LESS FROM MORE: REINFORCING SPARSE VIDEO REASONING FROM DENSE REFERENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

Video-language models commonly assume that more temporal observations lead to more reliable reasoning. We question this assumption and argue that the key challenge is not merely processing more video frames efficiently, but learning to reason reliably under limited temporal evidence. We propose SAVER, a dense-to-sparse post-training framework that uses dense video views as training-time references for sparse-frame inference. During reinforcement post-training, paired dense and sparse views are optimized with grounding rewards and a reliability-gated reference reward, encouraging sparse view predictions to preserve task-relevant temporal evidence. Notably, SAVER is trained only on 1,250 randomly sampled temporal grounding examples, without using any video question answering annotations. Across three temporal grounding benchmarks and six video question-answering benchmarks, SAVER consistently improves performance across frame budgets. In particular, SAVER can match or surpass dense-frame Qwen3.5 baselines while using substantially fewer frames. These results show that temporal grounding can serve as an effective evidence-localization proxy for learning sparse video reasoning that transfers to broader video understanding tasks.

1 INTRODUCTION

A common recipe for improving video reasoning is to show multimodal large language models (MLLMs) more frames. Recent MLLM families, including Qwen-VL (Bai et al., 2023; Wang et al., 2024; Bai et al., 2025b;a; Team, 2026), InternVL (Chen et al., 2024b;a), and LLaVA-style models (Liu et al., 2023; Li et al., 2024a; Lin et al., 2024), have substantially strengthened visual understanding, while video post-training further enhances reasoning on question answering, perception, and temporal tasks (Feng et al., 2025; Liu et al., 2026a; Feng et al., 2026). Yet most video reasoning pipelines still rely on a dense-input assumption: reliable predictions require observing many frames. This assumption is costly, as dense temporal inputs can involve tens or hundreds of frames, leading to large visual-token budgets, high memory usage, and increased latency. Such costs are especially problematic for streaming video, mobile assistants, robotics, surveillance, and long-form video search, where models must operate under limited compute or latency budgets. Efficient video reasoning therefore requires reducing the visual evidence processed in the first place, not merely decoding responses faster.

This raises a simple but underexplored question: *do video-language models truly need dense frame sampling at inference time, or can they be post-trained to reason reliably from sparse temporal observations?* The answer is non-trivial. Simply reducing the frame rate lowers visual cost, but it also removes temporal evidence that may be necessary for answering questions or localizing events (Li et al., 2025d). At the same time, dense sampling often introduces substantial redundancy, and its computational cost grows quickly with the number of observed frames. The central challenge we address in this paper is therefore not merely to process dense videos more efficiently, but to make sparse-frame inference itself reliable.

Figure 1 illustrates the accuracy–frame-budget trade-off. Lowering the frame rate reduces the number of sampled frames substantially, but it also degrades standard Qwen3.5 models, especially on temporal grounding where accurate localization depends on sufficient temporal evidence. SAVER changes this trade-off: at the same frame rate, it consistently improves over Qwen3.5, and at lower frame rates it can approach the accuracy of denser Qwen3.5 inference while using far fewer frames. For

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

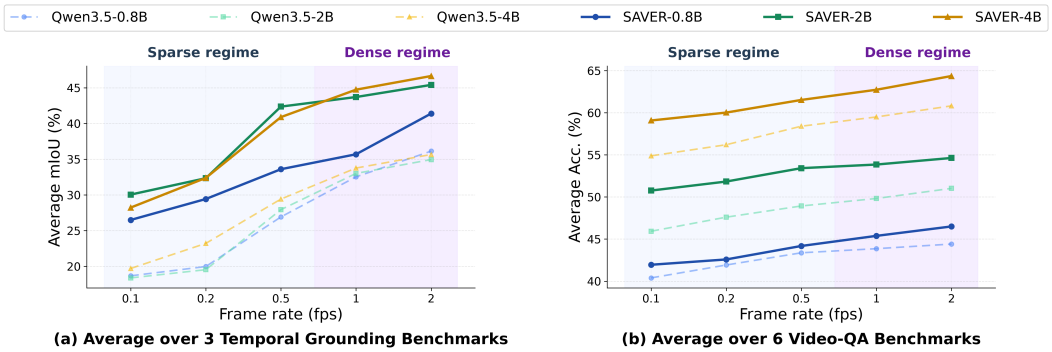


Figure 1: Performance–efficiency trade-off across frame rates. Lower frame rates reduce the number of sampled frames but enlarge the sparse–dense performance gap for Qwen3.5. SAVER mitigates this degradation; for example, for video-QA, SAVER-2B at 0.1 fps nearly matches dense Qwen3.5-2B at 2.0 fps.

example, on video-QA benchmarks, SAVER-2B at 0.1 fps nearly matches Qwen3.5-2B at 2.0 fps. Importantly, this improvement transfers from temporal grounding, the task used for post-training, to six video-QA benchmarks, suggesting that learning to preserve temporal evidence under sparse observations benefits broader video reasoning.

Existing efficiency approaches only partially address this problem. Some methods accelerate reasoning on the language side, for example by reducing redundant decoding or reasoning steps (Liu et al., 2026a), but they do not directly reduce the cost of visual input processing. Other methods reduce visual computation through token compression (Wu et al., 2025), adaptive keyframe sampling (Tang et al., 2025), or retrieval-based frame selection (Zou et al., 2026). These approaches are effective, but they typically assume a richer visual input and then reduce its cost by compressing, selecting, or retrieving visual evidence. Our work addresses a different part of the efficiency problem: given a prescribed low-frame-rate visual budget, can the video-language model itself be made reliable under sparse temporal evidence?

In this work, we study *sparse video reasoning*: the model receives only sparsely sampled frames at inference time and must reason reliably under limited temporal evidence. Sparse observations may miss key events, provide coarse temporal boundaries, or emphasize misleading static cues, making models trained mainly under dense observations fragile when the frame budget is reduced. Rather than designing another frame selector or token compressor, we take a training-centric view: dense frames should serve as training-time references for learning sparse-frame reasoning, but should not be required at inference time. Motivated by this view, we propose SAVER, a dense-guided sparse post-training framework for efficient video understanding. SAVER constructs paired dense and sparse views from the same video and uses temporal grounding as an evidence-localization proxy task. Dense views provide richer temporal reference signals, while sparse view predictions are optimized with grounding rewards and a reference reward that aligns them with reliable dense view predictions.

Notably, SAVER is trained only on 1,250 randomly sampled temporal grounding examples from Time-R1 (Wang et al., 2025), without using any video-QA annotations. Despite this small grounding-only training set, SAVER consistently improves across three temporal grounding benchmarks and six video question-answering benchmarks under different frame budgets, with particularly strong gains in the sparse regime. These results suggest that efficient video reasoning should not only focus on compressing dense-frame inference, but also on training models to remain reliable when dense temporal evidence is unavailable.

2 Related Work

Visual reasoning models. Recent multimodal large language models (MLLMs) have substantially advanced image and video understanding. Representative model families include Qwen-VL (Bai et al., 2023; Wang et al., 2024; Bai et al., 2025b;a; Team, 2026), InternVL (Chen et al., 2024b;a), LLaVA-style models (Liu et al., 2023; Li et al., 2024a), and video-oriented MLLMs such as Video-LLaVA (Lin et al., 2024), VideoChat (Li et al., 2025b), and TimeChat (Ren et al., 2024). Building on these models, reinforcement post-training has become a popular strategy for improving visual

reasoning. VisionReasoner (Liu et al., 2026b) and RegionReasoner (Sun et al., 2026) focus on image-level and region-grounded reasoning, while Video-R1 (Feng et al., 2025), VideoChat-R1 (Li et al., 2025c), VideoAuto-R1 (Liu et al., 2026a), and OneThinker (Feng et al., 2026) extend reasoning-oriented post-training to video. Other works further investigate temporal grounding (Wang et al., 2025; Li et al., 2025a), video reasoning segmentation (Gong et al., 2025), frame-referenced reasoning traces (Ghazanfari et al., 2025), and scalable video reasoning training (Chen et al., 2025; Li et al., 2025e; Fu et al., 2025b). These methods improve the model's reasoning behavior or supervision signal, but generally assume that sufficient visual evidence is available at inference time. In contrast, our work studies whether video reasoning can remain reliable when only sparse temporal observations are provided at test time.

Efficient video understanding. The high computational cost of video understanding has motivated methods that reduce either language-side or visual-side computation. Language-side methods reduce redundant reasoning or decoding steps; for example, VideoAuto-R1 (Liu et al., 2026a) avoids unnecessary repeated reasoning, while Sparse-to-Dense (Zhang et al., 2025b) accelerates decoding through sparse-attention speculation and dense-attention verification. Visual-side methods reduce the cost of processing long or dense videos by compressing visual tokens or selecting informative frames. MARC (Wu et al., 2025) studies memory-augmented token compression for efficient video understanding, STORM (Jiang et al., 2025) reduces visual tokens for long-video MLLMs, and frame selection or retrieval methods such as M-LLM-based frame selection (Hu et al., 2025a), Adaptive Keyframe Sampling (Tang et al., 2025), Generative Frame Sampler (Yao et al., 2025), VideoBrain (Zou et al., 2026), and ReViSe (Xu et al., 2026) select or retrieve useful frames from richer video contexts. Q-Frame (Zhang et al., 2025a) further combines query-aware frame selection with multi-resolution adaptation. FastVID (Shen et al., 2025) dynamically prunes redundant visual information, while WFS-SB (Chen et al., 2026) selects informative frames by detecting semantic boundaries. These works show that not all frames or visual tokens contribute equally to video understanding. Our setting differs in that we evaluate the model under a *fixed low-frame-rate input budget*: the model receives only uniformly sampled sparse frames at inference time, rather than first accessing a richer video representation and then selecting or compressing it. Accordingly, SAVER does not propose a new frame selector or token compressor; instead, it uses dense views only during training as scaffolds and aligns sparse view predictions with dense view signals through reinforcement post-training. This targets a different question: how can a video model remain reliable when dense temporal evidence is unavailable or too costly to process at inference time?

