VIDEOEVOHARNESS: SELF EVOLVING AGENT HARNESS FROM UNLABELED VIDEOS FOR VIDEO UNDERSTANDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Video understanding agents rely on a harness to acquire evidence, use tools, and reason, but improving it requires substantial effort. Recent self evolution methods seek to optimize harnesses through execution feedback to reduce human intervention. However, existing methods often rely on annotated video questions and answers or focus on updating model parameters, leaving harness evolution from unlabeled videos underexplored. VideoEvoHarness jointly evolves a Questioner and a Solver harness from unlabeled videos with frozen model parameters. The Questioner constructs questions and computes reference answers directly from video memory records. Question quality assessments and Solver execution trajectories guide question generation policy revisions. The Solver supports evolving skills, strategies, hyperparameters, and tools using execution feedback. Executable strategies define workflows in code, reducing repeated interpretation of natural language instructions. Each round, candidate harnesses are compared on identical validation questions, with validated improvements retained for later use. Experiments on benchmarks and a dedicated benchmark question set, using videos disjoint from evolution videos, show higher overall accuracy than initially and transfer gains across backbones. Overall accuracy on this set also improves progressively across rounds. This supports our foundational framework for harness evolution from unlabeled videos, opening a promising direction for self evolving video agents.

1 INTRODUCTION

Recent advances in large vision language models have improved their ability to understand video content and reason about events as they unfold over time (Bai et al., 2025; He et al., 2024; Ren et al., 2024; Shu et al., 2025; Diko et al., 2025). These advances have enabled the emergence of video agents. They use tools to retrieve evidence and select information for reasoning and answering questions (Wang et al., 2024; Fan et al., 2025; Long et al., 2026; Ma et al., 2025). Their performance therefore depends not only on the model but also on how effectively tools are used. These patterns of agent behavior are coordinated by the harness, which integrates instructions, reusable skills, tools, and execution procedures around the model (Yao et al., 2023; Cai et al., 2024; Khattab et al., 2024).

Designing an agent harness requires decisions across its components and their interactions. When these decisions remain fixed, newly observed failure trajectories require manual revision of the relevant harness components. This motivates agent self evolution, which enables an agent to analyze its own execution outcomes and automatically evaluate and revise its harness (Zhang et al., 2026b). Existing work on general agent systems illustrates this process. GEPA refines prompts by reflecting on execution traces (Agrawal et al., 2026). Meta Agent Search uses a model to write agent programs, evaluate them, and develop new versions from previous designs and their results (Hu et al., 2025). These developments motivate exploring self evolution for video agents, allowing task experience to inform subsequent evidence acquisition and reasoning.

Applying agent evolution to video understanding requires reliable evaluation, yet constructing video questions and verified answers is costly and demands careful quality control. The annotation process must examine video content, interpret temporal context, and verify answers against visual evidence

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.

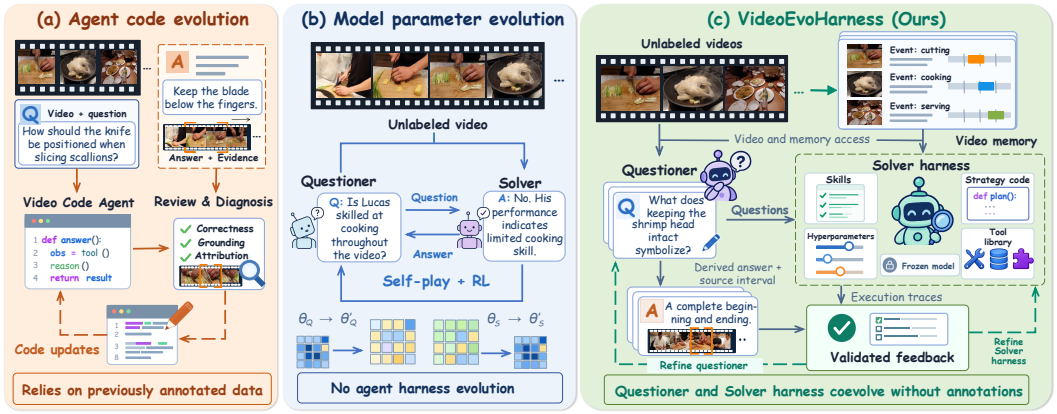


Figure 1: Comparison of video agent evolution paradigms. (a) Agent code evolution revises agent code through review and diagnosis, and relies on previously annotated questions, answers, and evidence. (b) Model parameter evolution builds supervision from unlabeled video through Questioner and Solver interaction, but updates model parameters and leaves the agent harness unchanged. (c) VideoEvoHarness derives questions, reference answers, and source intervals from video memory, and uses validated feedback to support Questioner refinement and Solver updates across skills, strategy code, hyperparameters, and tools, while the model stays frozen.

