Elephant in the Fridge: Constant-Memory Frame Packing for Long Video Understanding

Shuo Gao
Renmin University of China

Chenhao Zheng
University of Washington

Zhongzheng Ren
UNC-Chapel Hill

Abstract

Hour-long videos, streaming feeds, and movie-length content are pushing vision-language models (VLMs) past their breaking point—the standard recipe of encoding each frame into visual tokens yields sequences that grow linearly with video length, causing prohibitive memory consumption and inference latency. Existing approaches, such as token reduction and keyframe selection, attempt to shorten the token sequence but leave the $O(n)$ scaling intact, failing to resolve the fundamental bottleneck. We propose `FramePack-U`, *a strikingly simple, training-free frame packing framework that compresses arbitrarily long videos into a fixed-size visual representation at constant memory cost.* `FramePack-U` requires no fine-tuning or architectural modification — it plugs directly into off-the-shelf VLMs at inference time, yet consistently surpasses more elaborate baselines. Drawing inspiration from FramePack in generation literature, we propose a new content-aware compression strategy that operates under a fixed token budget for visual understanding. Specifically, `FramePack-U` proceeds in two steps: (1) scoring each frame by its semantic similarity to the text prompt and ranking frames in a diversity-driven manner, and (2) packing the selected frames at varying resolutions into a fixed-length context window. This design concentrates the memory budget on frames that are both instruction-relevant and visually diverse. While in theory `FramePack-U` can pack arbitrarily many frames into the fixed budget, we identify an empirical instantiation that generalizes robustly across different settings. Extensive experiments on LongVideoBench, MLVU, and Video-MME show that `FramePack-U` consistently outperforms full-resolution baselines as well as prior token reduction and keyframe selection methods, across diverse model families, parameter scales, and frame budgets. Notably, on Qwen3.5-35B-A3B with a 128-frame budget, `FramePack-U` yields gains of **+12.9%, +10.6%, and +9.7%** over the official model on the three benchmarks, respectively.

Project Page: `https://zeld0n.github.io/framepack-U/`.

1 Introduction

Long video understanding underpins a wide range of real-world applications, including surveillance analysis, autonomous driving, sports commentary, movie comprehension, and instructional or educational assistants. Many of these scenarios further demand streaming settings, where models must continuously ingest incoming frames and respond on the fly rather than process a bounded clip offline. Unlike images or short videos, hour-long or unbounded streams carry rich temporal dynamics, evolving narratives, and long-horizon causal dependencies, all of which are essential for higher-level reasoning. Recent vision-language models (VLMs) [2, 27, 10, 16, 5] offer a promising path forward by encoding visual content into tokens that are jointly processed with text instructions alongside an LLM backbone. Yet when scaled from images and short clips to long or streaming videos, this

Correspondence to Shuo Gao, Jason Ren <gaoshuo@ruc.edu.cn, jasonren@cs.unc.edu>.

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

paradigm encounters a fundamental obstacle: the number of visual tokens grows proportionally with the number of frames, which severely dilutes context, inflates memory consumption, and drives inference cost up linearly or higher with video duration.

To address this, a large body of work resorts to information compression, typically through token reduction [20, 34, 8] or keyframe selection [22, 28, 39]. By scoring visual features against handcrafted or learned criteria, these methods filter out non-essential content and retain only the clues deemed necessary for understanding. The VLM then operates on a much smaller set of visual tokens, yielding faster inference and reduced memory footprint without retraining the backbone. However, such compression schemes share two fundamental limitations. (1) They only postpone the scalability bottleneck rather than eliminate it: the retained token count still scales as $O(n)$ or higher with video duration, so memory and latency continue to grow without bound as videos get longer. (2) Hard-dropping tokens or frames is a discrete and inherently lossy operation. Forcing a binary keep-or-discard decision fragments temporal continuity and erases transient actions that are often pivotal for long-horizon reasoning – precisely the regime where long-video understanding matters most.

In this work we propose `FramePack-U`, *a strikingly simple, training-free frame compression and packing method that extracts a fixed-size representation from videos of arbitrary length, achieving constant $O(1)$ memory complexity.* `FramePack-U` requires no fine-tuning or architectural modification—it plugs directly into off-the-shelf VLMs at inference time. The key idea is to assign heterogeneous spatial resolutions to frames under a fixed total token budget regardless of video duration, so that the memory budget is spent on frames that matter most. Inspired by FramePack [36], which packs frames at varying resolutions for efficient video generation, we adapt this idea for long video understanding. However, the original formulation is designed for generation, with a schedule favoring temporal continuity and proximity, hence unsuitable for understanding tasks that must preserve task-relevant evidence throughout the video.

To mitigate this issue, `FramePack-U` is designed to be content-aware, and operates in two steps. (1) Relevance- and diversity-aware sampling. We compute the semantic similarity between the text prompt and each frame, producing a relevance score that identifies frames containing instruction-relevant evidence. Starting from the highest-scoring frame as a seed, we iteratively rank the remaining frames by selecting, at each step, the frame most visually distinct from those already chosen. This produces a single ordering that balances instruction relevance with visual diversity (Sec. 3.1). (2) Hierarchical resolution packing. We then assign frames along this ranking to resolution hierarchies, with top-ranked frames preserved at finer spatial resolutions and lower-ranked frames progressively downsampled (Sec. 3.2). In principle, all frames can be retained, since the tiered scheme packs them into a fixed-size context regardless of video length—concentrating the memory budget on diverse, instruction-relevant frames while preserving full temporal coverage. While many tiered packing schedules are possible in principle, we provide an empirical instantiation, identified through experiments, that generalizes robustly across architectures, scales, and benchmarks.

We conduct extensive experiments on major long video VQA benchmarks such as LongVideoBench [30], MLVU [38], and Video-MME [7]. Our results have verified the effectiveness of `FramePack-U` against a number of competitive baselines. Under the same inference memory budgets, `FramePack-U` consistently outperforms full-resolution baselines across a variety of model series, parameter scales, and frame number settings. For instance, on the Mixture-of-Expert (MoE) model `Qwen3.5-35B-A3B` with a 128-frame budget, `FramePack-U` yields significant gains of **12.9%, 10.6%, and 9.7%** on LongVideoBench, MLVU, and Video-MME, respectively. On an alternative backbone `Qwen3.6-27B`, `FramePack-U` yields consistent gains of **6.4%, 8.0%, and 2.7%**. Also, it surpasses existing token reduction and keyframe selection methods, demonstrating a better accuracy–efficiency trade-off in long and extreme-long video settings. It is worth noting that `FramePack-U` is training-free and directly applicable to off-the-shelf VLMs.

2 Related Work

Vision-Language Models. VLMs have emerged in recent years as the dominant approach for image understanding and grounding [6, 16, 23, 10]. The prevailing recipe pairs a CLIP-style [19, 25] visual encoder with an LLM backbone, bridging the two through a lightweight connector that projects image embeddings into the LLM token space [6, 12]. Video VLMs [3, 5, 2, 4, 13] typically inherit this design and apply the image encoder frame-by-frame to obtain per-frame embeddings. A handful of

efforts replace or augment the per-frame pipeline with pretrained video encoders, often operating on frame pairs or short clips [40, 26, 24], but scaling video encoders to longer temporal windows still trails per-frame image encoding with strong backbones such as SigLIP 2 [25].

Long Video VLM. To curb the token overhead of frame-wise encoding, a growing body of work pursues more compact video representations. One line of research performs *token reduction* inside the visual stream, merging or pruning redundant tokens based on attention scores, similarity, or learned importance [21, 11, 27]. The second line focuses on *keyframe selection*, scoring frames by query relevance or diversity and discarding the rest before they reach the LLM [22, 28, 39, 33]. Other works explore architectural changes, including memory banks for streaming video [35], sequential temporal aggregation [29], and slow–fast pathways that mix high-fidelity keyframes with coarsely encoded context frames [32, 40]. Despite their differences, these approaches share a common property: the resulting token sequence still scales proportionally with video duration, so memory and latency continue to grow as videos get longer.

Packing for Generation. FramePack [36] proposes a context packing strategy for next-frame prediction models in video generation. The key idea is to assign heterogeneous token budgets to input frames according to their importance, with frames closer to the prediction target receiving longer contexts via finer-grained patchifying kernels. As a result, the generation workload becomes invariant to video length, enabling long video synthesis on consumer GPUs. The packing principle has since been adopted beyond video generation: FastAvatar [31] applies it to 3D avatar reconstruction, retaining a sparse set of high-fidelity keyframes whilst compressing the remaining frames into an aggregated token representation, enabling feedforward reconstruction from hundreds of frames within a fixed memory footprint. Yume [15] and Yume1.5 [14] adopt related history-compression strategies for interactive world generation, where historical frames are compressed at different rates to support long-form autoregressive generation. `FramePack-U` shares the same packing philosophy, i.e., allocating more tokens to more important frames within a fixed budget. However, rather than generation, it targets *understanding*, adapting the packing strategy to the inference-time encoding of video inputs in VLMs.

3 `FramePack-U`

Let $\mathcal{V} = \{\mathbf{x}_t\}_{t=1}^{T}$ denote a video of T frames. Our goal is to construct a packed visual sequence $\mathcal{V}^* = \{\mathbf{x}_t^*\}_{t=1}^{N}$ whose total visual token count is bounded by a fixed memory budget, independent of the video length T. As shown in Fig 1, `FramePack-U` proceeds in 2 stages: (1) **relevance- and diversity-aware sampling**, which reduces $\mathcal{V}$ to $\mathcal{V}^*$ candidate frames via retrieval guided by instruction relevance and visual diversity (Sec. 3.1); and (2) **hierarchical resolution packing**, which assigns each selected frame a spatial downsampling factor according to its estimated priority, subject to the budget constraint (Sec. 3.2). The packed sequence is then passed to the VLM backbone as visual tokens.

Discussion of the necessity of N. In principle, N can be as large as T, so `FramePack-U` can pack a video of arbitrary length into the fixed budget. In practice, we treat N as a tunable parameter that controls how many candidate frames are considered for packing, giving the framework flexibility and enabling systematic study of how input scale affects performance. Later in experiments we show that performance is not monotonically increasing in N. Under a fixed token budget, larger N improves temporal coverage but forces lower per-frame resolution, and most VLMs degrade once frame resolution falls below the range seen during pre-training. The optimal N thus reflects a trade-off between temporal density and per-frame fidelity, and depends on model capacity and pre-training data; we report the setting that generalizes best across our evaluations in Sec. 4.

3.1 Relevance- and diversity-aware frame sampling

We adopt a relevance and diversity-aware selection strategy over these N candidates that favors frames that are both semantically relevant to the text instruction and visually non-redundant. Concretely, we first use CLIP [19] to compute the cosine similarity $r_i = \mathrm{CLIP}(\mathbf{x}_i, \mathbf{q})$ between frame $\mathbf{x}_i$ and the input text prompt $\mathbf{q}$.

