STORM: Token-Efficient Long Video Understanding for Multimodal LLMs

Jindong Jiang[1,2,†,⋆] **Xiuyu Li**[1,3,†,⋆] **Zhijian Liu**[1] **Muyang Li**[1,4,⋆] **Guo Chen**[1,5,⋆] **Zhiqi Li**[1,5,⋆] **De-An Huang**[1] **Guilin Liu**[1] **Zhiding Yu**[1] **Kurt Keutzer**[3] **Sungjin Ahn**[6] **Jan Kautz**[1] **Hongxu Yin**[1] **Yao Lu**[1] **Song Han**[1,4] **Wonmin Byeon**[1]

[1]NVIDIA [2]Rutgers University [3]UC Berkeley [4]MIT [5]Nanjing University [6]KAIST
[†]Equal contribution [⋆]Work performed during internship at NVIDIA

Recent advances in video-based multimodal large language models (Video-LLMs) have significantly improved video understanding by processing videos as sequences of image frames. However, many existing methods treat frames independently in the vision backbone, lacking explicit temporal modeling, which limits their ability to capture dynamic patterns and efficiently handle long videos. To address these limitations, we introduce STORM (**S**patiotemporal **TO**ken **R**eduction for **M**ultimodal LLMs), a novel architecture incorporating a dedicated temporal encoder between the image encoder and the LLM. Our temporal encoder leverages the Mamba State Space Model to integrate temporal information into image tokens, generating enriched representations that preserve inter-frame dynamics across the entire video sequence. This enriched encoding not only enhances video reasoning capabilities but also enables effective token reduction strategies, including test-time sampling and training-based temporal and spatial pooling, substantially reducing computational demands on the LLM without sacrificing key temporal information. By integrating these techniques, our approach simultaneously reduces training and inference latency while improving performance, enabling efficient and robust video understanding over extended temporal contexts. Extensive evaluations show that STORM achieves state-of-the-art results across various long video understanding benchmarks (more than 5% improvement on MLVU and LongVideoBench) while reducing the computation costs by up to 8× and the decoding latency by 2.4-2.9× for the fixed numbers of input frames. Project website is available at here.

1. Introduction

Recent advancements in video-based multimodal Large Language Models (Video-LLMs) have improved AI systems' ability to understand video content [1, 2, 3, 4, 5, 6]. These models typically process videos as sequences of individual frames, encoding each frame independently before feeding representations to a large language model (LLM) for temporal reasoning [3, 4, 2]. Despite leveraging powerful image encoders and LLMs [7, 8], these approaches exhibit fundamental limitations in video processing, particularly for long videos. The lack of explicit temporal encoding in image tokens forces LLMs to infer temporal relationships from static image sequences, creating substantial computational burden. Moreover, to manage computational overhead, existing methods like LongVILA [9] and LongVA [10] often use naive frame subsampling to reduce LLM processing tokens, resulting in significant information loss, potentially eliminating critical details necessary for comprehensive video understanding. Recently, LongVU [11] proposes token compression by adaptively selecting visual tokens based on user queries. However, unlike our approach, LongVU lacks a dedicated module for propagating spatiotemporal information before token selection, potentially limiting the model's ability to preserve critical information after token reduction.

In this paper, we propose STORM (**S**patiotemporal **TO**ken **R**eduction for **M**ultimodal LLMs), which introduces a novel temporal encoder between the image encoder and the LLM. This design integrates temporal dynamics earlier in the model pipeline, significantly enhancing the temporal reasoning capabilities of Video-LLMs while enabling substantial downstream token reduction. We employ Mamba [12] as the core of our temporal layer to allow both computation efficiency and generalization to extended temporal contexts. While there are existing Mamba-based video understanding models [13, 14, 15, 16, 17], unlike ours, these methods do not integrate Mamba with LLMs for multimodal tasks, and importantly, they mainly focus on replacing traditional backbones with Mamba architectures without explicitly leveraging Mamba's ability for reducing video redundancy and enabling visual token compression.

The key advantage of the Mamba layer is its ability to compress historical information into state representations. As consecutive frames in the video input often

Q: What is funny about the video?

- **Ours:** The video is funny because it shows a man driving a car with a chair on top of it, which is an unusual and unexpected sight. The man also appears to be enjoying himself, which adds to the humor of the situation.
- **LongVU:** The video is funny because it shows a man in a car who is driving erratically and bumping into things. He is also seen sitting on top of a chair and playing a musical instrument.
- **LongVILA:** The video is funny because it shows a man driving a car while sitting on a chair and holding a mop, which is an unusual and unexpected sight. The man's actions are also comical, as he is seen driving the car while sitting on the chair and holding the mop, which is not a typical way to drive a car. Additionally, the video ends with a man driving a car while sitting on a chair and holding a mop, which is a humorous and unexpected ending.

Figure 1 | **Open-Ended Video Understanding.** We show STORM's ability to handle free-form queries about complex long video scenes. By employing the Mamba-based temporal encoder to capture essential spatiotemporal cues while compressing redundant frame information, STORM enables efficient, accurate long-video understanding and outperforms existing methods on a wide range of video understanding tasks.

contain redundant information, our temporal encoder efficiently processes and propagates temporal information across the entire video sequence. The resulting visual tokens inherently encapsulate temporal history, allowing us to extract fewer tokens for LLM processing while preserving key information. We explore both training-free token subsampling and training-based compression methods (temporal and spatial token compression). Unlike previous frame-subsampling approaches, our method preserves essential temporal information in a compressed format, reducing computational load while improving performance through more comprehensive representation in a compact token space. Empirical evaluations demonstrate that our method significantly outperform baselines, especially on long-video inputs, while simultaneously achieving strong computational efficiency.

2. Related Work

Vision Language Model Architecture Vision language models (VLMs) primarily adopt two paradigms to process visual input. The first paradigm freezes language model weights and integrates visual information via cross-attention mechanisms [18], while the second paradigm utilizes a pre-trained image encoder, such as CLIP [7] or SigLIP [8], to convert images into tokens. These tokens are then concatenated with text tokens and input into the language model [2, 1, 19]. This approach can be naturally extended to video understanding by treating videos as sequences of images processed by the vision encoder [3, 4]. To enhance video processing, some works introduce specialized video encoders. For instance, InternVideo [5, 6] uses VideoMAE [20] as a video encoder, while Kangaroo [21] integrates depth-wise 3D convolution for fusing video tokens. In this work, we retain SigLIP as the vision encoder and focus on enhancing long video understanding by incorporating a linear-complexity temporal module in the Mamba [12] architecture. Positioned between the SigLIP vision encoder and the language model, this module efficiently improves spatial-temporal modeling effectiveness.

Long Video Understanding Understanding long videos with VLMs presents significant challenges in both accuracy and efficiency. Previous approaches have employed long-context language models trained on short-context video data to enable long video comprehension [10]. However, these methods lack sufficient long video training data and incur high computational costs during both training and inference as the number of frames increases. LongVILA [9] addresses these challenges through a multi-modal sequence parallelism system that directly handles long video data during training and inference, but this approach requires customized system implementations tailored for multi-GPU setups. Another line of research focuses on token reduction to shorten input sequences, thereby enabling efficient inference for long videos [22, 23, 24, 25, 26, 11]. For instance, VoCO-LLaMA [24] and VideoXL [26] use recursive KV cache compression learnt in an end-to-end manner, and LongVU [11] leverages DINO features for frame selection and inter-frame similarity to reduce tokens. Despite these diverse strategies, direct pooling along the temporal or spatial dimensions often performs sufficiently well, with additional gains being marginal. In this paper, we apply temporal and spatial pooling for token reduction, achieving superior performance when combined with our temporal projector.

Mamba for Video Understanding Recent advances in linear state space models such as Mamba [12, 27] have sparked extensive exploration in applying them to video understanding tasks. Due to their subquadratic computation complexity, Mamba models achieve significant efficiency improvements compared to transformer-based architectures while still delivering competitive performance [12, 27, 28, 29, 30]. These properties make Mamba particularly suitable for video processing as the models are required to process long sequence inputs. For example, VideoMamba in [13] and VideoMamba (identical model naming) in [14] use Mamba-based visual backbone in video models and demonstrates the model's strong ability to capture both local redundancy and long-term spatiotemporal dependencies. VideoMambaPro [15] proposes to improve Mamba's video understanding ability by applying masking and residual connection during the backward scan. The Video Mamba Suite [16] further explores various architectures to integrate Mamba into existing video understanding models, demonstrating favorable efficiency-performance tradeoffs for long sequence inputs. Mamba-ND [17] aims to improve Mamba's performance on multi-dimensional data by investigating design choices such as SSM layer structure and scanning order within and across dimensions. However, unlike our approach, these works do not directly apply Mamba for Multimodal LLMs. More importantly, they primarily focus on replacing traditional backbones with Mamba architectures without explicitly leveraging Mamba's unique ability to summarize historical information for reducing video redundancy and enabling visual token compression. Our paper addresses this research gap by proposing STORM, which proves to be both effective and efficient for video understanding while significantly reducing computational demands.

Concurrent Work Recently, BIMBA [31] explores a similar architecture for long-video understanding, reporting similar benefits through empirical evaluation. We are encouraged to see these independent findings further support the hypothesis.

3. Method

This section introduces the Mamba-based temporal projector architecture and explores several token compression techniques for efficient long video processing. We investigate both training-based and training-free token compression strategies across temporal and spatial dimensions. We begin with an overview of our Mamba-based temporal projector, followed by a discussion of the token compression methods. The overview of our method is shown in Figure 2.

3.1. Preliminaries

State Space Models (SSMs) A State Space Model (SSM) [32, 33, 12, 34] establishes a linear transformation between an input sequence $\boldsymbol{x}_{1:T} \in \mathbb{R}^{T\times D}$ and an output sequence $\boldsymbol{y}_{1:T} \in \mathbb{R}^{T\times D}$ through the following recurrent process:

$$\begin{aligned} \boldsymbol{h}_t &= \overline{\boldsymbol{A}}_t \boldsymbol{h}_{t-1} + \overline{\boldsymbol{B}}_t \boldsymbol{x}_t \ , \\ \boldsymbol{y}_t &= \boldsymbol{C}_t \boldsymbol{h}_t \ . \end{aligned} \tag{1}$$

Here, T is the sequence length; $\boldsymbol{x}_t, \boldsymbol{y}_t \in \mathbb{R}^D$ are input and output vectors at time t; and $\boldsymbol{h}_t \in \mathbb{R}^H$ is the hidden state summarizing the history $\boldsymbol{x}_{\leq t}$. The matrices $\overline{\boldsymbol{A}}_t \in \mathbb{R}^{H\times H}$, $\overline{\boldsymbol{B}}_t \in \mathbb{R}^{H\times D}$, and $\boldsymbol{C}_t \in \mathbb{R}^{D\times H}$ are parameterized with learnable weights designed to facilitate the modeling of long-term dependencies. When $\overline{\boldsymbol{A}}_t, \overline{\boldsymbol{B}}_t, \boldsymbol{C}_t$ are time-invariant (constant over t), the computation of $\boldsymbol{y}_{1:T}$ can be parallelized, enabling efficient training and inference.

