# LiteFrame: Efficient Vision Encoders Unlock Frame Scaling in Video LLMs

**Jihwan Kim**[1,2], **Nikhil Parthasarathy**[1], **Danfeng Qin**[1], **Junhwa Hur**[1], **Deqing Sun**[1], **Bohyung Han**[1,2], **Ming-Hsuan Yang**[1] **and Boqing Gong**[1]
[1]Google DeepMind, [2]Seoul National University

**The fundamental challenge in scaling Video Large Language Models (Video LLMs) to long-form video lies in managing the explosion of visual-token context length. Existing strategies predominantly focus on "post-hoc" token reduction—reducing visual tokens after feature extraction to alleviate the LLM's computational overhead. While these methods effectively reduce the number of visual tokens, we observe that the primary latency bottleneck then shifts from the LLM to the expensive per-frame processing of the vision encoder. To address this, we introduce *LiteFrame*, a strong, yet highly efficient video encoder backbone for Video LLMs. To train LiteFrame, we propose Compressed Token Distillation (CTD), a novel training framework that teaches a compact student vision encoder to directly predict information-dense, spatio-temporally compressed representations produced by a large teacher vision model, effectively bypassing redundant computation. When coupled with further Language Model Adaptation (LMA), this approach results in a new latency-accuracy Pareto frontier—compared with InternVL3-8B, LiteFrame provides a 35% reduction in end-to-end latency while processing 8× more frames *and* improves average video understanding accuracy across multiple benchmarks. Our results demonstrate a new potential path to unlocking longer-form video understanding under fixed compute budgets.**

**Project Page: jjihwan.github.io/projects/LiteFrame**

## 1. Introduction

Modern Multimodal LLMs (MLLMs) (Li et al., 2024a; Pichai et al., 2025; Qwen Team, 2025; Wang et al., 2025a; Zhu et al., 2025) have achieved remarkable progress in recent years in video understanding, parsing complex temporal dynamics for captioning (Fang et al., 2024), question answering (Fu et al., 2025; Zhou et al., 2025), and reasoning (Fu et al., 2026). Despite these strong capabilities, there remains a fundamental scaling problem when handling long-form video within the current paradigm—the computational cost of processing spatio-temporal video data grows prohibitively with increasing frame counts. To understand why this is, we first note that these models all typically follow a very similar multi-stage architecture consisting of an image encoder (e.g. Vision Transformer; Dosovitskiy et al., 2021) that processes a video frame by frame, an alignment projector, and an LLM that reasons over the interleaved visual and text tokens. Therefore, with each additional input frame, the computational cost increases due to processing demands in *both* the vision encoder and the LLM.

Existing works that try to alleviate this computational burden have largely focused on the LLM, attributing the primary bottleneck to the quadratic complexity of self-attention over an increasing number of visual tokens. Consequently, the dominant solution has been an "extract-and-reduce" paradigm, which maintains the frozen image encoder for frame-level feature extraction, and leverages post-hoc token reduction strategies (Figure 1 (b))—either spatially (Shang et al., 2025; Wang et al., 2025b; Yang et al., 2025b), spatio-temporally (Huang et al., 2025; Shao et al., 2025; Shen et al., 2025; Tao et al., 2025), or via query-guided pruning (Chen et al., 2024b; Shen et al., 2024; Xing et al., 2025; Yang et al., 2025a)—before feeding them to the LLM.

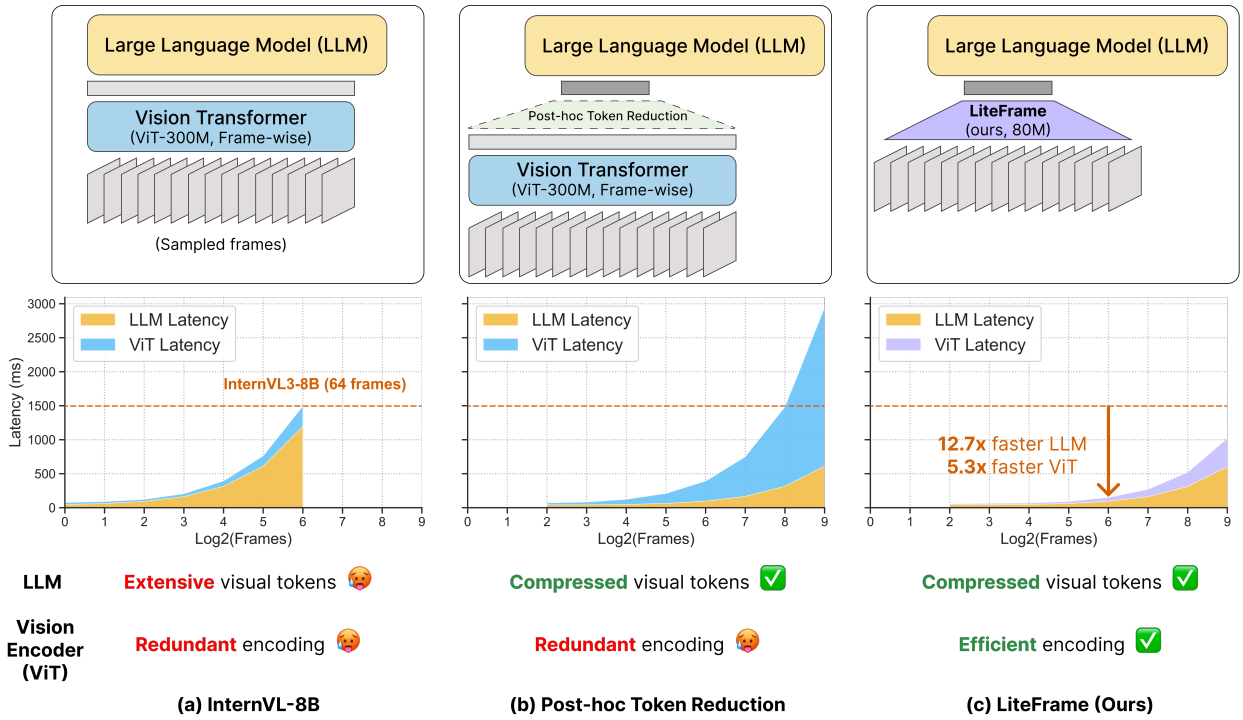



Figure 1 | We propose **LiteFrame**, a lightweight video encoder that reduces inefficiencies from both the LLM and the ViT. (a) Standard Video LLMs (Zhu et al., 2025) are bottlenecked by the LLM's quadratic complexity, strictly limiting context length to ~64 frames. (b) Post-hoc reduction alleviates LLM's burden, enabling more frames, yet ironically shifts the bottleneck to the ViT, causing latency to explode. (c) Our approach (LiteFrame) resolves both inefficiencies, enabling 12.7× faster LLM prefilling and 5.3× faster ViT encoding at 64 frames compared to the InternVL3-8B baseline.

We show that this class of methods ignores the cost of the per-frame feature extraction, which while seemingly lightweight, becomes cumulatively expensive. Specifically, our preliminary analysis in Section 3 reveals that while aggressive post-hoc token reduction (e.g., 16×) alleviates the LLM overhead, as the LLM compute decreases, the computational burden of the visual encoder begins to dominate. This remaining bottleneck prohibitively sets a floor for the achievable end-to-end inference efficiency. As illustrated in Figure 1 (b), once post-hoc token reduction is effectively applied, the vision encoder's latency becomes the new bottleneck as frame counts increase. Hence, unlocking the next generation of efficient MLLMs for long-video understanding requires a holistic approach that simultaneously optimizes both visual encoding and language model efficiency.

To this end, we introduce LiteFrame, a lightweight, efficient video encoder designed to reduce per-frame compute with a minimal decrease in video understanding accuracy. To achieve this, we propose Compressed Token Distillation (CTD), a novel strategy for training a compute-efficient, token-compressive encoder from a pretrained teacher image encoder. Specifically, CTD directly aligns the student with an information-dense, spatio-temporally compressed teacher output. Furthermore, we design the student encoder architecture to explicitly reduce spatio-temporal redundancies across frames.

When coupled with a lightweight Language Model Adaptation (LMA) stage (adapting the new encoder with the LLM), LiteFrame allows Video LLMs to achieve a new latency-accuracy Pareto frontier for video understanding. As illustrated in Figure 2, our model delivers superior accuracy with remarkably low latency when compared to existing baselines. Specifically, LiteFrame significantly outperforms the InternVL3-8B by processing 8× more frames with a 35% reduction in end-to-end latency, while using only 87M parameters (vs. 304M for the teacher).

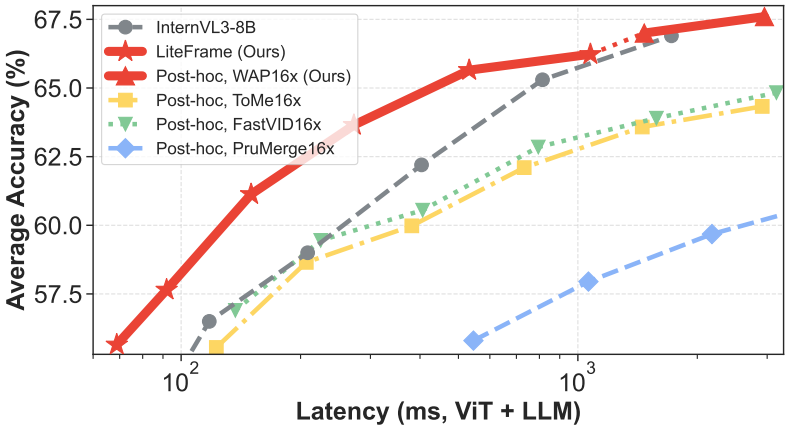



Figure 2 | **The pareto frontier of video understanding efficiency.** We illustrate the trade-off between average accuracy on four video benchmarks (Video-MME w/ and w/o subtitles, MLVU, and LongVideoBench) and end-to-end latency including vision encoding and LLM prefilling. Our proposed post-hoc primitive, Weighted Average Pooling (WAP, red triangles), and our efficient video encoder, LiteFrame (red stars), push the efficiency Pareto frontier, achieving superior accuracy compared to the teacher model (InternVL3, black dashed). Existing post-hoc methods (color dashed) fail to improve the trade-off as they neglect the encoder latency bottleneck. Note that the x-axis (latency) is log-scaled.