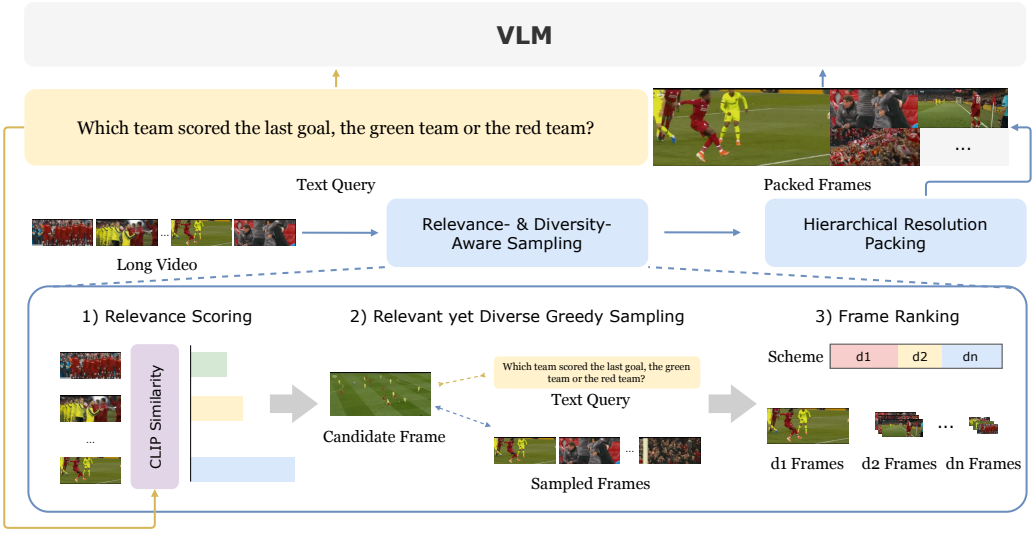

Figure 1: **Overview of `FramePack-U`.** Given a long video and a text query, `FramePack-U` selects query-relevant and visually diverse frames, then packs them with heterogeneous resolutions under a fixed memory budget. Higher-priority frames are kept at higher resolution, while lower-priority frames are progressively compressed. The packed frames are fed into the VLM for answer generation.

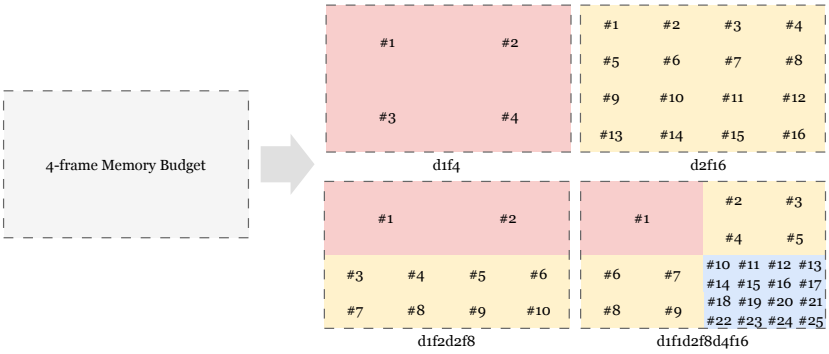

Figure 2: **Illustration of packing schemes under the same memory budget.** Using a 4-frame memory budget as an example, red, yellow, and blue denote frames with downsampling factors $d = 1, 2, 4$, respectively. Different resolution combinations can use the same memory while packing different numbers of frames.

We then start with the greedy sampling process. Let $n \in N$ denote the current sampling step, and $\tilde{\mathcal{V}}_n$ denote the set of chosen frames before the n-th step. The next frame is sampled as:

$$\mathbf{x}_n^* = \begin{cases} \arg\max_{\mathbf{x}_i \in \mathcal{V}} r_i, & n = 1, \\ \arg\max_{\mathbf{x}_i \in \mathcal{V} \setminus \tilde{\mathcal{V}}} \left[\lambda \cdot r_i - (1-\lambda) \max_{\mathbf{x}_j \in \tilde{\mathcal{V}}} \mathrm{sim}(\mathbf{x}_i, \mathbf{x}_j) \right], & n \in [2, N] \end{cases} \tag{1}$$

After each step, we update $\tilde{\mathcal{V}} = \tilde{\mathcal{V}} \cup \{\mathbf{x}_n^*\}$ until N frames are selected. This greedy procedure ensures that the retained frames cover the most instruction-relevant content while avoiding the over-selection of visually redundant frames.

3.2 Hierarchical resolution packing

Given the N ranked candidate frames from Sec. 3.1, `FramePack-U` assigns each one a spatial resolution under a fixed token budget. The guiding principle is that frames differ in their relevance to

Algorithm 1 `FramePack-U`

Require: Video frames $\mathcal{V} = \{x_i\}_{i=1}^{T}$, query q, packed frame number N, packing scheme $\mathcal{S} = \{(d, f_d)\}$, trade-off coefficient λ
Ensure: Packed visual sequence $\mathcal{V}^*$

Compute relevance scores $r_i \leftarrow \mathrm{CLIP}(x_i, q)$ for all $x_i \in \mathcal{V}$
Initialize selected set $\widetilde{\mathcal{V}} \leftarrow \emptyset$
for $n = 1, \ldots, N$ **do**
    Select $x_n^* \leftarrow \begin{cases} \arg\max_{x_i \in \mathcal{V}} r_i, & n = 1, \\ \arg\max_{x_i \in \mathcal{V} \setminus \widetilde{\mathcal{V}}} \left[\lambda r_i - (1-\lambda) \max_{x_j \in \widetilde{\mathcal{V}}} \mathrm{sim}(x_i, x_j) \right], & n > 1. \end{cases}$
    $\widetilde{\mathcal{V}} \leftarrow \widetilde{\mathcal{V}} \cup \{x_n^*\}$
end for
Sort packing levels in $\mathcal{S}$ by increasing downsampling factor d
for $n = 1, \ldots, N$ **do**
    Find d such that $\sum_{d' < d} f_{d'} < n \leq \sum_{d' \leq d} f_{d'}$
    $\hat{x}_n \leftarrow \mathrm{Resize}(x_n^*, 1/d)$
end for
$\mathcal{V}^* \leftarrow$ temporal-order sorting of $\{\hat{x}_n\}_{n=1}^{N}$
return $\mathcal{V}^*$

the query and their representativeness of the video: rather than encoding all frames at uniform fidelity, `FramePack-U` allocates higher spatial resolution to more informative frames and progressively compresses the remainder, maximizing temporal coverage within the budget. We next describe the packing scheme, the frame ranking it relies on, and the concrete resolution schedule we adopt.

Packing scheme. A packing scheme is a hierarchical resolution assignment

$$\mathcal{S} = \{(d,\ f_d)\}_{d \in \{1,2,4,8,\ldots\}}, \tag{2}$$

where d is a spatial downsampling factor and f_d is the number of frames assigned to level d, with $\sum_d f_d = N$. Downsampling a frame by d in each spatial dimension yields M/d^2 tokens per frame, where M is the token count of a full-resolution frame. The total memory footprint of a scheme is therefore constrained by

$$M \sum_d \frac{f_d}{d^2} \ \leq \ B, \tag{3}$$

where B is the fixed token budget.

For convenience, we denote a level of n frames at downsampling factor k as `d{`k`}f{`n`}`, and concatenate segments to describe mixed-resolution schemes. For example, `d1f16d2f448` allocates 16 frames at full resolution (d=1) and 448 frames at 2× downsampling (d=2), with a total cost of $16 + 448/4 = 128$ full-resolution frame equivalents.

Frame ranking. The packing scheme determines *how many* frames live at each resolution; a ranking determines *which* frames they are. As illustrated in Figure 2, different packing schemes can consume the same memory budget while retaining different numbers of frames by combining multiple resolution levels. We rank frames by the score in Eq. (1): frames selected earlier in the greedy procedure receive higher priority and are assigned to finer-resolution levels (smaller d). This concentrates the high-resolution quota on frames that are both instruction-relevant and visually diverse. The full procedure is summarized in Alg. 1.

As a query-agnostic alternative, we also consider a binary-tree midpoint ordering that recursively ranks frames by temporal midpoints, ensuring uniform temporal coverage without relying on query signals. We include it as a baseline in our ablations (Sec. 4).

4 Experiments

In this section, we evaluate the effectiveness of `FramePack-U` on long video understanding tasks. We first describe the experimental setup. We then compare `FramePack-U` with existing training-free

video compression methods (Sec. 4.1). Finally, we conduct ablation studies (Sec. 4.2) and show qualitative results (Sec. 4.3).

Benchmarks. We evaluate our method on three most popular long video understanding benchmarks: LongVideoBench [30], MLVU [38], and Video-MME [7]. LongVideoBench reports the overall accuracy as well as results under different video-duration ranges, i.e., 15s, 60s, 600s, and 3600s. MLVU reports the M-average score. For Video-MME, we report the overall accuracy and the performance on short, medium, and long videos, with particular emphasis on the long-video split.

Base models. To validate the effectiveness of our method across different scales and architectures, we conduct experiments on two Qwen VLMs: `Qwen3.6-27B` (dense) and `Qwen3.5-35B-A3B` (MoE).

Compared methods. We first compare against the vanilla uniform sampling strategy suggested by the official guidance. Additionally, we compare to representative training-free video token/frame reduction methods, including FastVID [20], VisionZip [34], and FrameFusion [8]. These methods reduce the visual computation by selecting or compressing visual tokens/frames before feeding them into the language model.

Implementation details. Given an input video, we uniformly sample T frames as the initial frame set. Our method then assigns each frame to different resolution levels under a fixed memory budget. Unless otherwise specified, we use the diversity-aware packing strategy as our default method. The strategy first measures query-frame relevance using CLIP similarity and then performs diversity-aware selection based on frame-level visual feature similarity, preventing visually redundant frames from occupying the high-resolution budget. For the CLIP model, we choose CLIP-ViT-B/32 for its lightweight nature. λ is set to 0.7. For the initial sampling rate, we sample all frames in an offline way for the benchmarking experiments. It could also be set in online mode for lower latency. We use the global feature extracted by ViT for frame-to-frame similarity. Regarding the resized dimensions, we maintain the original resolutions following the default settings for LongVideoBench and Video-MME. For MLVU, we proportionally resize candidate frames such that the longer edge is 1280 pixels before subsequent processing. We follow the practice of the benchmarks and directly use their testing text prompts.

4.1 Results

We compare `FramePack-U` with representative training-free video compression methods under the same visual memory budgets. Specifically, we evaluate all methods using 128-, 256-, and 512-frame memory, where each budget denotes the memory cost equivalent to processing the corresponding number of full-resolution frames. This setting allows us to assess whether different methods can improve video understanding accuracy without increasing the visual memory cost.

