PVC: Progressive Visual Token Compression for Unified Image and Video Processing in Large Vision-Language Models

Chenyu Yang[1,3]*, Xuan Dong[1,3]*, Xizhou Zhu[1,2]*, Weijie Su[2]*, Jiahao Wang[3], Hao Tian[3], Zhe Chen[2,4], Wenhai Wang[2,5], Lewei Lu[3], Jifeng Dai[1,2]✉

[1]Tsinghua University [2]OpenGVLab, Shanghai AI Laboratory [3]SenseTime Research
[4]Nanjing University [5]The Chinese University of Hong Kong

Abstract

Large Vision-Language Models (VLMs) have been extended to understand both images and videos. Visual token compression is leveraged to reduce the considerable token length of visual inputs. To meet the needs of different tasks, existing high-performance models usually process images and videos separately with different token compression strategies, limiting the capabilities of combining images and videos. To this end, we extend each image into a "static" video and introduce a unified token compression strategy called Progressive Visual Token Compression (PVC), where the tokens of each frame are progressively encoded and adaptively compressed to supplement the information not extracted from previous frames. Video tokens are efficiently compressed with exploiting the inherent temporal redundancy. Images are repeated as static videos, and the spatial details can be gradually supplemented in multiple frames. PVC unifies the token compressing of images and videos. With a limited number of tokens per frame (64 tokens by default), spatial details and temporal changes can still be preserved. Experiments show that our model achieves state-of-the-art performance across various video understanding benchmarks, including long video tasks and fine-grained short video tasks. Meanwhile, our unified token compression strategy incurs no performance loss on image benchmarks, particularly in detail-sensitive tasks. Code is released at https://github.com/OpenGVLab/PVC.

1. Introduction

Recently, large Vision-Language Models (VLM) [1, 3, 7, 8, 18, 24, 25, 38, 39, 48, 49, 52] have achieved remarkable success in various image-text tasks, demonstrating strong general capabilities. Further studies have extended these models to handle both images and videos. While videos can

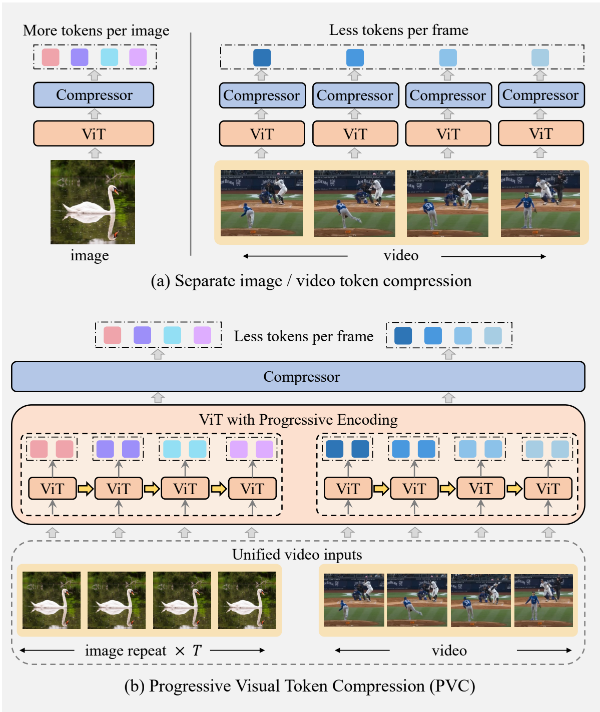

Figure 1. **Comparison of token encoding and compression in VLMs. (a)** Existing VLMs compress image and video tokens separately, leading to inconsistency: more tokens per image benefit image spatial perception, while videos tend to sacrifice some tokens per frame to accommodate more frames. **(b)** Our progressive compression (PVC) achieves unified compression of images and videos, allowing for the continuous supplementation of image details and temporal dynamic information in subsequent frames.

be seen as a natural extension of images over time, the additional temporal dimension poses new challenges. VLMs typically use visual tokens to represent visual information. A single image often requires hundreds or even thousands of tokens. If each frame of a video is encoded in the same way, a 100-frame video would generate over tens of thousands of tokens, causing a significant computational burden during training and inference. To address this, VLMs use a technique called *visual token compression* [7, 18, 52] to

*Equal contribution. This work is done when Chenyu Yang and Xuan Dong are interns at SenseTime Research. ✉Corresponding author: Jifeng Dai <daijifeng@tsinghua.edu.cn>.

reduce the number of tokens and lower computational complexity. The key challenge is finding a way to reduce the number of tokens while retaining essential information.

As shown in Fig. 1 (a), many high-performance VLMs usually process images and videos separately, using different visual token compression methods to meet the specific needs of each. For images, the models need to preserve fine-grained spatial details to support detail-intensive image tasks (e.g., OCR). For videos, current tasks often focus more on capturing dynamic changes over time and require fewer spatial details. This allows for fewer tokens per frame, sacrificing some spatial details to accommodate more frames. For example, Qwen2-VL [52] and LLaVA-OneVision [18] reduce the number of tokens per frame for videos compared to tokens per image to handle more frames. However, treating images and videos separately can hinder their combined capabilities, such as limiting the ability of video to recognize spatial details.

An ideal visual token compression method should support unified image and video processing, effectively capturing spatial details and temporal changes. To achieve this, as illustrated in Fig. 1 (b), we extend each image into a static video and introduce the concept of "progressive compression" for videos: the current frame only encodes new information not extracted from previous frames. This effectively reduces representation redundancy and lowers the number of tokens required for each frame. For the *video* modality, adjacent frames are often similar and naturally have temporal redundancy. Encoding each frame independently results in repeated information and wasted tokens. By applying progressive compression, token redundancy is minimized, and temporal changes can be captured efficiently. For the *image* modality, a single image is repeated into multiple frames to form a static video. Progressive compression can then be applied to these frames, allowing spatial details to be gradually supplemented in subsequent frames. This approach unifies the processing of images and videos. Even with a limited number of tokens per frame, spatial details and temporal changes can be preserved.