Mamba Recently, Mamba [12] propose to condition these matrices on the input $\boldsymbol{x}_t$ to enhance the sequence modeling capabilities of SSMs. Specifically, Mamba employs learnable functions $\overline{\boldsymbol{A}}\colon \mathbb{R}^D \to \mathbb{R}^{H\times H}$, $\overline{\boldsymbol{B}}\colon \mathbb{R}^D \to \mathbb{R}^{H\times D}$, and $\boldsymbol{C}\colon \mathbb{R}^D \to \mathbb{R}^{D\times H}$ to generate input-dependent matrices as follows:

$$\overline{\boldsymbol{A}}_t = \overline{\boldsymbol{A}}(\boldsymbol{x}_t) \ , \quad \overline{\boldsymbol{B}}_t = \overline{\boldsymbol{B}}(\boldsymbol{x}_t) \ , \quad \boldsymbol{C}_t = \boldsymbol{C}(\boldsymbol{x}_t) \ . \tag{2}$$

This approach allows the model to dynamically emphasize or suppress information based on the current input, thereby enabling more flexible and adaptive sequence modeling. Additionally, Mamba leverages a hardware-aware parallel algorithm to ensuring that the input-dependent matrices does not hinder the training and inference efficiency inherent to SSMs.

3.2. Mamba-Based Temporal Projector

Traditional Video-LLMs often process video frames independently, requiring the LLM to infer temporal relationships from sequences of static images. This approach is computationally inefficient, particularly when processing long videos. Additionally, this method fails to leverage the inherent temporal redundancy in video data, resulting in redundant processing of similar information across consecutive frames. To address these limitations, we introduce the **Mamba-based temporal projector**, which efficiently integrates temporal information across video frames while enabling effective temporal token compression.

Let $\mathbf{X}_t \in \mathbb{R}^{\hat{N}\times D}$ denote the image tokens for frame t by a Vision Transformer (ViT) encoder, where $\hat{N}$ is the number of tokens per frame and D is the token dimension. We first apply a linear layer to downsample

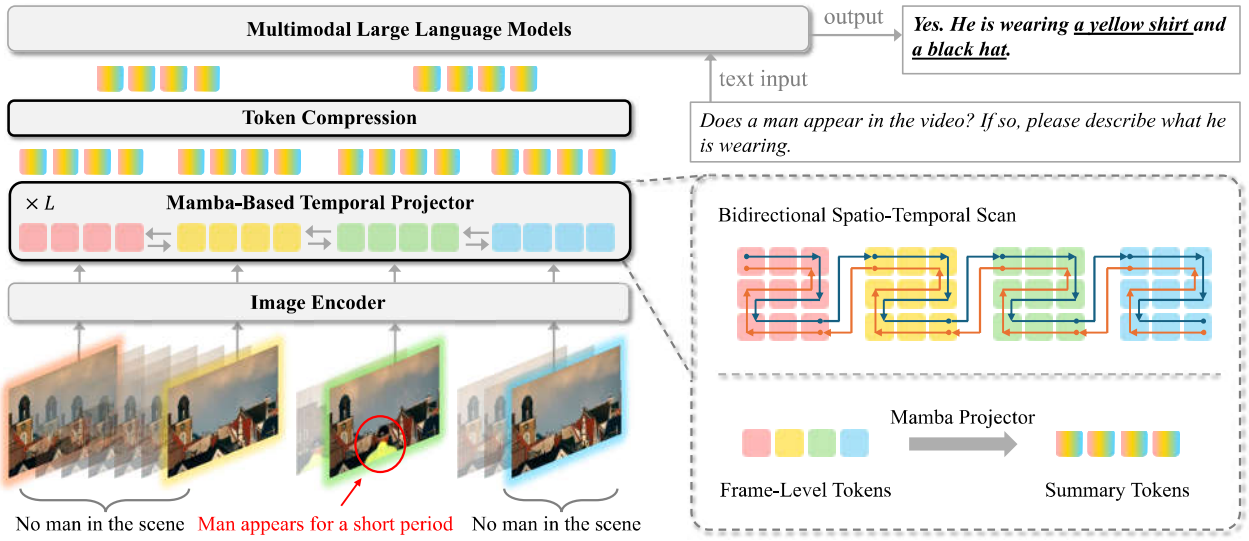


Figure 2 | **Overview of STORM pipeline.** STORM integrates a Mamba-based temporal projector between the image encoder and LLM. This projector performs spatiotemporal scanning to embed temporal information directly into visual tokens. The resulting Summary Tokens encapsulate temporal history, enabling efficient downstream token reduction while preserving essential video dynamics.

the tokens of each frame to $\frac{\hat{N}}{r}$ tokens:

$$\tilde{\mathbf{X}}_t = \text{Linear}\left(\mathbf{X}_t\right), \quad \text{for } t = 1, \ldots, T. \tag{3}$$

Where r is the downsample ratio. For simplicity, we define $N = \frac{\hat{N}}{r}$ and use N throughout the remainder of this paper. The downsampled tokens from all frames are stacked to form the input tensor for the temporal module:

$$\tilde{\mathbf{X}} = \left[\tilde{\mathbf{X}}_1; \tilde{\mathbf{X}}_2; \ldots; \tilde{\mathbf{X}}_T\right] \in \mathbb{R}^{T \times N \times D}. \tag{4}$$

The core of the temporal projector consists of L Mamba layers that iteratively integrate temporal dynamics into the tokens. In each layer $l = 1, \ldots, L$, we fuse temporal information into the visual tokens by:

$$\mathbf{X}^{(l)} = \mathbf{X}^{(l-1)} + \text{MambaMixer}\left(\text{Norm}\left(\mathbf{X}^{(l-1)}\right)\right), \tag{5}$$

where $\mathbf{X}^{(0)} = \tilde{\mathbf{X}}$, and $\text{Norm}(\cdot)$ denotes layer normalization. Each MambaMixer employs a bidirectional scanning module that captures dependencies in both spatial and temporal dimensions. Specifically, we apply a sweeping scan order within each frame and across frames, i.e., left-to-right, top-to-bottom, and frame-to-frame (see Figure 2). After L layers, we obtain tokens enriched with temporal information, denoted as $\mathbf{X}^{(L)} \in \mathbb{R}^{T \times N \times D}$.

3.3. Training-Time Token Compression

Processing long videos poses two major challenges for LLMs. First, handling all frames is computationally expensive, often requiring specialized system optimizations like sequence parallelization and multiple GPUs for training and inference [9]. Second, LLMs are inherently limited by their training context lengths constraints. For example, LLaMA 3 has a context length of 8K tokens, which corresponds to only 32 frames for video inputs when using 256 tokens per frame. Without token compression, video processing quickly exceeds the effective input capacity of LLMs, leading to significantly reduced model performance. In this work, we aim to address both computational cost and context length limitations by enabling efficient long-video processing through token compression along both temporal and spatial dimensions. Our approach eliminates the need for custom system optimizations and allows inference on a single GPU. We illustrate the training-time token compression in Figure 3 (left and middle).

Temporal Pooling Since consecutive frames often contain similar information, analyzing every frame may lead to redundant processing and potential overfitting. Additionally, having too many tokens can make it difficult for the LLM to identify important temporal patterns. Thus, we propose applying temporal average pooling to compress visual information efficiently [35, 36]. This method combines data from consecutive frames by averaging their enriched visual tokens. Specifically, for the tokens $\mathbf{X}^{(L)} \in \mathbb{R}^{T \times N \times D}$ from the temporal projector, we average every k consecutive frames:

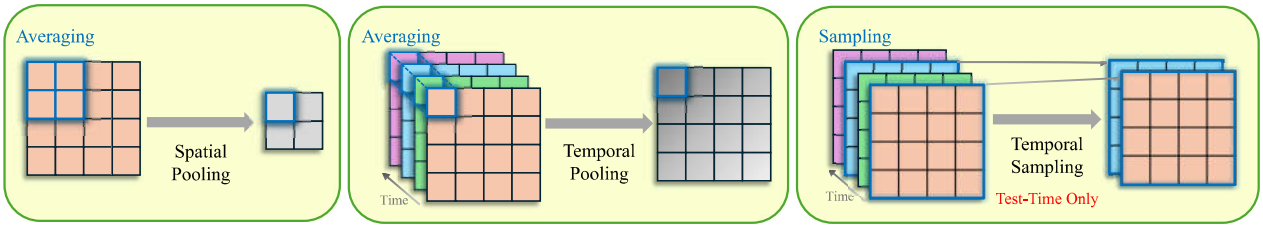


Figure 3 | **Token Compression Strategies.** This figure illustrates our token compression techniques: spatial average pooling (left), temporal average pooling (middle), and training-free temporal token sampling (right). These methods can be applied individually or in combination, depending on task requirements and computational budget constraints.

$$\mathbf{X}_{\text{time-avg}} = \frac{1}{k}\sum_{i=0}^{k-1}\mathbf{X}_{t+i}^{(L)}, \text{ for } t = 0, k, 2k, \ldots, T-k. \tag{6}$$

As a result, we obtain compressed tokens:

$$\mathbf{X}_{\text{time-avg}} \in \mathbb{R}^{\frac{T}{k}\times N\times D}, \tag{7}$$

Despite its simplicity, temporal averaging effectively decreases the number of tokens the LLM processes, with minimal loss of critical information. This motivates us to adopt such simple yet effective technique for efficient training in long video understanding.

Spatial Pooling In addition to temporal pooling, we also explore average pooling in the spatial domain. Formally, given the input $\mathbf{X}^{(L)} \in \mathbb{R}^{T\times N\times D}$ from the visual encoder and the spatial compression ratio p, we apply average pooling with a kernel size and stride of p on each frame, resulting in $\mathbf{X}_{\text{space-avg}} \in \mathbb{R}^{T\times \frac{N}{p}\times D}$.

3.4. Training-Free Temporal Token Sampling

After processing through the temporal projector, each visual token is enriched with spatiotemporal information, capturing features not only from its corresponding frame but also from other frames across the entire video. This encoding of global information allows us to subsample the visual tokens along the temporal dimension at test time, further reducing the number of tokens fed into the LLM without significant loss of information or performance. This temporal token subsampling strategy can be done with or without our pooling mechanisms. Note that comparing to methods that subsample raw frames and risk discarding critical temporal cues, our approach leverages the spatiotemporal richness of tokens post-temporal encoding. We show an illustration of the temporal token sampling in Figure 3 (right).

Formally, let $\mathbf{X}^{'} \in \mathbb{R}^{T'\times N'\times D}$ denote the token input to the subsampling layer which can be tokens from visual encoder $\mathbf{X}^{(L)}$, or tokens output from compression modules $\mathbf{X}_{\text{time-avg}}$ or $\mathbf{X}_{\text{space-avg}}$. Here, T' and N' are the temporal and spatial dimensions after any temporal or spatial pooling. We apply a uniform subsampling with rate s along the temporal dimension.

$$\mathbf{X}_{\text{time-skip}} = \{\mathbf{X}_t^{'} \mid t = 0, k, 2k, \ldots, T-1\} \in \mathbb{R}^{\frac{T'}{s}\times N'\times D} \tag{8}$$

Our empirical results demonstrate that this subsampling method not only maintains performance on various video understanding benchmarks but can also improve it by reducing noise from redundant frames.

4. Experiments

In this section, we extensively evaluate the proposed method on various video understanding benchmarks and provide empirical analysis demonstrating how the temporal projector enables efficient token reduction while delivering strong video reasoning abilities.

4.1. Experiment Details

We use the pre-trained vision encoder in PaliGemma [37] and the LLM model from Qwen2-VL [38], and fine-tune them to adapt to our video datasets. The temporal projector is initialized with random weights. Each image is always resized to a 448×448 resolution. Note that we also carried out experiments on more model configurations in Section 5 to showcase the universality of our design.