Results on Qwen3.6-27B. Table 1 left compares our method with existing training-free compression baselines on Qwen3.6-27B. We use `d1f16d2f448`, `d1f250d4f96` and `d1f128d2f1536` for 128-, 256- and 512-frame memory, respectively. Across three levels of memory usage, `FramePack-U` consistently achieves the best overall performance among all methods. (1) Compared with the vanilla baseline, `FramePack-U` brings clear improvements across every memory budget, with the largest relative gains appearing under the tighter 128-frame memory setting. (2) Compared with existing training-free compression methods, `FramePack-U` also shows stronger performance across benchmarks, demonstrating that preserving broad temporal coverage while adaptively allocating spatial resolution is more effective than directly dropping frames or compressing visual tokens. These results suggest that `FramePack-U` makes better use of a fixed visual memory budget by selecting query-relevant and visually diverse frames for higher-resolution processing.

Results on Qwen3.5-35B-A3B. Table 1 right reports the results on Qwen3.5-35B-A3B under different memory budgets. We use `d1f32d2f384`, `d1f64d2f768` and `d1f384d2f512` for 128-, 256- and 512-frame memory, respectively. Similarly, `FramePack-U` consistently outperforms the vanilla baseline and achieves the best overall performance among all compared methods. The improvements are particularly notable because Qwen3.5-35B-A3B has a weaker vanilla baseline than Qwen3.6-27B, yet `FramePack-U` substantially narrows the performance gap by better utilizing the fixed visual memory budget. These results further demonstrate the robustness of `FramePack-U` across different architectures and confirm the benefit of prioritizing query-relevant and visually diverse frames.

Method	LVB	MLVU	V-MME
Proprietary models			
GPT-5 [17]	72.6	77.7	83.3
GPT-5 mini [17]	69.7	69.1	77.3
Gemini 3 Pro [10]	75.9	75.7	88.6
Gemini 2.5 Flash [9]	73.1	75.1	84.2
Claude Sonnet 4.5 [1]	65.1	64.0	74.2
128 frames			
Vanilla	68.0	75.9	73.6
FastVID [20]	59.9$_{-8.1}$	67.3$_{-8.6}$	61.0$_{-12.6}$
VisionZip [34]	57.7$_{-10.3}$	67.4$_{-8.5}$	63.2$_{-10.4}$
FrameFusion [8]	57.8$_{-10.2}$	65.4$_{-10.5}$	63.5$_{-10.1}$
`FramePack-U` (Ours)	<u>74.4</u>$_{+6.4}$	**83.9**$_{+8.0}$	<u>76.3</u>$_{+2.7}$
256 frames			
Vanilla	70.9	76.7	75.9
FastVID [20]	60.8$_{-10.1}$	68.9$_{-7.8}$	62.2$_{-13.7}$
VisionZip [34]	62.8$_{-8.1}$	70.7$_{-6.0}$	65.2$_{-10.7}$
FrameFusion [8]	61.6$_{-9.3}$	67.7$_{-9.0}$	65.4$_{-10.9}$
`FramePack-U` (Ours)	74.3$_{+3.4}$	<u>83.2</u>$_{+6.5}$	**77.9**$_{+2.0}$
512 frames			
Vanilla	71.8	78.0	74.4
FastVID [20]	58.5$_{-13.3}$	68.4$_{-9.6}$	61.1$_{-13.3}$
VisionZip [34]	63.7$_{-8.1}$	69.9$_{-8.1}$	67.2$_{-7.2}$
FrameFusion [8]	62.8$_{-9.0}$	70.5$_{-7.5}$	66.9$_{-7.5}$
`FramePack-U` (Ours)	**74.9**$_{+3.1}$	82.1$_{+4.1}$	<u>76.3</u>$_{+1.9}$

Method	LVB	MLVU	V-MME
Proprietary models			
GPT-5 [17]	72.6	77.7	83.3
GPT-5 mini [17]	69.7	69.1	77.3
Gemini 3 Pro [10]	75.9	75.7	88.6
Gemini 2.5 Flash [9]	73.1	75.1	84.2
Claude Sonnet 4.5 [1]	65.1	64.0	74.2
128 frames			
Vanilla	59.4	71.8	66.7
FastVID [20]	50.3$_{-9.1}$	66.7$_{-5.1}$	61.5$_{-5.2}$
VisionZip [34]	52.2$_{-7.2}$	65.6$_{-6.2}$	60.7$_{-6.0}$
FrameFusion [8]	56.5$_{-2.9}$	63.9$_{-7.9}$	61.3$_{-5.4}$
`FramePack-U` (Ours)	72.3$_{+12.9}$	82.4$_{+10.6}$	**76.4**$_{+9.7}$
256 frames			
Vanilla	60.3	73.0	67.1
FastVID [20]	51.2$_{-9.1}$	68.4$_{-4.6}$	63.9$_{-3.2}$
VisionZip [34]	56.9$_{-3.4}$	67.5$_{-5.5}$	64.6$_{-2.5}$
FrameFusion [8]	57.5$_{-2.8}$	67.7$_{-5.3}$	64.8$_{-2.3}$
`FramePack-U` (Ours)	<u>72.6</u>$_{+12.3}$	<u>82.9</u>$_{+9.9}$	75.5$_{+8.4}$
512 frames			
Vanilla	59.8	74.3	67.3
FastVID [20]	57.9$_{-1.9}$	68.4$_{-5.9}$	64.5$_{-2.8}$
VisionZip [34]	55.2$_{-4.6}$	69.4$_{-4.9}$	64.4$_{-2.9}$
FrameFusion [8]	59.9$_{+0.1}$	69.3$_{-5.0}$	65.4$_{-1.9}$
`FramePack-U` (Ours)	**73.0**$_{+13.2}$	**84.7**$_{+10.4}$	<u>75.8</u>$_{+8.5}$

Table 1: Main results on Qwen3.6-27B (**left**) and Qwen3.5-35B-A3B (**right**) across different memory budgets. Subscripts indicate the gap to Vanilla at the same budget; **bold** and <u>underline</u> denote the best and second-best results among open-weight models, respectively. We find that: (1) `FramePack-U` outperforms all baselines substantially, (2) our top results are competitive with, and often surpasses, those of much larger proprietary VLMs.

4.2 Ablations

Ablation study on schemes. The hierarchical packing scheme determines the trade-off between temporal coverage and per-frame spatial fidelity. We conduct a broad search over mixed-resolution schemes and identify several configurations that work consistently well across benchmarks. Table 2 reports representative strong schemes under the 512-frame memory budget on Qwen3.6-27B.

Overall, `FramePack-U` is robust to the exact choice of scheme. All mixed-resolution schemes in Table 2 clearly outperform the vanilla baseline under the same memory budget. Different schemes slightly favor different benchmarks: for example, `d1f128d2f1536` performs best on LongVideoBench, `d1f384d2f512` performs best on MLVU, and `d1f480d2f128` performs best on Video-MME. This suggests that there is no single universally optimal allocation, as different tasks may require different balances between fine-grained spatial evidence and broader temporal coverage.

Based on this search, we select schemes that achieve strong overall performance while keeping a balanced allocation between high-resolution anchor frames and low-resolution contextual frames. Importantly, the gains are not tied to a single carefully tuned configuration: even when the number of packed frames varies substantially, from 608 to 1664 under the 512-frame budget, `FramePack-U` maintains consistent improvements over the vanilla baseline. This robustness supports our design choice of hierarchical resolution packing and shows that the key benefit comes from preserving broad temporal context while assigning higher fidelity to informative frames.

Ablation study on frame diversity. We further study the effect of the diversity-aware frame selection strategy on Qwen3.6-27B. As an alternative, we consider a *Relevance-only* strategy, which ranks frames solely according to their semantic similarity with the text query and assigns higher resolutions to the top-ranked frames. While this strategy can identify query-relevant frames, it does not explicitly consider visual redundancy among selected frames. As a result, multiple high-resolution slots may be allocated to visually similar frames, limiting the effective temporal coverage.

Table 4 compares the Relevance-only strategy with our diversity-aware strategy under different memory budgets. These results demonstrate that query relevance alone is insufficient for frame

Memory	Scheme	#Frame	LVB	MLVU	V-MME
512 frames	Vanilla	512	71.8	78.0	74.4
	d1f128d2f1536	1664	**74.9**	82.1	76.3
	d1f256d2f1024	1280	74.7	82.2	75.9
	d1f384d2f512	896	74.6	**82.9**	76.0
	d1f480d2f128	608	74.3	81.0	**76.5**

Table 2: **Ablation on different resolution allocation schemes using Qwen3.6-27B**. See Appendix B for ablations on 128- and 256-frame budgets.

Memory	Model	LVB	MLVU	V-MME
512 frames (d1f256d2f1024)	Qwen3.5-4B	+0.9	-1.0	-1.2
	Qwen3.5-9B	+1.6	-0.6	-0.4
	Qwen3.6-27B	**+2.5**	**+4.2**	**+1.5**

Table 3: **Ablation on model size.** We report the performance gain of FramePack-U over the vanilla baseline across different model sizes. See Appendix B for ablations on 128- and 256-frame budgets.

Memory	Scheme	Strategy	LongVideoBench	MLVU	Video-MME
512 frames	–	vanilla	71.8	78.0	74.4
	d1f128d2f1536	FramePack-U	**74.9**	82.1	**76.3**
		– Relevance	74.0 (-0.9)	82.3 (+0.2)	**76.3** (+0.0)
		– Diversity	73.9 (-1.0)	80.4 (-1.7)	69.8 (-6.5)
	d1f256d2f1024	FramePack-U	74.7	82.2	75.9
		– Relevance	74.6 (-0.1)	82.3 (+0.1)	76.0 (+0.1)
		– Diversity	73.8 (-0.9)	80.6 (-1.6)	69.6 (-6.3)
	d1f384d2f512	FramePack-U	74.6	**82.9**	76.0
		– Relevance	73.1 (-1.5)	81.1 (-1.8)	75.4 (-0.6)
		– Diversity	73.7 (-0.9)	80.8 (-2.1)	69.9 (-6.1)

Table 4: **Ablation on prompt relevance and frame diversity using Qwen3.6-27B**. – *Relevance* removes prompt relevance and uses a prompt-agnostic diversity-based ordering, while – *Diversity* ranks frames only by query-frame similarity. FramePack-U combines prompt relevance and visual diversity. Values in parentheses denote the difference between each ablated variant and FramePack-U. See Appendix B for ablations on 128- and 256-frame budgets.

packing. By jointly considering relevance and visual diversity, FramePack-U allocates the limited high-resolution budget to more complementary frames, leading to more reliable performance under fixed memory constraints.

Ablation study on prompt relevance. We further analyze the effect of prompt relevance in the packing strategy on Qwen3.6-27B. To isolate this factor, we construct a prompt-agnostic baseline, *Diversity-only*, which does not use the text prompt for frame selection. Instead, it uniformly samples N frames from the video to preserve coarse temporal coverage. It then assigns frame priorities using a binary-tree midpoint ordering: the temporal midpoint of the sampled sequence is ranked first, followed by the midpoints of the left and right sub-intervals, and this process is applied recursively in a breadth-first manner. Given this priority order, the same resolution allocation scheme is applied as in our method, with earlier-ranked frames receiving higher spatial resolutions. Thus, Diversity-only preserves global temporal coverage but does not explicitly identify query-relevant evidence.

