VIKEY: Enhancing Temporal Understanding in Videos via Visual Prompting Supplementary material

Contents

A. Analysis Details

A.1. Details of Positional Embedding Degradation

We now describe how we manipulate rotary position embeddings under the temporal-only degradation and full-collapse settings.

Standard RoPE. Standard rotary position embeddings (RoPE) encode token order using a single 1D rotational embedding over the concatenated text–vision sequence, such that each visual token receives a unique position index along the sequence dimension. Let L_t denote the number of text tokens, N the number of visual tokens per frame, k the (1-based) frame index, and j the spatial token index within a frame. In the original setting, the position index assigned to a visual token is

$$\text{pos}(k, j) = L_t + (k - 1)N + j,$$

where the offset $(k - 1)N$ encodes frame order and j preserves within-frame spatial structure.

Under *temporal-only degradation*, we remove the frame-dependent offset and assign the same index pattern to all frames:

$$\text{pos}(k, j) \leftarrow L_t + j.$$

This eliminates positional signals that encode frame order while preserving spatial ordering within each frame. Under *full collapse*, we further map all tokens to a single index,

$$\text{pos}(k, j) \leftarrow L_t,$$

removing both temporal and spatial positional distinctions.

M-RoPE. Multi-scale RoPE (M-RoPE) extends standard RoPE by decomposing positional information into multiple components, typically a temporal index and spatial indices, which are applied at different granularities to the Q/K representations. Each visual token is associated with a triplet

$$(t, h, w),$$

where t denotes the temporal index (frame) and (h, w) encode spatial coordinates within the frame.

In the original M-RoPE configuration, temporal ordering is represented by variation in t across frames, and (h, w) preserve the spatial layout. Under *temporal-only degradation*, we fix the temporal component across frames while keeping spatial components unchanged:

$$(t, h, w) \leftarrow (t_0, h, w),$$

for some constant t_0, thereby removing temporal order but preserving spatial structure. Under *full collapse*, we map all tokens to a single triplet,

$$(t, h, w) \leftarrow (t_0, h_0, w_0),$$

for fixed (t_0, h_0, w_0), eliminating both temporal and spatial positional cues in the embedding.

Relation to M-RoPE. This experiment also helps clarify the distinction between VP and M-RoPE. Although Qwen2.5 employs M-RoPE to encode temporal order within the embedding space, this mechanism operates implicitly through relative token positions. In contrast, VP introduces an explicit temporal signal in pixel space during visual encoding. As shown in Fig. 3(b) of the main paper, VP consistently improves performance even when positional information is degraded, indicating that its contribution goes beyond positional embeddings.

A.2. Details of Frame-Level Referencing

Settings. We construct a synthetic frame-referencing set from the temporal reasoning split of VideoMME by uniformly sampling $N \in 8, 16, 32, 64$ frames per video and overlaying a black-and-white panda image on exactly one randomly chosen frame. Frame-index prompts (frame #i) are inserted at a fixed corner (TL, TR, BL, or BR) with font size controlled as in Sec. 4.2 of the main paper (hyperparameter s=12), while all other visual content is kept unchanged.

Table 1. Results showing the accuracy of reverse lookup using VP, measured with a tolerance of ± 1 frame.

Position	8 frames	16 frames	32 frames	64 frames	average
–	69.01	35.67	17.54	7.60	32.46
TL	**100.0**	99.42	**100.0**	99.42	99.71
TR	**100.0**	99.42	**100.0**	97.66	99.27
BL	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
BR	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**

Additional Positional Bias Results. In Sec. 3.2 of the main paper, we demonstrated that inserting VP enables the model to recognize frame indices and use them as references for frame-level lookup and reverse lookup. In the reverse lookup experiments, we observed that placing the VP at the bottom-left (BL) or bottom-right (BR) corners led to perfect accuracy, whereas top-left (TL) and top-right (TR) placements resulted in reduced accuracy, with a notable tendency for the model to answer with adjacent frame numbers (either immediately before or after the target). Table 1 shows the accuracy under a relaxed criterion allowing an error margin of ± 1 frame. Under this setting, even TL and TR positions achieved near-perfect accuracy across all frame counts.

A.3. Details of Attention-Based Analysis

To better understand how visual prompting (VP) influences the attention mechanism in VideoLLMs, we perform a layer-wise attention analysis using the prefilling stage of the language model. Specifically, following prior analyses of cross-modal attention in vision-language models [4–6], we extract attention values from text tokens (queries) to image tokens (keys) before any autoregressive decoding takes place. This allows us to isolate how much attention the model allocates to the visual input based solely on the prompt.

Settings. We use the same input video and prompt across conditions with and without VP, and compare the resulting attention distributions. The prompt consists of the full user query as used in each benchmark [2, 10, 12, 17]. For each benchmark, we randomly sample 100 examples using seed 42 to ensure consistency across experiments.

Implementation. We modify the model's forward pass to cache the attention weights from all self-attention layers during the prefilling stage. The analysis is conducted on a LLaVA-Video [20] model with 32 transformer layers. For each layer, we average the attention values across all heads to obtain a single attention score per layer.

Table 2. **Statistics of the analysis subsets.** For each dataset, we report the number of questions, number of videos, and video length statistics (in seconds) for the 100-example evaluation subsets used in our analysis.