To summarize our contributions:

- We identify a critical scaling blindspot in current efficient Video LLM paradigms: while post-hoc token reduction effectively alleviates LLM computational costs, the vision encoder becomes the new latency bottleneck, preventing further efficient scaling to long videos.
- We propose LiteFrame, an efficient video encoder that resolves this bottleneck shift by integrating token compression directly within a lightweight visual backbone.
- We introduce Compressed Token Distillation (CTD), a novel training framework for maximizing the transfer of spatio-temporally dense information from a teacher to a compact student.
- Extensive experiments demonstrate that LiteFrame redefines the performance-latency trade-off. Our approach achieves a 1.53× acceleration in end-to-end inference compared to the InternVL3-8B teacher, while processing 8× more frames and outperforming the baselines on multiple video understanding tasks.

## 2. Related work

**Post-hoc token reduction.** The predominant method for making MLLMs efficient is the "extract-and-reduce" paradigm: applying post-hoc token reduction after heavy, pre-trained vision encoders extract dense features, aiming to reduce the cost attributed to the LLM's quadratic complexity. Early approaches focused on spatial redundancy within individual images, using adaptive selection or merging (Shang et al., 2025; Yang et al., 2025b). More recent efforts extend this to the temporal dimension for video inputs via dynamic pruning or holistic merging (Huang et al., 2025; Shao et al., 2025; Shen et al., 2025; Tao et al., 2025).

While these post-hoc methods reduce the computational burden on the LLM, they remain inefficient for long-form video understanding (hundreds or thousands of frames) because they miss a critical

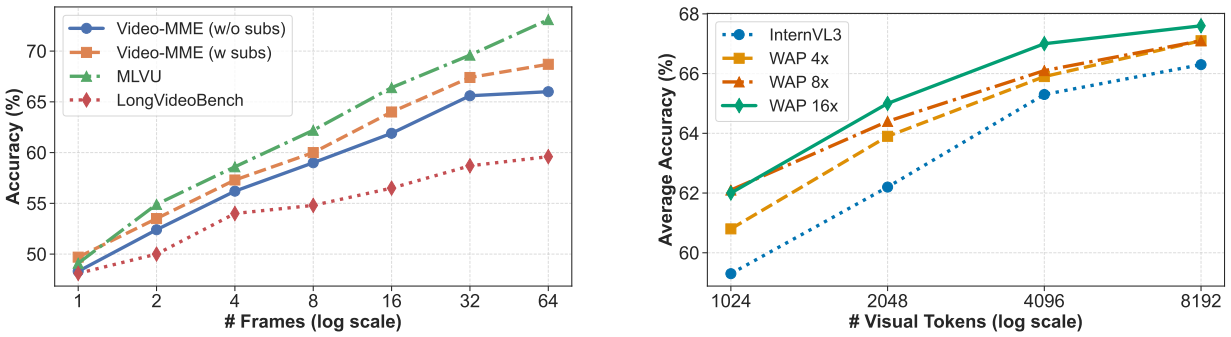



Figure 3 | **Impact of frame scaling and test-time compression. (Left)** Accuracy of Video LLMs on key benchmarks (Video-MME, MLVU, and LongVideoBench) scales logarithmically with input frames, highlighting the strict frame budget as a primary bottleneck. **(Right)** Under a fixed token budget, aggressive Weighted Average Pooling (up to 16×, green) enables InternVL3-8B to process significantly more frames at test time, maximizing accuracy gains.

scaling bottleneck. Because these methods rely on a heavy, frozen encoder to process every frame prior to compression, the latency bottleneck shifts from the LLM to vision encoding.

**Efficient vision encoders for MLLMs.** A parallel line of work aims to reduce the cost of visual encoding. MobileNet-v5 (Google, 2025; Qin et al., 2024) achieves high inference throughput on edge devices through aggressive architectural optimization. FastVLM (Vasu et al., 2025) introduces FastViTHD, a hybrid encoder that combines convolutional efficiency with transformer-based global modeling to better balance latency and input resolution. However, these methods focus on image-centric architectures that are highly effective for spatial encoding but do not explicitly exploit the strong temporal redundancy across frames.

In the video domain, Video-Panda (Yi et al., 2025) proposes an encoder-free paradigm, using a Spatio-Temporal Alignment Block to bypass a heavy visual backbone. This removes the visual backbone bottleneck but exposes the downstream LLM to dense, uncompressed token streams, shifting the bottleneck back to the LLM. More recently, AutoGaze (Shi et al., 2026) trains a lightweight module to pre-filter visual tokens before they are processed by the ViT. While it successfully reduces tokens, this method introduces additional latency overhead, including the cost of a heavy VideoViT and autoregressive decoding within the reduction module, ultimately degrading the latency-accuracy trade-off when evaluated on long videos.

## 3. Revisiting Post-Hoc Reduction

In this section, we motivate the core design choices for LiteFrame (Section 4). We revisit post-hoc token reduction to establish two critical design premises for our main approach: (1) **W**eighted **A**verage **P**ooling (WAP) serves as a simple and effective compression primitive compared to existing complex token merging or pruning strategies (Section 3.1), and (2) aggressive compression (up to 16×) is desirable because it trades off favorably with an increase in the number of frames that are processed at test time (Section 3.2). Moreover, we demonstrate that post-hoc reduction fails to reduce the base computational cost of the encoder, prompting us to instead "internalize" the token compression via a customized compact student network architecture.

Table 1 | **Evaluation of post-hoc token reduction strategies.** We evaluate various token reduction methods applied to InternVL3-8B with a fixed compression ratio of 16× (64 frames) on three video benchmarks (Fu et al., 2025; Wu et al., 2024; Zhou et al., 2025). The proposed **W**eighted **A**verage **P**ooling (WAP) achieves the highest average accuracy.

| Method | Video-MME (w/o subs) | Video-MME (w subs) | MLVU | LongVideo Bench | Avg. |
|---|---|---|---|---|---|
| InternVL3-8B (No comp.) | 66.0 | 68.7 | 73.1 | 59.6 | 66.9 |
| Average Pooling | 59.7 | 64.1 | 62.3 | **54.7** | 60.2 |
| Max Pooling | 59.7 | 63.9 | 62.0 | 54.2 | 60.0 |
| Subsampling | 60.4 | 64.6 | 65.1 | 54.5 | 61.2 |
| ToMe (Bolya et al., 2023) | 58.7 | 62.3 | 64.7 | 54.2 | 60.0 |
| PruMerge (Shang et al., 2025) | 60.3 | 63.2 | 64.6 | 50.6 | 59.7 |
| FastVID (Shen et al., 2025) | 59.3 | 63.4 | 65.1 | 54.4 | 60.6 |
| **WAP** (Ours) | **61.0** | **65.7** | **67.4** | 54.0 | **62.0** |

### 3.1. Spatio-temporal Weighted Average Pooling (WAP)

To reduce the number of visual tokens, existing literature often relies on attention-based pruning (Shang et al., 2025; Shen et al., 2025) or token merging via bipartite soft-matching (Bolya et al., 2023; Wang et al., 2025a). Since the attention and matching scores are mainly determined by the tokens' content rather than their positions, these methods disrupt the continuous spatio-temporal structure required for coherent video understanding. Recent findings (Liao et al., 2025; Wen et al., 2025) highlight this drawback, suggesting that simple average pooling or image downsampling outperforms complex reduction strategies. Extending this intuition, we propose Weighted Average Pooling (WAP), a primitive that harmonizes the structural regularity of pooling with attention-based weighting.

Let $\mathbf{X} \in \mathbb{R}^{T\times H\times W\times C}$ be the input feature tensor. We partition $\mathbf{X}$ into non-overlapping spatio-temporal blocks $\Omega_{u,v,s}$ to match a target compressed resolution $(t, h, w)$. The compressed token $\mathbf{Y}_{u,v,s}$, derived by WAP, is computed as:

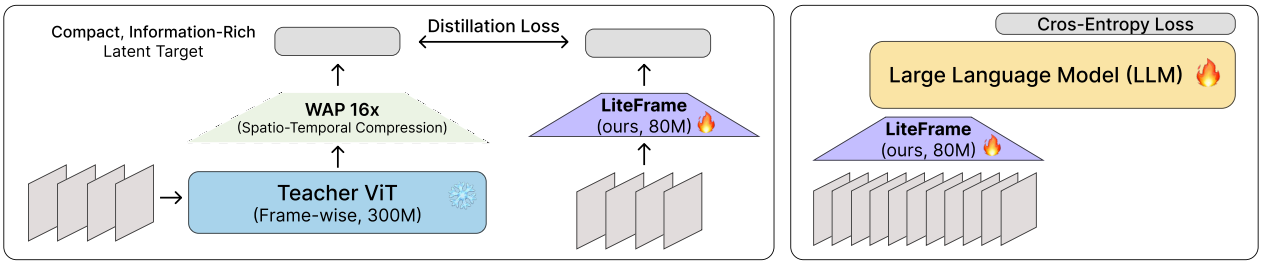
$$\mathbf{Y}_{u,v,s} = \sum_{(\tau,i,j)\in\Omega_{u,v,s}} \text{softmax}\left(\frac{\mathbf{x}_{\tau,\text{cls}}^{\top}\mathbf{x}_{\tau,i,j}}{\sqrt{C}}\right)\mathbf{x}_{\tau,i,j}, \tag{1}$$

where the softmax is computed within each block $\Omega_{u,v,s}$, $\mathbf{x}_{\tau,i,j} = \mathbf{X}[\tau, i, j, :]$, and $\mathbf{x}_{\tau,\text{cls}}$ is the class token of the $\tau^{\text{th}}$ frame. This operation effectively retains high-activation features while reducing the token count by a factor of $r = \frac{THW}{thw}$.