(Fu et al., 2025). In video agent evolution, one line of work depends heavily on previously annotated questions, answers, and evidence, and modifies the harness by evolving agent code, as illustrated in Figure 1(a) (Cui et al., 2026). Raw videos, by contrast, are abundant online and typically come without accompanying questions or answers. This gap motivates us to consider using unlabeled videos directly as a source of evolution feedback. To do so, an agent must construct evaluation tasks from unlabeled video. As illustrated in Figure 1(b), EvoVid introduces a Questioner and a Solver to derive supervision from raw video clips and iteratively update model parameters (Huang et al., 2026). However, sampled clips may omit parts of an event and its temporal context, making it difficult to generate informative questions. This leads to our first challenge: *How can video agents derive effective evolution feedback from unlabeled videos without externally supplied questions and answers?*

Beyond obtaining feedback, we must also consider how to evolve the agent harness. As shown in Figure 1(b), EvoVid updates only model parameters without evolving the agent harness (Huang et al., 2026). MetaVideoAgent evolves the harness, but limits its updates to agent code, as shown in Figure 1(a) (Cui et al., 2026). Another approach under exploration adapts agent behavior by revising prompts and reusable textual skills (Agrawal et al., 2026; Tu et al., 2026). In agent environments, these skills can build on established tools for search, file manipulation, and program execution. Video tasks, however, depend on temporal localization and the analysis of visual evidence. Their toolchains remain under active development, with different systems providing different combinations of these capabilities (Wang et al., 2024; Fan et al., 2025). Textual skill refinement alone may therefore be insufficient for complex video tasks. Executing these skills also requires the model to interpret instructions and translate them into tool calls. Variations in this interpretation can make an intended strategy difficult to reproduce consistently. This leads to our second challenge: *How can a video agent evolution framework be designed to evolve overall execution strategies rather than rely solely on textual skill optimization?*

To address these challenges, we introduce **VideoEvoHarness**, a framework in which a Questioner and a Solver evolve together from unlabeled video, as shown in Figure 1(c). The Questioner generates supervision for improvement, while the Solver modifies its harness under that supervision. Task construction thus adapts alongside task execution, allowing unlabeled videos to supply both evaluation tasks and directions for improvement. The underlying model parameters remain fixed throughout the process.

For the first challenge, **Questioner evolution** adapts the supervision constructed from structured visual memory. The Questioner generates questions about events and entities recorded in local video

segments and computes reference answers through operations on the memory. The associated video segments also provide a basis for analyzing the causes of Solver failures. We assess five aspects of question quality: answer accuracy without video evidence, answer accuracy with supporting memory, question type coverage, difficulty for the current Solver, and the proportion of questions removed by quality filtering. Solver execution trajectories provide additional feedback. The model uses this information to revise the question generation policy, allowing the supervision process to evolve with experience from successive task executions.

For the second challenge, **Solver harness evolution** supports updates to textual skills, executable strategy code, hyperparameters, and tools. Textual skills refine reasoning instructions, while strategy code specifies how evidence is retrieved and tools are composed. Writing these procedures as code makes their control flow explicit and reduces reliance on repeated model interpretation of natural language instructions. Hyperparameter updates adapt observation and evidence allocation, while the tool operation strategy expands the set of available operations or modifies existing tool code. The Solver applies a sampling strategy to short videos and uses memory to locate relevant frames in long videos, without being given the Questioner's source event. The model then uses execution feedback to propose changes and evaluates the revised harness on subsequent tasks. Validated improvements are retained for later iterations. Together, these mechanisms allow the agent to revise both what it can do and how it organizes those capabilities.

Our contributions are summarized as follows:

- We introduce **VideoEvoHarness**, a framework for evolving video agent harnesses from unlabeled videos without external question and answer annotations or updates to model parameters.
- We develop a feedback mechanism for **Questioner evolution**. Question quality assessment and Solver execution trajectories inform revisions to the generation policy, adapting the supervision derived from unlabeled video.
- We develop a mechanism supporting **Solver harness evolution** across textual skills, executable strategy code, hyperparameters, and tools. Candidate updates target reasoning and execution procedures, with executable strategies specifying how frames are sampled, memory is retrieved, and existing tools are used.
- Evaluations on standard video understanding benchmarks and a dedicated benchmark question set, using videos disjoint from evolution videos, show higher overall accuracy than the initial harness, progressive gains on the dedicated set across rounds, and transfer gains across backbones, supporting harness evolution.

2 Related Work

Video Understanding Agents. Video understanding agents extend Video-LLMs with active observation, structured memory, and role coordination. Observation-planning agents decide what to inspect: VideoAgent uses an LLM controller to iteratively identify and compile question-relevant information, with vision-language models serving as visual retrieval and perception tools (Wang et al., 2024). VideoZoomer learns coarse-to-fine temporal focusing, TimeSearch-R interleaves adaptive search with completeness self-verification, and LensWalk lets the reasoner select each observation's temporal range and sampling density (Ding et al., 2026; Pan et al., 2026; Li et al., 2026). Memory-structured agents organize video content for targeted retrieval. Fan et al. store event descriptions and object-centric tracks in memory accessed through segment-localization and object-query tools, while VideoTree and HOMER construct query-adaptive or hierarchical representations for multi-stage retrieval (Fan et al., 2025; Wang et al., 2025; Ji et al., 2026). Role-coordination agents modularize inference; VideoMind orchestrates planner, grounder, verifier, and answerer roles (Liu et al., 2026). Although these methods adapt evidence acquisition to each query, their prompts, tools, strategies, and hyperparameters generally remain fixed after design or training. **VideoEvoHarness** instead makes the video agent harness evolvable.

