a 16-core consumer CPU, codec-build takes 0.12s per video, about 7% of AdaCodec's 1.62s TTFT. Folding it back into TTFT yields 1.74s, still 5.3× faster than the per-frame RGB baseline's 9.26s, and E2EL retains a 3.4× gap. The overhead is small in absolute terms and does not change the latency advantage of the AdaCodec visual code.

Memory footprint. AdaCodec adds one ViT-sized visual branch for the P-frame tokenizer (about 576M parameters, 7% of the 8.14B-parameter backbone). Measured under FP16, AdaCodec increases peak GPU memory over the per-frame RGB baseline by 1.9 GB (5.4%).

F Adaptive GOP Behavior Details

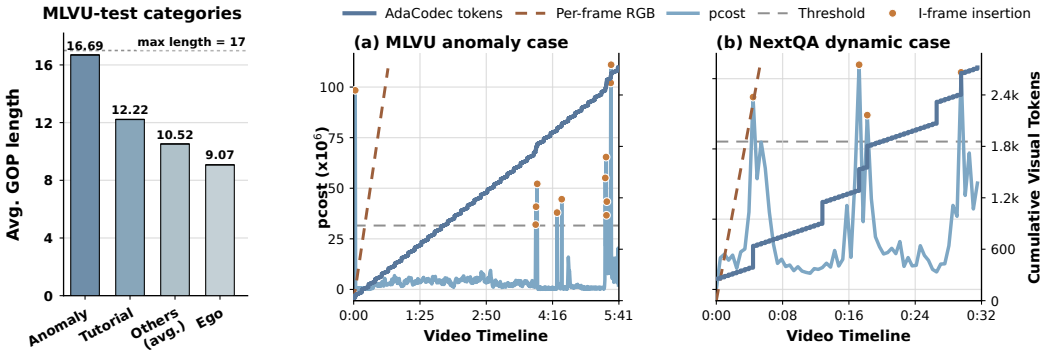

Figure 5: Adaptive GOP behavior in AdaCodec. Left: average GOP length for representative MLVU-test categories; "Others" averages the remaining official categories. Middle: an MLVU anomaly case. Right: a dynamic case from NextQA. The per-frame RGB trajectory uses 256 tokens per frame and quickly exits the plotting range, highlighting how AdaCodec avoids runaway token growth.

Per-category mechanism. We take MLVU `test` [10] as a case study. The left panel of Figure 5 reports the three representative categories discussed in Section 4.5 (anomaly recognition, tutorial-style videos, and ego reasoning) and averages the remaining official categories into "Others." The category-level variation is consistent with the composition of MLVU. Anomaly recognition is built from surveillance-style videos with relatively fixed cameras and slowly evolving scenes. Tutorial videos often exhibit a more stable camera setup and step-wise temporal structure, which also permits longer GOPs. Ego reasoning is derived from egocentric first-person videos with frequent viewpoint changes and stronger camera motion. The averaged "Others" category lies between these cases, which is expected because it mixes videos with different levels of camera motion and event density. Thus, the per-category trend provides an interpretable explanation for the adaptive behavior: it allocates more visual tokens to temporally unstable videos and compresses videos with higher inter-frame redundancy.

Case studies. The two case studies on the right of Figure 5 visualize the same mechanism at the video level. The middle panel shows an MLVU anomaly video with a long low-*pcost* interval and only late bursts, while the right panel uses a more dynamic example from NExT-QA, where multiple spikes trigger earlier I-frame refreshes.

From mechanism to accuracy. When videos remain visually stable for long periods, AdaCodec preserves much more of the original sequence within the same context budget, exposing a more complete temporal record to the MLLM. This explains the +20.6 MLVU anomaly-recognition gain reported in Section 4.5: the proposed *pcost*-guided construction allocates longer GOPs to temporally stable videos and shorter GOPs to content with frequent scene or motion changes, matching token budget to video complexity rather than enforcing a fixed schedule.

G Ablation Protocol and Per-Axis Analysis

Why subset-based ablations. Running the full AdaCodec training pipeline for every codec variant would be prohibitively expensive in both wall-clock time and GPU hours. We therefore perform ablations on a fixed subset that contains randomly sampled 1/3 of the full training corpus instead of repeating full-data training for every setting. The subset contains 1,301,438 examples, which is still large enough to support stable training and reveal the qualitative trends.

Protocol. All ablated variants use the same backbone, optimizer, learning-rate, input resolution, frame rate, and decoding protocol as the full model. We train each variant on the same reduced training split for the same number of optimization steps. This keeps the ablations focused on relative ranking among design choices rather than absolute leaderboard performance. After identifying the final operating point, we retrain that configuration on the full training data.