Empirically, Table 1 demonstrates that WAP significantly outperforms both standard pooling baselines (Average/Max Pooling, Subsampling) and state-of-the-art, more complex token reduction methods (Bolya et al., 2023; Shang et al., 2025; Shen et al., 2025) under a 16× (4× spatial and 4× temporal) compression ratio. Appendix A provides the evaluation setups for Table 1. While modern Video LLMs (Li et al., 2024a, 2025; Wang et al., 2025a) typically rely on simple pooling or ToMe (Bolya et al., 2023), we instead use WAP as a compression operator, not merely for preprocessing, but to *generate supervision targets for our distillation framework* in Section 4.2.

### 3.2. Frame-Count Bottleneck

The performance of Video LLMs depends critically on the number of input frames. As shown in Figure 3 (left), accuracy on the long video benchmarks, such as Video-MME (Fu et al., 2025), MLVU (Zhou et al., 2025), and LongVideoBench (Wu et al., 2024), exhibits logarithmic growth with respect to the input frame count. However, conventional models like InternVL3 are practically capped at ~64 input




(a) Compressed Token Distillation (b) Language Model Adaptation

Figure 4 | **Overview of our training framework for LiteFrame.** (a) Compressed Token Distillation employs WAP to compress the teacher's dense features into a compact, information-rich latent space, which serves as the prediction target for the student. (b) Language Model Adaptation fine-tunes the LLM and encoder on (video, text) pairs to further optimize the student's latent space and adapt LLM to the extended temporal context.

frames due to both the context length limits of the LLM and the large number of tokens per frame (e.g., 256). We argue that this dense per-frame tokenization is excessive, and that spatio-temporal token compression can overcome these bottlenecks.

To validate this, we compare a baseline without compression against three WAP variants with compression ratios of 4×, 8×, and 16× under a fixed visual token budget. Crucially, WAP enables high compression ratios, thereby allowing the model to process proportionally more frames. As seen in Figure 3 (right), all WAP variants outperform the baseline, with 16× compression (and thus 16× more frames) achieving the best results. These results demonstrate that aggressive compression effectively trades redundant tokens for richer temporal context. Appendix A describes the detailed experimental setup for Figure 3.

**Scaling paradox.** While post-hoc reduction effectively reduces the number of visual tokens fed to LLMs, the computational cost of the vision encoder remains the same. Therefore, as we scale the frame counts needed for high performance on long-form video understanding, the vision encoder latency explodes and becomes the new bottleneck (Figure 1 (b)). This insight drives the design of LiteFrame—we focus on achieving the aforementioned aggressive compression directly within the vision encoder, rather than as a post-hoc stage.

## 4. LiteFrame: Internalizing Spatio-Temporal Token Compression

We introduce LiteFrame, a video encoder designed to resolve the dual bottleneck of Video LLMs: the quadratic complexity of the LLM *and* the exploding latency of the vision encoder when scaling to high input frame counts. Unlike prior works that compress tokens post-hoc, we propose a lightweight encoder that internally compresses the tokens. To achieve this, our approach rests on two key ideas. First, we design a spatio-temporal encoder architecture that minimizes latency and FLOPs (Section 4.1). Second, we propose a novel distillation strategy where the student learns to directly predict the spatio-temporally compressed representations of a powerful teacher (Section 4.2).

### 4.1. Architecture: Spatio-temporal Token Compressive Encoding

We first design a lightweight student encoder to be significantly more compact than the corresponding teacher (87M vs. 304M parameters in our main experiments). We use a 12-layer, 768D ViT-Base (Dosovitskiy et al., 2021) backbone for the student while the teacher is a 24-layer, 1024D ViT-Large.

Moreover, we employ a low-latency video encoder backbone—instead of the standard image encoder—designed to progressively reduce spatio-temporal redundancies across frames. Specifically, to enable spatio-temporal encoding, we interleave standard spatial attention layers with lightweight, depth-wise (DW) 1D temporal convolution layers. To further reduce computation, we integrate DW strided convolution layers at strategic intervals, which gradually downsample the feature maps in both spatial and temporal dimensions as the network deepens. By progressively reducing the number of tokens, we ensure that the computational cost of the deeper layers is substantially lower than that of standard frame-wise image encoders. Section 5.1 describes the architecture in detail.

Table 2 | **Efficiency comparison for temporal modeling.** Compared to temporal attention (TempAttn), spatio-temporal full attention (SpatioTempAttn), or vanilla temporal convolutions (TempConv), Depth-Wise Temporal Convolutions (DWTempConv) achieves the lowest latency and FLOPs while introducing negligible parameter overhead (<1M). Latency and FLOPs are measured using 256 input frames.

| Architecture | Latency (ms) | TFLOPs | # Params (M) |
|---|---|---|---|
| ViT-Large-24L (Teacher) | 1043.33 | 158.80 | 304.01 |
| ViT-Base-12L (No comp.) | 338.01 | 44.81 | 86.31 |
| TempAttn | 348.29 | 32.77 | 143.83 |
| SpatioTempAttn | 204.35 | **17.92** | **87.15** |
| TempConv | 202.08 | 22.44 | 109.54 |
| **DWTempConv (Ours)** | **174.84** | **17.92** | **87.15** |

As demonstrated in Table 2, DW temporal convolutions allow the model to capture temporal dynamics with significantly lower latency and FLOPs, compared to other widely-used alternatives, such as interleaving temporal attention blocks, basic temporal convolution, or replacing the spatial attention with full spatio-temporal attention. Moreover, Table 5 demonstrates that DW temporal convolution consistently yields superior accuracy over full spatio-temporal attention across benchmarks. Appendix A details how the latency is measured.

### 4.2. Compressed Token Distillation (CTD)

Training a lightweight student to match the semantic richness of a large teacher while simultaneously reducing the token count is non-trivial. Standard distillation forces the student to learn redundant spatial details that it cannot effectively represent.

To address this, we propose Compressed Token Distillation (CTD), where we treat Weighted Average Pooling (WAP) as a strong post-hoc compression primitive (as seen in Section 3.1) and use it to generate supervision targets. As a result, rather than mimicking the teacher's dense output, the student is trained to predict the compressed representation produced by the teacher under WAP.

Formally, let $T(x) = Z_T \in \mathbb{R}^{N \times D}$ denote the teacher's dense features and $S_\theta(x) = Z_S \in \mathbb{R}^{(N/r) \times D}$ denote the student's output, where $r$ is the target compression ratio (e.g., 16×). We define a projection operator $\mathcal{P}(\cdot)$ based on WAP that aggregates dense tokens into compressed representations. The student is optimized to minimize the MSE loss between its output and the teacher's compressed representations:

$$\mathcal{L}_{\text{CTD}}(\theta) = \|S_\theta(x) - \mathcal{P}(T(x))\|_2^2. \tag{2}$$

By effectively transferring the attention-based weighting mechanism of WAP into the static parameters of the student network, the student can output the salient spatio-temporal information without the runtime overhead of computing attention over redundant patches.

### 4.3. Language Model Adaptation (LMA)

Although CTD effectively teaches the student to predict salient features, the resulting compressed latent space can be suboptimal for the LLM. Therefore, to bridge the modality gap and further optimize the student's latent space, we add a minimal Language Model Adaptation (LMA) stage.

Table 3 | **Latency and accuracy trade-off.** We evaluate LiteFrame as an efficient video encoder, and FastVID (Shen et al., 2025), one of the state-of-the-art post-hoc methods, both applied to InternVL3-8B under comparable total latency budgets. **LiteFrame (Ours)** denotes our final performance incorporating Compressed Token Distillation and Language Model Adaptation. By internalizing token compression within the vision backbone, LiteFrame processes 8× more frames than the baseline while achieving up to a 35% reduction in end-to-end latency and superior accuracy. In contrast, the post-hoc method is bottlenecked by the heavy original vision encoder.

| Method | Frames | Tokens/ Frame | Vision Params | Latency (ms) ↓ | | | Accuracy (%) ↑ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Vision | LLM | Total | V-MME† | V-MME‡ | MLVU | LongVideo | Avg |
| InternVL3-8B | 8 | 256 | 304M | 40.0 | 167.3 | 208.4 | 59.0 | 60.0 | 62.2 | **54.8** | 59.0 |
| +FastVID | 32 (4×) | 16 | 304M | 161.7 | 63.0 | 224.8 (+7.9%) | 58.7 | 62.5 | 64.0 | 52.6 | 59.5 (+0.5) |
| **+LiteFrame (Ours)** | **64** (8×) | 16 | **87M** | 54.8 | 94.9 | **150.1** (-28.0%) | **61.0** | **64.2** | **65.7** | 53.6 | **61.1** (+2.1) |
| InternVL3-8B | 16 | 256 | 304M | 74.0 | 329.3 | 403.6 | 61.9 | 64.0 | 66.4 | 56.5 | 62.2 |
| +FastVID | 64 (4×) | 16 | 304M | 310.6 | 95.4 | 406.2 (+0.6%) | 59.3 | 63.4 | 65.1 | 52.6 | 59.5 (-2.7) |
| **+LiteFrame (Ours)** | **128** (8×) | 16 | **87M** | 105.3 | 166.6 | **272.6** (-32.5%) | **63.9** | **66.8** | **66.7** | **57.2** | **63.7** (+1.5) |
| InternVL3-8B | 32 | 256 | 304M | 144.5 | 669.8 | 814.5 | **65.6** | 67.4 | 69.6 | **58.7** | 65.3 |
| +FastVID | 128 (4×) | 16 | 304M | 625.8 | 168.9 | 794.9 (-2.4%) | 60.4 | 66.1 | 69.3 | 55.6 | 62.9 (-2.4) |
| **+LiteFrame (Ours)** | **256** (8×) | 16 | **87M** | 204.0 | 327.4 | **532.3** (-34.6%) | 65.1 | **68.5** | **70.7** | 58.3 | **65.7** (+0.4) |

† without subtitles, ‡ with subtitles. (+)/(-) denotes improvement/degradation relative to Teacher.