Specifically, our proposed ***P**rogressive **V**isual Token **C**ompression (PVC)* strategy is achieved with the following architectural designs: (1) We standardize VLM inputs as videos. Each image is repeated multiple times (4 by default) to form a "static" video. (2) A *progressive encoding module* is introduced to the vision encoder. This module consists of a causal temporal attention to avoid redundant information and progressively extract information complementary to the representations of previous frames. An AdaLN operator injected with timestep information is adopted to differentiate different frames and avoid redundant encoding. (3) An *adaptive compression module* is introduced to compress the visual tokens without redundancy. Based on PixelShuffle [46] with shared MLP (i.e., the token compression method used by the open-source VLM InternVL [7, 49]), we incorporate an additional AdaLN operator to extract different spatiotemporal information at different time steps.

We evaluate our model on a wide range of multi-modal benchmarks. Our model achieves state-of-the-art results across various video understanding benchmarks, including long video tasks (e.g., VideoMME [13], MLVU [68]) and fine-grained short video tasks (e.g., MVBench [22]). Meanwhile, on image understanding benchmarks, especially for detail-intensive tasks (e.g., DocVQA [36], InfoVQA [37]), our method incurs no performance loss. This demonstrates that unifying visual inputs into video format and leveraging temporal-based token encoding and compression enables large VLMs to better handle diverse visual tasks.

Our contributions can be summarized as follows:

- To address the limitations of separate image and video processing in existing VLMs, we propose *Progressive Visual Token Compression (PVC)* that unifies visual inputs as videos, which allows spatial details and temporal dynamics to be preserved across different modalities.
- We introduce a novel *progressive encoding module* and an *adaptive compression module* to progressively encode complement information in visual tokens. This effectively minimizes temporal redundancy while capturing essential spatiotemporal information, allowing a lower token-per-frame in our model.
- With the proposed progressive token compression, our model achieves state-of-the-art results across long video and fine-grained short video benchmarks. Meanwhile, it incurs no performance loss on image benchmarks, particularly in detail-sensitive tasks.

2. Related Work

Vision-Language Models for Images and Videos. In the realm of large vision-language models (VLMs), proprietary models [1, 38, 39, 48] and open-source models [3, 7, 8, 18, 19, 24, 25, 49, 52, 64] have demonstrated exceptional performance across a variety of visual scenarios, including single-image, multi-image, and video contexts. Large VLMs typically represent an image with hundreds or thousands of visual tokens, but the additional temporal dimension in videos leads to a considerable number of tokens, posing a great challenge. Some methods [26, 53, 65] extent the context length of the large language model to process more tokens, but this causes a huge computational burden during training and inference. Other methods uses token compression techniques to reduce the number of tokens [18, 31, 52]. For instance, LLaVA-OneVision [18] spatially down-samples the video tokens by 2×2; Oryx MLLM [31] designs a dynamic compressor for different types of inputs, where videos are down-sampled to a smaller scale; Qwen2-VL [52] leverages pixel shuffle and 3D convs to down-sample vision tokens spatially and temporally, and reduces the resolution of video frames for long videos.

As mentioned above, existing methods usually compress image and video tokens with different strategies to meet the needs of different tasks, such as detailed images and long videos. Furthermore, the simple spatial or temporal downsampling does not take full advantage of the temporal redundancy in videos, leading to an inefficient compression. Our approach, otherwise, unifies the inputs into videos and compresses tokens by exploiting the temporal redundancy.

Vision Transformer with Temporal Attention. Modern video encoders typically enhance the basic image encoder (e.g., Vision Transformer, ViT [10]) with temporal attention to account for the temporal correlations in videos. Video Swin Transformer [30] applies joint spatio-temporal attention within localized 3D windows. TimeSformer [5], ViViT [2], TESTA[42] and EVLGen [15] apply self-attention mechanism along the spatial and temporal dimensions, respectively. We also introduce an additional temporal attention module to ViT layers, but the attention is causal to realize progressive feature encoding. Besides, we replace the Layer Normalization with Adaptive Layer Normalization (AdaLN) injected with timestep conditions to avoid redundant encoding for different input frames.

Visual Token Compression. Visual token compression is commonly used in VLMs to reduce the number of vision tokens to fit in the context length of LLMs and reduce computational overhead. Applying pooling [33, 66], down-sampling [18, 58], or convolution [9] along the spatial and temporal dimensions is straightforward but may cause information loss. Many other works [3, 11, 21, 22, 27, 60, 63] employ Q-Former [20] that aggregates visual features into a fixed number of tokens with learnable queries. However, learning a cross-attention layer is more difficult, and the compressed tokens may lose spatial and temporal awareness. Current high-performance VLMs such as InternVL2 [49] and Qwen2-VL [52] leverage PixelShuffle that aggregates a local grid of tokens into a single one by concatenating along the feature dimension. Meanwhile, some works also devise token compression modules specifically for video inputs, for instance, spatio-temporal attentional pooling [43], and token selection based on the similarity with text queries [23, 45]. Nevertheless, these approaches are not applicable to general image and video understanding tasks. Our progressive compression is based on the PixelShuffle module and incorporates AdaLN to avoid redundant representation.

3. Method

3.1. Standardize Vision Inputs of VLMs as Videos

Current high-performance VLMs typically use different methods to compress image and video tokens. Due to limited computational budget, allocating more tokens per image is beneficial for understanding image spatial details, whereas video processing often involves sacrificing the number of tokens per frame to include more frames.

To address this dilemma, we standardize vision inputs of VLMs as videos, and apply the proposed progressive visual token compression for all videos. Specifically, each image is repeated into multiple frames to create a "static" video. Then, progressive compression allows LLMs to revisit the image multiple times and capture richer spatial information in subsequent static frames. This approach enables a unified visual token compression for images and videos, balancing the distinct needs of both modalities. Even when the number of tokens per frame is limited, the necessary spatial details and temporal dynamics can be gradually supplemented in subsequent frame tokens.