Training In the first stage, known as the Alignment Stage, we freeze both the image encoder and the LLM, training only the temporal projector using a small

Models	Size	Comp. Ratio (%)	#Frames (train)	# Frames (test)	MVBench test	MLVU dev	LongVideoBench val	VideoMME (w/o sub.)
Duration					16 sec	3∼120 min	8 sec∼60 min	1∼60 min
Proprietary Models								
GPT4-V	-	-		1fps	43.7	-	-	60.7
GPT4-O	-	-		1fps	64.6	66.2	-	77.2
Open-Source Video-LLMs								
Video-LLaVA [39]	7B	-	8	8	41.0	47.3	-	40.4
LLaMA-VID [22]	7B	-	1fps	1fps	41.9	33.2	-	-
Chat-UniVi [40]	7B	-	64	64	-	-	-	45.9
ShareGPT4Video [41]	8B	-	16	16	51.2	46.4	-	43.6
LLaVA-NeXT-Video [42]	7B	-	16	32	33.7	-	-	46.5
VideoLLaMA2 [4]	7B	-	16	32	54.6	48.5	-	46.6
VideoChat2 [3]	7B	-	8	16	60.4	47.9	-	54.6
Long-LLaVA [43]	A13B (53B)	-	-	128	54.6	-	-	51.6
LongVA [10]	7B	-	-	128	-	56.3	-	54.3
LongVILA [9]	7B	-	2048	256	-	-	-	57.5
mPLUG-Owl3 [44]	8B	-	8	128	54.5	-	52.1	53.5
LLaVA-OneVision [45]	7B	-	32	32	56.7	64.7	56.5	58.2
Kangaroo [21]	8B	-	160	64	61.0	61.1	54.8	56.0
VideoXL [26]	7B	-	128	2048	55.3	64.9	49.5	55.5
Oryx-1.5 [46]	7B	-	256	64	67.6	67.5	56.3	58.8
LongVU [11]	7B	-	-	1fps	66.9	65.4	-	60.6
Qwen2-VL [38]	7B	-	2fps	2fps	67.0	-	55.6	63.3
Ours	7B	**25/12.5***	128	256	**71.3**	**72.9***	**60.5***	**63.4**

* test-time temporal sampling is applied, 2× additional compression.

Table 1 | **Comparison with Existing Video-LLMs.** We compare our best-performing configuration (STORM + T. Pooling) against existing Video-LLMs. Our approach outperforms all existing models while using significantly fewer tokens through 4× temporal pooling token compression. Additionally, values with asterisk (*) indicate that test-time token sampling is applied, which introduces additional 2× compression on visual inputs. Latency are based on 256 frames. # Frame (test) denote the maximum frames used in all benchmarks.

image-text dataset [47], containing 95K image-text pairs. Note that the Mamba layers perform not only temporal scan but also spatial scan within images, so video inputs are not strictly required to train it. For alignment stage, we find it sufficient to use only image-text pairs to pretrain the temporal projector. In the second stage, the supervised fine-tuning stage (SFT), we fine-tune all three components using a large and diverse dataset that includes text-only, image-text, and video-text data. There are around 12.5M samples in our SFT data mixture. due to space constraints. At this stage, we use 32 frames for each video input. For models with training-time token compression , we use a compression ratio of 4× — temporal pooling models compress 32 frames to 8 frames while spatial pooling models compress 256 tokens per image into 64 tokens. Moreover, for models with training-time token compression , we further employ a long video fine-tuning stage using 128-frames long-video inputs from the LLaVA-Video dataset [48]. We provide further details about the full SFT dataset and long video fine-tuning dataset in the appendix Section 8.4.

Evaluation We evaluate our STORM across multiple configurations on recent long-video understanding benchmarks specifically designed to rigorously assess video-language model capabilities. These benchmarks include MVBench [49], MLVU [50], LongVideoBench [51], and VideoMME [52]. We compare our approach against a wide range of representative video-language models, including recently proposed models tailored for long-video understanding [9, 11, 10, 26, 43, 21, 46]. See Table 1 for details.

Models We implement STORM based on the VILA codebase [2], a typical VLM pipeline consisting of a vision encoder, LLMs, and vision-language projector, and introduce our novel Mamba module and compression mechanisms into the architecture. We will refer all models trained with the VILA codebase as VILA-based models in our experiments. To thoroughly analyze our design, we evaluate STORM variants using all combinations of the three compression methods: temporal average pooling, spatial average pooling, and temporal token sampling. The full results are presented in Table 5. For comparison with existing Video-LLMs, we highlight the best-performing variants in Table 1. And include the detailed analysis of all variants of STORM in Table 2. To ensure fairness, we also include a baseline VILA model trained on the same dataset and training scheme but without the Mamba module. All of our models in Table 2 are trained under the same 8K token budget (corresponding to 32 frames of tokens), which represents the

VILA-Based Models	Size	Comp. Ratio (%)	Latency (s)	#Frames (train)	# Frames (test)	MVBench test	MLVU dev	LongVideoBench val	VideoMME (w/o sub.)
Duration						16 sec	3∼120 min	8 sec∼60 min	1∼60 min
Token Budget: 8K									
VILA Baseline	7B	100	4.31	32	256	69.5	70.2	55.9	60.1
STORM	7B	100	4.47	32	256	70.3	71.1	54.5	62.4
STORM + T. Pooling	7B	25	1.82	128	256	**71.3**	72.5	59.5	**63.4**
STORM + T. Sampling *	7B	50	2.50	32	256	70.1	70.8	54.8	63.1
STORM + T. Pooling + T. Sampling *	7B	12.5	1.51	128	256	70.6	**72.9**	**60.5**	62.4

* 2× additional compression at test time.

Table 2 | **Comparison between VILA-Based Models on the Same Token Budget.** We compare between STORM variants and the baseline VILA model, all trained with identical data and training pipelines. All models use the same 8K token budget, which is the maximum number of visual tokens provided to the LLM during training. Temporal pooling (T. Pooling) applies 4× compression and test-time token sampling (T. Sampling) applies additional 2× compression.

number of visual tokens fed to the LLM—a key factor affecting inference latency and memory consumption, particularly as the number of frames increases (see Section 5). Specifically, we report results for the standard STORM trained on 32-frame inputs and STORM with Temporal Pooling, which processes 128-frame inputs while reducing the token count to match the 32-frame variant. Additionally, we evaluate configurations where temporal sampling is applied at test time (+T. Sampling), which not only further enhances model efficiency but also improves performance on certain benchmarks.

4.2. Results on Video Understanding Benchmarks

STORM vs Existing Methods We start with comparing our configuration STORM + T. Pooling (+ T. Sampling) with existing Video-LLMs. As shown in Table 1 and detailed in Table 2, STORM + T. Pooling achieves new state-of-the-art performance across all long-video understanding benchmarks. Specifically, it achieves 71.3% accuracy on MVBench, 72.5% on MLVU, 59.5% on LongVideoBench, and 63.4% on VideoMME, outperforming all open-source Video-LLMs, including recent models specifically designed for long-context inputs such as LongVU and LongVILA. Additionally, our method significantly narrows the performance gap with proprietary models, outperforming GPT4-V and GPT4-O on MVBench and MLVU, as well as GPT4-V on VideoMME. Notably, STORM + T. Pooling achieves computational efficiency by compressing visual tokens to 25% of their original number before processing them through the LLM. We can further enhance efficiency by applying test-time temporal sampling in STORM + T. Pooling + T. Sampling while still achieving competitive results, reducing the token count to just 12.5% while maintaining competitive performance. In fact, this additional compression even improves results of certain benchmarks, yielding the best overall performance on MLVU and LongVideoBench.

STORM vs Baseline VILA Next, we provide controlled comparisons within VILA-based models to reveal advantages of the proposed Mamba-based temporal module in Table 2. We first compare the baseline VILA model with our STORM. By incorporating the Mamba module, STORM achieves performance gains on three out of four benchmarks, including a notable 2.3% improvement on VideoMME. Moreover, augmenting STORM with test-time temporal token sampling (STORM + T. Sampling) further enhances efficiency, reducing inference time by 43% while, surprisingly, maintaining or slightly boosting performance (an additional 0.8% gain on VideoMME). This advantageous behavior emerges because the Mamba module's ability to effectively propagate temporal information across video frames, enabling redundant tokens to be discarded without compromising the model's overall understanding.

The temporal pooling variant (STORM + T. Pooling) extends these benefits to long-context training, by applying temporal average pooling after the Mamba layer, which allows the model to process 128-frame inputs while compressing the token count to match that of the 32-frame setting. This approach not only improves performance, achieving 63.4% (+3.3%) on VideoMME, 59.5% (+3.6%) on LongVideoBench, 72.5% (+2.3%) on MLVU, and 71.3% (+1.8%) on MVBench, but also significantly reduces inference latency by 58.4%. By combining this model with test-time temporal token sampling (STORM + T. Pooling + T. Sampling), we further reduce the inference time by 65.5% and use only 12.5% of the visual tokens compared to VILA without sacrificing performance. We observe that this test-time temporal token sampling particularly benefits MLVU and LongVideoBench compared to MVBench and VideoMME. This different impact across benchmarks

Models	LongVideoBench		VideoMME	
	32F	128F	32F	128F
S. Pooling	**56.0**	55.9(-0.1)	61.1	58.3 (-2.8)
T. Pooling	54.2	**59.5** (+5.3)	**61.2**	**63.4** (+2.2)

Table 3 | **Spatial Pool vs Temporal Pooling on Frame Extension.** Models were initially trained on 32 frames, then fine-tuned on 128 frames using a small long video dataset (see Section 8.4). F indicates frame count during training. Both methods compress visual tokens to 25% of the original count. Spatial pooling performance degrades with increased frame length, while temporal pooling shows consistent improvements.

likely stems from the nature of the underlying tasks. MLVU and LongVideoBench require global understanding across long videos. We believe that test-time compression with the Mamba module better summarizes the essential contextual information. On the other hand, MVBench and VideoMME require visual details from specific frames. Our pooling-only method with the Mamba module maintains more detailed frame information throughout the sequence. Section 5 includes a detailed discussion about retaining visual information with our token compression.

Spatial Pooling vs Temporal Pooling on Long Video Inputs Table 3 provide comparison between spatial average pooling and temporal average pooling on 32 frames training and 128-frames extension. All models use STORM as the base model. We see that spatial pooling is effective when 32 frames are during training, outperforming temporal pooling and on LongVideoBench while achieving on par results on VideoMME. However, when applying 128-frames input, even though both methods use the same token budget, temporal pooling results in sigificantly better performance. In fact, spatial pooling can not benefit from longer video inputs, results in degrated performance on both LongVideoBench and VideoMME, while temporal pooling successfully achieves stronger performance on both benchmarks from the extended video length.

5. Analysis

Mamba Module Improves Simple Token Compression As illustrated in Figure 2, the Mamba-based temporal projector performs a spatiotemporal scan over input tokens. This enables refinement of visual tokens before compression operations, thereby preserving key temporal and spatial cues—such as the positional information of frames being pooled—that would otherwise be lost in naive compression strategies. Consequently, the Mamba temporal module allows simple token compression methods to work more effectively for long videos compared to the baseline, as evidenced in Table 4 and further detailed in Table 10.

Models	8F	32F (T. Pooling)	128F (T. Pooling)
VILA (w/o Mamba)	**62.0**	63.5 (+1.5)	64.3 (+0.8)
STORM (w/ Mamba)	61.6	**64.2** (+2.6)	**66.7** (+2.5)

Table 4 | **The Critical Role of the Mamba Module.** Results show average performance across all video benchmarks. Numbers in parentheses show improvements over shorter version of the same models. For 32F and 128F results, models were initially trained on the full dataset with 32 frame inputs, then fine-tuned on 128 frame inputs using a smaller long-video dataset (see Section 8.4). STORM demonstrates substantially better utilization of longer temporal contexts, with the performance gap between VILA baseline and STORM widening as input length increases (+0.7% at 32F and +2.4% at 128F compared to baseline). Full details of the comparisons are provided in Table 10.