3 METHOD

SAVER targets efficient video inference under a low-frame-rate protocol. At test time, the model receives frames at a prescribed fps and predicts from this sparse input. It does not perform dense video pre-scanning, adaptive frame selection, or test-time optimization. Thus, the goal is not to introduce a sampling strategy, but to make the model reliable when only sparse frames are processed.

3.1 OVERVIEW

For each training example, SAVER follows three simple steps. First, we sample two views from the same video: a dense view V^d with more frames and a sparse view V^s with fewer frames. Second, both views are fed into the same MLLM policy π_θ, producing a dense view response y^d and a sparse view response y^s. Third, we optimize both responses with temporal grounding rewards and add a dense-to-sparse reference reward, so that the sparse prediction learns from the dense prediction only when the dense prediction is more reliable. After training, only the sparse view is used for inference.

Given a video V and a temporal grounding instruction q, we construct:

$$V^d = \mathcal{S}_d(V), \qquad V^s = \mathcal{S}_s(V), \tag{1}$$

where $\mathcal{S}_d$ and $\mathcal{S}_s$ denote dense and sparse frame sampling functions, respectively. Both views are fed into the same MLLM policy:

$$y^d \sim \pi_\theta(\cdot \mid V^d, q), \qquad y^s \sim \pi_\theta(\cdot \mid V^s, q). \tag{2}$$

During post-training, each response is expected to predict a temporal interval. Although SAVER is trained only with temporal grounding supervision, we later evaluate the resulting sparse view policy on both temporal grounding and video question answering to test cross-task transfer.

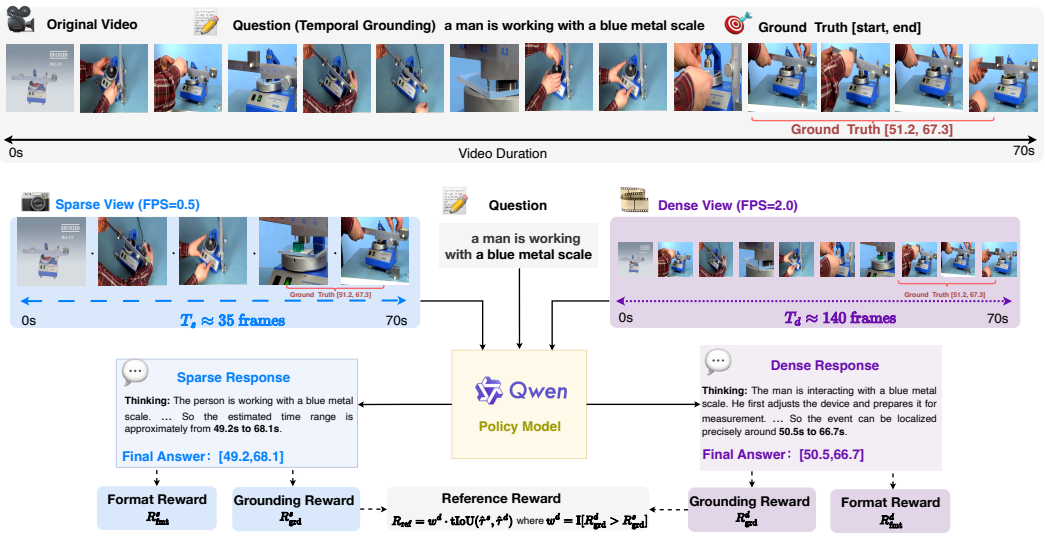


Figure 2: Overview of SAVER. SAVER performs dense-to-sparse post-training with paired views from the same video. Dense views provide training-time references, while sparse views expose the policy to reduced temporal observations. Format and grounding rewards supervise both views, and a reliability-gated reference reward aligns sparse predictions with reliable dense predictions. At inference time, the model uses sparse frames.

3.2 Dense-to-Sparse Post-Training

SAVER performs reinforcement post-training with paired dense and sparse video views, where dense views provide training-time guidance for improving sparse-frame inference. We use temporal grounding as the training task because it provides explicit supervision over when an event occurs, making it a natural signal for learning task-relevant temporal evidence. In our implementation, we randomly sample 1,250 training examples from Time-R1 (Wang et al., 2025). No video-QA annotations are used during post-training. Therefore, improvements on video-QA benchmarks come from cross-task transfer rather than task-specific QA post-training.

Given a video-instruction pair (V, q) and a ground-truth temporal segment $\tau^\star = (t_s^\star, t_e^\star)$, the model predicts temporal segments from both views:

$$\hat{\tau}^d = (\hat{t}_s^d, \hat{t}_e^d) \leftarrow y^d, \qquad \hat{\tau}^s = (\hat{t}_s^s, \hat{t}_e^s) \leftarrow y^s. \tag{3}$$

The dense and sparse responses are optimized with temporal grounding rewards. We then use the dense prediction as a training-time reference for the sparse prediction only when the dense view is more reliable. This dense-to-sparse alignment encourages the sparse view to preserve temporal evidence that is available in the dense view, while avoiding imitation of unreliable dense predictions.

3.3 Reward Design

SAVER uses three rewards during post-training: a format reward that ensures valid temporal outputs, a grounding reward that evaluates each predicted temporal interval against the ground truth, and a reference reward that transfers reliable dense view signals to the sparse view.

Format reward. The format reward $R_{\text{fmt}}(y)$ encourages the model to generate a valid and parseable temporal grounding response. It checks whether the response contains a start timestamp and an end timestamp in the required format. This reward stabilizes reinforcement post-training by preventing malformed outputs.

Grounding reward. The grounding reward evaluates the temporal segment predicted by each view. Given a predicted segment $\hat{\tau} = (\hat{t}_s, \hat{t}_e)$ and the ground-truth segment $\tau^\star = (t_s^\star, t_e^\star)$, we define:

$$R_{\text{grd}}(y, \tau^\star) = \text{tIoU}(\hat{\tau}, \tau^\star) - \lambda_b \left(\frac{|\hat{t}_s - t_s^\star|}{T} + \frac{|\hat{t}_e - t_e^\star|}{T} \right), \tag{4}$$

where tIoU denotes temporal intersection-over-union, T is the video duration, and λ_b controls the boundary penalty. The tIoU term encourages the predicted segment to overlap with the target

event, while the boundary term provides denser feedback for accurately localizing the start and end timestamps.

Reference reward. The grounding reward supervises each view independently. We additionally introduce a reference reward that encourages agreement between the predicted temporal intervals when the dense prediction receives a higher grounding reward. We therefore introduce a reference reward. Let $\hat{\tau}^d$ and $\hat{\tau}^s$ be the temporal intervals predicted from the dense and sparse views, respectively. We define:

$$R_{\text{ref}} = w^d \cdot \text{tIoU}(\hat{\tau}^s, \hat{\tau}^d), \tag{5}$$

where w^d is a relative reliability gate:

$$w^d = \mathbb{I}\left[R_{\text{grd}}(y^d, \tau^\star) > R_{\text{grd}}(y^s, \tau^\star)\right]. \tag{6}$$

The gate activates the reference reward only when the sampled dense prediction receives a higher grounding reward than its paired sparse prediction. Otherwise, the reference reward is zero. This is a relative, sample-level reward-shaping criterion: it neither certifies the correctness of the dense prediction nor guarantees that the combined reward increases monotonically with grounding quality.

3.4 Training Objective and Inference

The dense view reward contains the format reward and the grounding reward:

$$R^d = \lambda_{\text{fmt}} R_{\text{fmt}}(y^d) + \lambda_{\text{grd}} R_{\text{grd}}(y^d, \tau^\star). \tag{7}$$

The sparse view reward additionally includes the reference reward:

$$R^s = \lambda_{\text{fmt}} R_{\text{fmt}}(y^s) + \lambda_{\text{grd}} R_{\text{grd}}(y^s, \tau^\star) + \lambda_{\text{ref}} R_{\text{ref}}. \tag{8}$$

We optimize the model with reinforcement post-training over paired dense and sparse views:

$$\max_{\theta} \ \mathbb{E}\left[R^d + \lambda_s R^s - \beta \mathcal{D}_{\text{KL}}\right], \tag{9}$$

where λ_s controls the relative weight of the sparse view reward and β controls KL regularization to the reference policy π_{ref}. The KL term is defined as:

$$\mathcal{D}_{\text{KL}} = D_{\text{KL}}\big(\pi_\theta(\cdot \mid V^d, q) \,\|\, \pi_{\text{ref}}(\cdot \mid V^d, q)\big) + D_{\text{KL}}(\pi_\theta(\cdot \mid V^s, q) \,\|\, \pi_{\text{ref}}(\cdot \mid V^s, q)). \tag{10}$$

In practice, we follow the reinforcement post-training recipe of Time-R1 (Wang et al., 2025) and train on 1,250 randomly sampled temporal grounding examples. We pair $G = 4$ dense and sparse rollouts index-wise and normalize advantages over their combined rewards. At inference time, only the sparse view is used. Given a prescribed frame rate, the input video is sampled into a sparse frame sequence $V^s = \mathcal{S}_s(V)$, and the model directly predicts $y^s \sim \pi_\theta(\cdot \mid V^s, q)$.

No dense video pre-scanning, adaptive frame selection, or test-time optimization is performed. This makes the inference protocol compatible with low-fps streaming or resource-constrained settings, where only a sparse frame stream is decoded or processed.

Relation to distillation and risk intuition. SAVER is related to self-distillation, but differs in both the input views and the use of the teacher signal. Conventional self-distillation usually transfers predictions under the same input distribution, whereas SAVER performs reward-guided cross-view distillation from dense temporal views to sparse temporal views. The dense view provides privileged training-time information, while the sparse view matches the inference-time condition. Importantly, dense predictions are not treated as unconditional pseudo-labels: the alignment reward is activated only when the dense view grounding reward is higher than the sparse view grounding reward. This prevents sparse predictions from imitating inferior dense view outputs.