In contrast, our full strategy incorporates prompt relevance during frame sampling. Table 4 shows the comparison, where our full strategy achieves stronger performance on most benchmarks and memory settings, especially on MLVU. For example, under 512-frame memory, our strategy improves LongVideoBench accuracy from 74.0 to 74.9 with the same d1f128d2f1536 scheme, and from 73.1 to 74.6 with d1f384d2f512. These results show that temporal diversity alone is not sufficient: incorporating prompt relevance helps allocate high-resolution visual capacity to frames that are more likely to contain task-relevant evidence.

Ablation study on model size. We further study how the effectiveness of FramePack-U changes with the scale of the base model. To this end, we compare Qwen3.5-4B, Qwen3.5-9B, and Qwen3.6-27B. For each model and memory budget, we select the best-performing configuration from the evaluated resolution schemes. The results are shown in Table 3.

Overall, the benefit of FramePack-U becomes more pronounced as the base model scales up. On the smaller 4B and 9B models, FramePack-U can improve temporal coverage by packing more frames into the same memory budget, but the gains are less stable, especially under larger memory budgets. In some cases (eg., Qwen3.5-4B on MLVU), increasing the number of packed frames does not lead to better performance and may even result in marginal degradation on certain benchmarks. We attribute

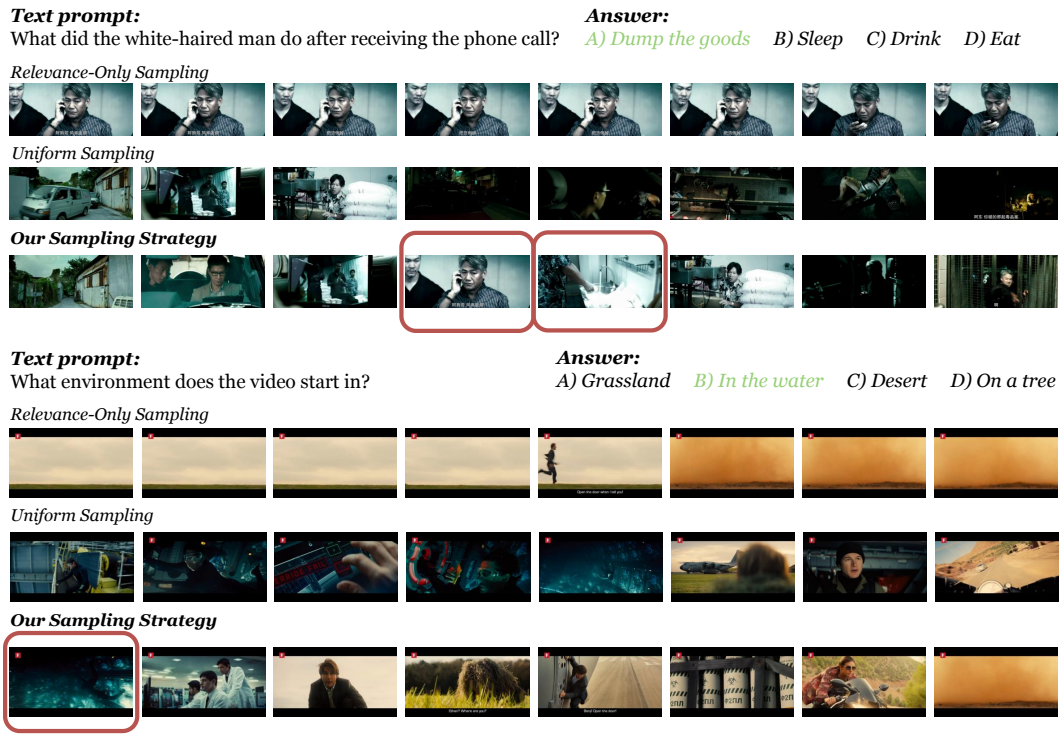

Figure 3: **Qualitative comparison.** Relevance-only sampling selects frames based only on query-frame similarity and often produces redundant frames, while uniform sampling preserves temporal coverage but may miss question-relevant evidence. Our strategy combines prompt relevance and visual diversity, allowing it to select frames that better cover the video while retaining key evidence for answering the question. Red boxes highlight frames that contain important visual evidence.

this to the capability boundary of smaller VLMs: although `FramePack-U` provides more diverse visual evidence, these models may not have sufficient capacity to effectively integrate and reason over a large number of visual observations. This suggests that stronger VLMs are better able to exploit the additional temporal context and selectively attend to high-resolution, task-relevant evidence. The advantage is especially evident when the memory budget is large, where `FramePack-U` can include more diverse contextual frames while still preserving fine-grained details for important frames.

4.3 Qualitative results

Fig 3 visualizes the sampled frames produced by different strategies. Relevance-only sampling tends to repeatedly select frames that are highly similar to the text prompt, but these frames are often visually redundant and cover only a narrow temporal span. Uniform sampling preserves coarse temporal coverage, but it is prompt-agnostic and may allocate visual budget to frames that are irrelevant to the question. In contrast, our sampling strategy selects frames that are both instruction-relevant and visually diverse. As shown in the examples, our method captures the key evidence needed to answer the question, such as the white-haired man dumping the goods after the phone call and the underwater scene at the beginning of the video. These qualitative results further illustrate why combining prompt relevance with visual diversity leads to more effective frame packing for long video understanding.

To analyze how different strategies cover visual content, Fig. 4 shows t-SNE feature distributions under a fixed 128-frame memory budget. Uniform sampling selects frames strictly by temporal position at full resolution ($d = 1$), yielding a sparse subset of 128 frames. Consequently, it under-represents query-critical regions and wastes budget on less informative content (right thumbnails).In contrast, `FramePack-U` allocates multi-tier downsampling factors ($d \in \{1, 2, 4\}$) to pack 848 frames

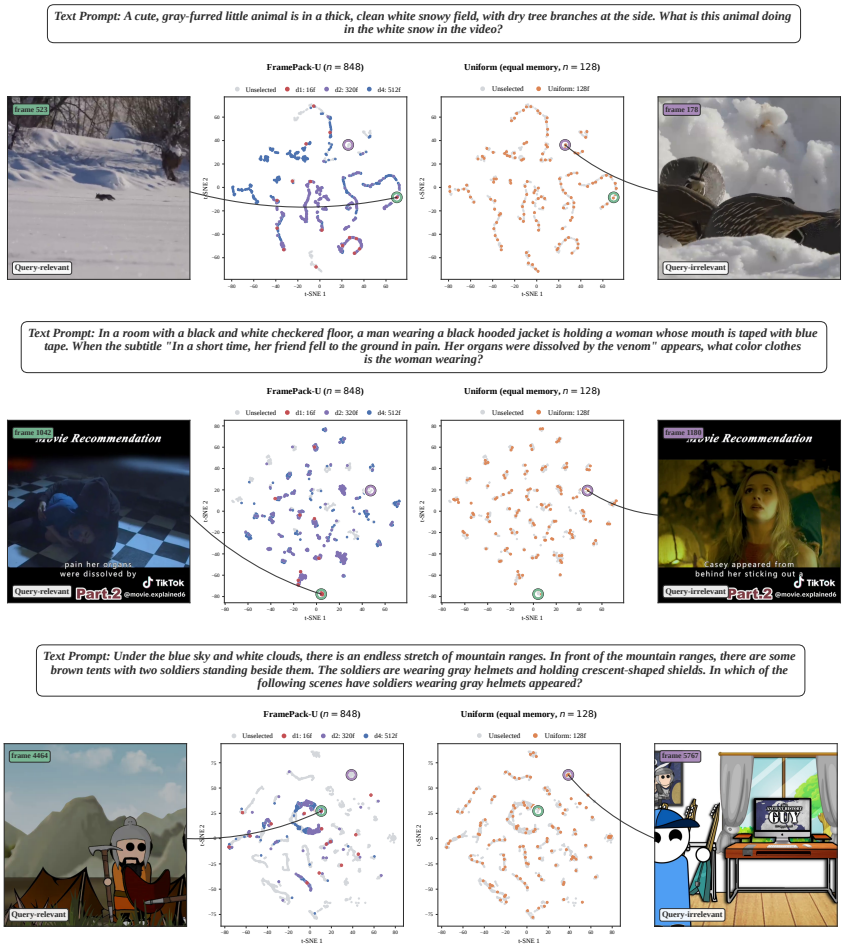

Figure 4: **t-SNE visualization.** We compare the feature distributions of frames selected by `FramePack-U` ($n = 848$, scheme `d1f16d2f320d4f512`) against *Uniform Sampling* ($n = 128$, full resolution) across three distinct queries. Circles mark representative frames identified by each strategy, whose corresponding visual frames are displayed in the callout thumbnails. Points are colored according to their downsampling tiers ($d = 1, 2, 4$).

within the same memory footprint, achieving much denser feature coverage. Guided jointly by query relevance and visual diversity, `FramePack-U` prioritizes query-relevant semantic clusters at higher resolution ($d = 1$) while compressing background context to $d = 2$ or 4 (left thumbnails). Thus, `FramePack-U` provides substantially broader contextual coverage while dynamically prioritizing task-relevant frames.

5 Conclusion

We introduced `FramePack-U`, a simple, training-free approach to long video understanding that achieves constant $O(1)$ memory complexity by packing frames into a fixed token budget at heterogeneous spatial resolutions. `FramePack-U` ranks frames by jointly considering instruction relevance and visual diversity, then allocates resolutions hierarchically so that the memory budget is concentrated on the most informative frames while full temporal coverage is preserved. Across LongVideoBench, MLVU, and Video-MME, `FramePack-U` consistently outperforms full-resolution baselines as well as prior token reduction and keyframe selection methods, and frequently matches or exceeds substantially larger proprietary VLMs. As a plug-and-play module for off-the-shelf VLMs, `FramePack-U` offers a practical step toward scalable long and streaming video understanding.

Acknowledgments and Disclosure of Funding

Funding. The authors received no third-party funding or support for this work.

Competing Interests. The authors declare no competing interests.

References

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

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

[3] Guo Chen, Zhiqi Li, Shihao Wang, Jindong Jiang, Yicheng Liu, Lidong Lu, De-An Huang, Wonmin Byeon, Matthieu Le, Max Ehrlich, Tong Lu, Limin Wang, Bryan Catanzaro, Jan Kautz, Andrew Tao, Zhiding Yu, and Guilin Liu. Eagle 2.5: Boosting long-context post-training for frontier vision-language models. In *NeurIPS*, 2025.