Our results demonstrate that STORM consistently improves with increased video input length during training, achieving substantial gains of +2.6% over all benchmarks when extending from 8 frames to 32 frames and an additional +2.5% when extending from 32 frames to 128 frames. In contrast, the baseline VILA model shows smaller improvement when extending from 8 frames to 32 frames, and demonstrates only +0.8% performance gains when fine-tuned on 128 frames. In fact, as shown in Table 10, the baseline model exhibits performance degradation on MVBench and MLVU benchmarks when extending video length from 32 frames to 128 frames. These results fully demonstrate the importance of the Mamba module in enabling both efficient and effective token compression, particularly for long-video input

Ablation of Mamba and token compression modules Table 5 presents an ablation study comparing various token compression strategies for models trained on a fixed 32-frame input. Without any compression, our standard STORM delivers improved performance across benchmarks comparing to the baseline VILA while utilizing a similar inference latency and number of visual tokens. When test-time temporal sampling is applied, the model maintains its performance while reducing the visual token count to 50% and lowering inference latency to 58.7% of the uncompressed STORM. Employing temporal average pooling during both training and testing further com-

Model	Mamba Module	T. Sampling	T. Pooling	S. Pooling	Token budget	Compression Ratio (%)	Latency (s)	MVBench	MLVU	LongVideoBench	VideoMME w/o sub.
VILA	✗	✗	✗	✗	8K	100	4.31	69.5	70.2	55.9	60.1
STORM	✓	✗	✗	✗	8K	100	4.47	**70.3**	**71.1**	54.5	62.4
	✓	✓	✗	✗	8K	50	2.50	70.1	70.8	54.8	**63.1**
	✓	✗	✓	✗	2K	25	1.82	**70.4**	**71.0**	54.2	61.2
	✓	✗	✗	✓	2K	25	1.82	68.9	69.2	**56.0**	61.1
	✓	✓	✓	✗	2K	12.5	1.51	**70.1**	**71.0**	54.0	60.8
	✓	✓	✗	✓	2K	12.5	1.51	68.9	69.5	**56.3**	60.9
	✓	✗	✓	✓	0.5K	6.25	1.36	68.5	68.2	53.7	60.2
	✓	✓	✓	✓	0.5K	3.13	1.29	67.2	68.6	55.3	60.1

Table 5 | **Ablation of token compression methods.** All models are trained on 32 frames. The pooling methods are added during training, and the temporal sampling is executed at test-time.

presses the token count to 25% and cuts latency down to 42.7%, which is particularly effective for MVBench and MLVU but slightly degrades LongVideoBench and VideoMME. Similarly, spatial average pooling provides the same efficiency improvements and proves particularly effective on LongVideoBench, but with compromises on other benchmarks.

We further apply temporal token sampling with either temporal or spatial pooling to improve the test-time efficiency. For instance, STORM + T. Pooling + T. Sampling reduces the visual tokens to only 12.5% and latency to 35.4% of the original STORM, while maintaining comparable accuracy to STORM + T. Pooling. The STORM + S. Pooling + T. Sampling even achieves the best LongVideoBench performance with this strong token reduction. Finally, we combine the temporal and spatial average pooling with/without test-time temporal token sampling, leading to even more aggressive compression. Interestingly, our STORM + T. Pooling + S. Pooling + T. Sampling variant, which has the strongest compression ratio, utilizing only 3.13% visual token and 29.5% of inference latency, already achieves a competitive results. Comparing to the baseline VILA, it achieves 100% performance on VideoMME, 98.9% on LongVideoBench, 97.7% on MLVU, and 96.7% on MVBench, making it quite attractive for efficiency-oriented scenarios.

We note that, although the uncompressed STORM and its test-time sampling variant achieve the highest overall performance when using the same 32 frames input, their computational demands limit scalability to train on longer sequence. In contrast, as demonstrated in Table 2, the STORM + T. Pooling allows extending the temporal context to 128 frames with temporal pooling, which enables improved performance without requiring additional LLM computation.

Token Compression Improves Model Efficiency We provide analysis on model efficiency in Figure 4. In Figure 4 left, we compare the inference latency of STORM with and without token compression and across varying numbers of input frames. Due to the quadratic computation of the LLM, the latency gap between the two configurations widens significantly as the sequence length increases. Figure 4 middle provides a detailed breakdown of inference latency of different modules for processing 256 frames. Without token compression, approximately 80% of the total latency is attributed to the LLM module. By applying a token compression ratio of 4, the latency associated with the LLM is reduced to roughly 35% of its original value, demonstrating a substantial improvement in efficiency.

Importantly, the temporal projector enables token compression without compromising performance. In fact, for some benchmarks, it even improves model performance while reducing computation cost, as demonstrated in Table 2 and detailed in Table 9. Figure 4 right presents a detailed analysis using the temporal sampling method on the VideoMME benchmark. We compare three configurations: the VILA baseline, STORM without compression, and STORM with test-time temporal token sampling Using up to 128 input frames and a compression ratio of 2. The results indicate that: (1) The VILA baseline's performance improves up to 32 frames but declines beyond this point, (2) STORM generalizes better to longer sequences, maintaining performance up to 64 frames before a slight drop, and (3) STORM with temporal sampling continues to improve performance up to 128 frames while providing a 50% token compression ratio, meaning STORM + T. Sampling at 128 frames uses the same number of visual tokens as the other two models at 64 frames.

We speculate that phenomenon is because of the limitation of the pre-trained LLM models which, despite having large context lengths on paper, operate with a smaller effective context length in practice. This limitation hinders their performance on very long sequences. The combination of the temporal

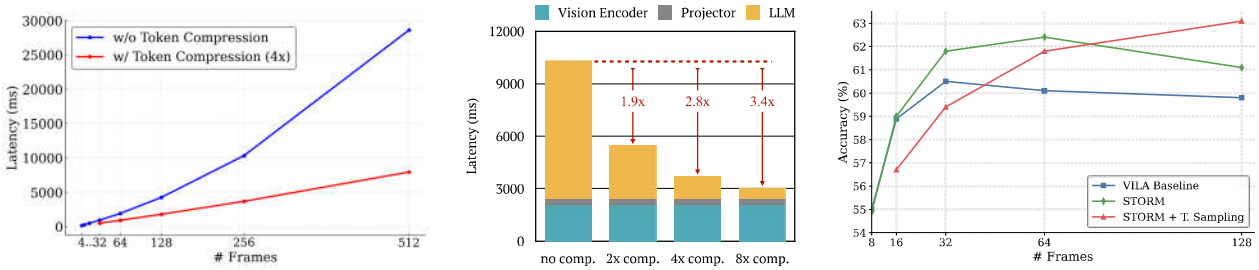


Figure 4 | **Model Efficiency and Effectiveness on Long Video Inputs. (left)** Profiling results of token compression as the number of frames increases during inference. **(middle)** Profiling results for 256 input frames with different compression ratios on a single A100. **(right)** The accuracy of Video-MME (without subtitles) across different numbers of frames during inference. While STORM with test-time temporal sampling showed consistent performance improvements, both VILA and STORM without token compression demonstrated decreased performance beyond 64 frames.

Q: Whom is the poem in the video written by?

Ours: The poem in the video is written by Edgar Allan Poe.

Q: How many times do news segments appear in this video? The best answer is:
A. 2. B. 4. C. 6. (GT answer) D. 8.

Ours: C. 6.

Q: What is unique about the last performance? The best answer is:
A. The last performance features background dancers, distinguishing it from the others.
B. The stage lighting is noticeably dimmer in the last performance compared to the others.
C. Musical instruments are incorporated into the last performance, unlike the others. (GT answer)
D. The last performance showcases a collective group of performers, while the others consist of individual acts.

Ours: C. Musical instruments are incorporated into the last performance, unlike the others.

Figure 5 | **Qualitative Examples of STORM + T. Pooling.** Our model effectively processes complex video content across various tasks requiring fine-grained temporal and visual understanding while reducing computational overhead through efficient token compression. The example videos can be found in our website.

projector and temporal sampling ensures that the number of tokens fed into the LLM remains within its effective context length. Simultaneously, the enriched visual tokens retain comprehensive temporal information beyond the sampled frames.

Mamba Module Has Minimal Training Overhead The Mamba projector introduces minimal training overhead: even when no compression is applied, it adds only $\sim 5\%$ latency in full training (19.1 hours with VILA (without Mamba module) and 20.1 hours with STORM (with Mamba module)). More importantly, since the Mamba module enables effective token compression, it in fact provides significant training / inference speedups and performance gains (e.g., +3.6% on MLVU with 4× fewer tokens), eventually reducing training the costs.

Our Token Compression Retains Visual Information The qualitative results in Figure 5 demonstrate that STORM's token compression preserves critical visual information while significantly reducing computational overhead. Even with 4× compression ratio, our model accurately extracts task-relevant information across diverse video understanding tasks. For example, questions like 'Whom is the poem in the video written by?', 'How many times do news segments appear in this video?', and 'What is unique about the last performance?' require fine-grained visual information and focus on the specific content from the prompt. Detailed category-level analysis on VideoMME (Figure 7) also reveals that STORM + token compression consistently outperforms baseline model across nearly all task categories. Even in OCR-heavy tasks that require detailed visual analysis, our compressed model maintains performance comparable to uncompressed versions while using only a fraction of the computational resources (only 25% visual tokens). These findings demonstrate that our Mamba-based temporal encoder enables effective token compression by encoding spatiotemporal relationships directly into visual tokens, allowing the model to maintain high-level understanding while drastically reducing token count. Additional qualitative results are provided in Figure 10-13.

Our Modules Generalize to Various Architectures Our core design, temporal module + token compression, is model-agnostic and can be integrated into various video-LLM architectures. Table 6 shows the experiments on models utilizing different LLM models (Qwen2 [38] vs Llama3 [53]), vision encoders (PaliGemma [37] vs SigLip [8]), model sizes (8B vs 8B vs 1.5B), and input resolutions (448×448 vs 384×384).

Architectures	Resolution	Latency (ms)		VideoMME	
		VILA	Ours	VILA	Ours
Qwen2 7B + PaliGemma	448×448	1980	**926**	59.7	**61.2**
Llama3 8B + SigLip	384×384	1560	**724**	54.6	**56.8**
Qwen2 1.5B + SigLip	384×384	724	**486**	49.6	**52.6**

Table 6 | **Performance Comparison across Model Architectures, Model Sizes, and Input Resolutions.** We provide results of benchmark VideoMME, without subtitles version.

Models	32F (T. Pooling)	128F (T. Pooling)
VILA	58.9	61.7
Uni-dir STORM	**62.2**	62.5
Bi-dir STORM	61.2	**63.4**

Table 7 | **Support for Streaming/Online Settings.** We evaluate a uni-directional variant of STORM designed for streaming video inputs. Results show that the Uni-dir STORM consistently outperforms the VILA baseline, highlighting the potential of our design to support streaming scenarios.

Our design shows consistent improvements in both performance and latency for all configurations, clearly showcasing the universality of the model.

Support for Streaming/Online Settings. To evaluate the applicability of our method in streaming scenarios, we replaced the default bi-directional Mamba with a uni-directional variant, allowing the model to reuse prior states for constant-time computation as new frames arrive. The results are provided in Table 7. We find that both uni-directional and bi-directional variants significantly outperform the baseline without temporal modeling. Notably, the uni-directional Mamba performs competitively and even slightly surpasses the bi-directional counterpart when trained on 32-frame inputs. On the other hand, the bi-directional model demonstrates stronger performance as video length increases. These results highlight the critical role of the Mamba module and suggest its potential in enabling future designs that support streaming video input for video-LLMs.