For each image x in the inputs, we repeat it T_{img} times to form a static frame sequence $[x, x, \ldots, x]$ (by default $T_{\text{img}} = 4$). For each native video input, we uniformly sample T frames, where T is randomly chosen from the range of $[16, 96]$ by default during training. In scenarios where dynamic resolution [7] is used, it is applied to every frame of both static and native videos.

3.2. Network Architecture

The architecture of our proposed PVC is outlined in Fig. 2, comprising a Vision Transformer (ViT) with Progressive Encoding, an Adaptive Token Compression module, and a Large Language Model (LLM). To fully leverage the redundancy between video frames, we integrate temporal attention mechanism into the ViT for progressive encoding. Additionally, the adaptive compression module is introduced to effectively reduce the number of visual tokens per frame.

Vision Transformer with Progressive Encoding. To avoid redundant encoding of video frames and give the current frame the potential to only encode new information not extracted from previous frames, we add a temporal attention module within the ViT. To minimize overhead, temporal attentions are applied only in the last $\tilde{L}$ layers of the ViT (with L layers in total). Let x be the video feature with dimensions [B, T, N, C], where B is the batch size, T is the number of frames, N is the number of patches per frame, and C is the hidden dimension. The ViT layer with progressive encoding is defined by:

$$x := x + \text{S-MHA}(\text{LayerNorm}(x)), \tag{1}$$

$$x := x + \alpha \odot \text{T-MHA}(\text{AdaLN}(x; x + \text{TE})), \tag{2}$$

$$x := x + \text{FFN}(\text{LayerNorm}(x)). \tag{3}$$

Spatial Multi-Head Attention (S-MHA) is the original multi-head self-attention [50] in ViT, which is applied on the patch tokens of each frame respectively. x is reshaped to dimensions [$B \times T$, N, C] when input into S-MHA, where N is the sequence length here.

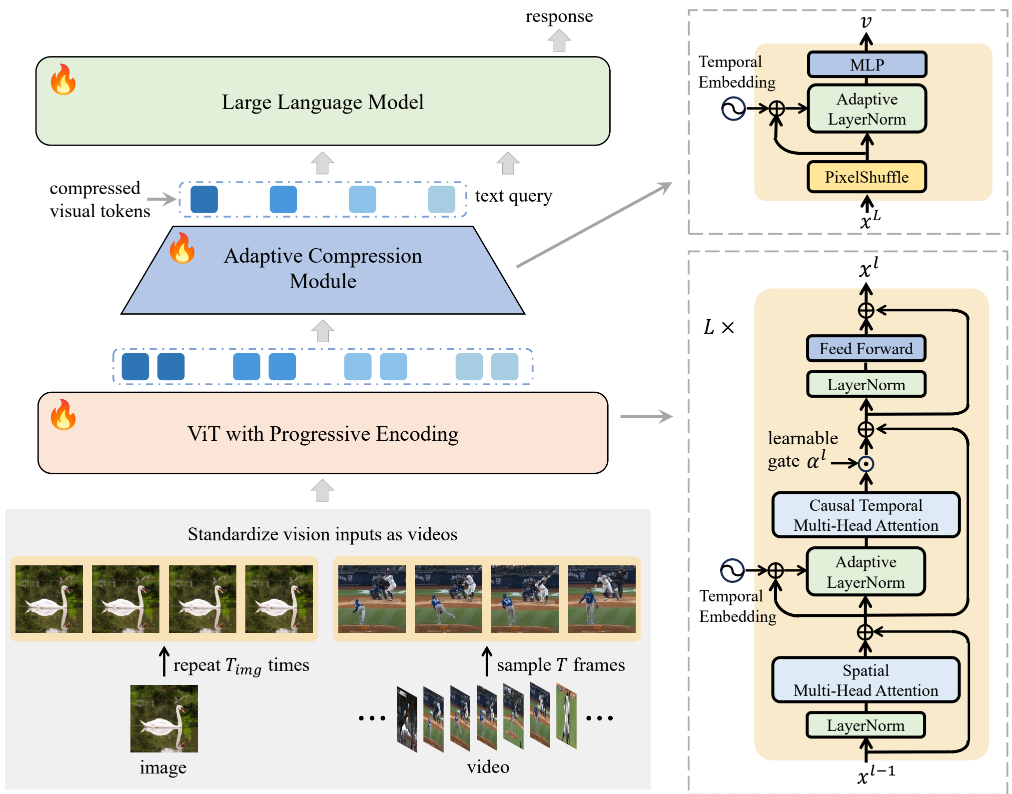

Figure 2. **Network architecture of progressive visual token compression (PVC).** The inputs are standardized as videos, with images repeated to form static videos. A causal temporal attention and an AdaLN layer are incorporated into the ViT layers to progressively encode visual tokens across timesteps. The adaptive compression module, based on PixelShuffle, includes an AdaLN layer to reduce redundancy in visual tokens.

Temporal Multi-Head Attention (T-MHA) is an additional *causal* multi-head self-attention [50] applied on the temporal dimension of the video feature. Each patch token attends to the tokens at the same spatial position in the previous frames. Specifically, x is rearranged to dimensions $[B \times N, T, C]$ when input into T-MHA, where T is the sequence length here.

Temporal Embedding (TE) is added to encode temporal position information. First, we use relative timestamps to indicate the video frames: $t = \left[0, \frac{1}{T-1}, \dots, \frac{T-2}{T-1}, 1\right]$. Then we encode all timestamps by a 256-dimension sinusoidal position embeddings [50], denoted as $\tilde{t} \in \mathbb{R}^{T \times 256}$. After that, we obtain the temporal embedding via an MLP:

$$\text{TE} = W_2 \cdot \text{SiLU}\left(W_1 \tilde{t}\right) \tag{4}$$

where W_1 and W_2 are learnable parameters, SiLU refers to Sigmoid Linear Unit [14].