Self Evolving Agents. Self evolving agents adapt tasks, instructions, or executable systems from interaction feedback (Agrawal et al., 2026; Hu et al., 2025; Fernando et al., 2024). WebRL evolves curricula from failed web trajectories (Qi et al., 2025); GEPA converts trajectory reflection into

prompt updates, while automated agent design searches over executable agent programs (Agrawal et al., 2026; Hu et al., 2025; Fernando et al., 2024). Vision-Zero extends label-free multi-agent self-play to visual tasks (Wang et al., 2026). In video, data- and model-evolution methods improve training supervision: Video-STaR cycles between instruction generation and fine-tuning using source labels, while ReWatch-R1 synthesizes video-grounded reasoning traces for post-training (Zohar et al., 2025; Zhang et al., 2026a). Video-Zero and EvoVid instead coevolve Questioner and Solver policies from raw videos without human annotations, but primarily update Video-LLM parameters rather than the runtime harness (Zhang et al., 2026c; Huang et al., 2026). Harness-oriented methods are closer to our setting: VisualClaw accumulates deployment failures into memory and skill banks, whereas MetaVideoAgent evolves modular agent code but requires supplied questions, gold answers, and annotated evidence intervals for failure diagnosis (Tu et al., 2026; Cui et al., 2026). **VideoEvoHarness** instead enables task construction and the execution harness to coevolve from unlabeled videos without human-authored questions, answers, or evidence annotations, while keeping model parameters frozen.

3 METHOD

We introduce **VideoEvoHarness**, a framework for evolving video agents through tasks derived from unlabeled video. Its central idea is to improve both the questions that reveal an agent's limitations and the harness that determines how it addresses them. A Questioner refines task construction from video observations, while a Solver evolves its evidence acquisition and reasoning procedures. We first describe their interaction, then present the evolution mechanism for each component.

3.1 FRAMEWORK OVERVIEW

Given unlabeled videos $\mathcal{V}$ and a pretrained model f_θ, we seek to improve video understanding by evolving the surrounding harness while keeping θ fixed. The Questioner constructs evaluation tasks from the videos, and the Solver tests its harness on these tasks. Their results guide evolution. Figure 2 summarizes this cycle, from memory construction and question generation to answering, diagnosis, and harness updates (Algorithm 1). We represent the system at iteration t as

$$\Phi_t = (\phi_t, H_t), \qquad H_t = (P_t, C_t, \lambda_t, \mathcal{T}_t), \tag{1}$$

where ϕ_t specifies the Questioner's generation policy and H_t denotes the Solver harness. Its components are textual skills and prompts P_t, executable strategies C_t, hyperparameters λ_t, and a tool library $\mathcal{T}_t$. Together, they determine what evidence the Solver observes and how it uses that evidence to answer a question.

We build on Qwen-MM-Plugins (Qwen Team, 2026b) for video reading and memory construction (Appendix A.1). The Questioner derives questions, reference answers, and source segments from hierarchical memory containing events, entities, and visible text. The Solver uses two evidence acquisition paths according to video duration. For short videos, it uses a single overview or chunk-wise coarse sampling followed by targeted detail inspection. Long videos are searched through memory to locate relevant segments before inspecting their original frames. Hyperparameters control temporal windows, sampling density, and resolution, as detailed in Appendix A.5. The reference answer and designated source segment are withheld from the Solver. Question evaluation and Solver execution trajectories inform updates to task generation and the Solver harness.

3.2 QUESTIONER EVOLUTION

Question generation. The central challenge in learning from unlabeled video is to derive an evaluation signal from observations alone. We address this by generating questions whose reference answers can be computed from structured video records. The visual memory associates event summaries, entities, actions, and visible text with temporal intervals. The Questioner selects a source segment and constructs a question supported by its records. Temporal questions use event timestamps. Counting questions derive reference answers by counting entities of a given type in the memory of the selected segment. Text questions retrieve recorded visual content. The resulting reference is derived from the memory rather than inferred from the Solver's response. Each question also retains its source segment, providing a reference for analyzing evidence retrieval. A video