[4] Jang Hyun Cho, Andrea Madotto, Effrosyni Mavroudi, Triantafyllos Afouras, Tushar Nagarajan, Muhammad Maaz, Yale Song, Tengyu Ma, Shuming Hu, Hanoona Rasheed, Peize Sun, Po-Yao Huang, Daniel Bolya, Suyog Jain, Miguel Martin, Huiyu Wang, Nikhila Ravi, Shashank Jain, Temmy Stark, Shane Moon, Babak Damavandi, Vivian Lee, Andrew Westbury, Salman Khan, Philipp Krähenbühl, Piotr Dollár, Lorenzo Torresani, Kristen Grauman, and Christoph Feichtenhofer. Perceptionlm: Open-access data and models for detailed visual understanding. *arXiv preprint arXiv:2504.13180*, 2025.

[5] Christopher Clark, Jieyu Zhang, Zixian Ma, Jae Sung Park, Mohammadreza Salehi, Rohun Tripathi, Sangho Lee, Zhongzheng Ren, Chris Dongjoo Kim, Yinuo Yang, Vincent Shao, Yue Yang, Weikai Huang, Ziqi Gao, Taira Anderson, Jianrui Zhang, Jitesh Jain, George Stoica, Winston Han, Ali Farhadi, and Ranjay Krishna. Molmo2: Open weights and data for vision-language models with video understanding and grounding. In *CVPR*, 2026.

[6] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, Andrew Head, Rose Hendrix, Favyen Bastani, Eli VanderBilt, Nathan Lambert, Yvonne Chou, Arnavi Chheda, Jenna Sparks, Sam Skjonsberg, Michael Schmitz, Aaron Sarnat, Byron Bischoff, Pete Walsh, Chris Newell, Piper Wolters, Tanmay Gupta, Kuo-Hao Zeng, Jon Borchardt, Dirk Groeneveld, Crystal Nam, Sophie Lebrecht, Caitlin Wittlif, Carissa Schoenick, Oscar Michel, Ranjay Krishna, Luca Weihs, Noah A. Smith, Hannaneh Hajishirzi, Ross Girshick, Ali Farhadi, and Aniruddha Kembhavi. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *CVPR*, 2025.

[7] 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. In *CVPR*, 2025.

[8] Tianyu Fu, Tengxuan Liu, Qinghao Han, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang. Framefusion: Combining similarity and importance for video token reduction on large visual language models. *CoRR*, abs/2501.01986, 2025.

[9] Google. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025. URL `https://arxiv.org/abs/2507.06261`.

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

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

[12] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *CVPR*, 2024.

[13] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-ChatGPT: Towards detailed video understanding via large vision and language models. In *ACL*, 2024.

[14] Xiaofeng Mao, Zhen Li, Chuanhao Li, Xiaojie Xu, Kaining Ying, Tong He, Jiangmiao Pang, Yu Qiao, and Kaipeng Zhang. Yume-1.5: A text-controlled interactive world generation model, 2025. URL https://arxiv.org/abs/2512.22096.

[15] Xiaofeng Mao, Shaoheng Lin, Zhen Li, Chuanhao Li, Wenshuo Peng, Tong He, Jiangmiao Pang, Mingmin Chi, Yu Qiao, and Kaipeng Zhang. Yume: An interactive world generation model. *arXiv preprint arXiv:2507.17744*, 2025.

[16] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[17] OpenAI. GPT-5 system card, 2025. URL https://openai.com/index/gpt-5-system-card/.

[18] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=a68SUt6zFt.

[19] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[20] Leqi Shen, Guoqiang Gong, Tao He, Yifeng Zhang, pengzhang liu, Sicheng Zhao, and Guiguang Ding. FastVID: Dynamic density pruning for fast video large language models. In *NeurIPS*, 2025.

[21] 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.

[22] Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive keyframe sampling for long video understanding. In *CVPR*, 2025.

[23] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

[24] V Team, Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, Shuaiqi Duan, Weihan Wang, Yan Wang, Yean Cheng, Zehai He, Zhe Su, Zhen Yang, Ziyang Pan, Aohan Zeng, Baoxu Wang, Bin Chen, Boyan Shi, Changyu Pang, Chenhui Zhang, Da Yin, Fan Yang, Guoqing Chen, Jiazheng Xu, Jiale Zhu, Jiali Chen, Jing Chen, Jinhao Chen, Jinghao Lin, Jinjiang Wang, Junjie Chen, Leqi Lei, Letian Gong, Leyi Pan, Mingdao Liu, Mingde Xu, Mingzhi Zhang, Qinkai Zheng, Sheng Yang, Shi Zhong,

Shiyu Huang, Shuyuan Zhao, Siyan Xue, Shangqin Tu, Shengbiao Meng, Tianshu Zhang, Tianwei Luo, Tianxiang Hao, Tianyu Tong, Wenkai Li, Wei Jia, Xiao Liu, Xiaohan Zhang, Xin Lyu, Xinyue Fan, Xuancheng Huang, Yanling Wang, Yadong Xue, Yanfeng Wang, Yanzi Wang, Yifan An, Yifan Du, Yiming Shi, Yiheng Huang, Yilin Niu, Yuan Wang, Yuanchang Yue, Yuchen Li, Yutao Zhang, Yuting Wang, Yu Wang, Yuxuan Zhang, Zhao Xue, Zhenyu Hou, Zhengxiao Du, Zihan Wang, Peng Zhang, Debing Liu, Bin Xu, Juanzi Li, Minlie Huang, Yuxiao Dong, and Jie Tang. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025.

[25] Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

[26] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-VL: A stronger and more general multimodal LLM. *arXiv preprint arXiv:2409.12191*, 2024.

[27] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

[28] Ziyang Wang, Shoubin Yu, Elias Stengel-Eskin, Jaehong Yoon, Feng Cheng, Gedas Bertasius, and Mohit Bansal. Videotree: Adaptive tree-based video representation for LLM reasoning on long videos. *CoRR*, abs/2405.19209, 2024. doi: 10.48550/ARXIV.2405.19209. URL `https://doi.org/10.48550/arXiv.2405.19209`.

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

[30] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. In *NeurIPS*, 2024.

[31] Yue Wu, Xuanhong Chen, Yufan Wu, Wen Li, Yuxi Lu, and Kairui Feng. Fastavatar: Towards unified and fast 3d avatar reconstruction with large gaussian reconstruction transformers. In *ICLR*, 2026.

[32] Mingze Xu, Mingfei Gao, Zhe Gan, Hong-You Chen, Zhengfeng Lai, Haiming Gang, Kai Kang, and Afshin Dehghan. Slowfast-llava: A strong training-free baseline for video large language models. *arXiv preprint arXiv:2407.15841*, 2024.

[33] Biao Yang, Bin Wen, Boyang Ding, Changyi Liu, Chenglong Chu, Chengru Song, Chongling Rao, Chuan Yi, Da Li, Dunju Zang, et al. Kwai keye-vl 1.5 technical report. *arXiv preprint arXiv:2509.01563*, 2025.

[34] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. In *CVPR*, 2025.

[35] Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, Jifeng Dai, and Xiaojie Jin. Flash-vstream: Memory-based real-time understanding for long video streams. *arXiv preprint arXiv:2406.08085*, 2024.

[36] Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. Frame context packing and drift prevention in next-frame-prediction video diffusion models. In *NeurIPS*, 2025.

[37] Shaojie Zhang, Jiahui Yang, Jianqin Yin, Zhenbo Luo, and Jian Luan. Q-frame: Query-aware frame selection and multi-resolution adaptation for video-llms. In *IEEE/CVF International Conference on Computer Vision, ICCV 2025, Honolulu, HI, USA, October 19-25, 2025*, pages 22056–22065. IEEE, 2025. doi: 10.1109/ICCV51701.2025.02048. URL `https://doi.org/10.1109/ICCV51701.2025.02048`.

[38] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Zhengyang Liang, Shitao Xiao, Minghao Qin, Xi Yang, Yongping Xiong, Bo Zhang, et al. Mlvu: Benchmarking multi-task long video understanding. In *CVPR*, 2025.

[39] Zirui Zhu, Hailun Xu, Yang Luo, Yong Liu, Kanchan Sarkar, Zhenheng Yang, and Yang You. FOCUS: efficient keyframe selection for long video understanding. *CoRR*, abs/2510.27280, 2025.

[40] Orr Zohar, Xiaohan Wang, Yann Dubois, Nikhil Mehta, Tong Xiao, Philippe Hansen-Estruch, Licheng Yu, Xiaofang Wang, Felix Juefei-Xu, Ning Zhang, Serena Yeung-Levy, and Xide Xia. Apollo: An exploration of video understanding in large multimodal models. In *CVPR*, 2025.

In the appendix, we first provide detailed description of the alternative strategies in Sec. A. We also present additional analysis on `FramePack-U` in Sec. B and fine-grained evaluation results in Sec. C. Finally, we discuss the limitations and failure cases in Sec. F.

A Alternative strategies

Binary-tree midpoint ordering. In the absence of query information, frame importance can be estimated purely from the temporal structure of the video. Inspired by the frame ordering of FramePack [36], the most naïve approach assigns ranks in chronological order, giving full-resolution quota to the earliest frames. This is problematic because the first frames of a long video are not necessarily more informative than later ones; in practice, this strategy reduces to uniform sampling of the early segment, discarding later temporal context entirely.

A more principled alternative is *binary-tree midpoint ordering*. It assigns ranks recursively: the temporally central frame of the full video is ranked first, followed by the midpoints of the left and right sub-intervals, and so on, proceeding breadth-first across the resulting binary tree. More formally, let the frames be indexed as $\{1, \ldots, N\}$. The rank-k frame is the midpoint of the sub-interval defined by the k-th node in a breadth-first traversal of the complete binary tree over the temporal axis, as detailed in Alg. 2. Such a design ensures that, for any budget $n_1 < N$, the n_1 anchor frames are distributed across the full temporal extent of the video rather than concentrated at one end, thus preserving global temporal coverage whilst allocating higher fidelity to temporally representative moments.

Algorithm 2 Binary-Tree Midpoint Ordering

Require: Number of candidate frames N
Ensure: Query-agnostic priority index list $R = [R_1, R_2, \ldots, R_N]$
Initialize queue $Q \leftarrow [(1, N)]$ ▷ Each entry is a temporal interval $[l, r]$
Initialize priority list $R \leftarrow [\,]$
while Q is not empty **do**
  $(l, r) \leftarrow Q.\mathrm{dequeue}()$
  $m \leftarrow \lfloor (l + r)/2 \rfloor$ ▷ Temporal midpoint
  Append m to R
  if $l \leq m - 1$ **then**
    $Q.\mathrm{enqueue}((l, m - 1))$ ▷ Left sub-interval
  end if
  if $m + 1 \leq r$ **then**
    $Q.\mathrm{enqueue}((m + 1, r))$ ▷ Right sub-interval
  end if
end while
return R

CLIP-guided sampling. Another straightforward alternative is to rely only on text–frame relevance for frame sampling and ranking. Given a text query q and video frames $\{x_i\}_{i=1}^{T}$, we use CLIP [19] to compute the cosine similarity between the text embedding of q and the visual embedding of each frame:

$$r_i = \cos\left(\phi_{\mathrm{img}}(x_i), \phi_{\mathrm{text}}(q)\right),$$

where $\phi_{\mathrm{img}}(\cdot)$ and $\phi_{\mathrm{text}}(\cdot)$ denote the CLIP image and text encoders, respectively. Frames are then sorted in descending order according to r_i, and the top N frames are selected as candidates for packing. The same ranking order is also used for resolution allocation: frames with higher CLIP similarity are assigned smaller downsampling factors and thus receive higher spatial resolutions, while lower-ranked frames are more aggressively compressed.