Adaptive Layer Normalization (AdaLN) was first proposed by [41] and is now widely used to adjust the normalization parameters according to given conditions, making the model more adaptable to different task requirements. In order to adapt to the "static" videos where different frames are the same, we use AdaLN here so that the network can extract different spatial information at different time steps and avoid redundant encoding. $\text{AdaLN}(x; z)$ normalizes the input x and applies an affine operation with the coefficients conditioned on z:

$$\text{AdaLN}(x; z) = \gamma(z) \odot \text{LayerNorm}(x) + \beta(z). \tag{5}$$

The affine scale $\gamma(z)$ and bias $\beta(z)$ are obtained via:

$$\gamma(z) = W_4 \cdot \text{SiLU}(W_3 z), \ \beta(z) = W_6 \cdot \text{SiLU}(W_5 z), \tag{6}$$

where W_3, W_4, W_5 and W_6 are learnable parameters.

Additionally, a *learnable gate* $\alpha \in \mathbb{R}^C$ is introduced to scale the output of T-MHA. It is zero-initialized to make sure that the ViT is initialized as the pre-trained checkpoint.

Adaptive Compression. To reduce the computational budget, current VLMs [7, 18, 52] commonly shorten the sequence length by compressing visual tokens. For instance, InternVL-1.5 [7] employs a PixelShuffle [46] operation [1] followed by an MLP, reducing the visual token length by a factor of 4. However, this relies on a shared MLP to compress different visual sequences, which hinders reducing the representation redundancy between different frames.

[1]Originally from the super-pixel area [46]; here, it refers to concatenating 4 tokens from each adjacent 2×2 region along the channel dimension to form a single token.

To this end, we introduce an adaptive compression module by integrating an AdaLN [41] layer before the shared MLP. The reason for using AdaLN here is the same as in the progressive encoding module, allowing the network to extract different spatiotemporal information at different time steps to avoid redundant representation. The input of token compression is the output video tokens x with dimensions $[B, T, N, C]$ from the ViT. It compresses the number of tokens per frame from N to M as follows:

$$\tilde{x} = \text{PixelShuffle}(x) \in \mathbb{R}^{B\times T\times M\times(\frac{N}{M}\cdot C)} \tag{7}$$

$$v = \text{MLP}(\text{AdaLN}(\tilde{x}; \tilde{x} + \text{TE})). \tag{8}$$

where AdaLN and TE is the same as mentioned above.

We set $M = \frac{1}{16}N$, and correspondingly, the kernel size for PixelShuffle is 4×4, *i.e.*, it aggregates a 4×4 grid of tokens into a single token by concatenating along the feature dimension. The compression ratio of 16 is higher than that of previous approaches (typically 4). For the image modality, we repeat each image $T_{\text{img}} = 4$ times by default, so the total number of visual tokens remains the same for image input. See Appendix C for the ablations of these settings.

3.3. Training Strategies and Data

Training Strategies. Following common practices from previous VLMs [3, 7, 18, 25, 49, 52], the training of PVC consists of two stages: the pre-training stage and the instruction-tuning stage. The main difference is that all parameters of PVC are jointly trained in both stages, without freezing ViT or LLM as in previous approaches [3, 7, 18, 25, 49, 52]. We empirically find that this strategy works better for PVC (see Appendix C for ablation experiments). Additionally, in the pre-training stage, we use a larger learning rate (10×) for randomly initialized parameters compared to pre-trained parameters.

Pre-training Data. The pre-training data used in PVC comprises two parts: (1) 45M image-text pairs used in InternVL2's [49] pre-training stage, including LAION [44], COYO [6], *etc*. (2) 23M video-text pairs, including InternVid [54], WebVid [4], *etc*. The sampling ratio of image-text to video-text data is set to 1:1.

Instruction Tuning Data. The instruction tuning data used in PVC consists of two parts: (1) 7.2M image-text and video-text data used in InternVL2's [49] fine-tuning stage. (2) 3.4M video-text data collected and refined from Vript [59], LLaVA-Video [67], *etc*. The sampling ratio of image-text to video-text data is set to 3:1.

The details of training data are provided in Appendix B.

4. Experiment

4.1. Experiment Settings

Our implementation mainly follows InternVL2 [49], referred to as $\text{PVC}_{\text{InternVL2}}$. The progressive encoding is introduced in the last 8 layers of the 24-layer ViT (*i.e.*, $\tilde{L} = 8$ and $L = 24$).Our model is pre-trained on 50M samples and fine-tuned for 1 epoch. More implementation details are provided in Appendix A.

Evaluation Benchmarks. The image-language benchmarks include AI2D [16], ChartQA [35], DocVQA [36], InfoVQA [37], SQA [32], TextVQA [47], MMB [29], MME [12], MMMU [62], SEED-I [17], and OCR-Bench [28]. For video-language benchmarks, we use MVBench [22], VideoMME [13], MLVU [68], LongVideoBench [56], NextQA [57], Egoschema [34], PercepTest [40], and ActNet-QA [61].

4.2. Comparison with State-of-the-art Methods

We compare our PVC with state-of-the-art VLMs within 8B parameters, including commercial VLMs [38, 39, 48], open-source general VLMs [18, 31, 49, 52, 64, 65], and open-source video VLMs [9, 11, 22, 26, 43, 45, 51, 55, 58, 67]. As shown in Tab. 1 and Tab. 2, PVC achieves superior performance on long video tasks (*e.g.*, VideoMME, MLVU) and fine-grained video tasks (*e.g.*, MVBench), while maintaining very competitive performance on image tasks.

Video-Language Benchmarks. The results on video-language benchmarks are presented in Tab. 1, which can be summarized as the following:

(1) Our $\text{PVC}_{\text{InternVL2}}$ utilizes 64 tokens per frame, fewer than most existing methods (*e.g.*, LLaVA-Video [67] uses 169 tokens, InternVL2 [49] uses 256 tokens), while it still achieves strong performance on all video tasks. In contrast, previous approaches such as LongVU [45] and Video-CCAM [11] also reduce tokens per frame to 64 but exhibit lower performance on fine-grained video tasks, *e.g.*, MVBench. This shows that our PVC can effectively maintain detailed information and exploit temporal redundancy.