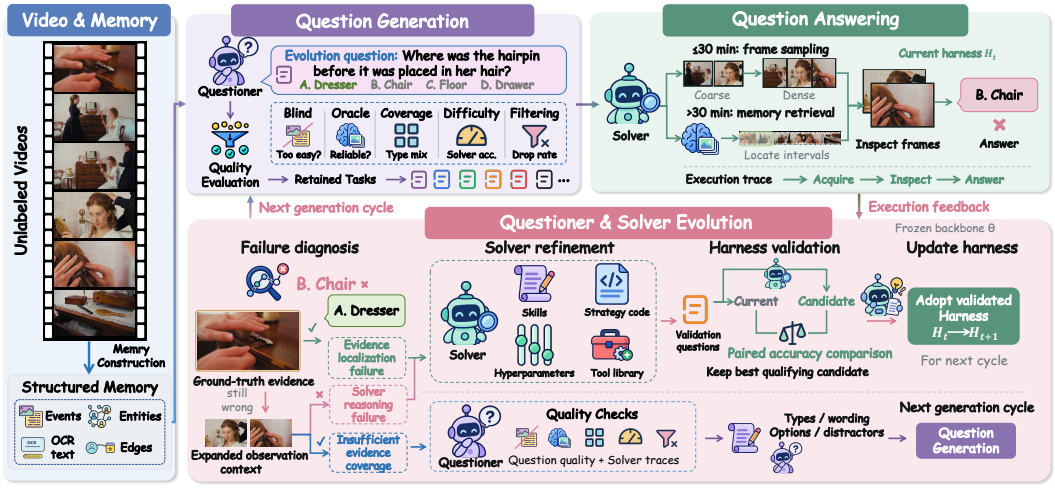


Figure 2: Overview of the VideoEvoHarness framework. The Questioner constructs and filters tasks from structured video memory, while the Solver acquires visual evidence through frame sampling or memory retrieval. Execution feedback supports revisions to the question generation policy and the Solver's skills, strategy code, hyperparameters, and tool operations. Validated updates enable both agents to evolve from unlabeled videos while keeping model parameters fixed.

therefore supplies both an answer evaluation signal and a basis for diagnosing how the answer was obtained.

Question quality evaluation. The questions generated from visual memory should reveal gaps in the Solver's video understanding. However, some can be answered from their wording or answer options without consulting the video. We measure accuracy without video evidence using a Blind probe before filtering and exclude questions it answers correctly from the Solver's task set. An Oracle probe measures accuracy with supporting memory; questions it answers incorrectly are also excluded. We assess question type coverage against target proportions specified in the question generation configuration. We estimate difficulty from Solver accuracy within each question type and favor types with intermediate accuracy. Finally, we track the proportion of generated questions removed by filtering, since excessive filtering reduces the number available for evolution. We combine these five criteria into a question quality score:

$$\mathcal{J}_Q = -w_{\mathrm{b}} L_{\mathrm{b}} + w_{\mathrm{o}} A_{\mathrm{o}} - w_{\mathrm{c}} C + w_{\mathrm{d}} D - w_{\mathrm{r}} R, \tag{2}$$

where L_{b} is normalized accuracy above chance without video evidence, and A_{o} is the fraction of questions answered correctly by the Oracle with supporting memory. C is the distance between the retained and target question type distributions. D is the fraction of retained questions whose type has intermediate Solver accuracy. R is the discarded fraction above the level expected from random Blind guesses. The weights balance these criteria. Detailed definitions are given in Appendix A.2.

Generation policy refinement. We provide the model with the question evaluation results, Solver execution trajectories, and the current generation policy. The model identifies weaknesses in question generation and proposes revisions to ϕ_t. These revisions act on question types, question forms, answer options, and distractor selection. For example, frequent success without visual evidence motivates replacing descriptive cues with relations between observations or drawing distractors from similar temporal contexts. Solver performance provides additional context about which task types are already mastered and which remain difficult. The revised policy changes how subsequent questions are constructed. This makes question generation an adaptive source of supervision rather than a fixed preprocessing step.

3.3 Video Agent Harness Evolution

Failure analysis. Questioner evolution revises how tasks are constructed, while Solver harness evolution revises how they are solved. The Solver obtains evidence by sampling frames under its

strategy or by retrieving memory. A wrong answer alone does not indicate whether the relevant evidence was missed or incorrectly used. We compare evidence conditions as appropriate for each path, using source segments retained during question construction. We preserve each path's evidence representation during these comparisons. A correct answer after supplying source evidence suggests an evidence localization failure by the Solver. A correct answer only after expanding evidence coverage suggests insufficient evidence coverage for the Questioner's question and reference answer. If the Solver still answers incorrectly under both conditions, we examine possible errors in its reasoning. We combine these comparisons with execution trajectories to identify limitations in evidence acquisition and reasoning.

Harness refinement. Failure analysis supports updates to any Solver harness component. Textual skills in P_t can shape task interpretation and reasoning through reusable prompts, such as instructions to verify entity counts against relevant frames. For executable strategies in C_t, the model can write and revise code for frame sampling, evidence retrieval, tool composition, and answer formation. Hyperparameter updates in λ_t can control retrieval breadth, sampling density, resolution, and evidence allocation. Tool updates can extend $\mathcal{T}_t$ with reusable operations or revise existing code. New or modified tools must pass verification and candidate harness validation before being retained.

Given the current harness and evaluation feedback, the model proposes changes to these components, including revised strategy and tool code. We apply the updated harness to subsequent tasks and use the evaluation results to refine it further. Combining these four forms of adaptation enables the Solver to adjust both the operations available for reasoning and the procedures that organize their use.