This strategy provides a simple query-aware baseline, as it explicitly prioritizes frames that are semantically aligned with the instruction. However, since it only considers query relevance, it may repeatedly select visually similar frames when several adjacent frames all match the text query. As a result, the selected high-resolution frames can become redundant and fail to cover complementary evidence across the video. In contrast, our default strategy augments CLIP relevance with a diversity-

Memory	Scheme	#Frame	LongVideoBench	MLVU	Video-MME
128 frames	Vanilla	128	68.0	75.9	73.6
	d1f16d2f448	464	74.4	83.9	76.3
	d1f64d2f256	320	73.5	83.1	77.2
	d1f96d2f128	224	73.6	83.1	**77.4**
	d1f16d2f320d4f512	848	**75.2**	**84.9**	76.7
256 frames	Vanilla	256	70.9	76.7	75.9
	d1f16d2f960	976	74.5	**84.5**	76.1
	d1f64d2f768	832	**74.8**	83.7	76.3
	d1f128d2f512	640	74.7	83.7	76.4
	d1f250d4f96	346	74.3	83.2	**77.9**
512 frames	Vanilla	512	71.8	78.0	74.4
	d1f128d2f1536	1664	**74.9**	82.1	76.3
	d1f256d2f1024	1280	74.7	82.2	75.9
	d1f384d2f512	896	74.6	**82.9**	76.0
	d1f480d2f128	608	74.3	81.0	**76.5**

Table 5: **Additional ablation study on resolution allocation schemes using Qwen3.6-27B.** We report results under 128-, 256-, and 512-frame memory budgets, where each budget denotes the cost equivalent to processing the corresponding number of full-resolution frames. All mixed-resolution schemes use the diversity-aware sampling strategy. Bold numbers indicate the best result within each memory budget.

aware greedy selection term, encouraging the packed frames to be both instruction-relevant and visually non-redundant.

Algorithm 3 CLIP-Guided Sampling and Ranking

Require: Video frames $\mathcal{V} = \{x_i\}_{i=1}^{T}$, text query q, number of packed frames N, packing scheme $\mathcal{S} = \{(d, f_d)\}_{d \in \{1,2,4,8,\dots\}}$
Ensure: Packed visual sequence $\mathcal{V}^*$
for $i = 1, \dots, T$ **do**
    Compute relevance score $r_i \leftarrow \mathrm{CLIP}(x_i, q)$
end for
Sort frame indices by descending relevance score:

$$R \leftarrow \mathrm{argsort}_{i \in \{1,\dots,T\}}(r_i)[:N]$$

Sort packing levels in $\mathcal{S}$ by increasing downsampling factor d
Initialize packed visual sequence $\mathcal{V}^* \leftarrow [\,]$
for $n = 1, \dots, N$ **do**
    Find d such that $\sum_{d'<d} f_{d'} < n \le \sum_{d' \le d} f_{d'}$.
    Pack frame x_{R_n} as $\hat{x}_n \leftarrow \mathrm{Resize}(x_{R_n}, 1/d)$
    Append $\hat{x}_n$ to $\mathcal{V}^*$
end for
Reorder $\mathcal{V}^*$ according to the original temporal order of the corresponding frames
return $\mathcal{V}^*$

B Additional analysis on `FramePack-U`

Additional ablation study on schemes. We further evaluate our method on various schemes across 128- and 256- frame memories. The full results are shown in Table 5.

Under the 128-frame memory budget, all diversity-aware schemes bring clear improvements over the vanilla baseline. Among them, `d1f16d2f320d4f512` achieves the best LongVideoBench and MLVU results, while `d1f96d2f128` obtains the best Video-MME score. This suggests that increasing temporal coverage with more low-resolution frames is helpful, but allocating a small number of higher-resolution frames remains important for preserving fine-grained visual evidence.

A similar trend can be observed under the 256-frame memory budget. Schemes such as `d1f16d2f960`, `d1f64d2f768`, and `d1f128d2f512` achieve comparable performance on LongVideoBench and MLVU, while d1f250d4f96 performs best on Video-MME. This indicates that there is no single

Memory	Scheme	Strategy	LongVideoBench	MLVU	Video-MME
128 frames	–	Vanilla	68.0	75.9	73.6
	d1f16d2f448	FramePack-U	**74.4**	**83.9**	76.3
		− Relevance	73.2 (-1.2)	81.5 (-2.4)	76.1 (-0.2)
		− Diversity	72.8 (-1.6)	81.7 (-2.2)	68.9 (-7.4)
	d1f64d2f256	FramePack-U	73.5	83.1	77.2
		− Relevance	72.2 (-1.3)	80.1 (-3.0)	76.2 (-1.0)
		− Diversity	72.5 (-1.0)	81.0 (-2.1)	69.4 (-7.8)
	d1f96d2f128	FramePack-U	73.6	83.1	**77.4**
		− Relevance	72.2 (-1.4)	78.3 (-4.8)	76.6 (-0.8)
		− Diversity	72.1 (-1.5)	80.6 (-2.5)	68.8 (-8.6)
256 frames	–	Vanilla	70.9	76.7	75.9
	d1f16d2f960	FramePack-U	74.5	**84.5**	76.1
		− Relevance	74.5 (+0.0)	82.0 (-2.5)	**77.0** (+0.9)
		− Diversity	73.6 (-0.9)	82.0 (-2.5)	69.7 (-6.4)
	d1f32d2f896	FramePack-U	**74.8**	83.7	76.0
		− Relevance	74.6 (-0.2)	82.2 (-1.5)	76.7 (+0.7)
		− Diversity	73.9 (-0.9)	81.8 (-1.9)	70.2 (-5.8)
	d1f128d2f512	FramePack-U	74.7	83.7	76.4
		− Relevance	74.3 (-0.4)	81.9 (-1.8)	76.2 (-0.2)
		− Diversity	73.7 (-1.0)	81.6 (-2.1)	70.0 (-6.4)
512 frames	–	Vanilla	71.8	78.0	74.4
	d1f128d2f1536	FramePack-U	**74.9**	82.1	**76.3**
		− Relevance	74.0 (-0.9)	82.3 (+0.2)	**76.3** (+0.0)
		− Diversity	73.9 (-1.0)	80.4 (-1.7)	69.8 (-6.5)
	d1f256d2f1024	FramePack-U	74.7	82.2	75.9
		− Relevance	74.6 (-0.1)	82.3 (+0.1)	76.0 (+0.1)
		− Diversity	73.8 (-0.9)	80.6 (-1.6)	69.6 (-6.3)
	d1f384d2f512	FramePack-U	74.6	**82.9**	76.0
		− Relevance	73.1 (-1.5)	81.1 (-1.8)	75.4 (-0.6)
		− Diversity	73.7 (-0.9)	80.8 (-2.1)	69.9 (-6.1)

Table 6: **Additional ablation study on prompt relevance and frame diversity using Qwen3.6-27B.** − *Relevance* removes prompt relevance and uses a prompt-agnostic diversity-based ordering, while − *Diversity* ranks frames only by query-frame similarity. `FramePack-U` combines prompt relevance and visual diversity. Values in parentheses denote the difference between each ablated variant and `FramePack-U`.

universally optimal allocation for all benchmarks; different tasks may prefer different balances between temporal coverage and spatial detail.

Additional ablation study on frame diversity. We provide additional ablation results on frame diversity under 128- and 256-frame memory budgets in Table 6. Here, the - *Diversity* variant removes the diversity-aware term and ranks frames only by query-frame similarity. Although this variant remains query-aware, it tends to select visually similar frames when multiple adjacent frames are relevant to the prompt, causing redundant high-resolution allocation.

Under the 128-frame memory budget, removing diversity leads to clear degradation, especially on Video-MME. For example, with `d1f16d2f448`, Video-MME drops from 76.3 to 68.9. The same trend holds for `d1f64d2f256` and `d1f96d2f128`, where Video-MME decreases by 7.8 and 8.6 points, respectively. Under the 256-frame memory budget, the effect remains consistent: removing diversity reduces Video-MME from 76.1 to 69.7 with `d1f16d2f960`, from 76.0 to 70.2 with `d1f32d2f896`, and from 76.4 to 70.0 with `d1f128d2f512`. These results show that query relevance alone is insufficient for frame packing. Visual diversity is critical for preventing redundant frames from occupying the limited high-resolution budget, thereby improving temporal coverage and robustness across benchmarks.

Additional ablation study on prompt relevance. We further analyze the effect of prompt relevance using the - *Relevance* variant, which removes the text-query signal and uses a prompt-agnostic diversity-based ordering. This baseline can preserve coarse temporal coverage, but it does not explicitly prioritize frames that contain task-relevant evidence.

Memory	Model	LongVideoBench	MLVU	Video-MME
128 frames (d1f96d2f128)	Qwen3.5-4B	+1.7	+4.0	+1.1
	Qwen3.5-9B	+2.8	+3.0	+0.6
	Qwen3.6-27B	+5.6	+7.2	+3.8
256 frames (d1f64d2f768)	Qwen3.5-4B	+0.2	+1.6	-0.9
	Qwen3.5-9B	+1.1	+1.4	-0.2
	Qwen3.6-27B	+3.9	+7.0	+0.4
512 frames (d1f256d2f1024)	Qwen3.5-4B	+0.9	-1.0	-1.2
	Qwen3.5-9B	+1.6	-0.6	-0.4
	Qwen3.6-27B	+2.5	+4.2	+1.5

Table 7: **Additional ablation study on model size.** We report the performance gain of `FramePack-U` over the vanilla baseline across different Qwen-series base models and memory budgets. Larger models generally obtain larger gains, suggesting that stronger VLMs can better exploit the packed multi-resolution visual evidence.