We fine-tune the LLM and the encoder with video-text pairs, minimizing the standard cross-entropy loss for text generation conditioned on videos. To ensure training efficiency and preserve the LLM's reasoning capabilities, we employ LoRA (Hu et al., 2022). In addition to aligning the student with the LLM, we also find that this stage helps with *long-context adaptation*, allowing the LLM to handle the extended temporal context (up to 512 frames) enabled by our encoder.

## 5. Experiments

### 5.1. Implementation details

We utilize InternVL3-8B as our primary baseline, leveraging its image encoder, InternViT-300M (304M parameters, 1024 hidden dim), as the teacher model. To measure the average accuracy, we employ four widely used video benchmarks—Video-MME (with and without subtitiles; Fu et al., 2025), MLVU (Zhou et al., 2025), and LongVideoBench (Wu et al., 2024)—as primary evaluation suites.

For the student model, we adopt a significantly more efficient ViT-Base backbone (87M parameters, 768 hidden dimensions). As described in Section 4.1, we interleave depth-wise 1D temporal convolutions after every spatial layer where the temporal dimension is greater than 1. In addition, we integrate depth-wise strided convolution layers after the $4^{\text{th}}$ and $8^{\text{th}}$ blocks, with strides of $[t, h, w] = [2, 2, 2]$ and $[2, 1, 1]$, respectively. Further details regarding training, datasets, and evaluation are provided in Appendix A.

### 5.2. Quantitative analysis

#### *5.2.1. Redefining the Pareto frontier*

We evaluate LiteFrame by analyzing the trade-off between video understanding accuracy across multiple benchmarks (Fu et al., 2025; Wu et al., 2024; Zhou et al., 2025) and end-to-end inference latency under varying frame counts. As detailed in Table 3, our approach establishes a new Pareto frontier, surpassing the baselines in both latency and accuracy. Applied to InternVL3-8B, LiteFrame reduces total inference latency by up to 35% while improving accuracy by 0.4%p (65.7% vs. 65.3%)

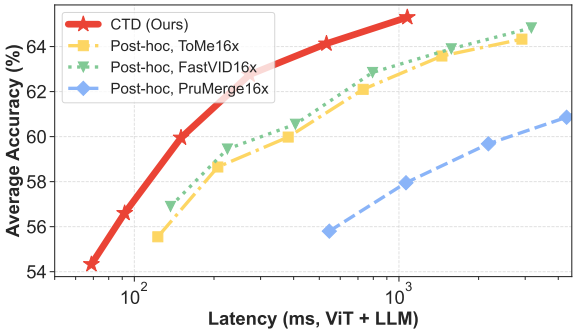



Figure 5 | **Comparison with SOTA post-hoc token compression methods.** Compressed Token Distillation (CTD) achieves superior accuracy compared to the baselines by avoiding the computational floor that limits the efficiency.

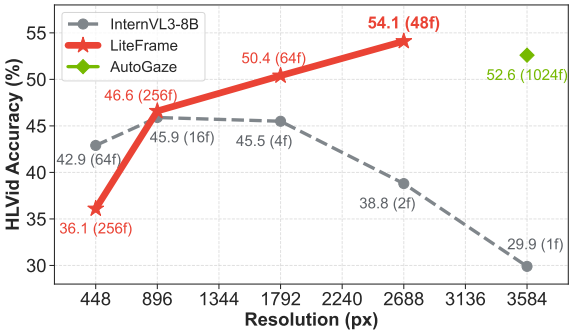



Figure 6 | **Zero-shot scaling in spatial dimension.** We measure maximum accuracy across spatial resolutions, utilizing the maximum context length of the LLMs. LiteFrame achieves a state-of-the-art score of **54.1** on HLVid.

on average. Notably, the accuracy gap widens by 2.1%p (61.1% vs. 59.0%), when we restrict the total latency budget (8 frames for InternVL3-8B). Moreover, as shown in Figure 2, LiteFrame significantly outperforms state-of-the-art post-hoc compression methods such as FastVID (Chen et al., 2024b), PruMerge (Shang et al., 2025), and ToMe (Bolya et al., 2023). The results demonstrate that LiteFrame effectively trades spatio-temporal redundancy for significantly richer temporal context, allowing the model to process 8× more frames within a fixed compute budget.

### *5.2.2. Comparison with post-hoc methods*

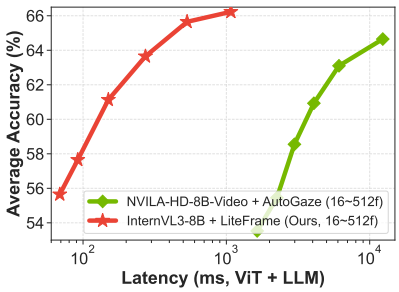
To ensure a fair comparison with training-free post-hoc baselines, we evaluate LiteFrame utilizing only CTD without subsequent LMA, keeping the LLM entirely frozen. As illustrated in Figure 5, simply swapping the original heavy ViT with LiteFrame surpasses all post-hoc methods—including ToMe (Bolya et al., 2023), LLaVA-PruMerge (Shang et al., 2025), and FastVID (Shen et al., 2025)—in both efficiency and accuracy, by effectively distilling the WAP primitive into the student model. In contrast, as expected, existing post-hoc methods are severely bottlenecked by the inevitable computational cost incurred prior to the compression, causing ViT latency to explode when frame counts increase.

### *5.2.3. Zero-shot spatial resolution scaling*

Beyond scaling temporal resolution for long-form video, the inherent token efficiency of LiteFrame naturally facilitates scaling in the spatial dimension, particularly for tasks requiring fine-grained visual perception. To highlight this, we implement a zero-shot tiling strategy that splits high-resolution frames into 448px sub-tiled clips, which are then processed independently by LiteFrame. We evaluate this on the HLVid benchmark (Shi et al., 2026), which requires high-fidelity spatial understanding across video frames (see Figure 6). Notably, InternVL3-8B exhibits a performance stagnation as input resolution increases—we attribute this to the LLM's fixed context length that forces a sacrifice in temporal resolution as token counts grow due to the increased spatial resolution. In contrast, the token efficiency of LiteFrame allows the model to maintain a better balance of spatial and temporal resolution, achieving a state-of-the-art score of 54.1 at 2688px with 48 frames. Remarkably, this surpasses the previous best method, AutoGaze (Shi et al., 2026) (52.6), despite AutoGaze requiring much higher resolutions (3584px and 1024 frames). Moreover, LiteFrame *achieves these results without any high-resolution training*, demonstrating its strong generalizability to higher resolutions.

Table 4 | **Comparison with efficient vision encoders for MLLMs.** We compare LiteFrame against state-of-the-art efficient vision encoders for VLMs. Our method shows significantly lower latency for both vision encoding and LLM prefilling, while achieving the best accuracy. "Total Tok." denotes the aggregate number of visual tokens fed to the LLM, and "Acc." denotes the average accuracy.

| Method | Vision Params | Tok./Frame | Frames | Total Tok. | Vis. (ms) | LLM (ms) | Total (ms) | Acc. (%) |
|---|---|---|---|---|---|---|---|---|
| FastVLM | 125M | 49 | 32 | 1568 | 98.3 | 132.9 | 231.5 | 47.6 |
| VideoPanda | 45M | 272 | 32 | 8704 | 36.5 | 345.7 | 382.4 | 49.2 |
| **LiteFrame** (Ours) | 87M | **16** | 32 | **512** | **30.1** | **61.5** | **91.9** | **58.0** |




| Method | Frames | Latency (ms) | | | | Acc. (%) |
|---|---|---|---|---|---|---|
| | | AutoGaze | ViT | LLM | Total | |
| NVILA-8B-Video | 32 | - | 451.0 | 329.9 | 780.8 | 63.1 |
| +AutoGaze | 256 (8×) | 2961.4 (+2961.4ms) | 2605.6 (+477.8%) | 539.5 (+63.5%) | 6106.5 (+682.1%) | 63.1 (0.00) |
| InternVL3-8B | 32 | - | 144.5 | 669.8 | 814.5 | 65.3 |
| +LiteFrame (Ours) | 256 (8×) | - | 204.0 (+41.2%) | 327.4 (-51.1%) | **532.3** (-34.6%) | **65.7** (+0.4) |

Figure 7 | **Comparison with AutoGaze. (Left)** LiteFrame (red) improves the efficiency frontier with lower latency and higher accuracy compared to AutoGaze (green). **(Right)** When scaling from 32 to 256 frames (8×), LiteFrame balances ViT and LLM scaling, while AutoGaze's pre-reduction module becomes a new bottleneck. "Acc." denotes the average accuracy.