6. Conclusion

We introduced STORM, a novel Video-LLM model that enhances long-video understanding using a Mamba-based temporal encoder and efficient token reduction. By explicitly integrating spatiotemporal dynamics into visual tokens early in the pipeline, STORM enables significant token compression while preserving critical information in the compressed inputs. Experiments demonstrate that STORM achieves new state-of-the-art results on long-video

understanding benchmarks while substantially improving computational efficiency.

7. Acknowledgments

We would like to thank Xin Dong, Matthieu Le, Yunhao Fang, and Dacheng Li for their valuable discussions on this work. Sungjin Ahn was supported by the NRF of Korea funded by the Ministry of Science and ICT (MSIT) (No. 2021H1D3A2A03103645).

References

[1] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 2024.

[2] Ji Lin, Hongxu Yin, Wei Ping, Yao Lu, Pavlo Molchanov, Andrew Tao, Huizi Mao, Jan Kautz, Mohammad Shoeybi, and Song Han. Vila: On pretraining for visual language models. *CVPR*, 2024.

[3] 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.

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

[5] Yi Wang, Kunchang Li, Yizhuo Li, Yinan He, Bingkun Huang, Zhiyu Zhao, Hongjie Zhang, Jilan Xu, Yi Liu, Zun Wang, et al. Internvideo: General video foundation models via generative and discriminative learning. *arXiv preprint arXiv:2212.03191*, 2022.

[6] Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng, Jilan Xu, Zun Wang, et al. Internvideo2: Scaling video foundation models for multimodal video understanding. *arXiv preprint arXiv:2403.15377*, 2024.

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 2021.

[8] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023.

[9] Fuzhao Xue, Yukang Chen, Dacheng Li, Qinghao Hu, Ligeng Zhu, Xiuyu Li, Yunhao Fang, Haotian Tang, Shang Yang, Zhijian Liu, Yihui He, Hongxu Yin, Pavlo Molchanov, Jan Kautz, Linxi Fan, Yuke Zhu, Yao Lu, and Song Han. Longvila: Scaling long-context visual language models for long videos. *arXiv preprint arXiv: 2408.10188*, 2024.

[10] 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.

[11] 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. *arXiv preprint arXiv: 2410.17434*, 2024.

[12] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024.

[13] Kunchang Li, Xinhao Li, Yi Wang, Yinan He, Yali Wang, Limin Wang, and Yu Qiao. Videomamba: State space model for efficient video understanding. In *European Conference on Computer Vision*, pages 237–255. Springer, 2024.

[14] Jinyoung Park, Hee-Seon Kim, Kangwook Ko, Minbeom Kim, and Changick Kim. Videomamba: Spatio-temporal selective state space model. In *European Conference on Computer Vision*, pages 1–18. Springer, 2024.

[15] Hui Lu, Albert Ali Salah, and Ronald Poppe. Videomambapro: A leap forward for mamba in video understanding. *arXiv preprint arXiv:2406.19006*, 2024.

[16] Guo Chen, Yifei Huang, Jilan Xu, Baoqi Pei, Zhe Chen, Zhiqi Li, Jiahao Wang, Kunchang Li, Tong Lu, and Limin Wang. Video mamba suite: State space model as a versatile alternative for video understanding. *arXiv preprint arXiv:2403.09626*, 2024.

[17] Shufan Li, Harkanwar Singh, and Aditya Grover. Mamba-nd: Selective state space modeling for multi-dimensional data. In *European Conference on Computer Vision*, pages 75–92. Springer, 2024.

[18] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 2022.

[19] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan

Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. In *International Conference on Machine Learning*. PMLR, 2023.

[20] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35, 2022.

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

[22] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. *European Conference on Computer Vision*, 2023.

[23] Yuetian Weng, Mingfei Han, Haoyu He, Xiaojun Chang, and Bohan Zhuang. Longvlm: Efficient long video understanding via large language models. In *European Conference on Computer Vision*, pages 453–470. Springer, 2025.

[24] Xubing Ye, Yukang Gan, Xiaoke Huang, Yixiao Ge, Ying Shan, and Yansong Tang. Voco-llama: Towards vision compression with large language models. *arXiv preprint arXiv:2406.12275*, 2024.

[25] Michael S. Ryoo, Honglu Zhou, Shrikant Kendre, Can Qin, Le Xue, Manli Shu, Silvio Savarese, Ran Xu, Caiming Xiong, and Juan Carlos Niebles. xgen-mm-vid (blip-3-video): You only need 32 tokens to represent a video even in vlms. *arXiv preprint arXiv: 2410.16267*, 2024.

[26] Yan Shu, Peitian Zhang, Zheng Liu, Minghao Qin, Junjie Zhou, Tiejun Huang, and Bo Zhao. Video-xl: Extra-long vision language model for hour-scale video understanding. *arXiv preprint arXiv: 2409.14485*, 2024.

[27] Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

[28] Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, et al. An empirical study of mamba-based language models. *arXiv preprint arXiv:2406.07887*, 2024.

[29] Junxiong Wang, Daniele Paliotta, Avner May, Alexander Rush, and Tri Dao. The mamba in the llama: Distilling and accelerating hybrid models. *Advances in Neural Information Processing Systems*, 37:62432–62457, 2024.

[30] Jingwei Zuo, Maksim Velikanov, Dhia Eddine Rhaiem, Ilyas Chahed, Younes Belkada, Guillaume Kunsch, and Hakim Hacid. Falcon mamba: The first competitive attention-free 7b language model. *arXiv preprint arXiv:2410.05355*, 2024.

[31] Md Mohaiminul Islam, Tushar Nagarajan, Huiyu Wang, Gedas Bertasius, and Lorenzo Torresani. Bimba: Selective-scan compression for long-range video question answering. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29096–29107, 2025.

[32] Daniel Y. Fu, Tri Dao, Khaled Kamal Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models. 2023.

[33] Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. 2023.

[34] Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning (ICML)*, 2024.

[35] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[36] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2740–2755, 2018.

[37] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[38] 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.

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

[40] Peng Jin, Ryuichi Takanobu, Caiwan Zhang, Xiaochun Cao, and Li Yuan. Chat-univi: Unified visual representation empowers large language models with

image and video understanding. *Computer Vision and Pattern Recognition*, 2023.

[41] Lin Chen, Xilin Wei, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Bin Lin, Zhenyu Tang, et al. Sharegpt4video: Improving video understanding and generation with better captions. *arXiv preprint arXiv:2406.04325*, 2024.

[42] Yuanhan Zhang, Bo Li, haotian Liu, Yong jae Lee, Liangke Gui, Di Fu, Jiashi Feng, Ziwei Liu, and Chunyuan Li. Llava-next: A strong zero-shot video understanding model, April 2024.

[43] Xidong Wang, Dingjie Song, Shunian Chen, Chen Zhang, and Benyou Wang. Longllava: Scaling multimodal llms to 1000 images efficiently via hybrid architecture. *arXiv preprint arXiv: 2409.02889*, 2024.

[44] Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. mplug-owl3: Towards long image-sequence understanding in multi-modal large language models. *arXiv preprint arXiv: 2408.04840*, 2024.

[45] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

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

[47] Bo Zhao, Boya Wu, Muyang He, and Tiejun Huang. Svit: Scaling up visual instruction tuning. *ArXiv preprint*, abs/2307.04087, 2023.

[48] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data, 2024.

[49] 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. *Computer Vision and Pattern Recognition*, 2023.

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

[51] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *arXiv preprint arXiv: 2407.15754*, 2024.

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

[53] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[54] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Austin Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *arXiv preprint arXiv: 2406.16860*, 2024.

[55] Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? *arXiv preprint arXiv: 2405.02246*, 2024.

[56] Ruohong Zhang, Liangke Gui, Zhiqing Sun, Yihao Feng, Keyang Xu, Yuanhan Zhang, Di Fu, Chunyuan Li, Alexander Hauptmann, Yonatan Bisk, and Yiming Yang. Direct preference optimization of video large multimodal models from language model reward. *arXiv preprint arXiv: 2404.01258*, 2024.

[57] Bin Zhu, Bin Lin, Munan Ning, Yang Yan, Jiaxi Cui, Hongfa Wang, Yatian Pang, Wenhao Jiang, Junwu Zhang, Zongwei Li, Wancai Zhang, Zhifeng Li, Wei Liu, and Liejie Yuan. Languagebind: Extending video-language pretraining to n-modality by language-based semantic alignment. *International Conference on Learning Representations*, 2023.

[58] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. MSR-VTT: A large video description dataset for bridging video and language. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 5288–5296. IEEE Computer Society, 2016.

[59] Jonathan Krause, Justin Johnson, Ranjay Krishna, and Li Fei-Fei. A hierarchical approach for generating descriptive image paragraphs. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3337–3345. IEEE Computer Society, 2017.

[60] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. Sharegpt4v: Improving large multi-modal models with better captions. *arXiv preprint arXiv: 2311.12793*, 2023.

[61] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In

2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 1988–1997. IEEE Computer Society, 2017.

[62] Ryota Tanaka, Kyosuke Nishida, and Sen Yoshida. Visualmrc: Machine reading comprehension on document images. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13878–13888. AAAI Press, 2021.

[63] Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 9127–9134. AAAI Press, 2019.

[64] Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Just ask: Learning to answer questions from millions of narrated videos. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 1666–1677. IEEE, 2021.

[65] Jianhao Shen, Ye Yuan, Srbuhi Mirzoyan, Ming Zhang, and Chenguang Wang. Measuring vision-language stem skills of neural models, 2024.

[66] Kushal Kafle, Brian L. Price, Scott Cohen, and Christopher Kanan. DVQA: understanding data visualizations via question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5648–5656. IEEE Computer Society, 2018.

[67] Ali Furkan Biten, Rubèn Tito, Andrés Mafla, Lluís Gómez i Bigorda, Marçal Rusiñol, C. V. Jawahar, Ernest Valveny, and Dimosthenis Karatzas. Scene text visual question answering. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 4290–4300. IEEE, 2019.

[68] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision (ECCV)*, 2022.

[69] Jimmy Carter. Textocr-gpt4v. https://huggingface.co/datasets/jimmycarter/textocr-gpt4v, 2024.

[70] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[71] Paul Lerner, Olivier Ferret, Camille Guinaudeau, Hervé Le Borgne, Romaric Besançon, José G. Moreno, and Jesús Lovón-Melgarejo. Viquae, a dataset for knowledge-based visual question answering about named entities. In Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai, editors, *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 3108–3120. ACM, 2022.

[72] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M. F. Moura, Devi Parikh, and Dhruv Batra. Visual dialog. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1080–1089. IEEE Computer Society, 2017.

[73] Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6700–6709. Computer Vision Foundation / IEEE, 2019.

[74] Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2263–2279, Dublin, Ireland, 2022. Association for Computational Linguistics.

[75] Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing Hong, Jianhua Han, Hang Xu, Zhenguo Li, and Lingpeng Kong. G-llava: Solving geometric problem with multi-modal large language model, 2023.

[76] Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. Mitigating hallucination in large multi-modal models via robust instruction tuning, 2023.

[77] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions, 2016.

[78] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209, 2021.

[79] Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric Xing, and Liang Lin. GeoQA: A geometric question answering benchmark towards multimodal numerical reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 513–523, Online, August 2021. Association for Computational Linguistics.

[80] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. OK-VQA: A visual question answering benchmark requiring external knowledge. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 3195–3204. Computer Vision Foundation / IEEE, 2019.

[81] Aniruddha Kembhavi, Michael Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. *ArXiv*, abs/1603.07396, 2016.

[82] Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *ArXiv preprint*, abs/2306.15195, 2023.