This view also gives a simple risk-decomposition intuition for dense-to-sparse alignment. Let $p_s(y) = \pi_\theta(y \mid V^s, q)$ and $p_d(y) = \pi_\theta(y \mid V^d, q)$ denote the predictive distributions under sparse and dense views, respectively. For a bounded task loss $\ell(y, y^\star) \in [0, 1]$, the sparse view risk can be upper-bounded by:

$$\mathbb{E}_{y \sim p_s}[\ell(y, y^\star)] \le \mathbb{E}_{y \sim p_d}[\ell(y, y^\star)] + D_{\text{TV}}(p_s, p_d), \tag{11}$$

where D_{TV} is the total variation distance between the two predictive distributions. This inequality suggests that sparse view performance depends on two factors: the reliability of dense view predictions and the discrepancy between sparse view and dense view predictions. SAVER directly targets both factors. The grounding rewards improve the quality of predictions under each view, while the relative reliability-gated reference reward reduces the sparse–dense discrepancy only when the dense view provides a better temporal reference. Thus, dense frames are used not to increase test-time computation, but to provide reliable training-time guidance for sparse-frame inference.

4 EXPERIMENTS

4.1 EXPERIMENT DETAILS

Implementation details. We instantiate SAVER on Qwen3.5 (Team, 2026) models with three different scales: 0.8B, 2B, and 4B. All models are trained with the same dense–sparse reinforcement post-training framework described in Sec. 3. Unlike large-scale video reasoning post-training methods, SAVER uses only a small temporal grounding training set. Specifically, we randomly sample 1,250 examples from Time-R1 (Wang et al., 2025) and use them as the only post-training data. No video-QA annotations are used during post-training, allowing us to evaluate whether sparse temporal alignment learned from grounding supervision transfers to broader video understanding tasks. For each training example, we construct a dense view and a sparse view from the same video. Both views are fed into the same policy model, and the model is optimized with grounding rewards and dense-to-sparse reference rewards. We set the coefficient of the sparse view reward to 2.0 to emphasize sparse-frame robustness, and set all other reward coefficients to 1.0. For reinforcement learning, the rollout size is set to $G = 4$. The global batch sizes are set to 64, 64, and 128 for the 0.8B, 2B, and 4B models. We train the 0.8B, 2B, and 4B models using 16, 16, and 32 A100 GPUs, with training times of approximately 4, 7, and 13 hours, respectively. We will release the code, model checkpoints, and sampled training subset to facilitate reproducibility.

Evaluation protocol. At test time, SAVER directly operates on sparsely sampled frames without using dense video inputs. We control the visual budget by varying the sampling frame rate over $\{0.1, 0.2, 0.5, 1, 2\}$ fps. This protocol allows us to evaluate the performance–efficiency trade-off from extremely sparse observations to denser temporal inputs. We cap the number of sampled frames to $[4, 256]$ per video for fair comparison. For each fps setting, we report both task performance and the corresponding average number of sampled frames. We evaluate with `lmms-eval` (Zhang et al., 2024) using greedy decoding unless noted.

Evaluation benchmarks. Following prior video reasoning evaluation protocols (Liu et al., 2026a), we evaluate SAVER on temporal grounding and video question answering benchmarks. For temporal grounding, we evaluate on Charades-STA (Gao et al., 2017) and ActivityNet (Caba Heilbron et al., 2015), reporting mean temporal intersection-over-union (mIoU). We further evaluate on NExT-GQA (Xiao et al., 2024), which combines question answering with temporal grounding and requires models to identify task-relevant video segments. Beyond the grounding task used for post-training, we evaluate transfer to video-QA benchmarks. For perception-oriented video-QA, we report accuracy on VideoMME without subtitles (Fu et al., 2025a), MVBench (Li et al., 2024b), LongVideoBench (Wu et al., 2024), and MMVU (Zhao et al., 2025). For reasoning-intensive video understanding, we evaluate on VideoMMMU (Hu et al., 2025b) and Minimal Video Pairs (MVP) (Krojer et al., 2025).

4.2 MAIN RESULTS

Temporal grounding benchmarks. Table 1 summarizes the results on Charades-STA, ActivityNet, and NExT-GQA. We first evaluate temporal grounding because it is the task used for post-training and directly measures whether the model can localize task-relevant temporal evidence. Temporal grounding is also highly sensitive to frame sparsity: accurate localization requires identifying event boundaries from limited temporal observations. This is reflected in the original Qwen3.5 baselines. When the frame rate is reduced from 2.0 fps to 0.1 fps, the average mIoU of Qwen3.5-2B drops from 34.9% to 18.4%, confirming that sparse-frame temporal grounding is a challenging setting rather than a trivial reduction of input frames. SAVER substantially alleviates this degradation under the same frame budgets. At 0.1 fps, SAVER improves average mIoU from 18.4% to 30.0% for the 2B model, yielding a large gain under extremely sparse observations. The same trend also holds for the 4B model, where SAVER improves average mIoU from 19.7% to 28.2% at 0.1 fps. Under denser inputs, SAVER remains effective, improving the 2B model from 34.9% to 45.4% at 2.0 fps. These results show that dense-to-sparse reference learning helps the model recover task-relevant temporal evidence even when only a few frames are observed, while also improving temporal localization under denser views. The prior temporal grounding methods in the upper block are included as reported-reference results, often with different training data, model scales, and evaluation settings. Our main comparison is the controlled frame-rate block, where Qwen3.5 and SAVER are evaluated under identical fps and frame budgets. Under this controlled setting, SAVER consistently improves Qwen3.5 across model scales and frame rates. This is notable because SAVER is trained with only 1,250 temporal

Table 1: Temporal grounding results under controlled frame budgets. For each Qwen3.5 setting, Qwen3.5 and SAVER are shown side by side under the same fps and average frame budget. Numbers in parentheses indicate the absolute improvement over Qwen3.5 at the same fps.

Method	FPS	Avg. Frames	Charades-STA	ActivityNet	NExT-GQA	Avg. mIoU
Prior temporal grounding models: 7B model						
Qwen2.5-VL-7B	2.0	110	52.9	26.9	20.2	33.3
TimeChat	2.0	110	31.2	30.4	17.4	26.3
Temporal-RLT	2.0	110	57.0	39.0	37.3	44.4
VideoAuto-R1-7B	2.0	110	60.0	47.6	36.7	48.1
Controlled frame-rate evaluation: 0.8B model						
Qwen3.5	2.0	110	43.9	38.0	26.6	36.1
SAVER	2.0	110	50.5 (+6.6)	45.3 (+7.3)	28.4 (+1.8)	**41.4** (+5.3)
Qwen3.5	0.5	31	33.4	28.4	19.0	26.9
SAVER	0.5	31	40.7 (+7.3)	42.4 (+14.0)	17.6 (-1.4)	**33.6** (+6.7)
Qwen3.5	0.1	7	24.0	21.7	10.3	18.7
SAVER	0.1	7	31.1 (+7.1)	32.7 (+11.0)	15.6 (+5.3)	**26.5** (+7.8)
Controlled frame-rate evaluation: 2B model						
Qwen3.5	2.0	110	44.0	38.3	22.6	34.9
SAVER	2.0	110	54.3 (+10.3)	44.8 (+6.5)	37.1 (+14.5)	**45.4** (+10.5)
Qwen3.5	0.5	31	32.3	29.8	21.7	27.9
SAVER	0.5	31	45.2 (+12.9)	43.0 (+13.2)	38.9 (+17.2)	**42.4** (+14.5)
Qwen3.5	0.1	7	17.6	20.7	16.9	18.4
SAVER	0.1	7	28.7 (+11.1)	37.2 (+16.5)	24.2 (+7.3)	**30.0** (+11.6)
Controlled frame-rate evaluation: 4B model						
Qwen3.5	2.0	110	54.2	44.7	8.1	35.7
SAVER	2.0	110	54.8 (+0.6)	46.9 (+2.2)	38.2 (+30.1)	**46.6** (+10.9)
Qwen3.5	0.5	31	43.9	40.5	3.9	29.4
SAVER	0.5	31	48.0 (+4.1)	43.4 (+2.9)	31.3 (+27.4)	**40.9** (+11.5)
Qwen3.5	0.1	7	25.0	29.8	4.3	19.7
SAVER	0.1	7	29.5 (+4.5)	32.5 (+2.7)	22.7 (+18.4)	**28.2** (+8.5)

Table 2: Video-QA results under controlled frame budgets. SAVER consistently improves Qwen3.5 across model sizes and frame rates, matching or surpassing dense-frame performance with far fewer frames.

Method	FPS	Avg. Frames	Video Perception				Video Reasoning		Avg. Acc.
			VideoMME	MVBench	LongVideoBench	MMVU	VideoMMMU	MVP	
Prior video reasoning models (reported settings)									
Qwen2.5-VL-7B	2.0	139	66.0	67.1	60.9	66.2	54.7	36.5	58.6
Qwen3-VL-8B	2.0	139	72.5	69.4	67.6	69.9	61.0	40.5	63.5
VideoAuto-R1-7B	2.0	139	67.3	71.0	60.5	69.7	58.6	39.4	61.1
Controlled frame-rate evaluation: 0.8B model									
Qwen3.5	2.0	139	55.0	49.2	54.2	50.4	32.7	25.0	44.4
SAVER	2.0	139	56.3 (+1.3)	52.9 (+3.7)	54.7 (+0.5)	51.2 (+0.8)	40.8 (+8.1)	23.4 (-1.6)	**46.5** (+2.1)
Qwen3.5	0.5	97	54.0	46.4	52.1	50.2	35.3	22.2	43.4
SAVER	0.5	97	55.0 (+1.0)	49.6 (+3.2)	52.3 (+0.2)	47.9 (-2.3)	39.1 (+3.8)	21.2 (-1.0)	**44.2** (+0.8)
Qwen3.5	0.1	38	49.0	43.7	49.1	49.0	31.6	20.1	40.4
SAVER	0.1	38	50.3 (+1.3)	46.0 (+2.3)	49.5 (+0.4)	50.1 (+1.1)	38.3 (+6.7)	17.6 (-2.5)	**42.0** (+1.6)
Controlled frame-rate evaluation: 2B model									
Qwen3.5	2.0	139	62.6	58.6	55.9	61.0	38.8	29.4	51.0
SAVER	2.0	139	61.1 (-1.5)	56.9 (-1.7)	60.5 (+4.6)	61.8 (+0.8)	57.0 (+18.2)	30.5 (+1.1)	**54.6** (+3.6)
Qwen3.5	0.5	97	61.7	55.7	52.4	58.7	40.2	24.9	48.9
SAVER	0.5	97	59.9 (-1.8)	57.0 (+1.3)	59.0 (+6.6)	60.2 (+1.5)	57.9 (+17.7)	26.5 (+1.6)	**53.4** (+4.5)
Qwen3.5	0.1	38	55.9	52.1	51.3	57.0	37.4	21.9	45.9
SAVER	0.1	38	53.6 (-2.3)	54.3 (+2.2)	56.0 (+4.7)	59.7 (+2.7)	56.1 (+18.7)	24.9 (+3.0)	**50.8** (+4.9)
Controlled frame-rate evaluation: 4B model									
Qwen3.5	2.0	139	70.9	64.4	64.7	68.2	54.8	41.9	60.8
SAVER	2.0	139	71.8 (+0.9)	65.3 (+0.9)	65.6 (+0.9)	70.9 (+2.7)	69.8 (+15.0)	43.1 (+1.2)	**64.4** (+3.6)
Qwen3.5	0.5	97	69.5	62.3	62.6	67.2	54.2	34.3	58.4
SAVER	0.5	97	70.3 (+0.8)	61.4 (-0.9)	63.8 (+1.2)	67.7 (+0.5)	71.2 (+17.0)	34.6 (+0.3)	**61.5** (+3.1)
Qwen3.5	0.1	38	63.2	57.9	59.3	66.6	52.4	29.9	54.9
SAVER	0.1	38	64.0 (+0.8)	59.1 (+1.2)	60.0 (+0.7)	68.0 (+1.4)	72.7 (+20.3)	30.8 (+0.9)	**59.1** (+4.2)