(2) Our $\text{PVC}_{\text{InternVL2}}$-8B attains superior performance on MVBench, VideoMME, MLVU, and LongVideoBench, demonstrating its strong capability in both fine-grained and long video understanding. Notably, $\text{PVC}_{\text{InternVL2}}$-8B reaches 73.8 on MVBench, significantly surpassing the previous best accuracy of 69.1 among open-source models within 8B parameters, set by IXC-2.5 [64].

(3) Our $\text{PVC}_{\text{InternVL2}}$-8B also obtains the best results in NextQA and ActNet-QA among image-video general VLMs. On Egoschema and PercepTest, our model is inferior to some existing models, which may be caused by the lack of training data in some specific domains.

(4) For a fair comparison, we implement the "+ PVC arch. only" setting where InternVL2 is integrated with our PVC architecture and its training data remains unchanged. Our PVC outperforms InternVL2 by a large margin, especially for long video tasks (e.g., VideoMME and MLVU). Meanwhile, incorporating new video-text data leads to significant improvements, since high-quality video data is relatively

Table 1. **Video-language benchmark results**. Our proposed PVC is compared with image-video general VLMs and video VLMs. "LongVideo." refers to LongVideoBench. The results of VideoMME are reported as "without subscript/with subscript". The best results among image-video general VLMs are **bold**, and the best results among all open-source VLMs are underlined. * Native resolution is used without fixed # token/frame. † 1024 tokens in total, with 16 or 96 frames used for different tasks. "+ PVC arch. only": only introduce the model architecture of PVC in InternVL2 while keeping the training data unchanged.

Model	Size	# token /frame	MVBench	VideoMME	MLVU	LongVideo.	NextQA	Egoschema	PercepTest	ActNet-QA
Commercial VLMs:										
Gemini-1.5-Pro [48]	-	-	52.6	75.0/81.3	-	64.0	-	64.5	-	57.5
GPT-4V [38]	-	-	43.7	59.9/63.3	49.2	61.3	-	55.6	-	57.0
GPT-4o [39]	-	-	47.8	71.9/77.2	64.6	66.7	-	72.2	-	-
Video VLMs:										
Video-CCAM [11]	4B	64/10.7†	62.8	50.1/51.2	56.5	-	-	-	-	58.0
BLIP-3-Video [43]	4B	16	-	-	-	-	77.1	-	-	56.9
VideoChat2 [22]	7B	96	60.4	39.5/43.8	47.9	-	-	54.4	-	49.1
PLLaVA [58]	7B	144	46.6	-	-	-	-	54.5	-	56.3
VideoLLaMA2 [9]	7B	72	54.6	47.9/50.3	48.5	-	-	51.7	51.4	50.2
Tarsier [51]	7B	576	62.6	-	-	-	71.6	49.9	59.5	
LongVU [45]	7B	64-144	66.9	- /60.6	65.4	-	-	67.6	-	-
LLaVA-Video [67]	7B	169	58.6	63.3/69.7	70.8	58.2	83.2	57.3	67.9	56.5
Kangaroo [26]	8B	1024	61.1	56.0/57.6	61.0	54.8	-	62.7	-	-
InternVideo2 [55]	8B	96	67.2	-	-	-	-	60.0	63.4	-
Video-CCAM [11]	9B	64/10.7†	64.6	50.3/52.6	58.5	-	-	-	-	59.7
Image-video general VLMs:										
Qwen2-VL [52]	2B	-*	63.2	55.6/60.4	-	-	-	54.9	53.9	-
InternVL2 [49]	2B	256	60.2	45.0/47.3	50.4	-	-	46.7	49.9	-
+ PVC arch. only	2B	64	62.0	47.7/49.8	54.1	46.3	67.6	47.5	54.2	49.5
$\text{PVC}_{\text{InternVL2}}$ (Ours)	2B	64	69.4	54.5/56.7	63.4	51.6	73.4	49.8	62.9	52.8
LongVA [65]	7B	144	-	52.6/54.3	56.3	-	68.3	-	-	50.0
IXC-2.5 [64]	7B	400	69.1	55.8/58.8	58.8	-	71.0	-	34.4	52.8
LLaVA-OV [18]	7B	196	56.7	58.2/61.5	64.7	56.5	79.4	60.1	57.1	56.6
Oryx MLLM [31]	7B	-*	63.9	58.3/62.6	67.5	55.3	81.9	-	**68.6**	-
Qwen2-VL [52]	7B	-*	67.0	63.3/69.0	-	-	-	**66.7**	62.3	-
InternVL2 [49]	8B	256	66.4	54.0/56.9	52.0	-	-	55.0	52.0	-
$\text{PVC}_{\text{InternVL2}}$ (Ours)	8B	64	**73.8**	**64.1/69.7**	**72.4**	**59.2**	**82.0**	59.6	68.4	**57.1**

scarce in the original dataset of InternVL2.

Image-Language Benchmarks. The results on image-language benchmarks are presented in Tab. 2 and can be summarized as follows:

(1) In a fair comparison (InternVL2 vs. + PVC arch. only), our PVC does not compromise performance on image tasks, even on OCR-related tasks (*e.g.*, DocVQA, InfoVQA, *etc.*) that heavily depend on spatial details (as shown in Tab. 4, directly reducing the tokens per patch would significantly hurt performance on DocVQA and InfoVQA). This validates that PVC can progressively encode spatial information from repeated images and effectively compress the visual tokens to fully preserve spatial details. Meanwhile, appending new video-text pairs in our training data does not have significant impacts on these image tasks.

(2) Compared to Qwen2-VL-7B [52], our $\text{PVC}_{\text{InternVL2}}$-8B achieves comparable or better performance on AI2D (+0.8), ChartQA (+1.1), and MMB (+0.9), but lags behind on DocVQA (-2.0), InfoVQA (-1.5), TextVQA (-4.3), MME (-45), MMMU (-3.2), and OCRBench (-59). The benchmarks where Qwen2-VL excels are primarily OCR-related (4 out of 6), which are highly sensitive to OCR-specific training data. Given the differences in model architecture and train-

Table 2. **Image-language benchmark results**. We compare our PVC with VLMs that support both image and video inputs. * Native resolution is used without fixed # token/image tile. "+ PVC arch. only": only introduce the model architecture of PVC in InternVL2 while keeping the training data unchanged.