Dataset	Category	Questions	Videos	Min (s)	Max (s)	Mean (s)	Std (s)
TempCompass	EO	55	45	3.04	27.20	14.59	6.99
	AC	45	39	3.00	24.68	10.46	5.19
	Total	100	84	3.00	27.20	12.67	6.55
MVBench	AS	51	49	8.20	43.20	21.12	8.67
	ST	49	49	20.00	20.00	20.00	0.00
	Total	100	98	8.20	43.20	20.56	6.16
VideoMME	TR	100	99	56.89	3540.14	1474.89	1078.71
LongVideoBench	E3E	31	28	11.50	2186.42	735.13	594.84
	O3O	30	24	7.98	2041.01	564.30	575.01
	SSS	39	29	8.57	1652.82	550.32	460.63
	Total	100	81	7.98	2186.42	618.35	550.89

Computation. Let $A^{(l,h)} \in \mathbb{R}^{T \times S}$ denote the attention matrix at layer l and head h, where T is the number of text tokens and S is the number of image tokens. We compute the mean attention over all query tokens and attention heads:

$$\bar{A}^{(l)} = \frac{1}{H} \sum_{h=1}^{H} \frac{1}{T} \sum_{t=1}^{T} \sum_{s=1}^{S} A_{t,s}^{(l,h)}, \tag{1}$$

where H is the total number of attention heads. We then calculate the relative change in attention between the VP and no-VP settings at each layer, as reported in Fig. 2(c) of the main paper.

Query Selection. Following the convention in the prefilling stage of VideoLLMs, we use the final token of the input prompt (i.e., the last token of the user query) as the attention query. This aligns with how the model typically attends to visual inputs before generating an answer.

Interpretation. The results show that VP increases attention to image tokens without changing the number of image tokens themselves. The increase is especially prominent in mid-to-late layers (layers 4–6, 11–14, and post-21), suggesting that VP enhances cross-modal alignment at deeper stages of processing, particularly relevant to temporal reasoning.

A.4. Details of the Analysis Subsets

Table 2 summarizes the evaluation subsets used in our analysis, where 100 examples were randomly sampled from each of the four datasets. Compared with the full temporal evaluation splits in Table 6, the sampled subsets show broadly similar distributions, especially in video length statistics across datasets and categories, indicating that the analysis subsets do not substantially deviate from the full benchmark distributions.

B. Method Details

B.1. Sequential Visual Prompting

Figure 1. **Effect of VP Design Parameters.** (a) Results with varying values of the font size control parameter s, and (b) results comparing the presence or absence of text outlines controlled by o.

Implementation Details. We implement sequential visual prompting with a script that iterates over each video folder, sorts frames in chronological order, and overlays a text label of the form "frame #i" on every frame at a fixed corner (TL, TR, BL, or BR). The font size is automatically scaled as $\text{fontsize} = \lfloor \min(\text{width}, \text{height})/s \rfloor$, and the index is rendered in red text following prior works [15, 18]. The resulting VP-injected frames are shown in Fig. 1.

B.2. Keyword-Frame Mapping (KFM)

We extract keywords exclusively from the question text. In cases where a prompt contains both the question and the answer options in a single string, we first separate them and apply keyword extraction only to the question portion. This keeps the mapping focused on the actual information request of the prompt and avoids introducing option-specific cues into the extraction process.

C. Additional Experiments

C.1. Additional Evaluation Results

C.1.1. Results on Non-Temporal Categories

To examine how VIKEY performs on a broader range of tasks beyond temporal reasoning, we report category-wise results on the general video understanding benchmark, VideoMME [2]. We observe that even for tasks not explicitly focused on

Algorithm 1 Keyword–Frame Mapping (KFM)

Input: user query q, sampled frames $\{\hat{\mathbf{I}}_i\}_{i=1}^{F}$, text tokens $\mathbf{Z}_{\text{text}}$, similarity threshold τ
Output: augmented text tokens $\hat{\mathbf{Z}}_{\text{text}}$

$\mathbf{W} \leftarrow \text{KeywordExtract}(q)$; // $\mathbf{W} = \{w_1, \ldots, w_m\}$
$\mathcal{K} \leftarrow \emptyset$; // set of mapped frame indices
for *each keyword* $w_j \in \mathbf{W}$ **do**
  for $i \leftarrow 1$ **to** F **do**
    $\mathbf{f}_i \leftarrow \text{CLIP}_{\text{img}}(\hat{\mathbf{I}}_i)$ $\mathbf{g}_j \leftarrow \text{CLIP}_{\text{text}}(w_j)$ $\text{sim}_i \leftarrow \cos(\mathbf{f}_i, \mathbf{g}_j)$
  $k_j^* \leftarrow \arg\max_i \text{sim}_i$ **if** $\text{sim}_{k_j^*} \geq \tau$ **then**
    $\mathcal{K} \leftarrow \mathcal{K} \cup \{k_j^*\}$; // map w_j to frame k_j^*
$\hat{\mathbf{Z}}_{\text{text}} \leftarrow \text{InsertIndex}(\mathbf{Z}_{\text{text}}, \mathcal{K})$ **return** $\hat{\mathbf{Z}}_{\text{text}}$

Table 3. **Results on Non-Temporal Categories.** Evaluation on the non-temporal categories of the VideoMME [2] dataset. The reported categories include: Action Reasoning (ARs), Action Recognition (ARc), Attribute Perception (AP), Counting Problem (CP), Information Synopsis (IS), OCR Problems (OP), Object Reasoning (ORs), Object Recognition (ORc), Spatial Perception (SP), Spatial Reasoning (SR), and Temporal Perception (TP).

Models	ARs	ARc	AP	CP	IS	OP	ORs	ORc	SP	SR	TP	mAcc
64 Frames (1 FPS)												
Baseline [20]	56.84	62.30	76.92	48.13	75.23	64.03	57.27	71.19	64.81	78.57	70.91	66.02
+ Ours	55.79	65.81	76.02	46.64	75.23	63.31	58.37	70.34	64.81	80.36	70.91	66.15 ▲
20% Frames												
Baseline [20]	55.09	57.51	67.42	41.79	73.99	58.99	55.73	61.02	57.41	83.93	60.00	59.44
+ Ours	54.74	58.15	68.78	41.04	73.68	54.68	55.95	62.71	59.26	82.14	63.64	61.34 ▲

temporal understanding, incorporating frame indices does not hinder performance and may offer auxiliary benefits for scene-level comprehension.