grounding examples from Time-R1 (Wang et al., 2025), yet the learned sparse view policy generalizes across grounding benchmarks and frame budgets.

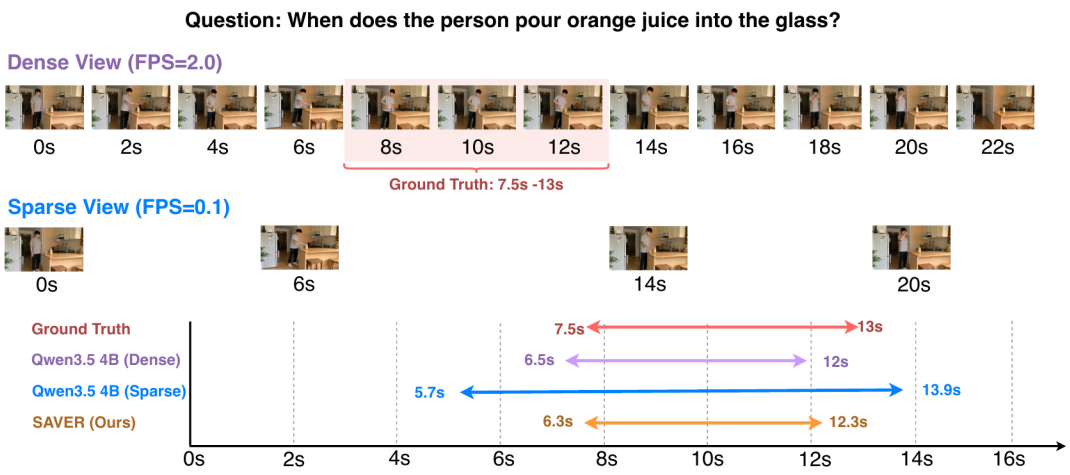


Figure 3: Qualitative comparison under sparse observations. Dense frames provide finer temporal evidence, whereas sparse frames contain only a few sampled moments. Under sparse input, Qwen3.5 predicts an overly broad interval, while SAVER produces a tighter prediction closer to the ground truth. This shows that dense references help preserve task-relevant temporal evidence under sparse frame budgets.

Transfer to video-QA benchmarks. We next evaluate transfer from grounding-only post-training to video question answering. Table 2 reports results on six video-QA benchmarks, none of whose QA annotations are used during post-training. Across the evaluated model scales and frame rates, SAVER improves macro-averaged QA accuracy over the corresponding Qwen3.5 baseline. At 2.0 fps, average accuracy increases from 51.0% to 54.6% for the 2B model and from 60.8% to 64.4% for the 4B model. At 0.1 fps, the 2B model improves from 45.9% to 50.8%. The gains vary substantially across benchmarks. VideoMMMU shows particularly large improvements, including an increase from 52.4% to 72.7% for the 4B model at 0.1 fps. Some settings show regressions, such as VideoMME for the 2B model and MVP for the 0.8B model. The results therefore support improved aggregate video-QA performance following grounding-only post-training, while also revealing task-dependent transfer. Sparse inference also reduces the visual input budget. Across the six video-QA benchmarks, changing the sampling rate from 2.0 to 0.1 fps reduces the average number of observed frames from 139 to 38, a 72.7% reduction. Separately, profiling the original Qwen3.5-2B model under the settings in Appendix C reports reductions of 86.8% in visual tokens, 49.3% in peak GPU memory, and 63.6% in end-to-end latency, averaged across nine benchmarks. These measurements characterize the cost of the profiled input settings on the base model; they do not establish a SAVER-specific speedup.

Qualitative analysis. Figure 3 provides a qualitative example of sparse-frame temporal grounding. When only a few frames are available, the original Qwen3.5 model captures the rough event context but predicts a much broader temporal interval, reflecting uncertainty caused by missing intermediate frames. In contrast, SAVER produces a more precise segment that is close to both the ground-truth interval and the dense view prediction. This suggests that the proposed dense-to-sparse alignment does not simply improve numerical scores, but also helps the model retain task-relevant temporal evidence when operating under sparse observations.

4.3 Analysis

Reward ablation. Table 3 reports reward ablations using Qwen3.5-2B evaluated at 0.1 fps. Joint dense-and-sparse training obtains 27.4 mIoU and 48.7% average video-QA accuracy, exceeding either single-view variant. Adding the reference reward increases these scores to 29.1 and 49.3%, respectively. The gated variant obtains the highest scores in this comparison, reaching 30.0 mIoU and 50.8% average video-QA accuracy. These ablations show incremental gains from the reference reward and its relative-quality gate under the evaluated configuration.

Comparison with alternative training strategies. We compare alternative post-training strategies in Table 4. All variants use the same Qwen3.5-2B initialization and 1,250 grounding examples and are evaluated at 0.1 fps. Time-R1 RL uses 2.0-fps training inputs and obtains 21.0 mIoU and

Table 3: Reward ablation.

Variant	R_{fmt}	R_{grd}	R_{ref}	w^d	TG mIoU	QA Avg.
Qwen3.5	–	–	–	–	18.4	45.9
Sparse-only	✓	✓	–	–	19.0	48.5
Dense-only	✓	✓	–	–	24.2	48.3
Dense+Sparse	✓	✓	–	–	27.4	48.7
+ Ref. reward	✓	✓	✓	–	29.1	49.3
SAVER	✓	✓	✓	✓	**30.0**	**50.8**

Table 4: Alternative training strategies.

Method	TG mIoU	QA Avg.
Qwen3.5	18.4	45.9
Time-R1 RL	21.0	49.1
Sparse SFT (GT labels)	20.3	48.1
Sparse SFT (dense pseudo-labels)	19.6	49.5
SAVER(GT ref.)	29.5	50.1
SAVER(dense ref.)	**30.0**	**50.8**

49.1% average video-QA accuracy. Sparse SFT uses 0.5-fps inputs with either ground-truth segments or fixed pseudo-labels generated by the original model at 2.0 fps, obtaining 20.3 and 19.6 mIoU, respectively. SAVER achieves the highest scores among these evaluated configurations, with 30.0 mIoU and 50.8% average video-QA accuracy. Replacing the dense reference with the ground-truth segment gives 29.5 mIoU and 50.1% average video-QA accuracy. The two reference variants perform similarly on grounding, while the dense-reference variant obtains a modestly higher QA score in this comparison. These results support the effectiveness of the proposed training configuration; the mechanism underlying its cross-task transfer remains an open question.

Effect of sparse view reward weight. Figure 4 shows the effect of the sparse view reward weight λ_s at 0.1 fps, where sparse-frame inference is most challenging. A small value of λ_s provides insufficient pressure for the model to improve sparse-frame robustness, leading to weaker performance under limited frame budgets. Increasing λ_s improves both sparse view temporal grounding mIoU and video-QA accuracy, showing that explicitly emphasizing the sparse view is important for sparse video reasoning. However, an overly large weight can overemphasize sparse view rewards and weaken the balance between dense view scaffolding and sparse view optimization. We therefore set $\lambda_s = 2$ as the default, which achieves the best overall trade-off across temporal grounding and video-QA. We further analyze the sensitivity to the reference reward weight λ_{ref} in Appendix E, where the results show an optimal intermediate weighting rather than monotonic gains from increasing the reward magnitude.

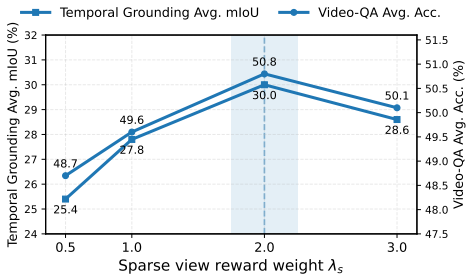


Figure 4: Effect of sparse view reward weight.

Limitations and failure cases. While SAVER improves sparse-frame inference across benchmarks, it does not eliminate all failures under limited temporal evidence. We observe two common failure modes. First, when the sparse frames completely miss the key event, the model may still predict an overly broad or shifted temporal interval. Second, when visually similar actions occur multiple times in a video, the model may align to the wrong occurrence, especially under very low frame rates. We provide additional success and failure cases in the Appendix H. A promising future direction is adaptive online sampling, where the model can request additional frames when sparse observations are insufficient.

5 Conclusion

We presented SAVER, a dense-to-sparse reinforcement post-training framework for sparse-input video understanding. The framework jointly trains dense and sparse views with grounding supervision and uses a relative-quality gate to control an interval-level reference reward. Using 1,250 grounding examples and no video-QA annotations during post-training, SAVER improves macro-averaged performance across the evaluated model scales and frame budgets. Its video-QA results demonstrate cross-task transfer, although the magnitude and direction of the gains vary across benchmarks. Importantly, inference requires neither dense video pre-scanning nor adaptive frame selection or test-time optimization, preserving a fixed sparse-input protocol. Rather than selecting more informative frames at inference time, SAVER learns to make better use of limited temporal evidence through post-training. These findings support training-time dense references as a practical approach to improving performance under limited temporal observations.