#### *5.2.4. Comparison with efficient vision encoders for MLLMs*

We further benchmark our approach against state-of-the-art efficient vision encoders designed for MLLMs, including FastVLM (Vasu et al., 2025) and VideoPanda (Yi et al., 2025). Based on InternVL3-8B, we fine-tune the LLM via LoRA with the respective frozen visual encoders for a fair comparison. As shown in Table 4, while these baselines achieve impressive parameter efficiency, LiteFrame is 1.2× faster than VideoPanda and 3.3× faster than FastVLM. Additionally, because our encoder is trained to output a highly compact set of tokens, it also significantly lowers the downstream computational cost (and thus latency) of the LLM.

Next, we compare LiteFrame against AutoGaze (Shi et al., 2026), a recent approach that similarly addresses the computational bottlenecks of both the ViT and LLM. We benchmark the latency-accuracy trade-offs when integrating LiteFrame and AutoGaze into their respective baselines. As shown in Figure 7 (left), LiteFrame significantly outperforms AutoGaze, achieving substantially lower total latency while improving in average accuracy. While initially surprising, the detailed breakdown in Figure 7 (right) reveals that the latency gap can be attributed to the AutoGaze pre-reduction auxiliary module that accounts for nearly half of the total inference time (3.0s out of 6.1s). A more detailed comparison and implementation details regarding AutoGaze are provided in Appendix D.

### 5.3. Ablation studies

To evaluate the effectiveness of each component of our approach, we conduct an ablation analysis isolating the contributions of 1) the token-compressive student architecture, 2) depth-wise (DW) temporal convolutions, 3) the Weighted Average Pooling (WAP) objective applied to the teacher's output, and 4) Language Model Adaptation (LMA).

Table 5 | **Ablation study.** We evaluate the impact of the token-compressive student architecture (TokComp.), Depth-Wise Temporal Convolutions (DWConv), the Weighted Average Pooling (WAP) objective, and Language Model Adaptation (LMA). "Acc" denotes the average accuracy. Our approach (bottom row) demonstrates the best trade-off, simultaneously reducing latency while improving accuracy.

| Ablations | TokComp. (Student) | DWConv (Student) | WAP (Teacher) | LMA | Frames | Latency (ms) | Acc (%) |
|---|---|---|---|---|---|---|---|
| InternVL3-8B (Teacher) | – | – | – | – | 16 | 403.6 | 62.2 |
| Distillation (ViT-Base-12L) | × | × | × | × | 16 | 362.9 | 60.3 |
| CTD (SpatioTempAttn) | ✓ | × | ✓ | × | 128 | 102.2 | 61.9 |
| CTD (DWTempConv) | ✓ | ✓ | ✓ | × | 128 | 87.4 | 62.8 |
| RTD | ✓ | ✓ | × | × | 128 | 87.4 | 43.8 |
| RTD + LMA | ✓ | ✓ | × | ✓ | 128 | 87.4 | 61.5 |
| **CTD + LMA (Ours)** | ✓ | ✓ | ✓ | ✓ | 128 | **87.4** | **63.4** |

Simple distillation into a standard ViT-Base-12L backbone without token compression shows marginal latency reduction and degrades accuracy, most likely due to the context limits, which restrict temporal resolution. Incorporating Compressed Token Distillation (CTD) significantly alleviates this bottleneck, however, utilizing full spatio-temporal attention in the student encoder underperforms DW temporal convolutions, highlighting the superior efficiency and efficacy of this form of temporal processing. Moreover, we explore an alternative training objective, Reconstructive Token Distillation (RTD), which replaces our WAP objective with an auto-encoding objective (detailed in Appendix C.1). RTD significantly lags behind CTD, demonstrating that the WAP primitive enables more effective distillation of strong spatio-temporal features into the student for downstream LLM reasoning. Finally, coupling CTD with the additional LMA stage ultimately gives the lowest latency and best accuracy. A more comprehensive ablation analysis is provided in Appendix C.

## 6. Conclusion

In this work, we identify and resolve a critical efficiency bottleneck in current Video LLMs. While post-hoc token reduction strategies effectively reduce the computational cost of the LLM, this leaves the vision encoder as the prohibitive latency bottleneck when scaling to high frame counts. We introduce LiteFrame, a lightweight video encoder that fundamentally addresses the full end-to-end efficiency problem by internalizing spatio-temporal compression in a compact encoder, trained with our novel Compressed Token Distillation (CTD) and Language Model Adaptation (LMA) methods. By teaching the student encoder to bypass redundant full-resolution computation and directly predict the information-dense (pooled) tokens of the heavy teacher, our approach redefines the efficiency-accuracy Pareto frontier. Specifically, across multiple long-video benchmarks, we achieve 35% faster end-to-end inference and better accuracy while processing 8× more frames, effectively trading spatio-temporal redundancy for significantly richer temporal context. While much of the community has been focused exclusively on pushing the limits of pure token reduction methods, our results demonstrate the principle that architectural internalization of token compression via distillation unlocks even more scalable, long-form video understanding.

## References

D. Bolya, C.-Y. Fu, X. Dai, P. Zhang, C. Feichtenhofer, and J. Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations (ICLR)*, 2023.

L. Chen, X. Wei, J. Li, X. Dong, P. Zhang, Y. Zang, Z. Chen, H. Duan, B. Lin, Z. Tang, L. Yuan, Y. Qiao, D. Lin, F. Zhao, and J. Wang. ShareGPT4Video: Improving video understanding and generation with better captions. In *European Conference on Computer Vision (ECCV)*, 2024a.

L. Chen, H. Zhao, T. Liu, S. Bai, J. Lin, C. Zhou, and B. Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision (ECCV)*, 2024b.

Z. Chen, W. Wang, Y. Cao, Y. Liu, Z. Gao, E. Cui, J. Zhu, S. Ye, H. Tian, Z. Liu, L. Gu, X. Wang, Q. Li, Y. Ren, Z. Chen, J. Luo, J. Wang, T. Jiang, B. Wang, C. He, B. Shi, X. Zhang, H. Lv, Y. Wang, W. Shao, P. Chu, Z. Tu, T. He, Z. Wu, H. Deng, J. Ge, K. Chen, K. Zhang, L. Wang, M. Dou, L. Lu, X. Zhu, T. Lu, D. Lin, Y. Qiao, J. Dai, and W. Wang. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024c.

D. Cores, M. Dorkenwald, M. Mucientes, C. G. M. Snoek, and Y. M. Asano. TVBench: Redesigning video-language evaluation. *arXiv preprint arXiv:2410.07752*, 2024.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

X. Fang, K. Mao, H. Duan, X. Zhao, Y. Li, D. Lin, and K. Chen. MMBench-Video: A long-form multi-shot benchmark for holistic video understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

M. Farré, A. Marafioti, L. Tunstall, L. Von Werra, and T. Wolf. FineVideo: A fine-grained dataset for video understanding. https://huggingface.co/datasets/HuggingFaceFV/finevideo, 2024.

C. Fu, Y. Dai, Y. Luo, L. Li, S. Ren, R. Zhang, Z. Wang, C. Zhou, Y. Shen, M. Zhang, P. Chen, Y. Li, S. Lin, S. Zhao, K. Li, T. Xu, X. Zheng, E. Chen, C. Shan, R. He, and X. 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)*, 2025.

C. Fu, H. Yuan, Y. Dong, Y.-F. Zhang, Y. Shen, X. Hu, X. Li, J. Su, C. Long, X. Xie, Y. Xie, X. Zheng, X. Yang, H. Cao, Y. Wu, Z. Liu, X. Sun, C. Shan, and R. He. Video-MME-v2: Towards the next stage in benchmarks for comprehensive video understanding. *arXiv preprint arXiv:2604.05015*, 2026.

Google. Gemini 3 Flash Preview. https://ai.google.dev/gemini-api/docs/models/gemini-3-flash-preview, 2025.

Google. Introducing Gemma 3n: A developer guide. https://developers.googleblog.com/en/introducing-gemma-3n-developer-guide/, 2025.

E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

X. Huang, H. Zhou, and K. Han. PruneVid: Visual token pruning for efficient video large language models. In *Findings of the Association for Computational Linguistics (ACL)*, 2025.

B. Li, Y. Zhang, D. Guo, R. Zhang, F. Li, H. Zhang, K. Zhang, Y. Li, Z. Liu, and C. Li. LLaVA-OneVision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024a.

K. Li, Y. Wang, Y. He, Y. Li, Y. Wang, Y. Liu, Z. Wang, J. Xu, G. Chen, P. Luo, L. Wang, and Y. Qiao. MVBench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024b.

Y. Li, C. Tang, J. Zhuang, Y. Yang, G. Sun, W. Li, Z. Ma, and C. Zhang. Improving LLM video understanding with 16 frames per second. *arXiv preprint arXiv:2503.13956*, 2025.

C. Liao, W. Wang, Z. Wen, X. Zheng, Y. B. Wang, H. He, Y. Lyu, L. Jiang, X. Zou, Y. Fu, B. Ren, L. Zhang, and X. Hu. Are we using the right benchmark: An evaluation framework for visual token compression methods. *arXiv preprint arXiv:2510.07143*, 2025.

K. Nan, R. Xie, P. Zhou, T. Fan, Z. Yang, Z. Chen, and Y. Tai. OpenVid-1M: A large-scale high-quality dataset for text-to-video generation. *arXiv preprint arXiv:2407.02371*, 2024.

S. Pichai, D. Hassabis, and K. Kavukcuoglu. A new era of intelligence with gemini 3. https://blog.google/products-and-platforms/products/gemini/gemini-3, 2025.

D. Qin, C. Leichner, M. Delakis, M. Fornoni, S. Luo, F. Yang, W. Wang, C. Banbury, C. Ye, B. Akin, V. Aggarwal, T. Zhu, D. Moro, and A. Howard. MobileNetV4 - universal models for the mobile ecosystem. *arXiv preprint arXiv:2404.10518*, 2024.