C.1.2. Results Combined with the Keyframe Selection Method

Table 4. **Integration with Keyframe Selection Methods.** Results showing the effect of combining our method with the state-of-the-art keyframe selection model AKS [16] on long video datasets, VideoMME [2] and LongVideoBench [17]. This evaluates performance when applied to sampled frames from lengthy video inputs.

Models	Sampling	VideoMME	LongVideoBench			
		TR	E3E	O3O	SSS	total
64 Frames (1 FPS)						
Baseline [20]	AKS [16]	51.41	67.02	57.58	41.24	54.86
+ Ours	AKS [16]	**53.67**	72.34	57.58	39.18	**56.03**
20% Frames						
Baseline [20]	AKS [16]	45.20	63.83	48.48	39.13	48.25
+ Ours	AKS [16]	**49.72**	63.83	57.58	31.96	**50.19**

Table 4 presents the results of applying keyframe-based sampling instead of uniform sampling in long-video scenarios. For this experiment, we used AKS [16], a state-of-the-art keyframe selection method. The results show a clear performance improvement over the baseline. This improvement likely stems from the increased inclusion of answer-relevant frames, particularly in the 64-frame setting, which raises the likelihood of retrieving frames aligned with extracted keywords and enables more accurate keyword-frame mapping. More broadly, this benefit can be extended to other keyframe selection methods [7, 11, 13, 19].

C.1.3. Additional Performance Comparision with Token-Level Method

In Section 5.2 of the main paper, we compare our method against token-level pruning approaches under various frame-sampling ratios. Table 5 shows the quantitative results for the 20% and 30% sampling settings. Overall, our approach consis-

Table 5. **Comparision with Token-Level Pruning Method.** Results on the temporal understanding subsets of the TempCompass [12], MVBench [10], VideoMME [2], and LongVideoBench [17] datasets.

Models	TempCompass			MVBench			VideoMME	LongVideoBench			
	EO	AC	total	AS	ST	total	TR	E3E	O3O	SSS	total
30% Frames (Tokens)											
Baseline [20]	71.24	69.32	70.27	67.00	90.00	78.50	44.63	63.83	45.45	36.08	48.64
+ Ours	73.63	71.95	72.78	67.50	93.00	80.25	50.28	64.89	46.97	38.14	50.19
+ Token Pruning [3]	71.97	63.14	67.51	65.50	91.00	78.25	45.20	64.89	45.45	32.99	47.86
20% Frames (Tokens)											
Baseline [20]	66.11	65.55	65.53	64.50	89.50	77.00	44.07	63.83	48.48	32.99	48.25
+ Ours	72.33	72.09	72.21	68.50	91.00	79.75	50.28	62.77	57.58	32.99	50.19
+ Token Pruning [3]	68.14	58.10	63.07	58.50	91.50	75.00	45.20	63.83	42.42	32.99	46.69

tently outperforms both the baseline and token-level methods across different sampling scenarios.

C.1.4. Additional Qualitative Results

In Fig. 2, we illustrate how our methodology jointly leverages frame-index visual prompting and keyword–frame mapping to improve temporal reasoning. In the first example, the keyword *"picks up the broom"* is mapped to frame 5, allowing the model to correctly identify what happens immediately before that moment. In the second example, the keywords *"poolside warm-up"* and *"pull-up scene"* are aligned to frames 2 and 4, respectively, enabling the model to reason about the events that take place between these points. In these cases, queries are also augmented with the corresponding frame indices, such as "What happens right before the scene where Mr. Bean picks up the broom (frame 5) in a room?" and "What happens between the poolside warm-up (frame 2) scene and the pull-up (frame 4) scene?", which encourages the model to explicitly reason over the numbered frames. In the third example, no keyword is mapped to any frame, so the model relies solely on the injected frame indices to recover the correct temporal order. These results highlight that our methodology effectively combines both visual prompting and keyword–frame correspondence to produce consistent temporal ordering across diverse open-ended queries.

C.2. Additional Ablation Results

C.2.1. Ablation on hyperparmeter τ

Fig. 3 illustrates the impact of varying the mapping threshold τ in the KFM module. When τ is set too low, irrelevant frames may be mapped to keywords, introducing noise and degrading performance. Conversely, a threshold that is too high reduces the number of successful mappings, thereby limiting the effectiveness of KFM. This trade-off often results in a single performance peak at an intermediate τ value, as observed in the graph. Notably, the TempCompass [12] dataset remains unaffected by changes in τ, as its questions lack identifiable keywords, rendering the KFM module inactive and resulting in stable performance across all threshold settings.

C.2.2. Ablation on hyperparmeter s

Fig. 4(a) illustrates how model performance varies with different values of the font size control parameter s. As shown in Fig. 4(b), larger fonts improve the visibility of the frame numbers but may occlude key objects or text in the scene, potentially hindering recognition. Conversely, excessively small fonts are difficult to discern. These results suggest that the optimal font size is dataset-dependent, balancing visibility with minimal interference.

C.2.3. Ablation on VP position

In Section 3.2 of the main paper, we observed that the VideoLLM exhibits a positional bias toward visual prompts (VP), which motivated our choice to place the VP at the bottom-left (BL) corner. Figure 5 presents the results of inserting the VP at alternative positions. Consistent with our analysis, the BL placement yields the highest performance across all datasets.

D. Implementation Details

D.1. Details of Evaluation Datasets

Our method works reliably across a broad range of temporal reasoning tasks, but is particularly effective in scenarios with frequent scene transitions or questions about what happens before or after specific moments. For this reason, we focus on

Query: What happens right before the scene where Mr. Bean picks up the broom in a room?