AI USE STATEMENT

In this work, we used generative AI tools to assist with developing conceptual frameworks, refining research hypotheses, discussing reward formulations and experimental designs, implementing and debugging code, reviewing literature, preparing figures, and drafting and editing the manuscript. We did not use AI to generate synthetic datasets, develop theoretical models, formulate mathematical claims, contribute to mathematical proofs, assist with translation, clean or reformat datasets, support qualitative or thematic data analysis, or interpret experimental results. All AI-assisted work was critically reviewed and verified by the authors. AI-assisted code was manually tested, figures and tables were checked against original experimental results, and literature references were verified against source publications. The authors take full responsibility for the methodology, results, claims, and final content of this work.

REPRODUCIBILITY STATEMENT

We provide the training and evaluation protocols, implementation details, and computational resources in Section 4 and the Appendix. We plan to release our code, model checkpoints, and sampled training subset to facilitate reproducibility.

REFERENCES

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

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

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*, 2025b.

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, June 2015.

Wang Chen, Yuhui Zeng, Yongdong Luo, Tianyu Xie, Luojun Lin, Jiayi Ji, Yan Zhang, and Xiawu Zheng. Wavelet-based frame selection by detecting semantic boundary for long video understanding. In *CVPR*, pp. 24052–24061, June 2026.

Yukang Chen, Wei Huang, Baifeng Shi, Qinghao Hu, Hanrong Ye, Ligeng Zhu, Zhijian Liu, Pavlo Molchanov, Jan Kautz, Xiaojuan Qi, et al. Scaling RL to long videos. *arXiv preprint arXiv:2507.07966*, 2025.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024a.

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

Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Junfei Wu, Xiaoying Zhang, Benyou Wang, and Xiangyu Yue. Video-r1: Reinforcing video reasoning in mllms. *NeurIPS*, 38:99114–99137, 2025.

Kaituo Feng, Manyuan Zhang, Hongyu Li, Kaixuan Fan, Shuang Chen, Yilei Jiang, Dian Zheng, Peiwen Sun, Yiyuan Zhang, Haoze Sun, et al. Onethinker: All-in-one reasoning model for image and video. In *CVPR*, pp. 5432–5443, 2026.

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 *CVPR*, pp. 24108–24118, 2025a.

Shenghao Fu, Qize Yang, Yuan-Ming Li, Xihan Wei, Xiaohua Xie, and Wei-Shi Zheng. LOVE-R1: Advancing long video understanding with an adaptive zoom-in mechanism via multi-step reasoning. *arXiv preprint arXiv:2509.24786*, 2025b.

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *ICCV*, pp. 5267–5275, 2017.

Sara Ghazanfari, Francesco Croce, Nicolas Flammarion, Prashanth Krishnamurthy, Farshad Khorrami, and Siddharth Garg. Chain-of-frames: Advancing video understanding in multimodal llms via frame-aware reasoning. *arXiv preprint arXiv:2506.00318*, 2025.

Sitong Gong, Lu Zhang, Yunzhi Zhuge, Xu Jia, Pingping Zhang, and Huchuan Lu. Reinforcing video reasoning segmentation to think before it segments. *arXiv preprint arXiv:2508.11538*, 2025.

Kai Hu, Feng Gao, Xiaohan Nie, Peng Zhou, Son Tran, Tal Neiman, Lingyun Wang, Mubarak Shah, Raffay Hamid, Bing Yin, and Trishul Chilimbi. M-LLM based video frame selection for efficient video understanding. In *CVPR*, pp. 13702–13712, 2025a.

Kairui Hu, Penghao Wu, Fanyi Pu, Wang Xiao, Yuanhan Zhang, Xiang Yue, Bo Li, and Ziwei Liu. Video-MMMU: Evaluating knowledge acquisition from multi-discipline professional videos. *arXiv preprint arXiv:2501.13826*, 2025b.

Jindong Jiang, Xiuyu Li, Zhijian Liu, Muyang Li, Guo Chen, Zhiqi Li, De-An Huang, Guilin Liu, Zhiding Yu, Kurt Keutzer, Sungjin Ahn, Jan Kautz, Hongxu Yin, Yao Lu, Song Han, and Wonmin Byeon. Storm: Token-efficient long video understanding for multimodal llms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5830–5841, 2025.

Benno Krojer, Mojtaba Komeili, Candace Ross, Quentin Garrido, Koustuv Sinha, Nicolas Ballas, and Mahmoud Assran. A shortcut-aware video-QA benchmark for physical understanding via minimal video pairs. *arXiv preprint arXiv:2506.09987*, 2025.

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*, 2024a.

Hongyu Li, Songhao Han, Yue Liao, Junfeng Luo, Jialin Gao, Shuicheng Yan, and Si Liu. Reinforcement learning tuning for videollms: Reward design and data efficiency. *arXiv preprint arXiv:2506.01908*, 2025a.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. MVBench: A comprehensive multi-modal video understanding benchmark. In *CVPR*, pp. 22195–22206, 2024b.

KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding. *Science China Information Sciences*, 68(10):200102, 2025b.

Xinhao Li, Ziang Yan, Desen Meng, Lu Dong, Xiangyu Zeng, Yinan He, Yali Wang, Yu Qiao, Yi Wang, and Limin Wang. VideoChat-R1: Enhancing spatio-temporal perception via reinforcement fine-tuning. *arXiv preprint arXiv:2504.06958*, 2025c.

Yixuan Li, Changli Tang, Jimin Zhuang, Yudong Yang, Guangzhi Sun, Wei Li, Zejun MA, and Chao Zhang. Improving LLM video understanding with 16 frames per second. In *ICML*, 2025d.

Yunxin Li, Xinyu Chen, Zitao Li, Zhenyu Liu, Longyue Wang, Wenhan Luo, Baotian Hu, and Min Zhang. VerIPO: Cultivating long reasoning in video-llms via verifier-guided iterative policy optimization. *arXiv preprint arXiv:2505.19000*, 2025e.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 5971–5984, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, volume 36, pp. 34892–34916, 2023.

Shuming Liu, Mingchen Zhuge, Changsheng Zhao, Jun Chen, Lemeng Wu, Zechun Liu, Chenchen Zhu, Zhipeng Cai, Chong Zhou, Haozhe Liu, et al. VideoAuto-R1: Video auto reasoning via thinking once, answering twice. *arXiv preprint arXiv:2601.05175*, 2026a.

Yuqi Liu, Tianyuan Qu, Zhisheng Zhong, Bohao Peng, Shu Liu, Bei Yu, and Jiaya Jia. Visionreasoner: Unified reasoning-integrated visual perception via reinforcement learning. In *ICLR*, volume 2026, pp. 94069–94086, 2026b.

Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. Timechat: A time-sensitive multimodal large language model for long video understanding. In *CVPR*, pp. 14313–14323. IEEE, 2024.

Leqi Shen, Guoqiang Gong, Tao He, Yifeng Zhang, Pengzhang Liu, Sicheng Zhao, et al. Fastvid: Dynamic density pruning for fast video large language models. *NeurIPS*, 38:123553–123581, 2025.

Wenfang Sun, Hao Chen, Yingjun Du, Yefeng Zheng, and Cees G Snoek. Regionreasoner: Region-grounded multi-round visual reasoning. In *ICLR*, volume 2026, pp. 80583–80603, 2026.

Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive keyframe sampling for long video understanding. In *CVPR*, pp. 29118–29128, 2025.

Qwen Team. Qwen3.5: Accelerating productivity with native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution, 2024.

Ye Wang, Ziheng Wang, Boshen Xu, Yang Du, Kejun Lin, Zihan Xiao, Zihao Yue, Jianzhong Ju, Liang Zhang, Dingyi Yang, et al. Time-R1: Post-training large vision language model for temporal video grounding. *arXiv preprint arXiv:2503.13377*, 2025.

Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. LongVideoBench: A benchmark for long-context interleaved video-language understanding. In *NeurIPS*, volume 37, pp. 28828–28857, 2024.

Peiran Wu, Zhuorui Yu, Yunze Liu, Chi-Hao Wu, Enmin Zhou, and Junxiao Shen. MARC: Memory-augmented RL token compression for efficient video understanding. *arXiv preprint arXiv:2510.07915*, 2025.

Junbin Xiao, Angela Yao, Yicong Li, and Tat-Seng Chua. Can I trust your answer? Visually grounded video question answering. In *CVPR*, pp. 13204–13214, 2024.

Chenwei Xu, Zhen Ye, Shang Wu, Weijian Li, Zihan Wang, Zhuofan Xia, Lie Lu, Pranav Maneriker, Fan Du, Manling Li, and Han Liu. Towards sparse video understanding and reasoning. In *CVPR*, pp. 11357–11368, June 2026.

Linli Yao, Haoning Wu, Kun Ouyang, Yuanxing Zhang, Caiming Xiong, Bei Chen, Xu Sun, and Junnan Li. Generative frame sampler for long video understanding. In *Findings of the Association for Computational Linguistics*, pp. 17900–17917, 2025.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. LMMs-Eval: Reality check on the evaluation of large multimodal models, 2024. URL https://arxiv.org/abs/2407.12772.

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 *ICCV*, pp. 22056–22065, 2025a.

Xuan Zhang, Cunxiao Du, Sicheng Yu, Jiawei Wu, Fengzhuo Zhang, Wei Gao, and Qian Liu. Sparse-to-dense: A free lunch for lossless acceleration of video understanding in llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 734–742, 2025b.

Yilun Zhao, Haowei Zhang, Lujing Xie, Tongyan Hu, Guo Gan, Yitao Long, Zhiyuan Hu, Weiyuan Chen, Chuhan Li, Zhijian Xu, et al. MMVU: Measuring expert-level multi-discipline video understanding. In *CVPR*, pp. 8475–8489, 2025.

Junbo Zou, Ziheng Huang, Shengjie Zhang, Liwen Zhang, and Weining Shen. VideoBrain: Learning adaptive frame sampling for long video understanding. *arXiv preprint arXiv:2602.04094*, 2026.