Table 10: Full ablation sweep over codec-design axes of AdaCodec. Each value is the mean score over the benchmarks in its category. The top row reports the subset-trained default; each subsequent row varies a single axis and shows the absolute score together with its change relative to the default in subscript. For the *pcost* threshold target, K denotes the number of P-frames per GOP, so the total GOP length is $K+1$ including the I-frame. The dynamic macroblock setting follows H.264 variable block size: 16×16, 16×8, 8×16, 8×8, selected per region by motion and residual complexity.

Axis	Setting	Long-video↑	Temporal↑	General↑
AdaCodec	N_P=16, $K_{\max}$=16, Med. K=8, MB=16×16, adaptive GOP	**60.3**	**56.2**	**74.1**
P-token count N_P	8	60.4 +0.1	55.8 -0.4	73.9 -0.2
	24	59.8 -0.5	56.1 -0.1	74.4 +0.3
Max #P-frames per GOP $K_{\max}$	8	59.4 -0.9	56.2 ±0.0	74.2 +0.1
	24	60.1 -0.2	56.0 -0.2	73.5 -0.6
pcost threshold target	Median $K=4$	59.2 -1.1	55.7 -0.5	73.8 -0.3
	Median $K=12$	60.0 -0.3	55.5 -0.7	73.6 -0.5
Macroblock size	Dynamic (H.264-style)	58.4 -1.9	54.8 -1.4	72.6 -1.5
GOP construction	Fixed, $n_P=8$	58.2 -2.1	56.0 -0.2	73.7 -0.4
	Fixed, $n_P=16$	59.7 -0.6	55.4 -0.8	72.3 -1.8

Per-axis analysis. Table 10 ablates the P-token count N_P (how many learned tokens represent one P-frame), the maximum number of P-frames per GOP $K_{\max}$, the *pcost* threshold target used to calibrate γ, the macroblock size used for motion and residual modeling (16×16 aligned with ViT patches versus H.264-style dynamic partitioning), and the GOP construction strategy (our adaptive *pcost*-guided policy versus fixed-length baselines with $n_P \in \{8, 16\}$ P-frames per GOP). AdaCodec is largely insensitive to the P-token count: N_P=16 is best, but 8 and 24 stay within 0.5 on every category. The largest hit appears on long-video at N_P=24 (-0.5), where each P-frame consumes more tokens and fewer frames fit the same budget. The same mechanism explains the chain-length sweep: $K_{\max}$=8 loses 0.9 on long-video because shorter chains insert more token-heavy I-frames and shrink the usable frame count. The *pcost* threshold sweep gives the strongest results at the default median $K=8$. A shorter target, median $K=4$, refreshes I-frames more often and reduces temporal coverage under the same context budget, while a longer target, median $K=12$, saves more tokens but increases the length of predictive chains, which hurts dynamic videos with larger residuals. Replacing the ViT-aligned 16×16 macroblocks with native H.264 dynamic partitioning costs 1.4–1.9 across all three categories, since the multiple block sizes break the per-patch motion-vector uniformity that the P-tokenizer's patch-embedding stem relies on. Adaptive *pcost*-guided GOP construction beats both fixed-length baselines on every category, with the largest gaps on long-video (2.1 for n_P=8) and general (1.8 for n_P=16); this directly matches the content-dependent GOP-length variation in Figure 5, where slow-content videos (e.g., anomaly recognition) receive longer GOPs and dynamic ones receive shorter, a regime no fixed schedule can capture.

H Broader Impacts and Responsible Release

AdaCodec pushes the accuracy-cost frontier for video MLLMs, which can reduce compute cost and make long-video analysis easier. These gains apply to benign uses such as educational video review, and assistive video analysis, but the same efficiency could also lower the cost of privacy-sensitive analysis. AdaCodec inherits, but does not amplify, the biases of its public video-instruction training data and the Qwen3-VL-8B base model; downstream deployments should apply standard fairness and privacy auditing, especially when the input contains identifiable individuals or sensitive scenes.

I Licenses and Terms for External Assets

Table 11 and Table 12 list the license names or source-specific terms for the public data assets used in training and evaluation. Each asset is cited in the corresponding data-source or benchmark table above, and each name below links to an official source when one is available. When a dataset combines released annotations with third-party videos, images, audio, or 3D scans, the table reports the annotation license and the stricter source-media terms stated by the dataset creators. We use all data for academic research, and non-commercially where required, through official splits and access paths. We do not redistribute raw videos, images, audio, 3D scans, or repackaged dataset files with this paper; any code or model release will preserve the relevant notices and will not include dataset contents. For sources without a standard public license, we treat the data as research-use only and do not redistribute it.