Improvement validation. The Solver tests each candidate harness on a shared validation subset of the Questioner's questions. We retain the candidate with the greatest validated improvement in accuracy. If no candidate is accepted, we preserve the current harness. Details are provided in Appendix A.3. Questions and reference answers remain fixed during this comparison. Questioner revisions take effect in a later generation cycle, separating changes in task difficulty from changes in Solver capability. Accepted skills, strategies, parameter settings, and tools are incorporated into the harness for subsequent iterations. Feedback from video tasks thus accumulates as reusable reasoning and execution procedures.

4 Experiments

4.1 Experimental Setup

Datasets and metrics. We evolve the harness on a pool of 130 videos sampled from six sources: the MLVU development split (Zhou et al., 2025), long videos from Vript, VideoMarathon (Lin et al., 2026), long TVQA episodes (Lei et al., 2018), ScaleLong (Ma et al., 2026), and the Video-Holmes training split (Cheng et al., 2025). We use only the raw videos and discard all accompanying questions, answers, and temporal annotations. Source counts and the video duration distribution are provided in Appendix A.8. These videos are disjoint from the evaluation videos, which we verify by video identifier and duration. We evaluate on three benchmarks: LongVideoBench (Wu et al., 2024), MMVU (Zhao et al., 2025), and Video-Holmes (Cheng et al., 2025). All evaluations use video frames without subtitles and report multiple-choice accuracy; for MMVU, we use its multiple-choice subset. Table 1 reports LongVideoBench results by question category under L1-Perception and L2-Relation, together with overall accuracy. Table 2 reports MMVU results across its four official disciplines and overall, alongside overall accuracy on Video-Holmes. Self generated questions support evolution and question quality analysis. Benchmark evaluations are conducted independently and do not guide evolution.

Baselines. Our primary comparison is the same frozen model under the initial harness before evolution, which separates the effect of evolution from the capability of the model. This harness follows the recommended configuration of the underlying toolkit (Qwen Team, 2026b) rather than a weakened setting. We compare Initial harness with VideoEvoHarness after the first and third evolution rounds, denoted R1 and R3, respectively. We evaluate two backbones: Qwen3.7 Plus (Qwen Team, 2026a) and Gemini 3.8 Flash (Doshi & Popa, 2026).

Table 1: Per-category results on LongVideoBench with frozen backbones. Question categories follow the official taxonomy and are grouped into L1-Perception and L2-Relation. Avg. reports accuracy over all questions within each group, and Overall reports accuracy over all questions. Scores are accuracy in percent, and the best result for each backbone is in bold.

Method	L1-Perception									L2-Relation										Overall
	S2E	S2O	S2A	E2O	O2E	T2E	T2O	T2A	**Avg.**	E3E	O3O	SSS	SOS	SAA	T3E	T3O	TOS	TAA	**Avg.**	
	Qwen3.7 Plus																			
Initial harness	69.9	**72.2**	85.1	75.4	65.5	64.5	72.0	65.3	71.4	74.7	74.2	55.8	72.8	59.7	52.1	59.5	31.9	53.7	59.6	65.1
VideoEvoHarness (R1)	72.0	70.8	86.2	73.8	**70.1**	62.9	**78.7**	64.0	72.7	75.8	77.3	**64.2**	75.3	58.3	**60.3**	**63.5**	31.9	43.9	61.5	66.7
VideoEvoHarness (R3)	**73.1**	66.7	**87.4**	**80.0**	65.5	**67.7**	76.0	**68.0**	**73.2**	**78.0**	**80.3**	61.1	**82.7**	**65.3**	53.4	54.1	**34.7**	**56.1**	**63.2**	**67.9**
	Gemini 3.8 Flash																			
Initial harness	83.9	**77.8**	**94.3**	**83.1**	**80.5**	75.8	74.7	**78.7**	81.5	79.1	72.7	63.2	**82.7**	75.0	**67.1**	64.9	47.2	**67.1**	69.0	74.8
VideoEvoHarness (R1)	83.9	**77.8**	**94.3**	**83.1**	79.3	**79.0**	80.0	74.7	**81.8**	**82.4**	77.3	**69.5**	79.0	73.6	65.8	67.6	**54.2**	**67.1**	**71.0**	**76.0**
VideoEvoHarness (R3)	**86.0**	75.0	93.1	**83.1**	79.3	74.2	**81.3**	74.7	81.3	**82.4**	**78.8**	65.3	76.5	**79.2**	61.6	**71.6**	48.6	**67.1**	70.3	75.4

Implementation details. We build the harness on Qwen-MM-Plugins (Qwen Team, 2026b) and keep all model parameters frozen. Videos of at least 30 minutes are searched through memory to locate relevant segments before their frames are inspected. Videos under 30 minutes use a single overview or segment-wise coarse sampling with targeted detail inspection. Evolution proceeds for three rounds. Each round draws 30 videos from this pool and generates about 600 usable questions: 200 for evolution and 400 for validating whether proposed modifications should be accepted. The initial and evolved harnesses start from the same initial observation budget. Remaining configurations are given in Appendix A.