Qwen Team. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang. NTU RGB+D: A large scale dataset for 3D human activity analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Y. Shang, M. Cai, B. Xu, Y. J. Lee, and Y. Yan. LLaVA-PruMerge: Adaptive token reduction for efficient large multimodal models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

K. Shao, K. Tao, C. Qin, H. You, Y. Sui, and H. Wang. HoliTom: Holistic token merging for fast video large language models. *arXiv preprint arXiv:2505.21334*, 2025.

L. Shen, G. Gong, T. He, Y. Zhang, P. Liu, S. Zhao, and G. Ding. Fastvid: Dynamic density pruning for fast video large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

X. Shen, Y. Xiong, C. Zhao, L. Wu, J. Chen, C. Zhu, Z. Liu, F. Xiao, B. Varadarajan, F. Bordes, Z. Liu, H. Xu, H. J. Kim, B. Soran, R. Krishnamoorthi, M. Elhoseiny, and V. Chandra. LongVU: Spatiotemporal adaptive compression for long video-language understanding. *arXiv preprint arXiv:2410.17434*, 2024.

B. Shi, S. Fu, L. Lian, H. Ye, D. Eigen, A. Reite, B. Li, J. Kautz, S. Han, D. M. Chan, P. Molchanov, T. Darrell, and H. Yin. Attend before attention: Efficient and scalable video understanding via autoregressive gazing. *arXiv preprint arXiv:2603.12254*, 2026.

K. Tao, C. Qin, H. You, Y. Sui, and H. 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)*, 2025.

P. K. A. Vasu, F. Faghri, C.-L. Li, C. Koc, N. True, A. Antony, G. Santhanam, J. Gabriel, P. Grasch, O. Tuzel, and H. Pouransari. FastVLM: Efficient vision encoding for vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

W. Wang, Z. He, W. Hong, Y. Cheng, X. Zhang, J. Qi, X. Gu, S. Huang, B. Xu, Y. Dong, M. Ding, and J. Tang. LVBench: An extreme long video understanding benchmark. *arXiv preprint arXiv:2406.08035*, 2024.

Y. Wang, X. Li, Z. Yan, Y. He, J. Yu, X. Zeng, C. Wang, C. Ma, H. Huang, J. Gao, M. Dou, K. Chen, W. Wang, Y. Qiao, Y. Wang, and L. Wang. InternVideo2.5: Empowering video MLLMs with long and rich context modeling. *arXiv preprint arXiv:2501.12386*, 2025a.

Z. Wang, S. Purushwalkam, C. Xiong, S. Savarese, H. Ji, and R. Xu. DyMU: Dynamic merging and virtual unmerging for efficient variable-length VLMs. In *International Conference on Learning Representations (ICLR)*, 2025b.

Z. Wen, Y. Gao, W. Li, C. He, and L. Zhang. Token pruning in multimodal large language models: Are we solving the right problem? *arXiv preprint arXiv:2502.11501*, 2025.

H. Wu, D. Li, B. Chen, and J. Li. LongVideoBench: A benchmark for long-context interleaved video-language understanding. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2024.

L. Xing, Q. Huang, X. Dong, J. Lu, P. Zhang, Y. Zang, Y. Cao, C. He, J. Wang, F. Wu, and D. Lin. PyramidDrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

C. Yang, Y. Sui, J. Xiao, L. Huang, Y. Gong, C. Li, J. Yan, Y. Bai, P. Sadayappan, X. Hu, and B. Yuan. TopV: Compatible token pruning with inference time optimization for fast and low-memory multimodal vision language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a.

S. Yang, Y. Chen, Z. Tian, C. Wang, J. Li, B. Yu, and J. Jia. VisionZip: Longer is better but not necessary in vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025b.

J. Yi, S. T. Wasim, Y. Luo, M. Naseer, and J. Gall. Video-Panda: Parameter-efficient alignment for encoder-free video-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

K. Yi, C. Gan, Y. Li, P. Kohli, J. Wu, A. Torralba, and J. B. Tenenbaum. CLEVRER: Collision events for video representation and reasoning. In *International Conference on Learning Representations (ICLR)*, 2020.

Y. Zhang, J. Wu, W. Li, B. Li, Z. Ma, Z. Liu, and C. Li. LLaVA-Video: Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024.

J. Zhou, Y. Shu, B. Zhao, B. Wu, Z. Liang, S. Xiao, M. Qin, X. Yang, Y. Xiong, B. Zhang, T. Huang, and Z. Liu. MLVU: Benchmarking multi-task long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

J. Zhu, W. Wang, Z. Chen, Z. Liu, S. Ye, L. Gu, H. Tian, Y. Duan, W. Su, J. Shao, Z. Gao, E. Cui, X. Wang, Y. Cao, Y. Liu, X. Wei, H. Zhang, H. Wang, W. Xu, H. Li, J. Wang, N. Deng, S. Li, Y. He, T. Jiang,

J. Luo, Y. Wang, C. He, B. Shi, X. Zhang, W. Shao, J. He, Y. Xiong, W. Qu, P. Sun, P. Jiao, H. Lv, L. Wu, K. Zhang, H. Deng, J. Ge, K. Chen, L. Wang, M. Dou, L. Lu, X. Zhu, T. Lu, D. Lin, Y. Qiao, J. Dai, and W. Wang. InternVL3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

## A. Implementation details

**Training (CTD).** For Compressed Token Distillation (CTD), we train the student encoder using the AdamW optimizer with a cosine learning rate schedule and linear warmup. Before distillation, we initialize the student's weights from those of the teacher, clipping them to match the student's dimensions. The global batch size is set to 512, distributed across 8× NVIDIA H100 GPUs. The maximum learning rate is set to 4e-5 with a 100 epochs warmup period; for specific variants susceptible to training instability, we reduce the learning rate to prevent loss explosion. The total training duration is 1800 epochs, requiring approximately 21 days. For the ablation studies (Tables 5, 6 and 7), we perform only 800 epochs of distillation for the training efficiency. We sample 4-frame clips with a frame rate (FPS) uniformly sampled from $[1, 4]$. To stabilize training, we apply an MSE outlier clipping strategy, clipping the target-prediction differences that exceed 3× the standard deviation. Furthermore, we employ gradient clipping with a maximum norm of 1.0.

**Training (LMA).** For Language Model Adaptation (LMA), we adapt the LLM using Low-Rank Adaptation (Hu et al., 2022) with rank $r = 4$, $\alpha = 8$, and lora_dropout = 0.05. Extensive experiments demonstrate that a lower rank (e.g. 4) performs better than higher ones (e.g. 8 and 16). We employ an effective batch size of 128 using gradient accumulation and train with a learning rate of 4e-5 following a cosine schedule. During this phase, we uniformly sample frame counts from $\{128, 256, 512\}$ with an FPS ranging from 1 to 4. This ensures that the total visual token volume matches that of the teacher's typical input (equivalent to 8–32 frames for the uncompressed teacher). We perform LMA on 8× NVIDIA H100 GPUs for 25K steps, which completes in a few hours.

**Datasets.** Our training pipeline utilizes a subset of the video data described in InternVL2.5 paper (Chen et al., 2024c). To be specific, we employ a diverse mix of datasets including ShareGPT4Video (Chen et al., 2024a), LLaVA-Video-178K (Zhang et al., 2024), FineVideo (Farré et al., 2024), CLEVRER (Yi et al., 2020), and NTURGB+D (Shahroudy et al., 2016) for CTD. Moreover, we employ high-quality video-question answering pairs from LLaVA-Video-178K and FineVideo, alongside captioning datasets from ShareGPT4Video and OpenVid-1M (Nan et al., 2024) to ensure robust visual-textual alignment. The datasets used in our work adhere to their respective license: ShareGPT4Video (CC-BY-NC-4.0), FineVideo (CC-BY), OpenVid-1M (CC-BY 4.0), and LLaVA-Video-178K (Apache License 2.0). Note that CLEVRER, and NTURGB+D are exclusively restricted to non-commercial, academic research purposes.

**Benchmarks.** We employ three widely used video benchmarks—Video-MME (Fu et al., 2025), MLVU (Zhou et al., 2025), and LongVideoBench (Wu et al., 2024)—as primary evaluation suites, measuring the average performance. Additionally, HLVid (Shi et al., 2026) is employed to evaluate high-fidelity spatial understanding capabilities for Figure 6. Furthermore, we report the latency-accuracy trade-offs on short video benchmarks, such as MVBench (Li et al., 2024b) and TVbench (Cores et al., 2024), as well as additional long video benchmarks, including LVBench (Wang et al., 2024) and MMBench-Video (Fang et al., 2024), in Appendix B. The datasets evaluated in this work strictly adhere to their respective licenses: MVBench (MIT), HLVid (Apache 2.0), TVBench and MMBench-Video (CC-BY-4.0), and LongVideoBench, MLVU, and LVBench (CC-BY-NC-SA-4.0). Note that Video-MME is exclusively restricted to non-commercial, academic research purposes.

**Evaluation setups.** For evaluating InternVL3-8B (Zhu et al., 2025) and the post-hoc methods in Figures 3 and 5 and table 1, we uniformly sample frames across the entire video and resize them to

448px. The three WAP variants—WAP 4×, 8×, and 16×—in Figure 3 (right) employ compression ratios of $(t, h, w) = (1, 2, 2)$, $(2, 2, 2)$, and $(4, 2, 2)$, respectively. For evaluating LiteFrame, we adopt a dense clip sampling strategy, unlike standard uniform frame sampling. We uniformly sample multiple clips across the video, where each clip consists of a fixed number of frames (4) extracted at a minimum of 1 FPS. All sampled frames are resized to 448px before being fed into the encoder to match the original evaluation setup of the baseline model, except for the spatial scaling experiments presented in Figure 6.