Table 11: Licenses and use terms for training data sources. "Source terms" denotes restrictions attached to underlying media or access-controlled data when those terms are stricter than the annotation or code license.

Source	License or Terms
ActivityNet-QA [72]	MIT-named README terms with non-commercial-only use; repository license file is Apache-2.0; source videos follow ActivityNet terms. Official source.
Charades [74]	AllenAI License for Non-Commercial Use. Official source.
CharadesEgo [77]	AllenAI License for Non-Commercial Use. Official source.
CLEVRER [80]	No standard license stated on the official project page or repository; treated as research-use public benchmark data. Official source.
DiDeMo [81]	BSD 2-Clause for annotations/code; source videos carry per-video Creative Commons licenses from Flickr/YFCC100M. Official source.
Ego4D [83]	Ego4D data-use agreement; redistribution and third-party access are restricted. Official source.
EgoClip [85]	MIT for EgoVLP code; EgoClip data derive from Ego4D and follow Ego4D terms. Official source.
EgoProceL [87]	MIT for code; videos come from multiple source datasets with separate terms. Official source.
EgoQA [90]	MIT for QA annotations; videos follow Ego4D terms. Official source.
HiREST [93]	MIT. Official source.
LongVILA [26]	No standard dataset license stated; project code is Apache-2.0, weights are CC BY-NC-SA 4.0, and source dataset/model/service terms apply. Official source.
LSMDC [75]	Access-controlled research dataset; no standard public license stated; clips derive from copyrighted movies. Official source.
LLaVA-Video-178K [69]	Apache-2.0 metadata with academic research and education terms on the dataset card; generator and source-video terms apply. Official source.
Mementos [78]	No explicit dataset license stated; mixed source terms from NExT-QA, Open X-Embodiment, and online image sequences apply. Official source.
Molmo2 [59]	ODC-BY for AskModelAnything, VideoCapQA, and VideoSubtitleQA; YouTube/source and generator terms apply. Official source.
NExT-QA [55]	MIT for code and annotations; source-video terms apply. Official source.
OneThinker [84]	Apache-2.0; component dataset terms may be stricter. Official source.
Perception Test [56]	Apache-2.0 for software; CC BY 4.0 for other released materials. Official source.
QuerYD [88]	No standard dataset license stated; YouTube and YouDescribe source terms apply. Official source.
SQA3D [94]	CC BY 4.0 for annotations and Apache-2.0 for code; ScanNet Terms of Use restrict scene data to non-commercial research and education. Official source.
ShareGPT4Video [91]	CC BY-NC 4.0. Official source.
STAR [96]	Apache-2.0 for benchmark code/annotations; Charades and Action Genome source terms apply to media and supporting annotations. Official source.
TextVR [73]	MIT for repository materials; no separate standard raw-video license stated. Official source.
TGIF-QA [76]	Academic-use-only terms in the official repository. Official source.
TVQA [79]	No standard public license stated; research/custom access terms apply to TV-show clips and annotations. Official source.
Video-ChatGPT [2]	CC BY-NC-SA 4.0 for VideoInstruct100K and related data. Official source.
Video-R1 [82]	Apache-2.0 metadata; component-source terms from CLEVRER, LLaVA-Video-178K, NExT-QA, Perception Test, and STAR apply. Official source.
VideoChat2 [54]	MIT for MVBench/VideoChat2 annotations; gated and source-video terms apply to videos. Official source.
VideoEspresso [86]	Apache-2.0 metadata for the test card; no clear standard license found for training-video cards; source-media terms apply. Official source.

Source	License or Terms
VideoGPT+ [89]	No standard license stated on the data cards; Video-ChatGPT-lineage and source-video terms apply. Official source.
Vript [92]	Custom academic-use-only terms; no distribution of dataset contents; source-platform and generator terms apply. Official source.
VTG-IT [95]	Apache-2.0; source-video terms may apply. Official source.
YouCook2 [97]	MIT for annotations; YouTube/source terms apply to raw videos. Official source.

Table 12: Licenses and use terms for evaluation benchmarks.