[83] Tianyu Yu, Jinyi Hu, Yuan Yao, Haoye Zhang, Yue Zhao, Chongyi Wang, Shan Wang, Yinxv Pan, Jiao Xue, Dahai Li, et al. Reformulating vision-language foundation models and datasets towards universal multimodal assistants. *arXiv preprint arXiv:2310.00653*, 2023.

[84] Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. Aligning large multi-modal model with robust instruction tuning. *arXiv preprint arXiv:2306.14565*, 2023.

[85] Fuxiao Liu, Xiaoyang Wang, Wenlin Yao, Jianshu Chen, Kaiqiang Song, Sangwoo Cho, Yaser Yacoob, and Dong Yu. MMC: Advancing multimodal chart understanding with large-scale instruction tuning. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1287–1310, Mexico City, Mexico, 2024. Association for Computational Linguistics.

[86] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024.

[87] Krishna Srinivasan, Karthik Raman, Jiecao Chen, Michael Bendersky, and Marc Najork. Wit: Wikipedia-based image text dataset for multimodal multilingual machine learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21. ACM, July 2021.

[88] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. *ICCV*, 2018.

8. Appendix

8.1. Qualitative Results

We present comprehensive qualitative evaluations in Figure 9 to Figure 13, which are segmented into three subsections

1. **Effective Long Video Understanding**: Demonstrating STORM's ability to effectively utilize long video inputs by comparing it with existing long-video LLMs.
2. **Importance of Long Video Context**: Highlighting the need for long video inputs by showcasing scenarios where 128-frame inputs (with token compression) enable accurate predictions, whereas 32-frame inputs fail.
3. **Showcase of Video Understanding Abilities**: Illustrating STORM's capabilities in various aspects such as OCR, spatial perception, temporal reasoning, and so on.

Effective Long Video Understanding. We compare our proposed STORM + Temporal Sampling with LongVILA and LongVU, both designed for long video understanding. We use a short film depicting a "moonfall disaster" from the VILA webpage [1]. The models are prompted to provide a narrative description of the video. The short film was chosen for its engaging and dramatic storyline that spans various interconnected scenarios, all contributing to a cohesive narrative. Understanding this video requires the models to comprehend each individual scene and effectively integrate temporal events to grasp the complete story. Both STORM and LongVILA use 128 input frames, while LongVU output was obtained from its online demonstration which uses 1fps input.

As shown in Figure 9, STORM delivers the most detailed and coherent summary of the video's narrative, effectively capturing key events and transitions throughout the entire film. Its response showcases a comprehensive understanding of the content, highlighting its ability to connect temporal events across different scenes. In contrast, the baseline models LongVILA and LongVU focus on some of the events but fail to cover all critical moments that contribute to the overall storyline. Their responses also highlight specific scenes without integrating them into the full context. Moreover, we observed that the baseline models often generate redundant content, repeating the same sentences with minimal new information, which reveals their limitations in handling open-ended queries. Notably, our STORM with Temporal Sampling is also computationally more efficient. By applying temporal sampling, we reduce the number of tokens to the equivalent of processing 32 frames. This comparison showcases STORM's superior ability to leverage long video inputs for in-depth visual understanding.

[1] https://vila.mit.edu/

Importance of Long Video Context. We further demonstrate the significance of incorporating long video context by providing qualitative examples where a 128-frame input yields more accurate predictions than a 32-frame input, as shown in Figure 10. Using samples from the VideoMME benchmark, we compare two configurations of our STORM: one with a 32-frame input without compression, and another with a 128-frame input employing a temporal sampling ratio of 4. In both settings, the number of tokens fed into the LLM remains the same; however, the STORM with temporal sampling encodes additional information into the compressed tokens due to the extended frame sequence.

The inclusion of more frames allows the model to capture richer temporal dynamics and contextual information. For example, the 128-frame input enables the model to develop a stronger understanding of the video's narrative (Figure 10 top). It also allows the 128-frame model to capture additional events that the 32-frame model misses (Figure 10 center). Finally, the additional information further improve model's ability to reason through different temporal events across the entire video to form a coherent understanding (Figure 10 bottom). These example demonstrate the crucial role of long video context in tasks that require detailed temporal reasoning and comprehensive content understanding.

Showcase of Video Understanding Abilities. Finally, we conclude our qualitative evaluation by showcasing the diverse video understanding capabilities of STORM, including OCR, attribute perception, spatial perception, information synopsis, and temporal reasoning. Results are shown in Figure 11 to Figure 13. We use the same setting of STORM + Temporal Sampling with 128-frame input and sampling ratio of 4. Utilizing videos from the VideoMME benchmark, we designed a more challenging assessment to thoroughly evaluate the model's proficiency. Instead of providing the model with multiple-choice questions accompanied by predefined answer options, we transformed these tasks into open-ended queries that require the model to generate answers in raw text form without any given choices. This modification significantly increases the task's difficulty, as it

demands a precise understanding of the content and the ability to accurately locate and extract specific information from the video input.

Our qualitative results demonstrate that STORM provides strong performance in these scenarios. Despite the increased complexity, the model effectively interprets intricate visual details, recognizes textual information within videos, and provides coherent summaries of temporal events. This showcases STORM's robust ability to handle various aspects of video understanding.

8.2. Additional Results

Ablation on Token Budget and Token Compression Strategies. Table 9 extends Table 5 in the main text by providing a comprehensive comparison of different compression method combinations across various token budgets during training. Overall, considering both compression ratio and inference latency, we find that STORM with temporal pooling (STORM + T. Pooling) is the most efficient and effective approach. Additionally, test-time temporal sampling offers a lossless way to further enhance inference efficiency in inference time.

Task-Level Analysis on VideoMME Table 12 shows the VideoMME results with different video lengths. The short is less than 2 minutes, the medium is up to 15 minutes, and the long is up to 60 minutes. Overall, our STORM with token compression outperforms the VILA baseline and STORM with no token compressions for all video lengths. Figure 7 compares the VideoMME results by task categories. We find that STORM with temporal pooling especially improves the object reasoning task accuracy, and STORM with test-time temporal sampling improves the attribute perception accuracy. Both token compression methods improve the temporal perception task accuracies compared to VILA and STORM. It indicates that the temporal perception task requires a longer video context, and our token compression methods are effective for such tasks.

Effect of Dataset Composition Table 11 shows how dataset composition affects model performance during 128 frames fine-tuning. We compare using the full LLaVA-Video dataset ($\sim$ 1.35M samples) versus only its longest 25% videos with at least 128 frames ($\sim$ 360K samples). Interestingly, while STORM improves with the larger dataset across all benchmarks, the baseline model actually performs worse on several benchmarks when trained on the full dataset.

Two key differences between these datasets are size and video length distribution, where the full dataset contains more data but with a mixture of short and long videos, whereas the long-video subset exclusively consists of longer videos. Since larger, more diverse datasets typically improve performance (assuming similar data quality), we attribute the baseline model's unexpected performance drop to its limited ability to generalize from shorter to longer videos. More specifically, when trained predominantly on shorter clips from the full dataset, the baseline overfits and can not effectively handle long contexts at inference time. Training solely on longer videos dataset variant better aligns with test conditions, partially addressing this limitation.

In contrast, STORM shows consistent performance gains in all benchmarks when trained on the larger and more diverse data set. This suggests that STORM is more robust in handling longer sequences and is capable of using a wide range of video lengths to enhance its overall performance.

8.3. Architecture Details

STORM is built on a standard multimodal pipeline but introduces key modifications for improved reasoning ability and token efficiency. Figure 6 illustrates the detailed composition of our models. Instead of an MLP projector, STORM uses a linear layer followed by a Mamba-based temporal module projector which integrates spatiotemporal information into visual tokens.

STORM incorporates three main components: (1) The Mamba-Based Temporal Projector captures and propagates spatiotemporal information within visual tokens. (2) **Temporal Token Compression Module** applies compression on temporal dimension using training-based average pooling and/or training-free sampling (applied only at test time). (3) **Spatial Token Compression** further reduces token number by performing training-based frame-level spatial average pooling. Both spatial and temporal compression methods—whether training-free or training-based—are independently applicable. Notably, spatial and temporal pooling can be applied in parallel after the Mamba module, while temporal sampling is performed separately at test time. These components enable STORM to process longer sequences more efficiently before passing them to the LLM.

8.4. Datasets Information

For all models, we begin with an alignment stage to align the multi-modal projector using the LLaVA-CC3M-Pretrain-595K dataset [1]. Following this, we proceed to the visual instruction fine-tuning stage,

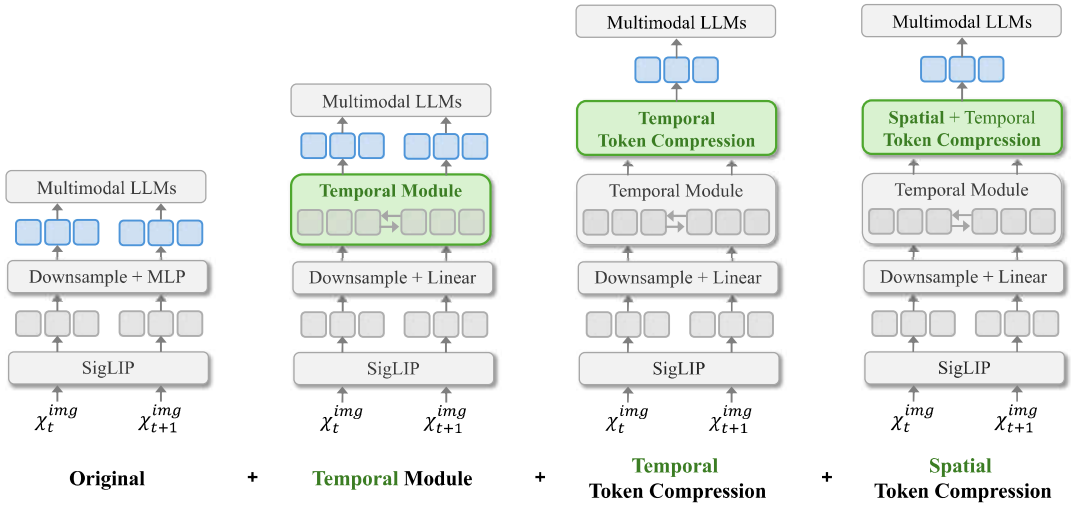


Figure 6 | **Breaking Down the STORM Architecture.** We begin with a standard multimodal pipeline that uses a pixel-shuffle downsampling layer and an MLP projector. In STORM, we replace the MLP with a linear layer and introduce our Mamba-based temporal module on top. Since the Mamba layer propagates spatiotemporal information in each visual tokens, the model can then perform temporal and spatial token compression of these tokens before passing them to the LLM, allowing STORM to handle longer sequences more efficiently.

experimenting with two different training data mixtures. These SFT mixtures incorporate both image and video data, encompassing three task types: captioning, open-ended question answering, and multiple-choice question answering. Further details are provided in the following:

- **SFT Data:** For most of our main experiments, we construct an expanded mixture by incorporating additional high-quality image datasets—such as Cambrian-1375K [54], Idefics2-SFT [55], and LLaVA-OneVision-Images-SFT [45]—along with video datasets including M4-Instruct-Video [56] and Youtube [57]. This enlarged dataset is used to scale up training and enhance overall performance. Detailed compositions of these mixtures are provided in Table 8.
- **Long Video SFT Data:** In the long video fine-tuning stage, our goal is to adapt models initially trained on full SFT data with 32-frame inputs to handle 128-frame inputs. Because processing 128-frame input incurs significant computational cost, we reduce training time by using a smaller dataset at this stage. Specifically, we select videos from only the LLaVA-Video dataset [48] that contains data amount roughly 11% of the full dataset (approximately 1.35M video-text pairs).
- **Long Video 25% Data:** As an ablation study to investigate the impact of dataset composition in the long video fine-tuning stage, we introduce an additional dataset derived from the LLaVA-Video dataset [48]. This subset consists exclusively of long videos with at least 128 frames, comprising approximately 25% of the full dataset (around 360K video-text pairs). Unlike the Long Video SFT Data, which includes both short and long videos, this dataset contains only long videos. Our experiments in Section 8.2 and Table 11 reveal distinct behaviors in our models and baselines in the composition of the dataset.