4.2 Main Results

Table 2: Results on MMVU and Video-Holmes.

Method	MMVU					V-Holmes
	Sci.	Health.	Hum.	Eng.	**Overall**	
	Qwen3.7 Plus					
Initial harness	76.5	79.3	**87.0**	80.9	80.0	56.5
VideoEvoHarness (R1)	78.4	82.6	85.9	81.9	81.4	57.1
VideoEvoHarness (R3)	**79.8**	**85.1**	83.7	**82.9**	**82.4**	**58.0**
	Gemini 3.8 Flash					
Initial harness	83.6	**90.1**	87.0	85.4	85.9	65.3
VideoEvoHarness (R1)	**85.9**	87.6	89.1	**87.4**	87.2	65.5
VideoEvoHarness (R3)	85.4	89.3	**90.2**	86.9	**87.4**	**65.8**

Evolution from unlabeled videos. Tables 1 and 2 report results on three benchmarks with two frozen backbones. VideoEvoHarness (R3) improves overall accuracy over the initial harness on every benchmark: Qwen3.7 Plus gains 2.8, 2.4, and 1.5 points on LongVideoBench, MMVU, and Video-Holmes, respectively; Gemini 3.8 Flash gains 0.6, 1.5, and 0.5 points. On LongVideoBench, Qwen's L2-Relation score improves by 3.6 points, compared with 1.8 for L1-Perception. Gemini similarly gains 1.3 points on L2 despite a 0.2-point decline on L1, suggesting larger benefits for relational questions. Table 2 groups MMVU into Science (Sci.), Healthcare (Health.), Humanities and Social Sciences (Hum.), and Engineering (Eng.), and reports Video-Holmes (V-Holmes) overall. MMVU also exhibits discipline-specific gains: Qwen improves most in Healthcare (5.8 points), while Gemini benefits most in Humanities and Social Sciences (3.2 points). These gains are obtained using tasks generated from unlabeled videos, without updating model parameters.

Development across evolution rounds. The first round already improves overall accuracy on all three benchmarks for both backbones. Qwen3.7 Plus achieves higher overall accuracy at R3 than at R1, with LongVideoBench rising from 66.7 to 67.9, MMVU from 81.4 to 82.4, and Video-Holmes from 57.1 to 58.0. This indicates additional gains beyond the first update. Gemini 3.8 Flash shows some fluctuation: LongVideoBench drops from 76.0 at R1 to 75.4 at R3, but remains above its initial 74.8. Meanwhile, MMVU and Video-Holmes improve from 87.2 to 87.4 and 65.5 to 65.8, respectively. Thus, although later rounds do not improve every benchmark monotonically, R3 retains gains over the initial harness on all three. Because the evolution videos are disjoint from all evaluation videos, these results support transfer of the evolved harness beyond its self generated tasks.

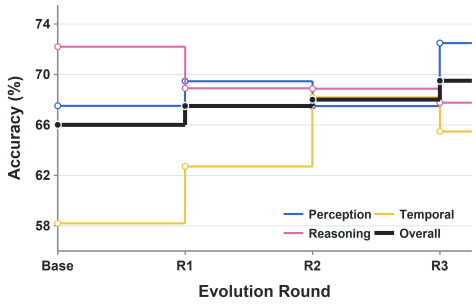


Figure 3: Accuracy on the dedicated benchmark question set across evolution rounds. Base denotes the initial harness.

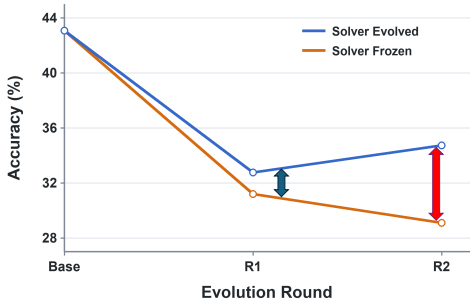


Figure 4: Accuracy on generated questions across evolution rounds for the evolving and initial Solver harnesses.

4.3 Evolution Effectiveness

Performance across evolution rounds. For a balanced assessment, we evaluate the initial harness (Base) and each round's saved harness with Qwen3.7 Plus on the same dedicated benchmark question set of 400 questions, grouped into perception, temporal grounding, and reasoning (Appendix A.7). These questions do not guide evolution. Figure 3 shows overall accuracy increasing across evolution rounds, indicating that later updates provide benefits beyond the first round. Perception improves overall despite intermediate fluctuations, while temporal grounding rises substantially before declining in the final round, remaining above its initial level. These trends show that the evolved harness improves both visual perception and temporal grounding, although progress within each group is not monotonic. In contrast, reasoning declines, mainly driven by lower performance on Topic and knowledge questions. This suggests that current evolution primarily improves fine-grained evidence localization. This provides a direction for subsequent evolution: strengthening global information integration and summarization while preserving these gains.