LLaVA-Video-7B ✗

Mr. Bean is seen in a theater, **sitting next to a woman with short hair and a green dress, watching a performance.**

LLaVA-Video-7B + Ours ✓

In **frame 4**, **Mr. Bean is on stage with a man in a purple suit and bow tie, holding a bouquet of yellow flowers**.

Query: What happens between the poolside warm-up scene and the pull-up scene?

LLaVA-Video-7B ✗

The scene transitions from the poolside warm-up to the pull-up scene, indicating a shift in activities.

LLaVA-Video-7B + Ours ✓

The person is seen eating breakfast at a cafeteria table (frame 3) between the poolside warm-up **(frame 2)** scene and the pull-up **(frame 4)** scene.

Query: Describe all the materials in chronological order.

LLaVA-Video-7B ✗

The materials that appear in the video in chronological order are a **small glass bottle with a cork stopper**, **a clear plastic cup filled with water**, **a tube of E6000 industrial adhesive, a blue food dye bottle, and a clear plastic container with a yellowish liquid**.

LLaVA-Video-7B + Ours ✓

The materials in chronological order are: **a clear glass bottle with a cork stopper, a small plastic bottle of blue food dye, a tube of E6000 industrial adhesive, a clear plastic cup filled with water, and a clear glass filled with water.**

Figure 2. **Additional qualitative temporal results.** Examples where our method provides temporally consistent answers, while the baseline VideoLLM fails to correctly capture the order of events.

benchmark categories that require explicit reasoning over event and scene changes, such as the temporal subsets of TempCompass and MVBench. To verify that these gains also hold in more complex and long-form settings, we additionally evaluate on the temporal reasoning splits of VideoMME and LongVideoBench.

TempCompass. TempCompass [12] is a diagnostic benchmark designed to probe fine-grained temporal perception in Video LLMs across diverse temporal aspects and task formats. It contains diverse videos paired with four query types: caption matching, captioning, multiple-choice QA, and yes/no QA. In our experiments, we use only the temporal reasoning subset, namely the Event Order (EO) and Attribute Change (AC) categories. EO evaluates whether the model can infer the chronological order of events, while AC tests whether it can track how object attributes or global scene states evolve over time.

MVBench. MVBench [10] is a comprehensive multi-modal video understanding benchmark that evaluates temporal skills beyond what can be captured from single static frames. It covers 20 video tasks derived from image-based benchmarks and

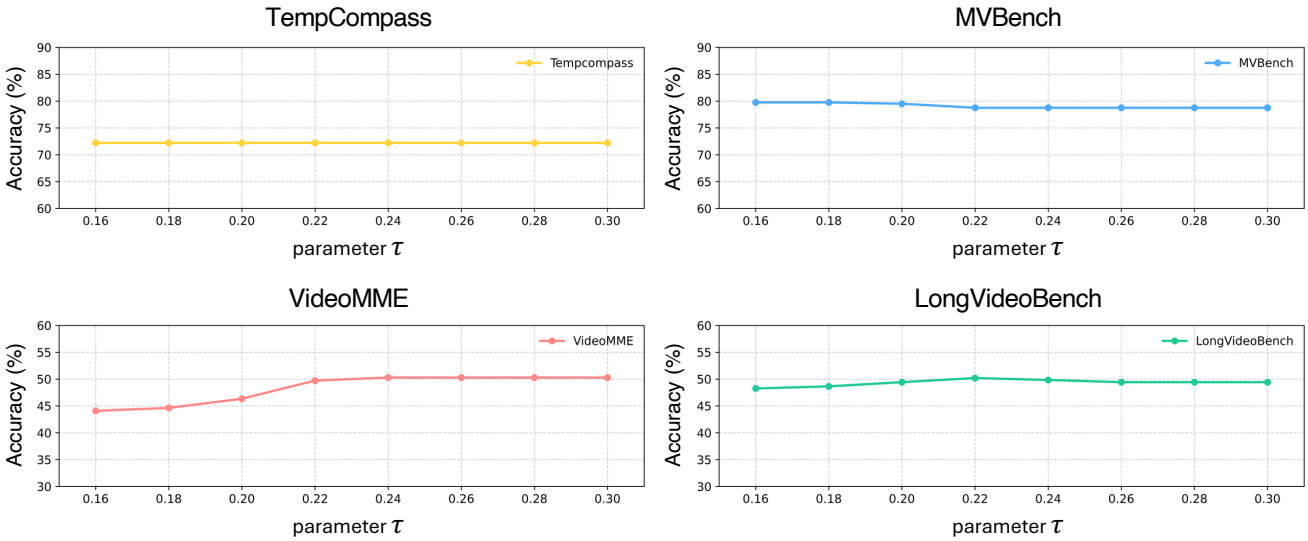

Figure 3. **Effect of Hyperparameter τ.** Performance variation across four datasets with different values of the mapping threshold τ.