8.5. Mamba Temporal Module Latency

In this section, we compare the latencies of the vanilla VILA architecture and STORM across varying numbers of frames without token compression and provide a breakdown of the percentage contribution of the multi-modal projector. All experiments are conducted on a single NVIDIA DGX A100-80G. The results, shown in Figure 8, demonstrate that STORM incurs negligible overhead compared to the vanilla VILA architecture, with the introduced Mamba Temporal Module accounting for no more than 3% of the total latency.

8.6. Inference Details

Table 15 summarizes the number of frames used for inference. We evaluate all models between 8 and 512 frames and select the number of frames with the best accuracy overall for each task and setup.

Datasets
LLaVA-SFT [1], Idefics2-SFT [55]
MSR-VTT [58], Image Paragraph Captioning [59], ShareGPT4V-100K [60]
CLEVR [61], NLVR, VisualMRC [62]
ActivityNet-QA [63], LLaVA-OneVision-Images-SFT [45],
iVQA [64], MSRVTT-QA, STEM-QA [65]
DVQA [66], ST-VQA [67], SynthDoG-en [68], TextOCR-GPT4V [69], MTWI
ScienceQA-train [70], VQAv2-train, ViQuAE [71], Visual Dialog [72],
GQA-train [73], ChatQA [74], Geo170K [75],
LRV-Instruction [76], RefCOCO-train [77],
DocVQA [78], GeoQA [79], KVQA [80], Cambrian-1375K [54]
AI2D [81], Shikra [82], Unimm-Chat [83]
LRV-Instruction [84], SVIT [47], MMC-Instruction [85],
M4-Instruct-Images [86], M4-Instruct-Video [56] , WIT [87], Youtube [57], etc

Table 8 | SFT data mixture.

Models	Size	Comp. Ratio (%)	Latency (s)	#Frames (train)	# Frames (test)	MVBench test	MLVU dev	LongVideoBench val	VideoMME (w/o sub.)
Duration						16 sec	3∼120 min	8 sec∼60 min	1∼60 min
Token Budget: 8K									
VILA Baseline	7B	100	4.31	32	256	69.5	70.2	55.9	60.1
STORM	7B	100	4.47	32	256	70.3	71.1	54.5	62.5
STORM + S. Pooling	7B	25	1.82	128	256	63.9	67.9	54.5	57.5
STORM + T. Pooling	7B	25	1.82	128	256	**71.3**	72.5	59.5	**63.4**
STORM + T. Sampling *	7B	50	2.50	32	256	70.1	70.8	54.8	**63.1**
STORM + S. Pooling + T. Sampling *	7B	12.5	1.51	128	256	65.2	68.3	55.0	57.6
STORM + T. Pooling + T. Sampling *	7B	12.5	1.51	128	256	70.6	**72.9**	**60.5**	62.4
Token budget: 2K									
STORM + S. Pooling	7B	25	1.82	32	256	68.9	69.2	56.0	61.1
STORM + T. Pooling	7B	25	1.82	32	256	**70.4**	71.0	54.2	61.2
STORM + S. Pooling + Sampling *	7B	12.5	1.51	32	256	68.9	69.5	56.3	60.9
STORM + T. Pooling + Sampling *	7B	12.5	1.51	32	256	68.9	69.5	56.3	60.9
Token budget: 0.5K									
STORM + S. Pooling + T. Pooling	7B	6.25	1.36	32	256	68.5	68.2	53.7	60.2

* 2x additional compression at test time.

Table 9 | **Ablation on Token Budget and Token Compression Strategies.** Both spatial and temporal pooling are with 4× compression. The number of frames used during testing is consistent across models but can differ across tasks. The # frames is the maximum number of frames during testing. We summarize the details of the number of frames for each task in Table 15. The temporal token sampling is with 2× additional compression.

8.7. STORM vs Other Temporal Fusion Strategies.

In this section, we present early explorations of temporal fusion strategies and show that the simple Temporal-Pooling design used in our final model is surprisingly effective when combined with the Mamba module. Specifically, we experimented with more complex fusion approaches, including TSM [35] and SlowFast [88], using LLaMA3-8B + SigLip. TSM incorporates temporal information by shifting tokens from neighboring frames into the image-based visual encoder. On the other hand, SlowFast encodes video using two token streams: one with high temporal but low spatial resolution, and the other with the opposite configuration.

The results of these experiments are shown in Table 14. These evaluations were conducted during the early stages of our study, and as such, the number of input frames was not consistently matched across variants, making exact comparisons difficult. However, the settings are generally favorable to the TSM and SlowFast variants, as they use the same or more frames than the baseline, which does not perform any temporal fusion. Despite this, both fusion methods fail to yield meaningful improvements over the VILA baseline. In contrast, our STORM design achieves a significant gain, outperforming all other variants.

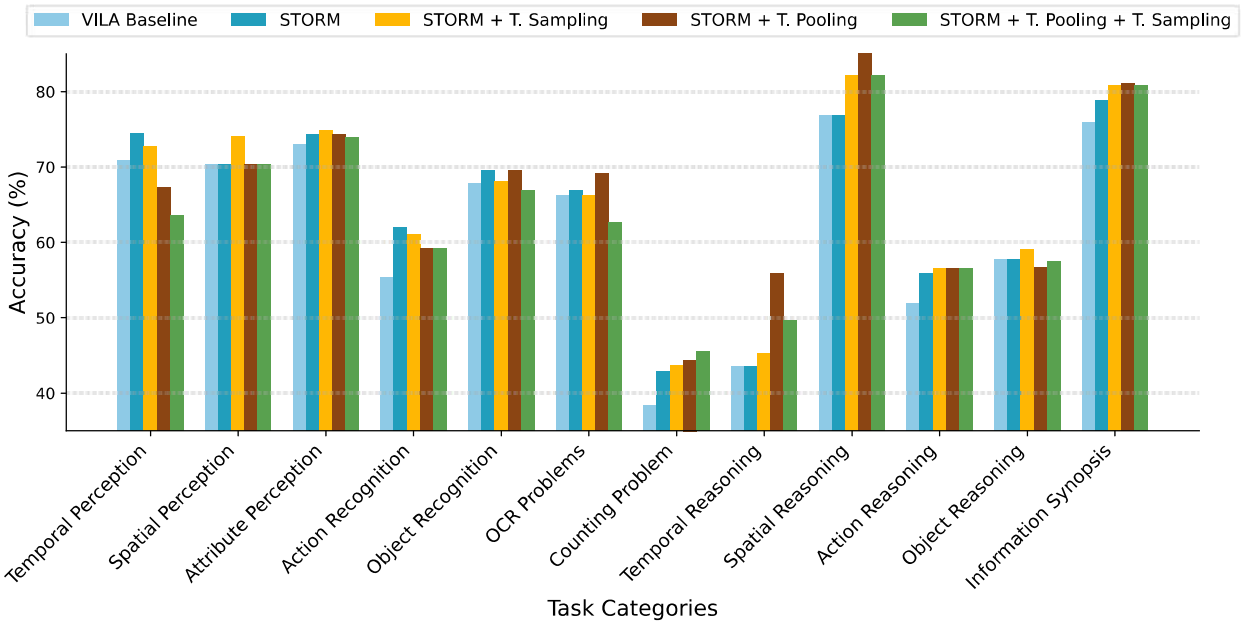


Figure 7 | **VideoMME Results by Task Categories.**

Models	8F	32F (T. Pooling)	128F (T. Pooling)
MVBench			
VILA (w/o Mamba)	67.9	68.7	68.1
STORM (w/ Mamba)	**68.8**	**70.4**	**71.3**
MLVU			
VILA (w/o Mamba)	**67.7**	**71.0**	69.9
STORM (w/ Mamba)	66.8	**71.0**	**72.5**
LongVidBench			
VILA (w/o Mamba)	**52.4**	**55.4**	57.4
STORM (w/ Mamba)	50.6	54.2	**59.5**
VideoMME			
VILA (w/o Mamba)	60.0	58.9	61.7
STORM (w/ Mamba)	**60.2**	**61.2**	**63.4**
Avg			
VILA (w/o Mamba)	**62.0**	63.5	64.3
STORM (w/ Mamba)	61.6	**64.2**	**66.7**

Table 10 | **Detailed Comparison across Benchmarks for Table 4.** STORM consistently improves performance across all benchmarks as the input video length increases from 8F to 32F to 128F. In contrast, the baseline VILA exhibits diminishing gains with longer inputs and even experiences performance degradation on certain benchmarks when extending from 32F to 128F. These results highlight the critical role of the Mamba module in effectively leveraging long-video inputs to enhance model performance.

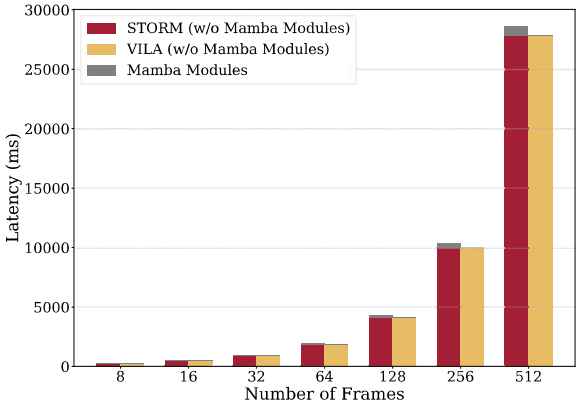


Figure 8 | Latency Comparison: VILA vs STORM. The multi-modal projector in VILA is a 2-layer MLP, while it is the Mamba Temporal Module in STORM.

Models	Dataset type	MVBench test	MLVU dev	LongVideoBench val	VideoMME (w/o sub.)
Duration		16 sec	3∼120 min	8 sec∼60 min	1∼60 min
VILA Baseline + T. Pooling	long-video only (25% of full)	67.2	71.4	59.2	62.2
VILA Baseline + T. Pooling	full LLaVA-Video [48]	68.1 (+0.9)	69.9 (-1.5)	57.4 (-1.8)	61.7 (-0.5)
VILA Baseline + T. Pooling + T. Sampling *	long-video only (25% of full)	64.5	71.4	59.2	61.0
VILA Baseline + T. Pooling + T. Sampling *	full LLaVA-Video [48]	67.8 (+3.3)	70.1 (-1.3)	57.7 (-1.5)	59.3 (-1.7)
STORM + T. Pooling	long-video only (25% of full)	69.4	71.7	57.6	63.2
STORM + T. Pooling	full LLaVA-Video [48]	**71.3** (+1.9)	72.5 (+0.8)	59.5 (+1.9)	**63.4** (+0.2)
STORM + T. Pooling + T. Sampling *	long-video only (25% of full)	68.8	72.7	59.2	62.6
STORM + T. Pooling + T. Sampling *	full LLaVA-Video [48]	70.6 (+1.8)	**72.9** (+0.2)	**60.1** (+0.9)	62.4 (-0.2)

* 2x additional compression at test time.