Benchmark	License or Terms
MLVU [10]	CC BY-NC-SA 4.0; research-only and non-commercial terms. Official source.
LongVideoBench [49]	CC BY-NC-SA 4.0; non-commercial use only; linked videos remain owned by source creators. Official source.
LVBench [50]	CC BY-NC-SA 4.0; academic research only; source-video copyrights remain with original owners. Official source.
TempCompass [51]	CC BY-NC 4.0; academic research only. Official source.
MotionBench [52]	CC BY-NC-SA 4.0; academic research only; source-video copyrights remain with original owners. Official source.
TOMATO [53]	CC BY-SA 4.0. Official source.
Video-MME [9]	Custom academic-research-only terms; commercial use and redistribution/modification without approval are prohibited. Official source.
MVBench [54]	MIT for benchmark metadata; component video dataset terms apply, including NTU RGB+D download terms. Official source.
NExT-QA [55]	MIT for code and annotations; source-video terms apply. Official source.
Perception Test [56]	Apache-2.0 for software; CC BY 4.0 for other released materials. Official source.
EgoSchema [57]	Ego4D access/license flow; platform-specific terms may also apply. Official source.

Model and software assets. We use Qwen3-VL-8B as the base model under its public Apache-2.0 release terms [48]. We use `lmms-eval` for evaluation; its main harness follows the MIT license, and the added multimodal task and model code follows Apache-2.0 [58]. We cite YaRN as the context-extrapolation method used for the all-RGB ablation [71]; no YaRN dataset is used.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and Introduction state the claim of predictive visual code, the AdaCodec components, the evaluation setting, and the measured gains in accuracy, token use, and latency. These claims are supported by Sections 3 and 4, with limits stated in the Conclusion.

   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: The Conclusion includes a Limitations paragraph covering fixed input resolution, a uniform per-P-frame token budget, and the lack of streaming evaluation.

   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: The paper does not present theoretical results. The equations define the predictive visual code and the two training losses rather than stating theorems or lemmas.

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: The paper specifies the method, model adaptation, public training sources, split handling, hyperparameters, benchmark splits, evaluation toolkit, and latency protocol in Sections 3–4 and Appendices B–E. We further commit to releasing source code and model checkpoints.

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: A public repository is available at `https://github.com/HaowenHou/AdaCodec`, and we are preparing to release the repository. Section 3, Appendix B, and Appendix C provide the architecture, public training-data sources, and hyperparameters needed to reproduce the main results.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 4.1 specifies the base model, fairness protocol, visual-token budgets, FPS settings, benchmark splits, metric, and evaluation toolkit. Appendix C lists public training sources, split handling, optimizer settings, learning rates, batch sizes, and step counts.

   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: Following the prevailing protocol of the open-source video-MLLM baselines we compare against (Qwen3-VL, LLaVA-Video, Molmo2, Eagle2.5, etc.), we report single-seed accuracy on each benchmark; rerunning Stage 2 across multiple seeds for all eleven benchmarks would multiply the reported compute by an order of magnitude. Robustness is supported instead by consistent gains across eleven benchmarks (Table 2), four token budgets (Figure 3), and the five-axis ablation sweep (Table 10).

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: Appendix C reports GPU type and count, wall-clock time, the latency-measurement hardware.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The study uses public datasets and benchmarks and does not collect private data or run human-subject experiments. We have reviewed the NeurIPS Code of Ethics and identify no deviation.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Appendix H discusses the positive impact of lower-cost long-video analysis and the negative risk.

    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: AdaCodec is a visual-tokenization interface trained on top of the public Qwen3-VL-8B with public video-instruction data; it introduces no new generative model, scraped dataset, or other high-misuse-risk artifact, so model-gating or content-filter safeguards are not applicable.

    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: Every external asset is cited at first use: Qwen3-VL-8B in Section 4.1, the lmms-eval evaluation toolkit in Section 4.1, YaRN context extrapolation in Section 4.6, and the training-data source families in Appendix C. Appendix I lists the licenses or source-specific use terms for all training sources, evaluation benchmarks, and external model/software assets. We use each asset for academic research under its public-release terms and do not redistribute raw videos, images, audio, 3D scans, or repackaged datasets.

    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 do not introduce a new dataset or benchmark.

    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 conduct crowdsourcing experiments or new research with human subjects. It uses existing public video datasets and benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [N/A]

    Justification: The paper does not conduct new research with human subjects and therefore does not require participant instructions, risk disclosure, compensation, or IRB approval for a new study.

    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: [Yes]

    Justification: The core method is a video MLLM interface built on Qwen3-VL-8B, and the paper describes how AdaCodec inserts I-frame and P-frame tokens into the MLLM. Sections 3 and Appendix B describe the visual-token interface, Qwen3-VL adaptation, and language-model tuning in Stage 2.

    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.