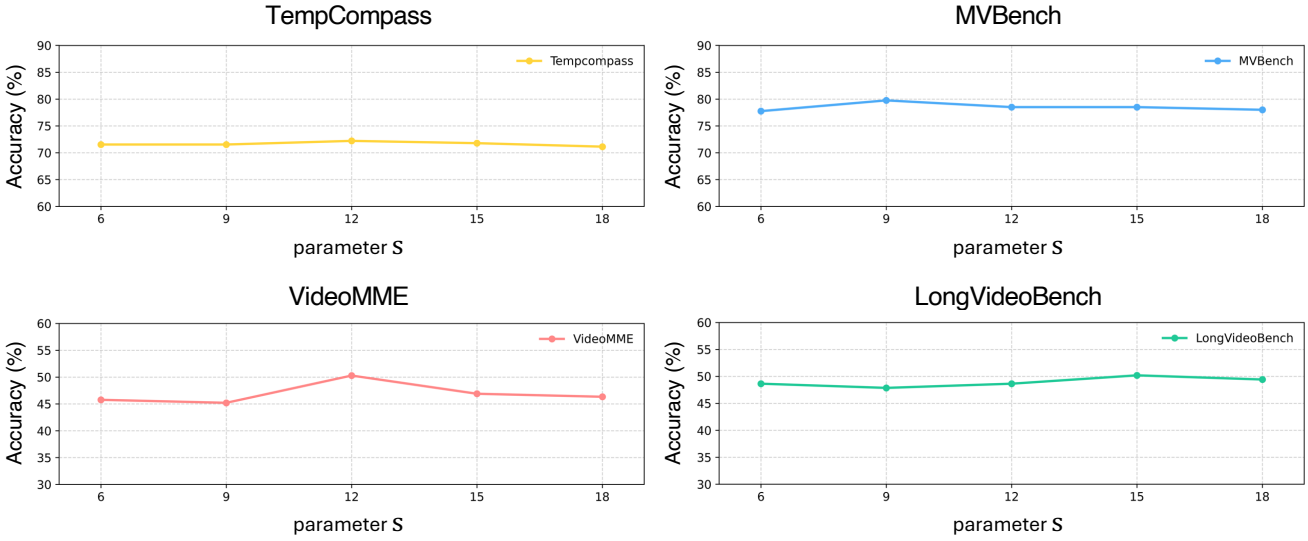

Figure 4. **Effect of Hyperparameter s.** Performance variation across four datasets as the font size control parameter s changes.

converted into temporal, video-level multiple-choice QA. In our experiments, we use only the temporal reasoning subset, namely the Action Sequence (AS) and Scene Transition (ST) categories. AS evaluates whether the model can retrieve the events occurring before or after a specific action, while ST tests whether it can determine how the scene transitions over the course of the video.

VideoMME. VideoMME [2] is a large-scale video benchmark designed to evaluate multimodal LLMs on fine-grained, video-grounded question answering across diverse domains. It includes a dedicated temporal reasoning (TR) subset, where questions require understanding event progression, temporal relations, and dependencies over extended video clips rather than single frames. In our evaluation, we use only this TR subset to specifically assess how well models can reason about when events occur and how they unfold over time.

LongVideoBench. LongVideoBench [17] is a long-form video question answering benchmark with video–language interleaved inputs of up to an hour. In our evaluation, we use the temporal reasoning subsets E3E, O3O, and SSS, which focus on before/after relations between events, the order of entities or concepts, and the correct sequence of scenes across the video.

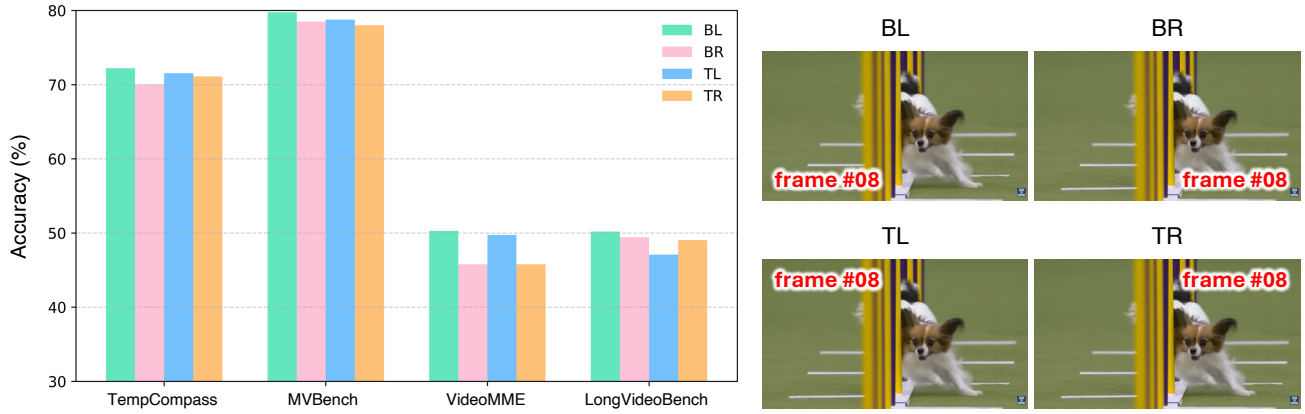

Figure 5. **Effect of VP Position.** Results on four datasets [2, 10, 12, 17] when the visual prompt is inserted at different frame positions: bottom-left (BL), bottom-right (BR), top-left (TL), and top-right (TR).

D.2. Hyperparameter Settings

We apply optimized values for three key hyperparameters based on the dataset, model size, and number of frames used in each experimental setting, as summarized in Table 7.

Font Size Controller s. The parameter s, introduced in Section 4.2 of the main paper, controls the relative font size of the visual prompt (VP) text with respect to the resolution of each video frame. This ensures that the VP is clearly visible regardless of frame size or aspect ratio.

Mapping Threshold τ. The threshold τ, as defined in Section 4.3 of the main paper, determines whether a keyword-to-frame mapping is applied during the KFM process. Mapping is only performed if the frame with the highest similarity to a given keyword exceeds this threshold, thereby preventing incorrect associations with semantically irrelevant frames. This condition assumes that meaningful keywords can be extracted from the user query. In the case of the TempCompass [12] dataset, however, keyword extraction did not yield any informative terms from the question (e.g., *"Which scene is the most suspenseful?"*). As a result, the mapping mechanism was effectively bypassed, and we set $\tau = 1.0$ uniformly for all instances in this dataset. For all other benchmarks, keywords are extracted only from the question portion of the user query, excluding any answer options or candidates.

Outline Usage o. The parameter o is a boolean flag that determines whether an outline is applied to the VP text to enhance visibility. This feature was introduced to improve legibility in cases where red-colored VP text overlaps with red or visually

Table 6. **Statistics of the temporal evaluation splits.** For each dataset, we report the number of questions, number of videos, and video length statistics (in seconds).