A Controlled Frame-Rate Evaluation

We provide detailed results under the controlled frame-rate protocol in Table 5 and Table 6. For each fps setting, we report both the task performance and the corresponding average number of observed frames for each benchmark individually. To ensure stable and meaningful video evaluation, the total number of sampled frames is capped within $[4, 256]$ per video. We additionally report the average frame usage both for each individual benchmark and across all temporal grounding or video-QA benchmarks. The detailed results further demonstrate that reducing the sampling rate substantially lowers the visual input budget across all benchmarks, but also makes sparse-frame reasoning significantly more challenging for the original Qwen3.5 models. In contrast, SAVER improves average performance under matched frame budgets across all model scales and fps settings, for both temporal grounding and video-QA. In most settings, the largest gains appear at 0.5 fps or below, where temporal evidence becomes highly sparse and incomplete. Notably, SAVER substantially narrows the sparse–dense performance gap, and in several cases sparse SAVER inference approaches or even surpasses denser Qwen3.5 baselines while using far fewer observed frames. These results suggest that dense-to-sparse post-training improves the robustness and reliability of video reasoning under limited temporal evidence, rather than merely reducing the number of processed frames.

Table 5: Temporal grounding results under controlled frame budgets. For each fps setting, we report grounding performance and the corresponding number of observed frames for three model scales (0.8B, 2B, and 4B). Avg. Frames denotes the average number of observed frames across all grounding benchmarks, and Avg. mIoU denotes the average grounding performance.

Model	Size	FPS	Avg. Frames	Temporal Grounding						Avg. mIoU
				Charades-STA	Frames	ActivityNet	Frames	NExT-GQA	Frames	
Controlled frame-rate evaluation: 0.8B model										
Qwen3.5	0.8B	2.0	110	43.9	59	38.0	191	26.6	80	36.1
Qwen3.5	0.8B	1.0	63	39.1	29	34.4	121	24.0	39	32.5
Qwen3.5	0.8B	0.5	31	33.4	14	28.4	60	19.0	19	26.9
Qwen3.5	0.8B	0.2	12	24.4	5	23.1	24	12.5	8	20.0
Qwen3.5	0.8B	0.1	7	24.0	4	21.7	12	10.3	5	18.7
SAVER	**0.8B**	**2.0**	**110**	50.5	59	45.3	191	28.4	80	**41.4**
SAVER	**0.8B**	**1.0**	**63**	47.2	29	44.3	121	15.6	39	**35.7**
SAVER	**0.8B**	**0.5**	**31**	40.7	14	42.4	60	17.6	19	**33.6**
SAVER	**0.8B**	**0.2**	**12**	31.5	5	38.8	24	18.0	8	**29.4**
SAVER	**0.8B**	**0.1**	**7**	31.1	4	32.7	12	15.6	5	**26.5**
Controlled frame-rate evaluation: 2B model										
Qwen3.5	2B	2.0	110	44.0	59	38.3	191	22.6	80	34.9
Qwen3.5	2B	1.0	63	39.9	29	34.5	121	24.6	39	33.0
Qwen3.5	2B	0.5	31	32.3	14	29.8	60	21.7	19	27.9
Qwen3.5	2B	0.2	12	17.9	5	22.6	24	18.2	8	19.5
Qwen3.5	2B	0.1	7	17.6	4	20.7	12	16.9	5	18.4
SAVER	**2B**	**2.0**	**110**	54.3	59	44.8	191	37.1	80	**45.4**
SAVER	**2B**	**1.0**	**63**	52.0	29	44.2	121	34.9	39	**43.7**
SAVER	**2B**	**0.5**	**31**	45.2	14	43.0	60	38.9	19	**42.4**
SAVER	**2B**	**0.2**	**12**	31.2	5	40.4	24	25.5	8	**32.3**
SAVER	**2B**	**0.1**	**7**	28.7	4	37.2	12	24.2	5	**30.0**
Controlled frame-rate evaluation: 4B model										
Qwen3.5	4B	2.0	110	54.2	59	44.7	191	8.1	80	35.7
Qwen3.5	4B	1.0	63	51.5	29	43.5	121	6.3	39	33.8
Qwen3.5	4B	0.5	31	43.9	14	40.5	60	3.9	19	29.4
Qwen3.5	4B	0.2	12	29.3	5	36.4	24	4.1	8	23.2
Qwen3.5	4B	0.1	7	25.0	4	29.8	12	4.3	5	19.7
SAVER	**4B**	**2.0**	**110**	**54.8**	59	**46.9**	191	**38.2**	80	**46.6**
SAVER	**4B**	**1.0**	**63**	53.4	29	46.4	121	34.5	39	**44.7**
SAVER	**4B**	**0.5**	**31**	48.0	14	43.4	60	31.3	19	**40.9**
SAVER	**4B**	**0.2**	**12**	33.8	5	38.4	24	25.0	8	**32.4**
SAVER	**4B**	**0.1**	**7**	29.5	4	32.5	12	22.7	5	**28.2**

B Effect of sparse view Training Frame Rate

Table 7 studies the effect of the sparse view frame rate used during post-training, while fixing the dense view frame rate to 2.0 fps. We evaluate all variants under 0.1-fps inference, where sparse-frame reasoning is most challenging. Using an excessively sparse training view (0.1 fps) weakens the temporal evidence available to the sparse branch during training, making dense-to-sparse alignment less effective. In contrast, using a less sparse training view (1.0 fps) reduces the discrepancy between

Table 6: Video-QA results under controlled frame budgets. For each fps setting, we report benchmark performance and the corresponding number of observed frames across video perception and video reasoning tasks for three model scales (0.8B, 2B, and 4B). Avg. Frames denotes the average number of observed frames across all benchmarks, and Avg. Acc. denotes the average performance.

Model	Size	FPS	Avg. Frames	Video Perception								Video Reasoning				Avg. Acc.
				VideoMME	Frames	MVBench	Frames	LongVideoBench	Frames	MMVU	Frames	VideoMMMU	Frames	MVP	Frames	
Controlled frame-rate evaluation: 0.8B model																
Qwen3.5	0.8B	2.0	139	55.0	224	49.2	35	54.2	200	50.4	99	32.7	253	25.0	25	44.4
Qwen3.5	0.8B	1.0	118	54.8	197	47.6	17	52.8	190	50.1	51	33.1	240	24.7	12	43.9
Qwen3.5	0.8B	0.5	97	54.0	171	46.4	9	52.1	170	50.2	25	35.3	200	22.2	6	43.4
Qwen3.5	0.8B	0.2	61	51.6	124	44.3	5	50.9	122	49.4	10	34.1	99	21.1	5	41.9
Qwen3.5	0.8B	0.1	38	49.0	96	43.7	4	49.1	71	49.0	6	31.6	50	20.1	4	40.4
SAVER	**0.8B**	**2.0**	**139**	56.3	224	52.9	35	54.7	200	51.2	99	40.8	253	23.4	25	**46.5**
SAVER	**0.8B**	**1.0**	**118**	56.2	197	51.1	17	53.3	190	50.6	51	38.9	240	22.3	12	**45.4**
SAVER	**0.8B**	**0.5**	**97**	55.0	171	49.6	9	52.3	170	47.9	25	39.1	200	21.2	6	**44.2**
SAVER	**0.8B**	**0.2**	**61**	52.5	124	47.1	5	50.5	122	45.8	10	39.7	99	19.9	5	**42.6**
SAVER	**0.8B**	**0.1**	**38**	50.3	96	46.0	4	49.5	71	50.1	6	38.3	50	17.6	4	**42.0**
Controlled frame-rate evaluation: 2B model																
Qwen3.5	2B	2.0	139	62.6	224	58.6	35	55.9	200	61.0	99	38.8	253	29.4	25	51.0
Qwen3.5	2B	1.0	118	62.4	197	56.2	17	54.5	190	60.0	51	38.3	240	27.4	12	49.8
Qwen3.5	2B	0.5	97	61.7	171	55.7	9	52.4	170	58.7	25	40.2	200	24.9	6	48.9
Qwen3.5	2B	0.2	61	59.3	124	53.3	5	51.9	122	57.8	10	40.3	99	22.8	5	47.6
Qwen3.5	2B	0.1	38	55.9	96	52.1	4	51.3	71	57.0	6	37.4	50	21.9	4	45.9
SAVER	**2B**	**2.0**	**139**	61.1	224	56.9	35	60.5	200	61.8	99	57.0	253	30.5	25	**54.6**
SAVER	**2B**	**1.0**	**118**	61.0	197	56.6	17	59.8	190	61.4	51	56.0	240	28.3	12	**53.9**
SAVER	**2B**	**0.5**	**97**	59.9	171	57.0	9	59.0	170	60.2	25	57.9	200	26.5	6	**53.4**
SAVER	**2B**	**0.2**	**61**	56.9	124	54.5	5	56.8	122	60.6	10	57.7	99	24.5	5	**51.8**
SAVER	**2B**	**0.1**	**38**	53.6	96	54.3	4	56.0	71	59.7	6	56.1	50	24.9	4	**50.8**
Controlled frame-rate evaluation: 4B model																
Qwen3.5	4B	2.0	139	70.9	224	64.4	35	64.7	200	68.2	99	54.8	253	41.9	25	60.8
Qwen3.5	4B	1.0	118	70.1	197	62.6	17	64.3	190	68.7	51	54.8	240	36.5	12	59.5
Qwen3.5	4B	0.5	97	69.5	171	62.3	9	62.6	170	67.2	25	54.2	200	34.3	6	58.4
Qwen3.5	4B	0.2	61	66.9	124	58.7	5	59.8	122	67.4	10	54.1	99	30.3	5	56.2
Qwen3.5	4B	0.1	38	63.2	96	57.9	4	59.3	71	66.6	6	52.4	50	29.9	4	54.9
SAVER	**4B**	**2.0**	**139**	**71.8**	224	**65.3**	35	**65.6**	200	**70.9**	99	69.8	253	**43.1**	25	**64.4**
SAVER	**4B**	**1.0**	**118**	71.3	197	62.4	17	64.5	190	70.2	51	71.1	240	36.5	12	**62.7**
SAVER	**4B**	**0.5**	**97**	70.3	171	61.4	9	63.8	170	67.7	25	71.2	200	34.6	6	**61.5**
SAVER	**4B**	**0.2**	**61**	67.0	124	59.6	5	62.0	122	67.8	10	72.6	99	30.6	5	**60.0**
SAVER	**4B**	**0.1**	**38**	64.0	96	59.1	4	60.0	71	68.0	6	**72.7**	50	30.8	4	**59.1**

Table 7: Effect of sparse view training frame rate. We fix the dense view training frame rate to 2.0 fps and vary the sparse view frame rate used during post-training. Results are evaluated under 0.1-fps inference, where sparse-frame reasoning is most challenging.