Under the 128-frame memory budget, removing relevance consistently hurts LongVideoBench and MLVU. For example, with `d1f16d2f448`, LongVideoBench decreases from 74.4 to 73.2 and MLVU decreases from 83.9 to 81.5. With `d1f96d2f128`, the drop on MLVU becomes larger, from 83.1 to 78.3. Under the 256-frame memory budget, - *Relevance* occasionally matches or slightly exceeds `FramePack-U` on a single metric, such as Video-MME for `d1f16d2f960`, but it is less stable across benchmarks and usually performs worse on MLVU. For instance, MLVU drops from 84.5 to 82.0 with `d1f16d2f960`, and from 83.7 to 82.2 with `d1f32d2f896`. These results indicate that prompt relevance helps allocate high-resolution capacity to frames that are more likely to contain instruction-relevant evidence. Combined with the diversity analysis above, this confirms that `FramePack-U` benefits from both components: prompt relevance identifies useful evidence, while frame diversity ensures non-redundant temporal coverage.

Additional ablation study on model size. We provide additional results on different model sizes under 128-, 256-, and 512-frame memory budgets in Table 7. We compare three Qwen-series backbones, including Qwen3.5-4B, Qwen3.5-9B, and Qwen3.6-27B, and report the performance gain of `FramePack-U` over the vanilla baseline under the same memory budget.

Overall, the benefit of `FramePack-U` becomes more pronounced as the base model scales up. Under the 128-frame memory budget, all three models benefit from `FramePack-U`, but the gain is much larger on Qwen3.6-27B. For example, Qwen3.6-27B improves by +5.6, +7.2, and +3.8 on LongVideoBench, MLVU, and Video-MME, respectively, while the gains on Qwen3.5-4B and Qwen3.5-9B are more moderate. This suggests that stronger VLMs are better able to exploit the additional temporally diverse evidence introduced by frame packing.

The same trend is more evident under larger memory budgets. At 256-frame memory, Qwen3.6-27B still obtains clear gains on LongVideoBench and MLVU, whereas smaller models show weaker or less stable improvements. At 512-frame memory, Qwen3.6-27B consistently improves across all benchmarks, while Qwen3.5-4B and Qwen3.5-9B improve on LongVideoBench but show slight degradation on MLVU or Video-MME. This indicates that when many more packed frames are provided, smaller VLMs may reach their capacity boundary and fail to effectively integrate the increased visual context. Therefore, the occasional degradation on smaller models is likely due to the limited reasoning and visual integration capability of the base model, rather than the packing strategy itself. These results further support that `FramePack-U` is especially beneficial for stronger VLMs, which can better leverage broad temporal coverage together with high-resolution evidence frames.

C Fine-grained results

We further report fine-grained results on the duration-based subsets of LongVideoBench and Video-MME. For LongVideoBench, we report the overall accuracy and the results on the 15s, 60s, 600s, and 3600s subsets. For Video-MME, we report the overall accuracy and the results on the short, medium, and long subsets.

As shown in Table 8, `FramePack-U` improves performance consistently across most fine-grained duration splits. The gains are especially pronounced on longer videos, where preserving broader

Model	Setting	Method	LVB Overall	LVB 15 / 60 / 600 / 3600	V-MME Overall	V-MME S / M / L
Qwen3.6-27B	128-frame memory `d1f16d2f448`	Vanilla	68.0	82.0 / 83.1 / 68.9 / 58.0	73.6	85.2 / 74.1 / 61.4
		`FramePack-U`	74.4	82.5 / **84.9** / 73.5 / **69.1**	76.3	85.4 / 77.7 / 65.7
		Δ	+6.4	+0.5 / +1.8 / +4.6 / +11.1	+2.7	+0.2 / +3.6 / +4.3
	256-frame memory `d1f250d4f96`	Vanilla	70.9	83.1 / 83.7 / 72.8 / 61.5	75.9	85.7 / 76.6 / 65.6
		`FramePack-U`	74.3	83.6 / 83.7 / 73.8 / 68.6	**77.9**	86.1 / **79.2** / **68.3**
		Δ	+3.4	+0.5 / +0.0 / +1.0 / +7.1	+2.0	+0.4 / +2.6 / +2.7
	512-frame memory `d1f128d2f1536`	Vanilla	71.8	82.0 / 83.1 / 72.3 / 64.5	74.4	**86.2** / 76.4 / 60.4
		`FramePack-U`	**74.9**	83.1 / 82.6 / **76.5** / 68.8	76.3	84.7 / 77.4 / 66.7
		Δ	+3.1	+1.1 / -0.5 / +4.2 / +4.3	+1.9	-1.5 / +1.0 / +6.3
Qwen3.5-35B-A3B	128-frame memory `d1f32d2f384`	Vanilla	59.4	81.0 / 75.6 / 55.3 / 50.2	66.7	79.6 / 66.4 / 54.1
		`FramePack-U`	72.3	83.6 / 82.6 / 73.1 / 64.7	76.4	84.8 / 77.7 / 66.9
		Δ	+12.9	+2.6 / +7.0 / +17.8 / +14.5	+9.7	+5.2 / +11.3 / +12.8
	256-frame memory `d1f64d2f768`	Vanilla	60.3	78.3 / 76.2 / 59.0 / 50.4	67.1	78.9 / 67.2 / 55.2
		`FramePack-U`	72.6	82.5 / 80.8 / 74.3 / 65.4	75.5	84.1 / 76.9 / 65.6
		Δ	+12.3	+4.2 / +4.6 / +15.3 / +15.0	+8.4	+5.2 / +9.7 / +10.4
	512-frame memory `d1f384d2f512`	Vanilla	59.8	74.6 / 75.0 / 60.0 / 50.2	67.3	78.4 / 67.0 / 56.3
		`FramePack-U`	73.0	**85.7** / 81.4 / 73.1 / 66.1	75.8	84.1 / 76.9 / 66.4
		Δ	+13.2	+11.1 / +6.4 / +13.1 / +15.9	+8.5	+5.7 / +9.9 / +10.1

Table 8: **Fine-grained results on LongVideoBench and Video-MME.** For LongVideoBench, we report the overall accuracy and the results on the 15s / 60s / 600s / 3600s subsets. For Video-MME, we report the overall accuracy and the short / medium / long subsets. Only the default diversity-aware packing strategy of `FramePack-U` is reported, with the vanilla baseline included as reference. Δ denotes the difference between `FramePack-U` and Vanilla under the same setting. Bold indicates the best result in each column or sub-column.

temporal context becomes more important. On Qwen3.6-27B, under the 128-frame memory budget, `FramePack-U` improves LongVideoBench 3600s accuracy from 58.0 to 69.1, yielding a gain of +11.1. It also improves the Video-MME long split from 61.4 to 65.7. Under the 256-frame memory budget, the gains remain clear on long videos, with LongVideoBench 3600s improving from 61.5 to 68.6 and Video-MME long improving from 65.6 to 68.3. These results indicate that multi-resolution packing is particularly helpful for long-duration understanding, where relevant evidence may appear sparsely across the video.

The improvement is even more significant on Qwen3.5-35B-A3B. For example, under the 128-frame memory budget, `FramePack-U` improves LongVideoBench 600s and 3600s accuracy from 55.3 and 50.2 to 73.1 and 64.7, respectively. It also improves Video-MME long from 54.1 to 66.9. Under the 512-frame memory budget, `FramePack-U` still brings large gains on long-duration subsets, improving LongVideoBench 3600s from 50.2 to 66.1 and Video-MME long from 56.3 to 66.4. These results suggest that `FramePack-U` effectively converts a fixed memory budget into broader temporal coverage, enabling the VLM to access more long-range evidence without increasing the visual memory cost.

D Runtime analysis

To clarify the preprocess overhead impact and study the end-to-end latency of our method, we conduct evaluations on LongVideoBench using Qwen3.5-4B, Qwen3.5-9B, and Qwen3.6-27B. To understand the scaling pattern as the number of frames increases, we report the average preprocess overhead of different video duration tiers and discuss the relativity of different inference stages. For both our method and vanilla models, we set fps=1.0 for candidate frames. We use the `d1f96d2f128` scheme for our method. All samples are evaluated on a single NVIDIA A800 GPU. Due to memory constraint, we set the maximum sampled frames of Qwen3.6-27B to 224 frames.

Memory overhead. As shown in Table 9, the memory overhead introduced by the CLIP-based scoring process is constant and relatively trivial. In comparison, the vanilla models suffer from a huge increase of the input tokens with frame scaling, while our method keeps a constant memory upper bound with even better performance.

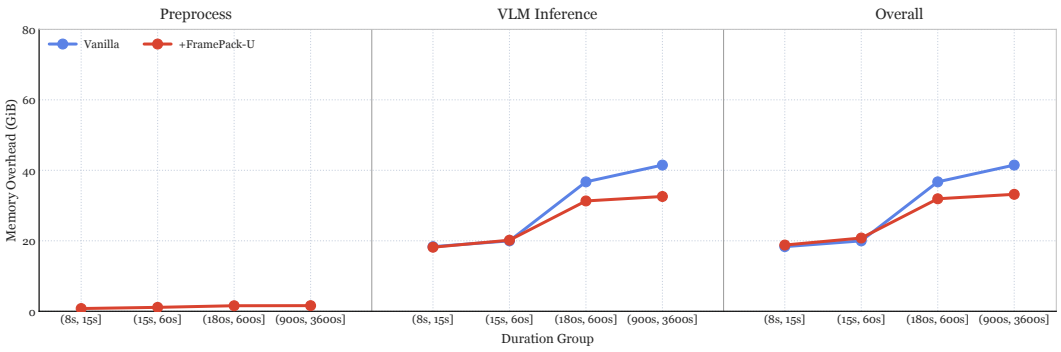

Figure 5: **Memory overhead of `FramePack-U` on Qwen3.5-9B.** We evaluate the preprocess, VLM inference and overall memory overhead of both vanilla models and `FramePack-U` on LongVideoBench. Results of different video duration tiers are reported. More results can be found at Table 9.

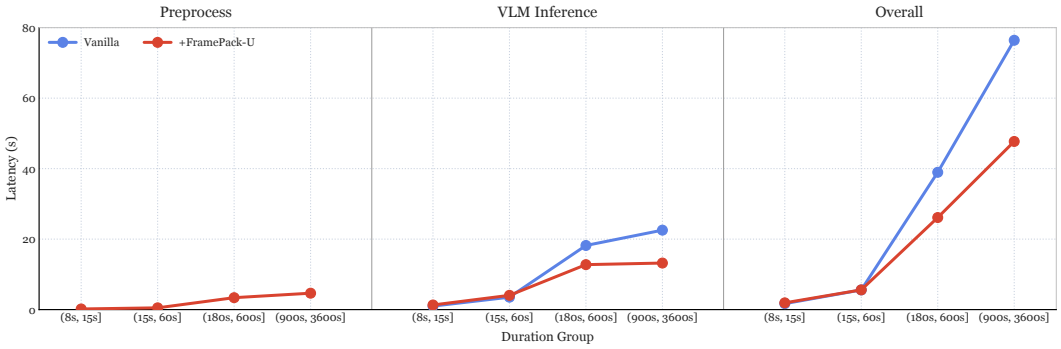

Figure 6: **Latency evaluation of `FramePack-U` on Qwen3.5-9B.** We evaluate the preprocess, VLM inference and overall latency of both vanilla models and `FramePack-U` on LongVideoBench. Results of different video duration groups are reported. More results can be found at Table 10.