Dataset	Category	Questions	Videos	Min (s)	Max (s)	Mean (s)	Std (s)
TempCompass	EO	1384	100	3.04	29.36	13.98	6.67
	AC	1408	96	3.00	24.70	10.18	4.99
	Total	2792	195	3.00	29.36	12.08	6.19
MVBench	AS	200	179	7.10	43.20	20.57	7.93
	ST	200	200	20.00	20.00	20.00	0.00
	Total	400	379	7.10	43.20	20.27	5.46
VideoMME	TR	177	171	56.89	3540.14	1459.26	1049.67
LongVideoBench	E3E	94	49	8.01	3485.12	587.69	719.64
	O3O	66	38	7.98	2041.01	404.31	521.95
	SSS	97	52	8.57	1995.88	469.75	501.24
	Total	257	139	7.98	3485.12	493.44	597.11

Table 7. **Hyperparameter Settings.** Hyperparameters selected for four different VideoLLM baselines [1, 9, 14, 20] across four datasets [2, 10, 12, 17], under two frame sampling conditions (64 frames and 20% frames), as reported in Table 2 of the main paper. Notation is provided Section D.2.

Models	TempCompass			MVBench			VideoMME			LongVideoBench		
	τ	s	o	τ	s	o	τ	s	o	τ	s	o
64 Frames (1 FPS)												
Ours *GPT4.1*	1.00	12	✗	0.18	9	✗	-	-	-	-	-	-
Ours *QwenVL*	1.00	12	✗	0.20	9	✗	0.15	12	✗	0.15	15	✗
Ours *LLaVA-Video*	1.00	12	✗	0.18	9	✗	0.23	12	✗	0.22	12	✓
Ours *LLaVA-OneVision*	1.00	12	✗	0.18	9	✓	0.28	15	✓	0.18	12	✓
20% Frames												
Ours *GPT4.1*	1.00	12	✗	0.18	9	✗	-	-	-	-	-	-
Ours *QwenVL*	1.00	12	✗	0.20	9	✗	0.32	12	✗	0.22	15	✗
Ours *LLaVA-Video*	1.00	12	✗	0.18	9	✗	0.24	12	✗	0.22	16	✗
Ours *LLaVA-OneVision*	1.00	12	✗	0.20	12	✓	0.27	9	✓	0.24	12	✓

complex regions within a frame. In datasets such as VideoMME [2] and LongVideoBench [17], which contain relatively cluttered scenes, the default setting without outlines often led to poor visibility. In addition, recognition performance varied depending on the specific VideoLLM model, particularly in models like LLaVA-OneVision [9], so the outline option was selectively used to improve visual clarity.

D.3. Prompt Templates

D.3.1. System Prompt Templates

We adopt the default system instruction (`You are a helpful assistant.`), and extend it with an additional guideline that encourages temporal grounding. Specifically, we append the following sentence to the instruction: `Focus on the temporal relationships by referring to the number written in the {position} corner of each frame.` This prompt steers the model to explicitly use the injected visual prompts when interpreting temporal relationships across frames.

D.3.2. User Prompt Templates

User prompts follow the standard formatting used in LMMs-Eval [8], ensuring that each model receives questions in a consistent and comparable manner. Detailed examples of the user prompts can be found in Table 8.

D.3.3. Keyword Extractor Prompt Templates

We extract text keywords from user queries using a unified LLM-based procedure across all benchmarks. The extractor uses Qwen2.5-7B-Instruct with a fixed system prompt and dataset-specific instruction templates. In particular, we employ a single system prompt that enforces keyword-only outputs and a shared user prompt that specifies generic extraction rules. On top of this shared template, we attach short dataset-specific user instructions and examples that mirror the question style of each benchmark. This allows the extractor to adapt to different temporal reasoning formats while preserving a unified keyword output, as summarized in Table 9.

Role / Dataset	Prompt
System (shared across all datasets)	
system	You are a helpful assistant. Focus on the temporal relationships by referring to the number written in the bottom-left corner of each frame.
TempCompass	
user	${question} ${options} Task-specific post-prompt: - Multi-choice: 'Please directly give the best option.' - Yes/No: 'Please answer yes or no.' - Caption matching: 'Please directly give the best option.' - Captioning: 'Please directly give the best option.'
Example	Which event happens first to the skillet? A. Burning in fire B. None of both C. Smoking Please directly give the best option.
MVBench	
user	${question} ${options} Only give the best option.
Example	What happened after the person took the food? A. Ate the medicine. B. Tidied up the blanket. C. Put down the cup/glass/bottle. D. Took the box. Only give the best option.
VideoMME	
user	${question} ${options} The best answer is:
Example	What kind of communication is listed before Semaphore? A. Telephone. B. Homing pigeon. C. Telegraph. D. Pony express. The best answer is:
LongVideoBench	
user	${question} ${options} Answer with the option's letter from the given choices directly.
Example	What is the color of the first piece of clothing shown in the video? A. white B. purple C. red D. olive E. black Answer with the option's letter from the given choices directly.

Table 8. System prompt and dataset-specific user prompt templates with examples.