Model	Size	Sparse View FPS	TG Avg. mIoU	Video-QA Avg. Acc.
SAVER	2B	0.1	29.6	49.7
SAVER	2B	**0.5**	**30.0**	**50.8**
SAVER	2B	1.0	28.4	48.8

dense and sparse observations, weakening the robustness gains under sparse inference. We find that a moderate sparse view frame rate (0.5 fps) provides the best trade-off, achieving the strongest performance on both temporal grounding and video-QA benchmarks.

C Practical Efficiency Profiling

To complement the sampled-frame analysis in the main paper, we further profile the practical inference cost of sparse video processing in terms of visual-token count, peak GPU memory, end-to-end latency, GPU inference latency, and estimated computation. We evaluate Qwen3.5-2B at 2.0 and 0.1 fps on the same three temporal-grounding and six video-QA benchmarks used in the main experiments. All measurements are collected on a single 96-GB NVIDIA H100 GPU with batch size 1 and BF16 precision. We perform one warm-up generation before measurement and use CUDA synchronization for latency profiling. Table 8 reports each metric as 2.0 fps / 0.1 fps.

Averaged across all nine benchmarks, reducing the frame rate from 2.0 to 0.1 fps decreases the number of visual tokens by 86.8%. End-to-end latency decreases by 63.6%, corresponding to a 2.74× speedup, while GPU inference latency decreases by 63.9%. Peak GPU memory usage is reduced by 49.3%. These results show that the reduction in sampled visual observations translates into substantial practical inference savings rather than only a smaller nominal frame count. MVBench is the only benchmark for which latency increases at 0.1 fps. This behavior is primarily associated with longer output generation rather than increased visual processing. At 0.1 fps, the model generates 43.3 output

Table 8: Practical efficiency profiling of Qwen3.5-2B at 2.0 and 0.1 fps. Lower frame rates substantially reduce visual-token count, peak GPU memory, and inference latency on average across nine evaluation benchmarks.

	Visual tokens ↓		Peak memory (GiB) ↓		E2E latency (s) ↓		GPU inference (s) ↓	
Benchmark	**2.0 fps**	**0.1 fps**	**2.0 fps**	**0.1 fps**	**2.0 fps**	**0.1 fps**	**2.0 fps**	**0.1 fps**
Charades-STA	3,746	251	5.04	4.24	1.72	0.97	1.49	0.89
NExT-GQA	10,402	515	8.94	4.37	2.12	0.70	1.63	0.60
ActivityNet	8,039	376	6.65	4.30	2.41	0.73	1.62	0.65
MMVU	8,689	474	7.70	4.33	2.16	1.78	1.93	1.67
VideoMME	69,750	3,060	18.72	4.80	10.03	1.41	8.17	1.12
LongVideoBench	92,160	13,950	20.65	7.89	16.59	4.31	11.57	3.04
MVBench	1,449	268	4.66	4.24	0.59	1.31	0.49	1.22
MVP	14,625	1,440	7.77	4.41	2.96	2.34	1.88	1.79
VideoMMMU	89,160	19,137	21.54	12.99	20.62	8.02	12.03	3.74
Macro average	**33,113**	**4,386**	**11.30**	**5.73**	**6.58**	**2.40**	**4.53**	**1.63**

Table 9: Comparison with inference-time frame/token selection methods. We evaluate Qwen3.5-2B, SAVER, FastVID, and WFS-SB on one temporal-grounding benchmark and two video-QA benchmarks. SAVER and the selection-based approaches target complementary aspects of efficient video reasoning, and combining SAVER with WFS-SB yields the strongest overall performance. The average is computed over the three reported benchmarks.

Method	**ActivityNet TG**	**LongVideoBench VQA**	**MMVU VQA**	**Avg.**
Qwen3.5-2B (0.1 fps)	20.7	51.3	57.0	43.0
Qwen3.5-2B + FastVID	31.1	51.0	59.3	47.1
Qwen3.5-2B + WFS-SB	30.0	56.6	60.3	49.0
SAVER (0.1 fps)	37.2	56.0	59.7	51.0
SAVER + WFS-SB	**42.1**	**62.3**	**60.7**	**55.0**

tokens on average, compared with only 8.8 output tokens on average at 2.0 fps. Despite the shorter visual input, the longer generation increases end-to-end latency from 0.59 to 1.31 seconds and GPU inference latency from 0.49 to 1.22 seconds.

Measurement details. Visual-token counts are computed from the processor-provided video and image grids after spatial merging. End-to-end latency includes video loading, frame sampling, preprocessing, host-to-device transfer, and generation. GPU inference latency measures synchronized `model.generate()` time. Peak GPU memory is measured using `torch.cuda.max_memory_allocated()`.

D Comparison with Inference-Time Selection Methods

We compare SAVER with two representative inference-time efficiency methods: FastVID (Shen et al., 2025) and WFS-SB (Chen et al., 2026). Both methods reduce inference cost by pruning visual tokens or selecting informative frames from richer visual inputs. In contrast, SAVER improves sparse-frame reasoning through dense-to-sparse post-training and operates directly under a prescribed sparse-input budget without dense pre-scanning or additional selection at inference time.

Table 9 shows that adaptive selection and SAVER improve efficiency from different directions. FastVID and WFS-SB improve the quality of the visual input by selecting or retaining informative observations from a richer candidate pool, whereas SAVER improves the model itself so that it can reason more reliably from a fixed sparse input. Under the same Qwen3.5-2B backbone, SAVER at 0.1 fps achieves an average score of 51.0, compared with 47.1 for FastVID and 49.0 for WFS-SB across the three reported benchmarks.

The two approaches are also naturally compositional. When a richer candidate pool is available, WFS-SB can first select informative frames, after which SAVER processes the resulting sparse observations. Combining SAVER with WFS-SB improves the average performance from 51.0 to 55.0, including gains from 37.2 to 42.1 on ActivityNet and from 56.0 to 62.3 on LongVideoBench. This result indicates that training-time sparse robustness and inference-time adaptive selection are complementary rather than competing approaches.

Table 10: Practical inference cost of SAVER and inference-time selection methods. Lower values indicate lower inference cost. SAVER directly operates on sparse inputs, while FastVID and WFS-SB introduce additional inference-time processing for pruning or frame selection.

Method	Visual tokens ↓	Peak memory (GiB) ↓	GPU inference (s) ↓	E2E latency (s) ↓
SAVER	**4,933**	**5.51**	**1.79**	**2.27**
FastVID	5,292	13.39	2.22	4.42
WFS-SB	5,393	6.96	1.94	6.36

Table 11: Effect of the reference reward weight λ_{ref}. Temporal-grounding results are averaged over three benchmarks and video-QA results over six benchmarks.

λ_{ref}	TG mIoU	QA Avg.
0.0	27.4	48.7
0.5	29.2	50.0
1.0	30.0	**50.8**
2.0	**30.1**	50.2

Practical inference cost. We further compare the practical inference cost of SAVER, FastVID, and WFS-SB under matched profiling settings, using the Qwen3.5-2B backbone and averaging over the three benchmarks in Table 9. Table 10 reports visual-token count, peak GPU memory, GPU inference latency and end-to-end latency. The practical-cost comparison further highlights the different inference trade-offs. SAVER processes fewer visual tokens and uses less peak GPU memory than the two selection-based methods, while also requiring lower GPU inference latency and end-to-end latency under the matched profiling setting. This advantage follows from the different inference pipeline: SAVER improves sparse-input capability during post-training and therefore does not require dense pre-scanning, intermediate feature extraction, or an additional selection stage at inference time.

FastVID and WFS-SB instead spend additional inference-time computation to identify informative visual content from a richer candidate representation. Such processing can be worthwhile when a richer input pool is available, as demonstrated by the improved performance of SAVER + WFS-SB in Table 9. Taken together, these results suggest that SAVER provides a complementary training-time approach to efficient video reasoning: it reduces the model's dependence on dense observations, while adaptive selection methods can further improve the sparse input when additional inference-time computation is available.

E Effect of Reference Reward Weight

We study the sensitivity of SAVER to the reference reward coefficient λ_{ref}. This analysis examines whether the observed gains can be explained simply by increasing the relative contribution of the reference reward during optimization. All results use Qwen3.5-2B and are evaluated at 0.1 fps.

As shown in Table 11, reward weighting affects performance, but the gains are not explained by monotonically increasing the reference reward weight. Starting from $\lambda_{\text{ref}} = 0$, introducing the reference objective substantially improves both temporal grounding and video-QA. Performance improves as λ_{ref} increases to 1.0, whereas further increasing it to 2.0 yields only a marginal improvement in temporal grounding (30.0 to 30.1 mIoU) while reducing video-QA accuracy from 50.8 to 50.2. These results indicate that the reference reward contributes a distinct cross-view alignment objective, while λ_{ref} controls the strength of this objective during optimization. The benefit therefore comes from both the dense-to-sparse alignment signal and an appropriate balance among reward components, rather than from simply increasing the magnitude of a single reward term.

Table 12: Effect of grounding post-training data scale. We vary the number of Time-R1 grounding training examples used during post-training. Results are evaluated under 0.1-fps inference.

Model	Size	Training Samples	TG Avg. mIoU	Video-QA Avg. Acc.
SAVER	2B	0	18.4	45.9
SAVER	2B	625	25.7	47.6
SAVER	2B	1250	**30.0**	**50.8**
SAVER	2B	1875	29.5	49.7
SAVER	2B	2500	28.6	48.2

Table 13: Confidence-based sparse-to-dense fallback. Inference starts from the 0.1-fps sparse view, and dense 2.0-fps inference is triggered when the sparse prediction confidence is below γ. Increasing γ routes more examples to dense inference and therefore increases the average number of processed frames.