**Latency.** Latency is measured end-to-end including ViT processing and LLM prefilling. We focus exclusively on the visual token encoding and its prefilling stage, as these constitute the primary bottleneck addressed by our contributions. We report the median latency over 100 iterations, following a 40 iterations of warmup phase (140 iterations total), measured on a single NVIDIA A100-80GB GPU.

## B. Additional video benchmarks

### B.1. Short video benchmarks

The efficacy of LiteFrame extends beyond long-form video understanding, demonstrating natural applicability to short video tasks. Specifically, LiteFrame reduces end-to-end latency by 28% and 63% on MVBench (Li et al., 2024b) and TVBench (Cores et al., 2024), respectively, while maintaining the accuracy of the baseline.

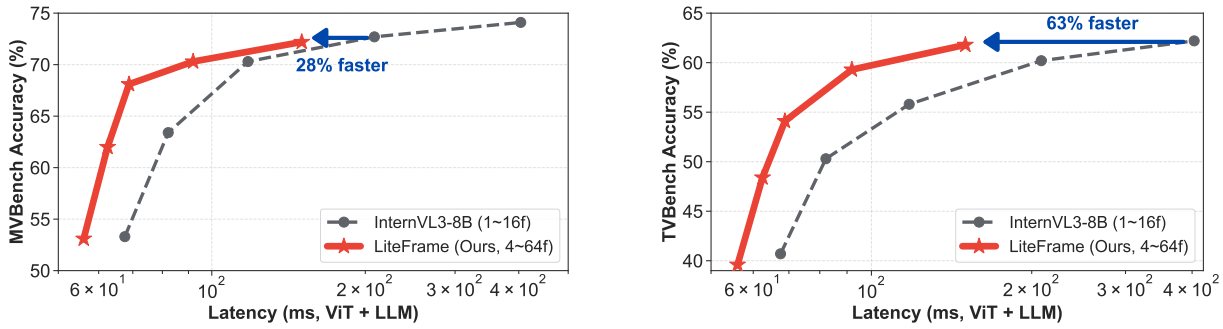



Figure 8 | **Latency-accuracy trade-offs on short video benchmarks.** Evaluation on MVBench **(Left)** and TVBench **(Right)**. Even across standard short-form evaluation setups, LiteFrame achieves significant latency reductions while preserving accuracy.

### B.2. Long video benchmarks

We report additional results on two long video benchmarks, LVBench (Wang et al., 2024) and MMBench-Video (Fang et al., 2024). Notably, on LVBench, LiteFrame utilizing 512-frame input achieves a superior score of 43.9 compared to the 64-frame baseline (43.5) while operating 38% faster, successfully leveraging the extended temporal context. On MMBench-Video, a free-form QA benchmark, LiteFrame demonstrates improved efficiency, particularly within the low-latency regime (16–128 input frames). To evaluate the response quality on the MMBench-Video, we employ the Gemini 3 Flash Preview API (Google, 2025).

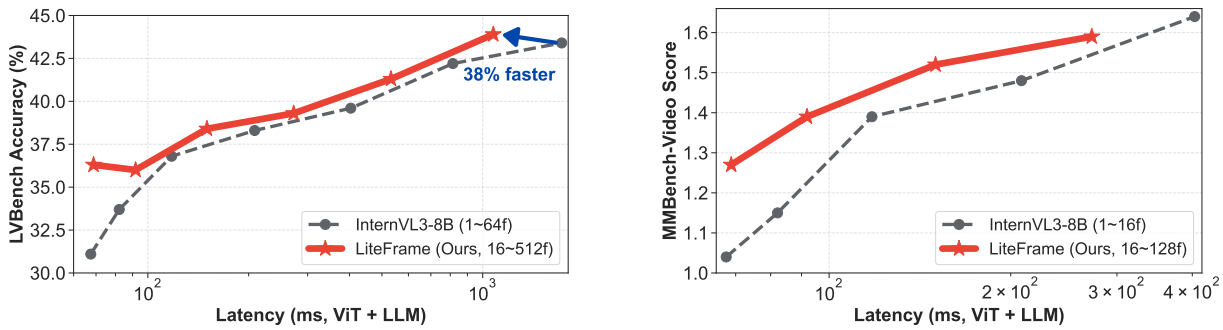



Figure 9 | **Latency-accuracy trade-offs on long video benchmarks.** Additional evaluation results on LVBench **(Left)** and MMBench-Video **(Right)**. LiteFrame achieves the best score of 43.9 (vs. 43.5) on LVBench, while improves efficiency within the low-latency region on MMBench-Video.

## C. Details on ablation studies

### C.1. Reconstructive training objective

While Weighted Average Pooling (WAP) serves as our primary and highly effective mechanism for token compression, we explore whether a purely learned compression paradigm could surpass distilling WAP. To this end, we introduce Reconstructive Token Distillation (RTD), an exploratory variant that removes the constraints of a pre-defined latent space. Instead, RTD employs an autoencoding objective where the student acts as the encoder and lightweight auxiliary transformer blocks serves as the decoder (Dec). The exact objective is to reconstruct the teacher's full dense feature map $T(x)$ from the student's compressed latent representation $S(x)$:

$$\mathcal{L}_{\text{RTD}} = ||T(x) - \text{Dec}(S(x))||_2^2$$

This objective encourages the student to learn a compression manifold that preserves the maximum amount of general visual information from the teacher, theoretically allowing the network to discover non-trivial spatio-temporal dependencies.

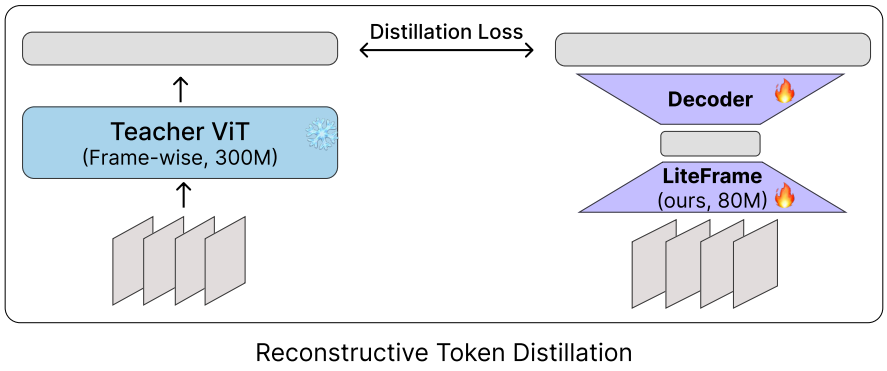



Figure 10 | **Reconstructive Token Distillation (RTD).** RTD enables the student to learn a compressed latent space via an auxiliary auto-encoding strategy, serving as an ablation to our primary WAP-based distillation.

However, as shown in Table 6, empirical results strongly validate our primary design choice (i.e. CTD). While the learnable compression (RTD) yields competitive results when paired with LMA, CTD

Table 6 | **Ablation analysis of compressive token distillation variants.** We evaluate the average performance (%) of different distillation strategies. **CTD**: Compressed Token Distillation; **RTD**: Reconstructive Token Distillation; **LMA**: Language Model Adaptation. **CTD + LMA** achieves the best overall trade-off.

| | Average Accuracy (%) | | |
|---|---|---|---|
| **Method** | **64 frames** | **128 frames** | **256 frames** |
| **RTD** | 43.6 | 43.8 | 43.9 |
| **RTD + LMA** | 59.4 | 61.5 | 63.1 |
| **CTD** | 60.0 | 62.8 | 64.1 |
| **CTD + LMA** | **61.0** | **63.4** | **65.3** |

consistently achieves superior performance. Notably, CTD without LMA already surpasses RTD with LMA. This suggests that explicitly aligning the student with the WAP primitive provides a much more robust, task-relevant semantic foundation than a generic reconstruction objective.

Furthermore, the combination of CTD and LMA delivers the highest accuracy across all frame budgets, reaching 65.3% at 256 frames. This underscores a critical synergy between CTD and LMA. CTD effectively distills the high-saliency feature maps of the teacher into the student's weights, while the subsequent lightweight fine-tuning phase (LMA) is essential for properly aligning the pretrained LLM with the spatio-temporally compressed, information-dense representations produced by our student encoder.

### C.2. Distillation without compression

To isolate the performance gains attributable to our token-compressive student from those of simple model distillation, we compare LiteFrame against a standard baseline where the heavy teacher is distilled into a ViT-Base-12L. For a fair comparison, we evaluate performance without subsequent LMA, keeping the LLM frozen. As demonstrated in Table 7, our "Distill (No Comp.)" baseline suffers from the exact same latency paradox as the teacher: despite a drastic reduction in encoder computational cost (18.5 ms vs. 40.0 ms at 8 input frames), the absence of token compression forces the LLM to process an excessive volume of visual tokens (256 per frame), which severely bottlenecks overall efficiency. Consequently, when constrained to a fixed latency budget, the uncompressed student is forced to process significantly fewer frames, resulting in suboptimal performance.

In contrast, LiteFrame (CTD) drastically reduces the visual token volume to just 16 tokens per frame. This efficiency offloads the critical prefilling bottleneck from the LLM, unlocking the capability to process 8× more frames within much lower latency (272.6ms vs. 393.3ms at 256 input frames for CTD).

### C.3. Spatio-temporal vs. Spatial-only compression

We further scrutinize the necessity of temporal compression by comparing our proposed spatio-temporal reduction (Spatial 4× and Temporal 4×) against a spatial 16× baseline. This baseline applies internal pooling solely along the spatial dimension without temporal layers (i.e. ImageViT), and is trained to predict the teacher's compressed features using spatial 16× WAP.

While the Spatial 16× variant theoretically adheres to the same overall token budget, it significantly underperforms our spatio-temporal approach across most of the latency region. Furthermore, the performance gap widens as the input frame count increases, as shown in Appendix C.3.