Model	Size	# token /image tile	AI2D test	ChartQA test	DocVQA test	InfoVQA test	SQA test	TextVQA val	MMB en-test	MME sum	MMMU val	SEED-I -	OCRBench -
Commercial VLMs:													
Gemini-1.5-Pro [48]	-	-	94.4	87.2	93.1	81.0	-	-	-	-	62.2	-	-
GPT-4V [38]	-	-	78.2	78.5	88.4	-	75.7	-	-	1926	56.8	49.9	-
GPT-4o [39]	-	-	94.2	85.7	92.8	-	-	-	83.4	2329	69.1	76.2	736
Image-video general VLMs:													
Qwen2-VL [52]	2B	-*	74.7	73.5	90.1	65.5	-	79.7	74.9	1872	41.1	-	809
InternVL2 [49]	2B	256	74.1	76.2	86.9	58.9	94.0	73.4	73.2	1877	34.3	71.6	784
+ PVC arch. only	2B	64	75.8	78.8	87.8	59.6	94.1	76.3	75.3	1967	35.6	72.8	791
PVC$_{\text{InternVL2}}$ (Ours)	2B	64	76.3	78.9	87.7	59.9	94.9	76.2	75.8	1960	36.9	73.2	796
LongVA [65]	7B	144	70.7	70.4	80.8	49.4	-	-	-	-	42.6	-	-
IXC-2.5 [64]	7B	400	81.5	82.2	90.9	69.9	-	78.2	82.2	2229	42.9	75.4	690
LLaVA-OV [18]	7B	729	81.4	80.0	87.5	68.8	96.0	-	-	1998	48.8	75.4	-
Oryx MLLM [31]	7B	-*	78.5	-	89.0	-	-	75.0	81.4	-	43.9	-	672
Qwen2-VL [52]	7B	-*	83.0	83.0	**94.5**	**76.5**	-	**84.3**	83.0	**2327**	**54.1**	-	**866**
InternVL2 [49]	8B	256	**83.8**	83.3	91.6	74.8	97.1	77.4	81.7	2210	49.3	76.2	794
PVC$_{\text{InternVL2}}$ (Ours)	8B	64	**83.8**	**84.1**	92.5	75.0	**97.7**	80.0	**83.9**	2282	50.9	**77.2**	807

ing data, a direct comparison is challenging. However, our PVC is orthogonal and complementary to the advances in model architecture and training data. The ablation studies in Tab. 4 show the effectiveness of our PVC.

Table 3. **Ablation of AdaLN conditions.** The ablation is conducted on 2B model with shortened pre-training on 10M samples.

AdaLN condition	MVBench	VideoMME	InfoVQA	MMB
-	59.2	46.4	54.0	73.7
TE	62.0	46.5	57.8	74.6
$x + \text{TE}$	62.4	46.7	58.5	74.9

4.3. Ablation Study

Tab. 4 ablates the key components of our PVC, where setting (a) is the baseline corresponding to InternVL2 . Tab. 3 ablates the conditions used in AdaLN layers. Ablation studies of the number of tokens per frame and the number of progressive encoding layers are provided in Appendix C.

Standardize Vision Inputs as Videos. The standardization reduces tokens per image from 256 to 64 while maintaining the overall number of visual tokens by repeating each image and increasing the number of video frames. As shown in settings (a) and (b) of Tab. 4, directly standardizing vision inputs as video significantly reduces performance on OCR-related image tasks (DocVQA -3.7 and InfoVQA -6.5) and fine-grained video tasks (MVBench -3.9). This performance drop occurs because repeating the image four times for the vanilla ViT and vision compression module only produces four groups of identical visual tokens, with repeated information encoded. Consequently, spatial details are lost compared to using 256 distinct visual tokens. However, long-video understanding tasks (VideoMME +1.1 and MLVU +2.5), which do not require as much visual detail, can benefit from the increase in video frames.

Progressive Encoding. Progressive encoding involves adding temporal causal attention in ViT. As shown in Tab. 4, settings (b) and (c) demonstrate that adding temporal attention in ViT mitigates the information loss observed in setting (b) and maintains comparable performance with setting (a) on tasks requiring spatial details (MVBench, DocVQA, and InfoVQA). Furthermore, temporal attention significantly boosts performance on long video tasks (MLVU +1.2 and VideoMME +0.5). These results validate that our proposed progressive encoding techniques can effectively encode both spatial details and temporal correlations.

Adaptive Compression. As shown in Tab. 4 setting (c) and PVC, the performance on all tasks is further improved by adding the adaptive compression module, especially on long video tasks (*e.g.*, MLVU +0.9). This demonstrates our proposed adaptive compression mechanism can better exploit representation redundancy within video frames and model long-term temporal correlations.

AdaLN Conditions. As shown in Tab. 3, without any condition, AdaLN degrades to LN where repeated frames are redundantly encoded, leading to poor results. Using tempo-

Table 4. **Ablation of key components in PVC.** The ablation study is conducted on 2B model with shortened pre-training on 10M samples. "Adapt. compress" refers to adaptive compression. VideoMME is tested under "without subscript" setting.