Policy	γ	Temporal Grounding		Video-QA	
		Avg. Frames	Avg. mIoU	Avg. Frames	Avg. Acc.
Qwen3.5 (dense)	–	110	34.9	139	51.0
SAVER (sparse only)	–	7	30.0	38	50.8
Fallback	0.3	8	29.1	39	50.5
Fallback	0.5	11	31.0	43	50.9
Fallback	0.7	47	38.3	54	52.0
SAVER (dense)	–	110	**45.4**	139	**54.6**

F Effect of Grounding Post-Training Data Scale

We study the effect of the grounding post-training data scale in Table 12. Time-R1 contains approximately 2500 temporal grounding training examples, while our main experiments use only a randomly sampled subset of 1250 examples. We observe that increasing the amount of grounding supervision initially improves both temporal grounding and video-QA performance. However, the gains saturate beyond 1250 examples, and using the full 2500-example training set slightly degrades performance under sparse inference. We hypothesize that excessive grounding-focused post-training may over-specialize the model toward narrow localization patterns and reduce the diversity of temporal reasoning behaviors. Overall, these results suggest that effective dense-to-sparse reasoning transfer can emerge from a relatively small amount of grounding supervision.

G Confidence-Based Sparse-to-Dense Fallback

SAVER is designed for direct sparse-frame inference and does not require dense video access at test time. Nevertheless, when additional computation is available, the sparse prediction can optionally be used to decide whether a denser observation should be requested. We therefore study a simple confidence-based sparse-to-dense fallback strategy.

We first perform inference using sparse frames sampled at 0.1 fps. If the confidence of the sparse prediction falls below a threshold γ, we rerun the same example using frames sampled at 2.0 fps and use the dense-view prediction instead. Table 13 reports the resulting performance–computation trade-off for temporal grounding and video-QA.

As γ increases, more examples are routed to dense inference, increasing the average frame budget and generally moving performance toward the dense-view setting. However, the fallback strategy does not improve performance monotonically. At lower thresholds, the confidence signal is not sufficiently calibrated to identify examples that truly require additional temporal evidence. As a result,



Locate the start and end timestamps of the video segment corresponding to the description: a person opens a small cabinet door. Please provide the start and end timestamps (in seconds, precise to one decimal places) in the format boxed [start, end].

Ground Truth: [0.0, 4.1015625]

SAVER 4B (Reasoning) :

The action “a person opens a small cabinet door” begins immediately at the start of the clip. The person is already positioned at the cabinet and starts the opening motion from the beginning. The door-opening action continues for the first few seconds and appears essentially completed by around 4.0s. After that, the clip mainly shows the post-action state rather than the core opening event.

Therefore, the most precise temporal segment for the target event is approximately [0.0, 4.0]. ✅

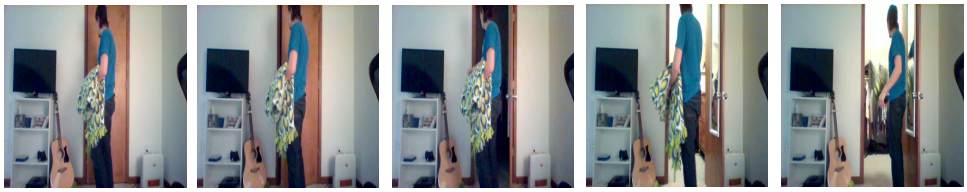
Figure 5: Temporal grounding success case on Charades-STA. Given the query “a person opens a small cabinet door,” SAVER correctly identifies that the action starts at the beginning of the clip and ends around 4 seconds. The predicted interval closely matches the ground-truth segment, showing that sparse-frame reasoning can succeed when the sampled frames preserve the key temporal evidence.

some correct sparse predictions are replaced by less helpful dense predictions, leading to degraded temporal-grounding performance.

At $\gamma = 0.7$, the fallback strategy improves temporal grounding from 30.0 to 38.3 average mIoU and video-QA from 50.8 to 52.0 average accuracy, while processing substantially fewer frames than dense-only inference (47 vs. 110 frames for temporal grounding and 54 vs. 139 frames for video-QA). These results indicate that selective access to denser observations can provide additional gains when sparse evidence is insufficient, but its effectiveness depends critically on reliable confidence calibration. We therefore treat confidence-based fallback as an optional extension rather than part of the default SAVER inference protocol.

H Additional Qualitative Cases

We provide additional qualitative examples for both temporal grounding and video question answering under the extremely sparse inference setting of 0.1 fps. Figures 5 and 6 show Charades-STA grounding cases, where SAVER predicts temporal segments from sparse frames. Figures 7 and 8 show VideoMMMU cases, where SAVER must solve reasoning-intensive video questions from sparse observations. Together, these examples show that SAVER can preserve task-relevant evidence when the sampled frames contain sufficient cues, but may still fail when the sparse observations miss critical information or when the reasoning chain contains an incorrect intermediate assumption.



Locate the start and end timestamps of the video segment corresponding to the description: person throws their blanket inside. Please provide the start and end timestamps (in seconds, precise to one decimal places) in the format boxed{[start, end]}.

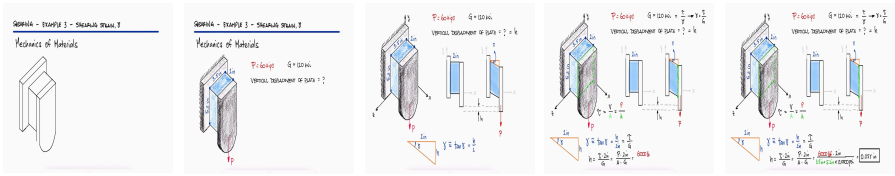
Ground Truth: [4.30078125, 8.0]

SAVER 4B (Reasoning) :

The person is standing near the doorway while holding a blanket. Around 3 seconds, they move closer to the open door and start extending the blanket forward. This appears to be the moment when they throw the blanket inside. By around 4 seconds, the blanket is no longer clearly visible in the person's hands, suggesting the throwing action has already finished.

Therefore, the segment corresponding to "person throws their blanket inside" is approximately from [3.0, 4.0]. ❌

Figure 6: Temporal grounding failure case on Charades-STA. Given the query "person throws their blanket inside," SAVER predicts an earlier interval than the ground truth. Although the reasoning identifies a plausible moment when the person moves near the doorway, the sparse observations do not provide enough evidence to precisely distinguish the target throwing action. This example highlights a common failure mode: sparse-frame inference can still localize an incorrect segment when the key action is ambiguous or insufficiently captured.



Based on your understanding of the solution, calculate the new value of the vertical displacement h if the length changes from 2 inches to 3 inches.

A. 0.075 inches B. 0.080 inches C. 0.092 inches D.0.082 inches E. 0.055 inches
F. 0.068 inches G.0.095 inches H. 0.078 inches I. 0.088 inches J. 0.102 inches

SAVER 4B (Reasoning) :

Let's analyze the problem step by step. We are given a shearing strain problem where the vertical displacement h is related to the shearing strain γ and the geometry of the setup.

From the video:

Shear modulus: G = 120 ksi = 120,000 psi

Applied load: P = 60 kips = 60,000 lb

Original dimensions: width = 2 in, height = 5.2 in, length = 3.5 in

Using the shear relation:

$\gamma \approx \tan(\gamma) = h / L = \tau / G = (V / A) / G$

So, $h = (V \times L) / (A \times G)$

where:

V = P = 60,000 lb

L is the sheared length dimension

$A = 3.5 \times 5.2 = 18.2$ in²

Original case (L = 2 in):

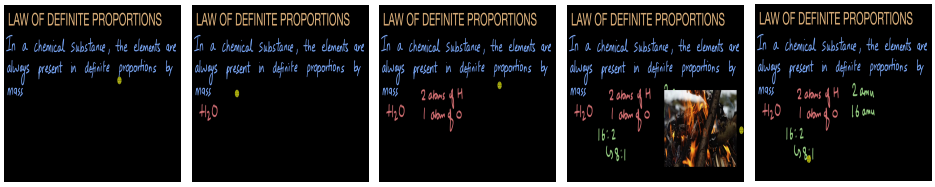
h_original = (60000 × 2) / (3.5 × 5.2 × 120000)
= 120000 / 2184000
= 0.054945 ≈ 0.055 in

Now the problem changes the 2-inch length to 3 inches, so displacement scales proportionally:

h_new = (3 / 2) × 0.054945
= 0.0824175 ≈ 0.082 in

Therefore, the correct answer is: D. 0.082 inches ✅

Figure 7: Video-QA success case on VideoMMMU. Given a mechanics question about how the vertical displacement changes when the length increases from 2 inches to 3 inches, SAVER correctly follows the visual solution and derives the proportional change. The model identifies the relevant formula, applies the new length, and selects the correct answer, showing that sparse video reasoning can succeed when the sampled frames retain the necessary visual and textual evidence.



Based on the description in the video, 4 kg of hydrogen atoms correspond to how many oxygen atoms?

A. 16 kg B. 48 kg C. 32 kg D. 64 kg E. 128 kg

SAVER 4B (Reasoning) :

Let's analyze the problem step by step.

The Law of Definite Proportions states that elements in a compound always combine in a fixed mass ratio. For water (H_2O):

Hydrogen atomic mass ≈ 1 amu
Oxygen atomic mass ≈ 16 amu

Since water contains 2 hydrogen atoms and 1 oxygen atom, the mass ratio is:

Hydrogen : Oxygen = 2 × 1 : 1 × 16 = 2 : 16 = 1 : 8

So oxygen has 8 times the mass of hydrogen.

Given 4 kg of hydrogen:Mass of oxygen = 4 × 8 = 32 kg

Therefore, the correct answer is C ❌

Figure 8: Video-QA failure case on VideoMMMU. Given a chemistry question about the mass of oxygen corresponding to 4 kg of hydrogen atoms, SAVER applies the standard hydrogen-to-oxygen mass ratio in water (1:8) and selects C (32 kg), whereas the ground-truth answer is D (64 kg), which corresponds to a 1:16 mass relation. Although the arithmetic is internally consistent, the model relies on a default prior assumption instead of the relation presented in the video. This example shows that sparse video reasoning can still fail when the reasoning chain is built on an incorrect premise that is not grounded in the visual evidence.