Inference latency. Table 10 compares the inference latency of `FramePack-U` and the vanilla models. It can be observed that compared to the VLM inference stage, the latency of the CLIP-based scoring (preprocess) stage is relatively negligible. It is also noticeable that `FramePack-U` effectively reduces VLM inference and overall latency, making itself more efficient compared to the vanilla model. As for the results of Qwen3.6-27B, the higher overall latency for `FramePack-U` on extremely long videos stems directly from the I/O stage due to the difference in the number of loaded frames. Due to the GPU memory constraints of the massive Qwen3.6-27B model, the vanilla model is strictly restricted to sample a maximum of 224 frames. In contrast, `FramePack-U` evaluates candidate frames at fps=1.0, meaning it actually decodes up to 3600 frames for a 3600s video, while still safely operating within the constant memory budget. Decoding 3600 frames from the disk incurs a significantly higher I/O overhead than decoding merely 224 frames. This massive hardware-bound I/O difference inevitably offsets the time saved during the VLM inference stage. In a strictly equivalent comparison where both methods sample candidate frames at fps=1.0, `FramePack-U` would exhibit drastically lower overall latency, whereas the vanilla model would simply crash due to Out-of-Memory or suffer from catastrophic inference delays.

E More comparisons with similar methods

Token compression and key frame selection is a recognized paradigm in efficient video LLMs. Methods like SlowFast-LLaVA [32], Q-Frame [37] and LongVU [21] explore how to identify tokens or frames with key information in different ways. `FramePack-U` differs from these works in the following aspects:

Stage	Model	15s	60s	600s	3600s	Avg.
Preprocess	Qwen3.5-4B	–	–	–	–	–
	+ FramePack-U	0.779	1.117	1.576	1.602	1.288
	Qwen3.5-9B	–	–	–	–	–
	+ FramePack-U	0.777	1.115	1.574	1.600	1.286
	Qwen3.6-27B	–	–	–	–	–
	+ FramePack-U	0.775	1.115	1.575	1.600	1.307
VLM Inference	Qwen3.5-4B	9.207	10.723	26.174	30.584	19.860
	+ FramePack-U	9.033	10.871	21.145	22.311	16.294
	Qwen3.5-9B	18.337	19.980	36.726	41.483	29.877
	+ FramePack-U	18.178	20.171	31.306	32.570	26.048
	Qwen3.6-27B	52.217	54.630	73.137	74.633	64.473
	+ FramePack-U	52.139	55.065	71.420	73.275	63.697
Overall	Qwen3.5-4B	9.207	10.723	26.174	30.584	19.860
	+ FramePack-U	9.650	11.488	21.762	22.928	16.911
	Qwen3.5-9B	18.337	19.980	36.726	41.483	29.877
	+ FramePack-U	18.794	20.786	31.922	33.185	26.664
	Qwen3.6-27B	52.217	54.630	73.137	74.633	64.473
	+ FramePack-U	52.755	55.680	72.035	73.890	64.313

Table 9: **Memory overhead (GiB) across different model sizes.** We evaluate the memory overhead of three inference stages on different video duration tiers of LongVideoBench. Preprocess stage represents the introduced CLIP-based frame ranking process. VLM Inference stage denotes the inference process of the off-the-shelf VLM. Overall results reflect the peak memory used during the end-to-end process.

SlowFast-LLaVA relies on a rigid, query-agnostic two-stream architecture. Its resolution assignment is strictly dictated by a predetermined temporal frame rate. The Slow pathway operates at a low frame rate with high spatial resolution, while the Fast pathway utilizes a high frame rate with aggressive spatial pooling. It assigns resolution simply based on temporal position, completely ignoring whether the visual content is relevant to the instruction. In stark contrast, `FramePack-U` dynamically and deterministically assigns resolutions based on content relevance and visual diversity.

Q-Frame assigns High, Medium, and Low resolutions using a probabilistic Gumbel-Max sampling trick applied to CLIP scores, followed by static hyperparameter thresholding to rank frames into resolution tiers. `FramePack-U` is mathematically governed by a strict constant memory constraint, formulated as Eq. 3. Our resolution packing is not a heuristic ranking strategy. It is a deterministic, hierarchical optimization that mathematically guarantees O(1) memory complexity regardless of the video length.

LongVU relies heavily on an extensive video-language fine-tuning stage, whereas `FramePack-U` operates entirely training-free on off-the-shelf VLMs. Methodologically, LongVU employs a disjointed pipeline that query-agnostically discards frames based on DINOv2 [18] visual similarity before introducing any cross-modal text queries. This creates a severe bottleneck that blindly destroys instruction-relevant evidence simply due to low-level visual redundancy. In contrast, `FramePack-U`'s deterministic greedy algorithm evaluates semantic text-relevance and visual diversity simultaneously in a single, unified step. This joint optimization ensures that our strict memory budget is precisely allocated without prematurely dropping query-critical frames.

Empirical Comparison. Empirically, we conduct controlled experiments to directly compare our strategy with Q-Frame and LongVU. For fairness, we adopt the same VLM (CLIP-ViT-B/32) for feature extraction and similarity calculation, and apply these methods on the same VLM (Qwen3.5-4B) under the same resolution scheme contract (`d1f96d2f128`). For the training-based method

Stage	Model	15s	60s	600s	3600s	Avg.
Preprocess	Qwen3.5-4B	–	–	–	–	–
	+ `FramePack-U`	0.183	0.518	3.460	4.729	2.354
	Qwen3.5-9B	–	–	–	–	–
	+ `FramePack-U`	0.181	0.502	3.383	4.655	2.309
	Qwen3.6-27B	–	–	–	–	–
	+ `FramePack-U`	0.169	0.480	3.210	4.366	2.178
VLM Inference	Qwen3.5-4B	0.787	2.964	14.322	17.854	9.486
	+ `FramePack-U`	1.027	3.370	9.878	9.933	6.336
	Qwen3.5-9B	0.993	3.523	18.196	22.543	11.965
	+ `FramePack-U`	1.311	4.049	12.758	13.208	8.213
	Qwen3.6-27B	1.857	6.544	33.778	35.847	20.710
	+ `FramePack-U`	2.335	7.503	31.914	34.030	20.022
Overall	Qwen3.5-4B	1.490	5.030	35.143	71.840	29.744
	+ `FramePack-U`	1.639	4.971	23.501	45.093	19.639
	Qwen3.5-9B	1.692	5.584	38.963	76.421	32.178
	+ `FramePack-U`	1.916	5.600	26.125	47.710	21.264
	Qwen3.6-27B	2.541	8.560	48.875	55.416	30.364
	+ `FramePack-U`	2.917	8.986	44.652	66.767	32.402

Table 10: **Inference latency (s) across different model sizes.** We evaluate the inference latency of three inference stages on different video duration tiers of LongVideoBench. Preprocess stage represents the introduced CLIP-based frame ranking process. VLM Inference stage denotes the inference process of the off-the-shelf VLM. Overall results reflect the total latency during the end-to-end process. Noticeably, we restrict the vanilla Qwen3.6-27B model to sample a maximum of 224 frames for the memory constraint, leading to a significantly lower I/O latency.

Method	LVB	MLVU	VideoMME	Overall
Vanilla Model	64.5	70.3	70.5	68.4
+ Q-Frame	62.6	**76.4**	66.1	68.4
Δ	-1.9	+6.1	-4.4	+0.0
+ LongVU-Style Ranking	58.5	71.5	63.9	64.6
Δ	-6.0	+1.2	-6.6	-3.8
+ `FramePack-U`	**66.2**	74.3	**71.6**	**70.7**
Δ	+1.7	+4.0	+1.1	+2.3

Table 11: **Performance comparison with similar token compression or key frame selection methods.** Δ denotes the performance gap with the vanilla model. Bold indicates the best result in each column.

LongVU, we implement a LongVU-style frame ranking strategy for resolution allocation. These experiments control the same resolution assignment process and isolate the ranking strategy.

Table 11 shows that our method achieves superior performance, while other methods demonstrate a lack of robustness or even degradation. These results prove that our ranking strategy effectively distinguishes the most critical frames from less informative ones.

F Limitations

Despite its simplicity and effectiveness, `FramePack-U` still has several limitations. First, the benefit of frame packing depends on the capacity of the underlying VLM. Although `FramePack-U` can pack more temporally diverse frames into the same memory budget, smaller models may not be able to effectively integrate and reason over the increased number of visual observations. As discussed in

Sec. 4.2, this can lead to unstable gains under larger memory budgets. For example, on smaller Qwen-series models, increasing the number of packed frames sometimes improves LongVideoBench but slightly degrades MLVU or Video-MME. We attribute this failure case to the capability boundary of the base model rather than the packing strategy itself: when the model lacks sufficient visual reasoning capacity, providing more contextual frames can introduce additional distraction instead of useful evidence.

Second, `FramePack-U` is most naturally applicable to VLMs that encode video inputs as a sequence of independently processed image frames. This design allows us to assign different spatial resolutions to different frames before visual encoding. However, some recent video VLMs adopt grid-based or tightly coupled video encoders, where frames are first organized into a fixed spatiotemporal grid or processed jointly by a video-specific visual module. For such models, e.g., Eagle2.5 [3]-style architectures, changing the resolution of individual frames may break the expected input format or require non-trivial modifications to the visual encoder and preprocessing pipeline. As a result, `FramePack-U` is currently less plug-and-play for these grid-based video architectures.

Third, the current implementation relies on CLIP similarity as a lightweight proxy for prompt relevance. While this choice keeps the method training-free and efficient, CLIP may fail to capture fine-grained temporal events, subtle interactions, or reasoning-heavy cues that are not easily recognized from individual frames. In such cases, the relevance score may under-prioritize frames that are important for answering the question. Similarly, the diversity term is based on visual feature similarity and may not fully reflect semantic complementarity. Future work could explore stronger query-frame scoring functions or lightweight temporal-aware relevance models while preserving the training-free nature of the framework.

Finally, `FramePack-U` introduces a scheme-selection hyperparameter that controls the trade-off between temporal coverage and per-frame spatial fidelity. Although our ablations show that the method is robust across a range of mixed-resolution schemes, the optimal allocation can still vary across models, benchmarks, and video domains. A more adaptive strategy that automatically selects the packing scheme based on video length, query type, and model capacity would further improve usability and robustness.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The paper's contributions and scope are claimed in the Abstract and Introduction sections.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discuss the limitations in Sec. F.

   Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not include theoretical results.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper fully discloses all the information needed in Sec. 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.

- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper provides open access to the data and code.

   Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: The paper specifies all the test details in Sec. 4. No training process is needed.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The paper does not report error bars.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper provides sufficient information on the computer resources.

Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [N/A]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite all papers that are mentioned.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We discuss the information in the Appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are used solely in grammar checking, and trivial works in the development.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.