Table 11 | **Effect of Dataset Composition on 128-frame Fine-Tuning.** We compare models trained on two variants: the "full LLaVA-Video" dataset [48] (∼1.35M video-text pairs) versus the "long-video only" subset using top 25% longest videos (minimum of 128 frames) (∼360K pairs). Values in parentheses show performance differences between training on the long-video subset vs the full dataset. STORM consistently benefits from the larger, more diverse dataset across benchmarks, while the baseline VILA model degrades on several benchmarks when trained on the full dataset.

Models	# frames (train)	Short < 2 min	Medium 4∼15 min	Long 30∼60 min	Avg.
Token Budget: 8K					
VILA Baseline	32	73.0	58.0	49.2	60.1
STORM	32	**75.6**	60.9	51.1	62.5
STORM + T. Sampling*	32	75.2	60.8	53.2	63.1
STORM + T. Pooling	128	72.4	**64.4**	**53.4**	**63.4**
STORM + T. Pooling + T. Sampling*	128	72.9	60.9	**53.4**	62.4

* 2x additional compression at test time.

Table 12 | **Breakdown of VideoMME Results by Input Video Length.**

# Frames	Compression Ratio	Overall (ms)	llm (ms)	vision_tower (ms)	mm_projector (ms)
4	1	162.92	103.80	52.41	6.71
8	1	270.87	174.61	85.11	11.15
16	1	486.37	321.73	144.47	20.17
32	1	933.99	623.41	269.49	41.09
64	1	1910	1310	515.17	82.41
128	1	4270	3090	1020	163.34
256	1	10340	7960	2030	348.22
512	1	28620	23710	4090	811.31
32	4	486.97	175.75	269.96	41.26
64	4	920.10	322.22	515.82	82.06
128	4	1800	622.29	1010	163.23
256	4	3680	1310	2020	348.52
512	4	7950	3080	4060	811.84
64	8	772.18	175.27	514.59	82.32
128	8	1500	322.73	1020	163.09
256	8	3000	622.56	2030	348.71
512	8	6200	1310	4070	815.08

Table 13 | **Full Latencies on Various Compression Ratios and Input Frames.**

Q: Please provide a description of the narrative of the video.

- **Ours:** The video begins with a person playing with a ball on a rooftop, which is disrupted by a large, glowing moon-like object in the sky. The person looks up in awe as the moon-like object grows larger and more detailed. The scene transitions to a close-up of the moon-like object, which then explodes, causing chaos on the rooftop. The person and a child run away from the explosion, and the scene shifts to a view of the moon-like object in space, surrounded by debris. The person and the child watch the moon-like object on a television, which then explodes, causing a massive fireball. The video then shows a cityscape with buildings exploding and debris flying everywhere, followed by a view of a city under attack with spaceships and debris in the sky. The scene transitions to a chaotic cityscape with buildings collapsing and debris flying everywhere. A person in a red shirt is seen running through the debris, followed by a view of a city under attack with spaceships and debris in the sky. The scene shifts to a close-up of a person in a spacesuit inside a spacecraft, looking out at the chaotic scene outside. The video concludes with a black screen displaying the text 'BUY TICKETS NOW IN THEATERS AND IMAX FRIDAY' along with a rating of 'PG-13' and the logo of the production company, Lionsgate.
- **LongVU:** The video starts with a scene of a man throwing a ball to another man who hits it back. The ball then flies towards the camera and hits the lens. The scene then shifts to a man looking through binoculars as a flock of birds flies overhead. The man then throws a ball that hits a bird, which then falls to the ground. The scene then shows a man throwing a ball that hits a bird, which then explodes into pieces. The video then shows a man throwing a ball that hits a bird, which then explodes into pieces…
- **LongVILA:** The video begins with a man in black short sleeves throwing a ball at a target on a rooftop, followed by a scene of a large explosion. Subsequently, a man in a spacesuit is seen flying through the air, transitioning to a scene of a car driving on a road. The video then shows a man and a boy on a balcony, the man pointing at the sky. The scene shifts to a large white object in the sky, with a man and a boy observing birds, and then to a man and a woman on the balcony. The video continues with a man and a boy on a staircase, the man holding a rope, and a large object falling from the sky. The video then shows a large object falling, a man and a boy on a balcony, and a man and a woman looking at birds. The scene changes to a man and a boy on a balcony, a man holding a wooden box, and a large object falling from the sky…

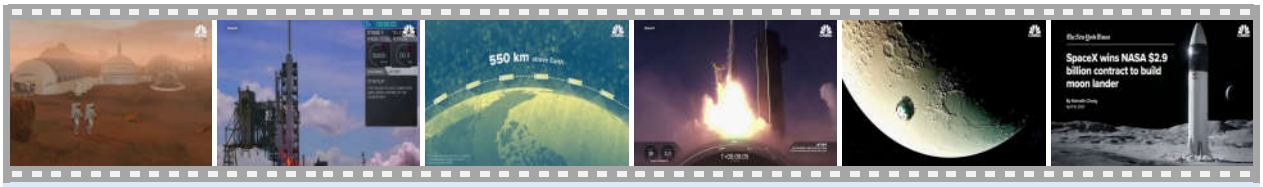
Figure 9 | **Effective Long Video Understanding.** We compare STORM + Temporal Sampling with existing long video LLMs. Reults show that STORM delivers a more detailed and coherent summary, effectively capturing key events and transitions throughout the film. The example videos can be found in our website.

Table 14 | Comparison of temporal fusion strategies.

	16 frames		32-64 frames	
Baseline	Baseline (T.pool)	TSM (T.pool)	SlowFast (T.pool)	STORM (T.pool)
52.0	50.0	49.0	51.3	**56.8**

Models	MVBench	MLVU	LongVidBench	VideoMME
8-frame-models	8	64	32	64
+ Temporal Sampling	16	256	256	128
32-frame-models	16	64	64	64
+ Temporal Sampling	32	256	256	128
32-frame-models + Temporal Pooling	32	64	128	64
+ Temporal Sampling	64	256	256	128

Table 15 | **The Number of Frames Used for Inference.** We evaluate all models for [8, 16, 32, 64, 128, 256, 512] frames and select the best overall for each task and setup.



Q: What does this video tell? The best answer is:
A. The process of building a starship. B. Why Starship is the holy grail for SpaceX.
C. Why Starlink is crucial to SpaceX's success. D. How SpaceX could Win The Space Race. **(GT answer)**

- **STORM (32-Frame Input, 8K Visual Tokens, No Compression) :** B. Why Starship is the holy grail for SpaceX.
- **STORM +Token Compression (128-Frame Input, 8K Visual Tokens, 4x Compression) :** D. How SpaceX could Win The Space Race.

Q: How many times do news segments appear in this video? The best answer is:
A. 2. B. 4. C. 6. **(GT answer)** D. 8.

- **STORM (32-Frame Input, 8K Visual Tokens, No Compression) :** B. 4.
- **STORM +Token Compression (128-Frame Input, 8K Visual Tokens, 4x Compression) :** C. 6.

Q: What is unique about the last performance? The best answer is:
A. The last performance features background dancers, distinguishing it from the others.
B. The stage lighting is noticeably dimmer in the last performance compared to the others.
C. Musical instruments are incorporated into the last performance, unlike the others. **(GT answer)**
D. The last performance showcases a collective group of performers, while the others consist of individual acts.

- **STORM (32-Frame Input, 8K Visual Tokens, No Compression) :** D. The last performance showcases a collective group of performers, while the others consist of individual acts.
- **STORM +Token Compression (128-Frame Input , 8K Visual Tokens, 4x Compression) :** C. Musical instruments are incorporated into the last performance, unlike the others.

Figure 10 | **Importance of Long Video Context.** We compare STORM with a 32-frame input to STORM + Temporal Sampling using a 128-frame input. Both configurations have negligible differences in computational cost; however, the latter encodes additional information into compressed tokens due to the extended frame sequence. The examples illustrate that processing more frames allows the model to capture richer temporal dynamics and contextual information. This leads to a stronger understanding of the video's narrative, reduces information loss, and enhances the ability to reason through temporal events across the entire video. The example videos can be found in our website.

a. OCR Problem

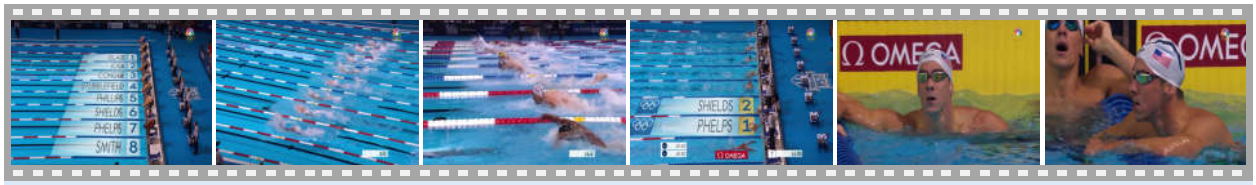
Q: Whom is the poem in the video written by?

Ours: The poem in the video is written by Edgar Allan Poe.

Q: What are the red texts recognized on the white wall?

Ours: The red texts recognized on the white wall are 'GO BOWLING'.

b. Attribute Perception



Q: Which country does the swimmer who was given a close-up in the video hail from?

Ours: The swimmer who was given a close-up in the video hails from the USA.

Q: What color are his gloves in the video?

Ours: His gloves are black.

Figure 11 | **Showcase of Video Understanding Abilities in Various Task Categories.** We provide additional examples to showcase model's video understanding capabilities in different aspects. This is done by providing the models with open-ended queries that require the model to generate answers in raw text form without any given choices. Part 1. The example videos can be found in our website.

c. Spatial Perception

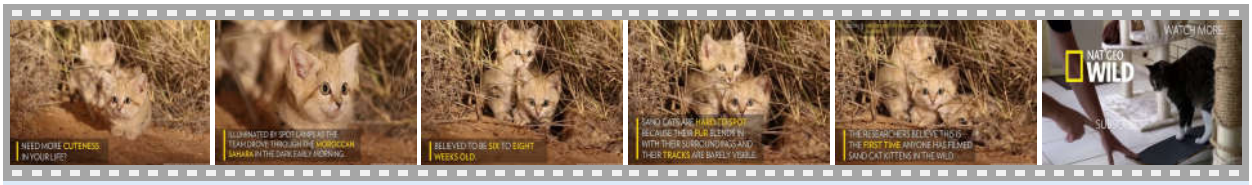
Q: Where is this video most likely shot?

Ours: The video is most likely shot inside a birdhouse.

Q: Which hand did the person in the video wear a glove on?

Ours: The person wore a glove on their right hand.

d. Information Synopsis



Q: According to the video, why is it very difficult to capture footage of sand cats?

Ours: Because their fur blends in with their surroundings and their tracks are barely visible.

Q: What is this video mainly about?

Ours: This video is mainly about demonstrating how to fold a shirt. The man in the video shows how to fold a shirt in a step-by-step manner, and he also demonstrates how to fold a t-shirt and a polo shirt.

Figure 12 | **Showcase of Video Understanding Abilities in Various Task Categories.** Continue 2. The example videos can be found in our website.

e. Temporal Reasoning

Q: What is the fourth-to-last news item in this news video?
A. Josh Liendo swims into the history books. B. Coming soon | Rising demand for pet psychics.
C. U.S. vice president calls for Gaza ceasefire. D. California storm drops 2 meters of snow.

Ours: A. Josh Liendo swims into the history books.

Q: In which order do the six tips are introduced in the video?
(a) Clip coupons. (b) Eat at home. (c) Freeze leftovers. (d) Cook once, eat twice. (e) Meal plan. (f) Buy in bulk.
A. b e d f a c. B. b e f c a d. C. e c d a b f. D. c e b f a d.

Ours: A. b e d f a c.

Figure 13 | **Showcase of Video Understanding Abilities in Various Task Categories.** Continue 3. The example videos can be found in our website.