Role / Component	Prompt
System Prompt (Shared)	
system	You are a helpful assistant that only extracts keywords and outputs them as a Python list.
User Prompt (Shared Core Instructions)	
user	Follow these rules carefully: 1. Identify Key Phrases: Your goal is to extract key phrases from the question that refer to specific scenes, events, actions, or distinct items. 2. Exact Extraction: The extracted phrases must appear exactly as they do in the question. Do not modify or rephrase them. 3. Empty List Condition: If no relevant key phrases (as defined in Rule 1) are found in the question, you must return an empty list []. Now: Question: {$question} Your Answer:
Dataset-Specific User Prompt Examples	
TempCompass	Example 1: Question: Which sentence better captures the essence of the video? Your Answer: [] Example 2: Question: Which description is a more suitable match for the video? Your Answer: []
MVBench	Example 1: Question: What happened after the person took the food? Your Answer: ["the person took the food"] Example 2: Question: What happened after the person closed the door? Your Answer: ["the person closed the door"]
VideoMME	Example 1: Question: When is the zodiacal light visible from the video? Your Answer: ["the zodiacal light"] Example 2: Question: Which GPT is introduced after Convert Anything? Your Answer: ["Convert Anything"]
LongVideoBench	Example 1: Question: In front of a blue background, a gentleman wearing a shirt with pink floral patterns is speaking. What did the gentleman do after becoming friends with the unicorn? Your Answer: ["gentleman wearing a shirt with pink floral patterns is speaking", "becoming friends with the unicorn"] Example 2: Question: In the movie scene, there is a man in gray-black clothes standing between a red door and wall on the left, and a silver-white window and yellow wall on the right. After this man appears, which person or object appears first? Your Answer: ["man in gray-black clothes standing", "a red door and wall on the left, and a silver-white window and yellow wall on the right"]

Table 9. System prompt, shared keyword-extraction instructions, and dataset-specific examples used for keyword extraction.

	VideoMME		LongVideoBench	
	1fps	20%	1fps	20%
Baseline [20]	47.46	44.07	56.42	48.25
Interleaved Text Token	0.00	44.07	22.96	47.86
Structured Timeline	47.46	45.76	55.64	46.69
Random Number VP	46.33	44.07	54.09	47.85
Ours	48.02	50.28	56.42	50.19

Table 10. Comparison with alternative prompting strategies.

	1fps	20%		
	Baseline [20]	Baseline [20]	Ours (w/o KFM)	Ours
FLOPs ($\times 10^{13}$)	5.47	2.05	2.07	2.08
Latency (sec)	12.86	1.53	1.63	2.31

Table 11. Computation cost under different settings.

E. Discussion

E.1. Clarification on Novelty

Prior works like Number it [18] utilize frame numbering for temporal grounding. That is, numbers recognized via OCR serve as frame indices to guide output formats and mitigate hallucinations. In contrast, we use VP for temporal reasoning, where the model needs to understand multiple events, infer their temporal relations, and derive high-level reasoning. Thus, simply treating VP as frame indices is insufficient for such complex reasoning tasks. Within our framework, VP is redefined as *dictionary keys* for our KFM, precisely aligning textual concepts with relevant frames. We note that our work is the first to identify and methodologically utilize VP as strong temporal cues for various complex temporal reasoning tasks.

E.2. Comparison with Alternative Prompting Methods

We compare ViKey with several alternative prompting strategies, including Interleaved Text Token, Structured Timeline, and Random Number VP. As shown in Table 10, sequential visual prompts consistently achieve the best performance under sparse-frame settings across both VideoMME and LongVideoBench. While some alternatives provide minor improvements over the baseline in specific cases, none outperform ViKey overall. In particular, text interleaving often leads to unstable behavior and incoherent outputs as the number of frames increases. These results support the use of sequential visual prompts as an effective and practical plug-and-play design for pretrained VideoLLMs.

E.3. Computational Cost

We further report the end-to-end computational cost in Tab. 11. Compared with the dense 1 fps baseline, our sparse setting substantially reduces FLOPs and latency, while adding KFM introduces only a small additional cost over the VP-only variant. Since the method is training-free, it also avoids any extra training overhead.

E.4. Occlusion Concerns

Because ViKey inserts frame indices directly into the visual input, an important question is whether the prompt may interfere with pre-existing textual elements or partially occlude task-relevant content. To examine this issue, we evaluated 163 VideoMME (TR) clips containing explicitly encoded subtitles. In this setting, the baseline achieves 39.88% accuracy, while our default VP configuration reaches 42.94%(+3.06), indicating that the added visual prompts remain effective even in the presence of real-world text. To further reduce possible overlap, we additionally consider a flexible zero-padding configuration, illustrated on the right. This variant improves accuracy to 44.17%(+4.29), suggesting that simple layout adjustments can further mitigate interference between the visual prompt and existing on-screen content.

Example of Padding-based VP Placement with Subtitles

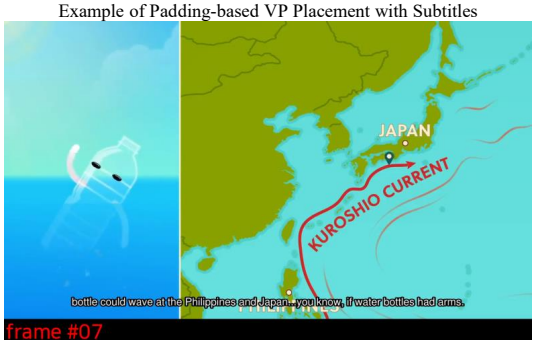

Figure 6. **Flexible zero-padding to reduce overlap with on-screen text.** This helps reduce interference with subtitles and task-relevant content.

E.5. Limitations

Although Keyword-to-Frame Mapping (KFM) generally aligns textual cues with the correct temporal moment, we observe that CLIP-based similarity can be ambiguous when consecutive frames are visually near-identical, as illustrated in Fig. 7. In such cases, multiple adjacent frames obtain nearly the same similarity score, causing the model to map the keyword (e.g., ate the sandwich) to an incorrect frame. Because the query is then augmented with this erroneous frame index, the