Question difficulty and Solver adaptation. Using Qwen3.7 Plus, we compare the evolving Solver harness with its initial version on questions generated by the evolving Questioner at each round. The Questioner is evolved in the first two of the three evolution rounds. Both Solver versions answer the same questions within a round and are scored against identical references. Figure 4 shows both declining from Base to R1; at R2, the evolving Solver recovers while the frozen Solver continues to decline. The continued decline of the unchanged Solver harness indicates that later question sets are more challenging, with a growing proportion of questions probing unresolved capability gaps in the initial harness. Meanwhile, the evolving Solver outperforms the frozen version at both R1 and R2, with a larger advantage at R2. Despite fluctuations in absolute accuracy, this widening gap indicates that the evolved Solver adapts more effectively to the changing question distribution. Together with the gains on the dedicated set in Figure 3, these results support the effectiveness of coevolving task generation and execution.

4.4 Ablation Studies

On the same dedicated benchmark question set, we examine cross-backbone harness transfer, filtering and validation, and Solver component evolution. Perc., Temp., and Reas. report accuracy for perception, temporal grounding, and reasoning and synthesis, respectively; All reports overall accuracy. Scores are percentages. Unless otherwise stated, we use Qwen3.7 Plus.

Transfer across backbones. Table 3 evaluates the transfer of the harness evolved for three rounds with Gemini 3.8 Flash to Qwen3.7 Plus. Initial uses Qwen's original harness; Transferred reuses Gemini's R3 harness without further evolution; Own evolved uses the harness evolved with Qwen itself. Transferred raises overall accuracy from 66.0 to 67.8, showing that an evolved harness can benefit another backbone without target-specific evolution. Temporal grounding gains 6.3 points, while reasoning drops by 2.2 points, indicating that transfer benefits differ across capabilities. Own evolved further improves overall accuracy to 69.5, exceeding Transferred by 1.7 points. Thus, direct

Table 3: Transfer of the R3 harness from Gemini 3.8 Flash to Qwen3.7 Plus.

Harness	Perc.	Temp.	Reas.	All
Initial	67.5	58.2	**72.2**	66.0
Transferred	68.5	64.5	70.0	67.8
Own evolved	**72.5**	**65.5**	67.8	**69.5**

Table 4: Ablation of question filtering and candidate validation over three evolution rounds.

Variant	Perc.	Temp.	Reas.	All	N
Full framework	**72.5**	**65.5**	67.8	**69.5**	3
No validation	68.0	61.8	68.9	66.5	**39**
No filtering	66.0	64.5	**71.1**	66.8	1

reuse provides cross-backbone gains, while evolving the harness with the target model itself yields the best overall performance.

Effectiveness of filtering and validation. Table 4 examines question filtering and candidate validation. N counts harness updates retained over three evolution rounds. Full framework retains three updates and achieves the highest overall accuracy (69.5). No validation retains all 39 proposed updates without checking their benefit, yet reaches only 66.5 overall accuracy, with temporal grounding falling from 65.5 to 61.8. This contrast shows that update quantity alone does not ensure progress; update quality and effectiveness matter more. Without question filtering, low-quality tasks weaken the feedback used to propose changes, and most candidates fail to improve validation accuracy. Only one strategy update is retained, yielding 66.8 overall accuracy. These results highlight complementary roles: filtering provides informative tasks to guide useful proposals, while validation selects updates with measured gains. Together, they support more effective evolution within the three-round budget.

Effectiveness of component evolution. Table 5 evaluates strategy and hyperparameter evolution. Initial harness denotes the reference configuration, and Full harness corresponds to Own evolved in Table 3. Strategy and Hyperparams. retain only the corresponding component changes. Skills and tools remain unchanged within the current three-round budget and are therefore omitted. Relative to Initial harness, Strategy improves perception and temporal grounding by 0.5 and 4.5 points, respectively, highlighting the effectiveness of execution logic for coordinating temporal evidence. Hyperparams. gains 2.0 and 1.8 points in these categories, demonstrating the contribution of sampling and retrieval adjustments to visual evidence acquisition. Combining both updates yields complementary benefits: Full harness reaches 69.5 overall accuracy, exceeding Strategy and Hyperparams. by 3.0 and 3.2 points, respectively. It gains 3.5 points overall over Initial harness, with improvements of 5.0 in perception and 7.3 in temporal grounding, while Initial harness retains higher reasoning accuracy (72.2 versus 67.8). The combined configuration thus achieves the strongest overall performance and the highest perception and temporal grounding scores, bringing together the benefits of execution logic and evidence allocation. These results demonstrate the effectiveness of component evolution in improving Solver performance.

Table 5: Ablation of Solver evolution. Column maxima are in bold.