Settings	# tokens per image	# image repeat	# video frames	Progressive encoding	Adapt. compress	MVBench	VideoMME	MLVU	DocVQA	InfoVQA	MMB
(a)	256	1	16			61.9	44.9	50.5	86.4	59.1	75.1
(b)	64	4	64			58.0	46.0	53.0	82.7	52.6	73.1
(c)	64	4	64	✓		61.8	46.5	54.2	85.9	58.1	74.8
PVC	64	4	64	✓	✓	62.4	46.7	55.1	86.7	58.5	74.9

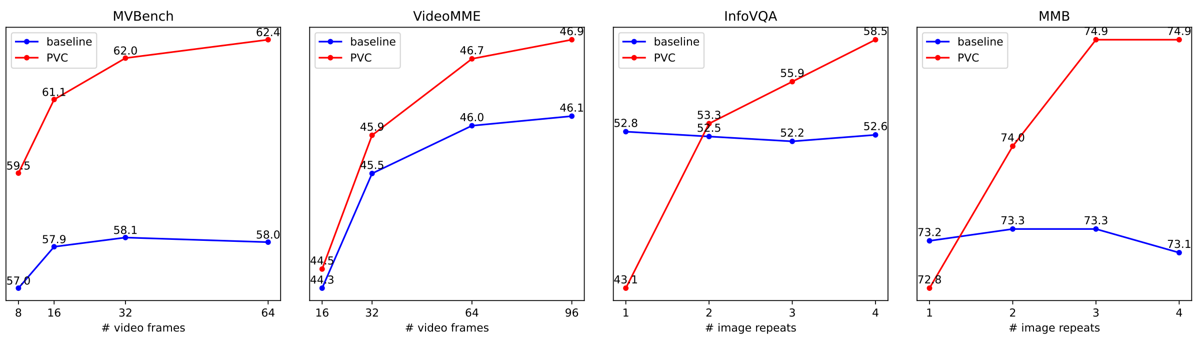

Figure 3. **Analysis of progressive compression.** We compare our PVC model with the baseline without progressive compression (setting (b)) in Tab. 4. For video tasks (MVBench and VideoMME), we test different number of input frames. For image tasks (InforVQA and MMB), we test different repetition times of the image.

ral embedding (TE) as the condition significantly boosts the performances. Adding the feature x provides further improvements. x contains temporal information aggregated from previous layers, which helps better extract complementary information and minimize redundancy.

4.4. Analysis

Effectiveness of Progressive Compression. We evaluate our model by gradually increasing the number of video frames and image repetitions. We use the model in Tab. 4 setting (b) (without progressive compression) as the baseline. The results are shown in Fig. 3.

For MVBench, which mainly contains videos shorter than one minute with fine-grained queries, the baseline model's score plateaus after increasing to 16 frames, while our model still gets better with complement information encoded. For the long-video task VideoMME, the scores of both models increase with the number of frames. Still, our model grows faster, indicating that it is better at removing temporal redundancy and capturing temporal dynamics.

On image tasks, increasing the number of image repetitions does not boost the baseline model's performance since its repeated visual tokens do not provide more information. On the detail-sensitive task InfoVQA, our model performs poorly when the image is repeated only once because only part of the information is extracted. When the image is repeated more times, the detailed information is supplemented, so the results get substantially better. For MMB, the images contain less spatial details, so the gain from repeating more times is relatively small for our model.

Table 5. **Speed of image repetition and progressive encoding.** We report the FLOPs and FPS (samples per second) for processing samples containing a 448×448 image and 2048 text tokens, measured on an NVIDIA A100.

Model	# tokens per image	# image repeats	Progressive encoding	FLOPs	FPS
InternVL2-8B	256	1		13.3T	4.8
$\text{PVC}_{\text{InternVL2}}$-8B	64	4	✓	14.1T	4.5
Δ_{relative}	-	-	-	+6.0%	-6.3%

Speed of Image Repetition and Progressive Encoding. As shown in Tab. 5, for the 8B model, repeating images introduces only a marginal overhead (+6.0% FLOPs relative, -6.3% FPS relative), since the visual feature computation can be partially reused and the main computational load is in the LLM. Thus, the inference speed of our $\text{PVC}_{\text{InternVL2}}$ on image tasks remains similar to that of InternVL2.

5. Conclusion

In conclusion, we introduce Progressive Visual Token Compression (PVC), a unified approach for effectively processing images and videos within large Vision-Language Models (VLMs). It standardizes visual inputs as videos, along with a progressive encoding module and an adaptive compression module. Extensive empirical analysis demonstrates that it can gradually capture both spatial details and temporal dynamics from the sequence of image repetitions and video frames. Our PVC not only achieves state-of-the-art performance on fine-grained short video tasks and long video tasks but also maintains accuracy on image benchmarks, particularly in detail-sensitive tasks.

Acknowledgment

This work is supported by the National Key R&D Program of China (NO. 2022ZD0161300), by the National Natural Science Foundation of China (62376134).

References

[1] Anthropic. Claude 3.5 sonnet, 2024. 1, 2

[2] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *ICCV*, pages 6836–6846, 2021. 3

[3] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 1(2):3, 2023. 1, 2, 3, 5

[4] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *ICCV*, pages 1728–1738, 2021. 5

[5] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, page 4, 2021. 3

[6] Minwoo Byeon, Beomhee Park, Haecheon Kim, Sungjun Lee, Woonhyuk Baek, and Saehoon Kim. Coyo-700m: Image-text pair dataset. https://github.com/kakaobrain/coyo-dataset, 2022. 5

[7] Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *arXiv preprint arXiv:2404.16821*, 2024. 1, 2, 3, 4, 5

[8] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *CVPR*, pages 24185–24198, 2024. 1, 2

[9] Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, et al. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv preprint arXiv:2406.07476*, 2024. 3, 5, 6

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3

[11] Jiajun Fei, Dian Li, Zhidong Deng, Zekun Wang, Gang Liu, and Hui Wang. Video-ccam: Enhancing video-language understanding with causal cross-attention masks for short and long videos. *arXiv preprint arXiv:2408.14023*, 2024. 3, 5, 6

[12] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, and Rongrong Ji. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023. 5

[13] Chaoyou Fu, Yuhan Dai, Yongdong 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. 2, 5

[14] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 4

[15] Yiren Jian, Tingkai Liu, Yunzhe Tao, Chunhui Zhang, Soroush Vosoughi, and Hongxia Yang. Expedited training of visual conditioned language generation via redundancy reduction. In *ACL*, pages 300–314, 2024. 3

[16] Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *ECCV*, pages 235–251, 2016. 5

[17] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023. 5

[18] 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. 1, 2, 3, 4, 5, 6, 7

[19] Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. *arXiv preprint arXiv:2407.07895*, 2024. 2