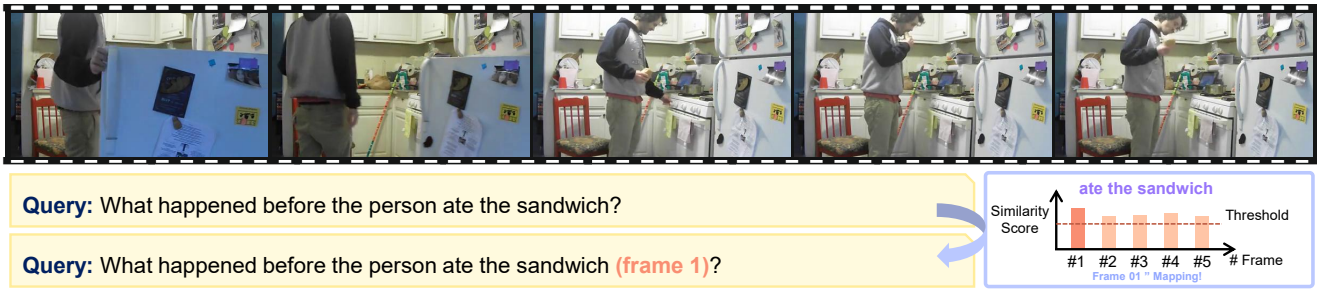

Figure 7. **Example of ambiguous keyword–frame mapping.** When a video contains many visually similar consecutive frames, CLIP-based similarity may not be sufficient for accurate mapping.

model may reason around the wrong temporal anchor, leading to subtle but consistent mistakes in temporal reasoning. In addition, because VIKEY inserts the frame index at a fixed corner of each frame, the visual prompt can occasionally occlude task-relevant content. Although the injected index occupies only a small region, certain videos place important objects or discriminative cues near the frame corners. When this occurs, the prompt may partially cover these regions and alter the appearance of the underlying content, which can in turn lead the model to misinterpret the scene or miss fine-grained details.

E.6. Future Works

Building on these observations, several extensions could further improve the stability and robustness of VIKEY. First, to mitigate the ambiguity observed when consecutive frames are visually near-identical, a promising direction is to incorporate sequence-aware or multi-frame similarity, where keyword matching considers short temporal windows rather than individual frames. Leveraging temporal gradients, motion cues, or lightweight event-boundary detectors may also help distinguish frames that are visually similar but temporally distinct, reducing the likelihood of attaching a keyword to a neighboring but incorrect frame. Second, to address the occasional occlusion caused by placing the prompt at a fixed corner, future work may explore adaptive prompt placement strategies. Simple heuristics based on visual saliency, object occupancy, or corner-level feature density could identify unobtrusive regions for inserting the frame index. Together, these extensions would preserve the simplicity of VIKEY while improving its reliability across diverse video layouts and scene structures.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 10

[2] Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 24108–24118, 2025. 3, 4, 5, 6, 8, 9, 10

[3] Tianyu Fu, Tengxuan Liu, Qinghao Han, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang. Framefusion: Combining similarity and importance for video token reduction on large vision language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22654–22663, 2025. 6

[4] Seil Kang, Jinyeong Kim, Junhyeok Kim, and Seong Jae Hwang. See what you are told: Visual attention sink in large multimodal models. *arXiv preprint arXiv:2503.03321*, 2025. 3

[5] Seil Kang, Jinyeong Kim, Junhyeok Kim, and Seong Jae Hwang. Your large vision-language model only needs a few attention heads for visual grounding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 9339–9350, 2025.

[6] Jinyeong Kim, Seil Kang, Jiwoo Park, Junhyeok Kim, and Seong Jae Hwang. Interpreting attention heads for image-to-text information flow in large vision-language models. *arXiv preprint arXiv:2509.17588*, 2025. 3

[7] Youngmin Kim and Hyeongboo Baek. Preprocessing for keypoint-based sign language translation without glosses. *Sensors*, 23(6): 3231, 2023. 5

[8] Bo Li, Peiyuan Zhang, Kaichen Zhang, Fanyi Pu, Xinrun Du, Yuhao Dong, Haotian Liu, Yuanhan Zhang, Ge Zhang aChunyuan Li, and Ziwei Liu. Lmms-eval: Accelerating the development of large multimoal models, 2024. 10

[9] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024. 10

[10] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206, 2024. 3, 6, 7, 9, 10

[11] Shuming Liu, Chen Zhao, Tianqi Xu, and Bernard Ghanem. Bolt: Boost large vision-language model without training for long-form video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3318–3327, 2025. 5

[12] Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. TempCompass: Do video LLMs really understand videos? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8731–8772, Bangkok, Thailand, 2024. Association for Computational Linguistics. 3, 6, 7, 9, 10

[13] Yongdong Luo, Wang Chen, Weizhong Huang, Shukang Yin, Haojia Lin, Jinfa Huang, Chaoyou Fu, Jiayi Ji, Xiawu Zheng, and Jiebo Luo. Quota: Query-oriented token assignment via cot query decouple for long video comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 24160–24168, 2026. 5

[14] OpenAI. Gpt-4.1: Openai's most advanced model. https://openai.com/index/gpt-4-1/, 2024. Accessed: 2025-11-14. 10

[15] Aleksandar Shtedritski, Christian Rupprecht, and Andrea Vedaldi. What does clip know about a red circle? visual prompt engineering for vlms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11987–11997, 2023. 4

[16] Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive keyframe sampling for long video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29118–29128, 2025. 5

[17] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 37:28828–28857, 2024. 3, 5, 6, 8, 9, 10

[18] Yongliang Wu, Xinting Hu, Yuyang Sun, Yizhou Zhou, Wenbo Zhu, Fengyun Rao, Bernt Schiele, and Xu Yang. Number it: Temporal grounding videos like flipping manga. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13754–13765, 2025. 4, 13

[19] Sicheng Yu, Chengkai Jin, Huanyu Wang, Zhenghao Chen, Sheng Jin, Zhongrong Zuo, Xiaolei Xu, Zhenbang Sun, Bingni Zhang, Jiawei Wu, et al. Frame-voyager: Learning to query frames for video large language models. *arXiv preprint arXiv:2410.03226*, 2024. 5

[20] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024. 3, 5, 6, 10, 13