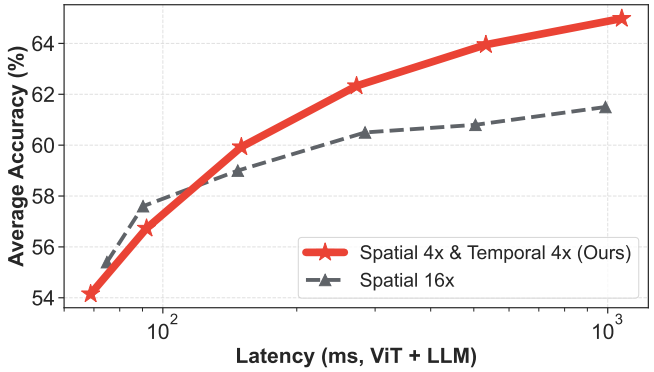
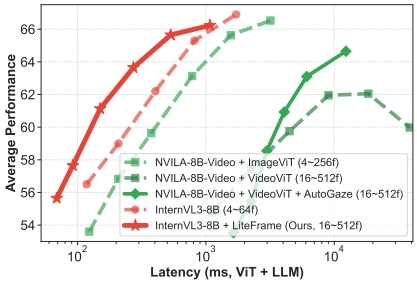



Figure 11 | **Comparison between spatio-temporal compression and spatial-only compression.** We compare the accuracy of 16× spatial-only compression and spatio-temporal compression (4× spatial and 4× temporal). Spatio-temporal compression consistently outperforms spatial-only compression by effectively eliminating temporal redundancies between adjacent frames while preserving spatial fidelity.

Table 7 | **Ablation analysis of compression strategies.** Comparison of our proposed spatio-temporal compression (Ours) against basic distillation without compression (ViT-Base-12L) and spatial-only compression. The best results are marked in **bold**. **Ours (CTD)** consistently outperforms the others.

| Method | Vision Params | Tokens/ Frame | Frames | Latency (ms) | | | Accuracy (%) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Vision | LLM | Total | V-MME† | V-MME‡ | MLVU | LongVideo | Avg |
| InternVL3-8B | 304M | 256 | 8 | 40.0 | 167.3 | 208.4 | **59.0** | 60.0 | 62.2 | 54.8 | 59.0 |
| Distill (No Comp.) | 86M | 256 | 8 | 18.5 | 166.5 | 185.2 | 55.9 | 58.1 | 58.6 | 53.8 | 56.6 |
| Spatial 16x | 86M | 16 | **64** (8×) | 49.8 | 96.2 | **147.4** | 56.2 | 62.0 | 63.2 | 54.6 | 59.0 |
| **Ours (CTD)** | **87M** | 16 | **64** (8×) | 54.8 | 94.9 | 150.1 | 58.7 | **62.1** | **65.1** | 53.9 | **60.0** |
| InternVL3-8B | 304M | 256 | 16 | 74.0 | 329.3 | 403.6 | **61.9** | 64.0 | **66.4** | 56.5 | 62.2 |
| Distill (No Comp.) | 86M | 256 | 16 | 34.3 | 328.6 | 362.9 | 59.4 | 61.4 | 64.5 | 55.7 | 60.3 |
| Spatial 16x | 86M | 16 | **128** (8×) | 93.4 | 166.5 | **264.8** | 57.0 | 64.1 | 64.6 | 56.4 | 60.5 |
| **Ours (CTD)** | **87M** | 16 | **128** (8×) | 105.3 | 166.6 | 272.6 | **61.9** | **65.6** | 65.8 | **57.7** | **62.8** |
| InternVL3-8B | 304M | 256 | 32 | 144.5 | 669.8 | 814.5 | **65.6** | **67.4** | **69.6** | **58.7** | **65.3** |
| Distill (No Comp.) | 86M | 256 | 32 | 63.9 | 670.0 | 733.8 | 60.6 | 64.2 | 67.5 | 58.2 | 62.6 |
| Spatial 16x | 86M | 16 | **256** (8×) | 185.2 | 319.1 | **504.4** | 57.2 | 64.7 | 65.4 | 55.8 | 60.8 |
| **Ours (CTD)** | **87M** | 16 | **256** (8×) | 204.0 | 327.4 | 532.3 | 62.7 | 67.3 | 68.8 | 57.7 | 64.1 |

Specifically, at the 128-frame budget, the spatial-only baseline achieves 60.5% average accuracy compared to our 62.8%, with a notable drop in fine-grained spatial understanding required for rigorous benchmarks like Video-MME (57.0% vs. 61.9%). We attribute this degradation to the excessive loss of spatial fidelity required to meet the compression target (e.g., destructively pooling a frame into a $4 \times 4$ grid). By distributing the compression load across both spatial and temporal dimensions, our approach preserves critical spatial details while effectively aggregating redundant temporal dynamics, resulting in a far more balanced and information-rich representation for downstream LLM reasoning.




| Method | Frames | Latency (ms) | | | | Avg. (%) |
|---|---|---|---|---|---|---|
| | | AutoGaze | ViT | LLM | Total | |
| NVILA-8B-Video (ImageViT) | 32 | - | 451.0 | 329.9 | 780.8 | 63.1 |
| NVILA-8B-Video (VideoViT) | 32 | - | 1916.0 | 330.8 | 2246.8 | 57.5 |
| +AutoGaze (VideoViT) | 256 | 2961.4 | 2605.6 | 539.5 | 6106.5 | 63.1 |
| (compared with ImageViT) | (8×) | (+2961.4ms) | (+477.8%) | (+63.5%) | (+682.1%) | (0.00) |
| InternVL3-8B | 32 | - | 144.5 | 669.8 | 814.5 | 65.3 |
| +LiteFrame (Ours) | 256 | - | 204.0 | 327.4 | **532.3** | **65.7** |
| | (8×) | | (+41.2%) | (-51.1%) | (-34.6%) | (+0.4) |

Figure 12 | **Detailed comparison with AutoGaze. (Left)** LiteFrame (red) improves the efficiency frontier with higher accuracy and lower latency, whereas AutoGaze (green) introduces auxiliary overhead compared with standard VLMs with ImageViT. **(Right)** When scaling from 32 to 256 frames (8×), LiteFrame balances ViT and LLM scaling, while AutoGaze's pre-reduction module becomes a new bottleneck. Note: Relative changes for +AutoGaze are computed against the standard ImageViT baseline, highlighting the architectural overhead.

## D. Detailed comparisons with AutoGaze

### D.1. Detailed comparison

We compare our method against AutoGaze (Shi et al., 2026), a recent approach that similarly addresses the dual computational bottlenecks of the ViT and LLM. We benchmark the latency-performance trade-offs when integrating LiteFrame and AutoGaze into their respective baselines. It is important to note that AutoGaze structurally requires a VideoViT backbone (full spatio-temporal attention across 16-frame clips), whereas standard VLMs utilizes a much lighter ImageViT. For a transparent comparison, we report two NVILA-8B-Video baseline using ImageViT and VideoViT. As illustrated in Figure 12 (left), while AutoGaze successfully accelerates its own heavy VideoViT baseline, it remains substantially slower than a standard baseline equipped with an ImageViT. The detailed breakdown in Figure 12 (right) reveals that the AutoGaze pre-reduction module introduces a severe latency bottleneck, accounting for nearly half of total inference time (3.0s out of 6.1s). In contrast, LiteFrame is designed directly for the standard VLM paradigm, without incurring auxiliary overheads. This allows LiteFrame to strictly advance the Pareto frontier, lowering the total latency without compromising performance.

### D.2. Evaluation setups for AutoGaze

For evaluating AutoGaze, we employ the optimal set of hyperparameters provided by the authors: task_loss_requirement_tile = 0.6, gazing_ratio = [1]+[0.3]∗15 and target_scales = [64, 128, 224, 448]. In addition, we fix the input resolution to 448px to directly match the input of our method, and we set num_video_frames_thumbnail = num_video_frames//16 to avoid using excessive thumbnail frames.

To benchmark the latency, we explicitly exclude video preprocessing overhead of AutoGaze (reading the video, constructing the image pyramid, and resizing frames). This ensures a fair comparison that focuses only on the neural-network execution time. Since AutoGaze's latency varies across videos, we measure the median latency on Video-MME.

Specifically, NVILA-8B-Video (both ImageViT and VideoViT) refer to the NVILA-HD-8B-Video checkpoints evaluated without incorporating AutoGaze module (i.e. setting task_loss_requirement_tile = 1.0). The ImageViT variant encodes videos in a frame-wise manner, whereas VideoViT variant concatenates all visual tokens from a 16-frame clip and proess them through a full spatio-temporal transformer.

## E. Limitations

While LiteFrame establishes a new efficiency-accuracy Pareto frontier for video understanding, we acknowledge a few limitations that present promising directions for future work. First, our Language Model Adaptation (LMA) was trained using a subset of existing video data. Incorporating higher-quality, extreme long-form video datasets could potentially maximize the benefits of our extended temporal context window, further elevating performance without requiring any architectural changes to our core contribution. Second, because our primary focus is mitigating the temporal scaling paradox and frame-count bottlenecks inherent to Video LLMs, we evaluated LiteFrame exclusively on video-centric benchmarks. Although the model exhibits promising zero-shot spatial scaling capabilities, its performance on purely static, traditional image benchmarks remains unexplored. Finally, while we successfully distilled the vision encoder from a heavy 304M parameters teacher into a lightweight 87M parameters student, efforts to scale down to even smaller student models were constrained by training instabilities, such as loss explosions. Advancing the Compressed Token Distillation (CTD) framework to stabilize the training of ultra-lightweight students remains a highly promising next step.