[20] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, pages 19730–19742, 2023. 3

[21] 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. 3

[22] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *CVPR*, pages 22195–22206, 2024. 2, 3, 5, 6

[23] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *ECCV*, pages 323–340, 2025. 3

[24] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *CVPR*, pages 26296–26306, 2024. 1, 2

[25] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *NeurIPS*, 36, 2024. 1, 2, 5

[26] 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. 2, 5, 6

[27] Ruyang Liu, Chen Li, Haoran Tang, Yixiao Ge, Ying Shan, and Ge Li. St-llm: Large language models are effective temporal learners. In *ECCV*, pages 1–18, 2025. 3

[28] Yuliang Liu, Zhang Li, Biao Yang, Chunyuan Li, Xucheng Yin, Cheng-lin Liu, Lianwen Jin, and Xiang Bai. On the

hidden mystery of ocr in large multimodal models. *arXiv preprint arXiv:2305.07895*, 2023. 5

[29] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *ECCV*, pages 216–233, 2025. 5

[30] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *CVPR*, pages 3202–3211, 2022. 3

[31] 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. 2, 5, 6, 7

[32] Pan Lu, Swaroop Mishra, Tanglin 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. *NeurIPS*, 35: 2507–2521, 2022. 5

[33] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *ACL*, pages 12585–12602, 2024. 3

[34] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. *NeurIPS*, 36:46212–46244, 2023. 5

[35] 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 *ACL*, pages 2263–2279, 2022. 5

[36] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *WACV*, pages 2200–2209, 2021. 2, 5

[37] Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. Infographicvqa. In *WACV*, pages 1697–1706, 2022. 2, 5

[38] OpenAI. Gpt-4v(ision) system card, 2023. 1, 2, 5, 6, 7

[39] OpenAI. Hello gpt-4o, 2024. 1, 2, 5, 6, 7

[40] Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Mateusz Malinowski, Yi Yang, Carl Doersch, et al. Perception test: A diagnostic benchmark for multimodal video models. *NeurIPS*, 36, 2024. 5

[41] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, pages 4195–4205, 2023. 4, 5

[42] Shuhuai Ren, Sishuo Chen, Shicheng Li, Xu Sun, and Lu Hou. Testa: Temporal-spatial token aggregation for long-form video-language understanding. In *EMNLP*, pages 932–947, 2023. 3

[43] 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. 3, 5, 6

[44] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *NeurIPS*, 35:25278–25294, 2022. 5

[45] Xiaoqian Shen, Yunyang Xiong, Changsheng Zhao, Lemeng Wu, Jun Chen, Chenchen Zhu, Zechun Liu, Fanyi Xiao, Balakrishnan Varadarajan, Florian Bordes, et al. Longvu: Spatiotemporal adaptive compression for long video-language understanding. *arXiv preprint arXiv:2410.17434*, 2024. 3, 5, 6

[46] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, pages 1874–1883, 2016. 2, 4

[47] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *CVPR*, pages 8317–8326, 2019. 5

[48] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: A family of highly capable multimodal models. *arXiv:2312.11805*, 2023. 1, 2, 5, 6, 7

[49] OpenGVLab Team. Internvl2: Better than the best—expanding performance boundaries of open-source multimodal models with the progressive scaling strategy, 2024. 1, 2, 3, 5, 6, 7

[50] A Vaswani. Attention is all you need. *NeurIPS*, 2017. 3, 4

[51] Jiawei Wang, Liping Yuan, Yuchen Zhang, and Haomiao Sun. Tarsier: Recipes for training and evaluating large video description models. *arXiv preprint arXiv:2407.00634*, 2024. 5, 6

[52] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. 1, 2, 3, 4, 5, 6, 7

[53] Xidong Wang, Dingjie Song, Shunian Chen, Chen Zhang, and Benyou Wang. Longllava: Scaling multi-modal llms to 1000 images efficiently via hybrid architecture. *arXiv preprint arXiv:2409.02889*, 2024. 2

[54] Yi Wang, Yinan He, Yizhuo Li, Kunchang Li, Jiashuo Yu, Xin Ma, Xinhao Li, Guo Chen, Xinyuan Chen, Yaohui Wang, et al. Internvid: A large-scale video-text dataset for multimodal understanding and generation. *arXiv preprint arXiv:2307.06942*, 2023. 5

[55] 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. 5, 6

[56] 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. 5

[57] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *CVPR*, pages 9777–9786, 2021. 5

[58] Lin Xu, Yilin Zhao, Daquan Zhou, Zhijie Lin, See Kiong Ng, and Jiashi Feng. Pllava: Parameter-free llava extension from images to videos for video dense captioning. *arXiv preprint arXiv:2404.16994*, 2024. 3, 5, 6

[59] Dongjie Yang, Suyuan Huang, Chengqiang Lu, Xiaodong Han, Haoxin Zhang, Yan Gao, Yao Hu, and Hai Zhao. Vript: A video is worth thousands of words. *arXiv preprint arXiv:2406.06040*, 2024. 5

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

[61] Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *AAAI*, pages 9127–9134, 2019. 5

[62] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *CVPR*, pages 9556–9567, 2024. 5

[63] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. In *EMNLP: System Demonstrations*, pages 543–553, 2023. 3

[64] Pan Zhang, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Rui Qian, Lin Chen, Qipeng Guo, Haodong Duan, Bin Wang, Linke Ouyang, et al. Internlm-xcomposer-2.5: A versatile large vision language model supporting long-contextual input and output. *arXiv preprint arXiv:2407.03320*, 2024. 2, 5, 6, 7

[65] 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. 2, 5, 6, 7

[66] Ruohong Zhang, Liangke Gui, Zhiqing Sun, Yihao Feng, Keyang Xu, Yuanhan Zhang, Di Fu, Chunyuan Li, Alexander Hauptmann, Yonatan Bisk, et al. Direct preference optimization of video large multimodal models from language model reward. *arXiv preprint arXiv:2404.01258*, 2024. 3

[67] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024. 5, 6

[68] 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. 2, 5