Harness	Perc.	Temp.	Reas.	All
Full harness	**72.5**	**65.5**	67.8	**69.5**
Strategy	68.0	62.7	67.8	66.5
Hyperparams.	69.5	60.0	66.7	66.3
Initial harness	67.5	58.2	**72.2**	66.0

5 Conclusion

VideoEvoHarness coevolves a Questioner and a Solver harness from unlabeled videos with frozen model parameters. The Questioner generates tasks from video memory and refines its generation policy using question quality assessments and Solver execution trajectories. Validated execution feedback supports updates to the Solver's textual skills, executable strategies, hyperparameters, and tools. Executable strategies encode reusable workflows, reducing repeated interpretation of natural language instructions. Within each round, candidates are compared on identical validation questions, with accepted improvements retained for subsequent tasks. Experiments on three benchmarks with two backbones and on a dedicated benchmark question set show higher overall accuracy than the initial harness and transfer gains across backbones. Overall accuracy on this set also improves progressively across rounds. These results highlight harness evolution from unlabeled videos as a promising direction for adaptable video understanding agents.

AI USE STATEMENT

Generative models are integral to the method studied in this work. They construct video memory, generate synthetic questions, and propose Questioner policies and Solver harness updates, including strategy code and hyperparameter changes, as described in Section 3 and Appendix B. Reference answers for the generated questions are computed programmatically from the memory records.

Separately, we used the CodeBuddy agent framework with Claude-Opus-5 and GPT-6-Astra for manuscript language polishing, translating research literature, generating icons used in figures, organizing and formatting implementation code, and standardizing names without changing method logic. Beyond the model-driven procedures described in the method, we did not use these assistants to develop theoretical models, conceptual frameworks, or hypotheses; design research methodology or experiments; formulate mathematical claims or proofs; clean or reformat datasets; interpret or present experimental results; or perform qualitative or thematic data analysis.

The authors manually reviewed the relevant text, references, code, experimental results, generated icons, and final figures on an item-by-item basis. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

To support reproducibility, we describe the Questioner and Solver update procedures in Section 3 and provide the overall algorithm in Appendix B. Appendix A documents memory construction, question quality assessment, candidate validation, tool verification, and evidence acquisition, with key hyperparameters summarized in Table 6. Section 4.1 specifies the model backbones, benchmarks, and evaluation protocol. Appendices A.7 and A.8 report the composition of the fixed analysis set and the sources and duration distribution of the evolution videos. Appendix C.3 further illustrates strategy code and configuration changes during evolution. The code is provided in the supplementary material.

REFERENCES

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 8479–8565, 2026.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. In *International Conference on Learning Representations*, volume 2024, pp. 54067–54089, 2024.

Junhao Cheng, Yuying Ge, Teng Wang, Yixiao Ge, Jing Liao, and Ying Shan. Video-Holmes: Can MLLM think like holmes for complex video reasoning? *arXiv preprint arXiv:2505.21374*, 2025.

Benlei Cui, Ruize Wang, Junjie Li, Jinhao Chen, Longtao Huang, Yinghao Chen, Yuwen Zhai, Jingqun Tang, Ruijian Jia, Weiwei Wu, Pengfei Sun, and Haiwen Hong. MetaVideoAgent: Automated video-agent evolution for long-form video understanding. *arXiv preprint arXiv:2608.04587*, 2026.

Anxhelo Diko, Tinghuai Wang, Wassim Swaileh, Shiyan Sun, and Ioannis Patras. ReWind: Understanding long videos with instructed learnable memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13734–13743, 2025.

Yang Ding, Xin Lai, Yizhen Zhang, Wei Li, Ruihang Chu, and Yujiu Yang. VideoZoomer: Reinforcement-learned temporal focusing for long video reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 20087–20111, 2026.

Tulsee Doshi and Raluca Ada Popa. Introducing Gemini 3.8 Flash and 3.8 Flash Cyber, September 2026. URL `https://blog.google/innovation-and-ai/models-and-research/gemini-models/3-8-flash-and-3-8-flash-cyber/`.

Yue Fan, Xiaojian Ma, Rujie Wu, Yuntao Du, Jiaqi Li, Zhi Gao, and Qing Li. VideoAgent: A memory-augmented multimodal agent for video understanding. In *European Conference on Computer Vision*, pp. 75–92. Springer, 2025.

Chrisantha Fernando, Dylan Sunil Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 13481–13544. PMLR, 2024.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, Peixian Chen, Yanwei Li, Shaohui Lin, Sirui Zhao, Ke Li, Tong Xu, Xiawu Zheng, Enhong Chen, Caifeng Shan, Ran He, and Xing Sun. Video-MME: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24108–24118, 2025.

Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. MA-LMM: Memory-augmented large multimodal model for long-term video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13504–13514, 2024.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Shiqi Huang, Ziyue Wang, Zhongrong Zuo, Han Qiu, Qi She, and Bihan Wen. EvoVid: Temporal-centric self-evolution for video large language models. *arXiv preprint arXiv:2605.21931*, 2026.

Yixin Ji, Fanghua Ye, Juntao Li, Bo Zhao, Zexuan Qiu, Zhaopeng Tu, Liefeng Bo, and Min Zhang. Homer: Understanding long-form videos with hierarchical memory and agentic reasoning. *arXiv preprint arXiv:2607.02588*, 2026.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling declarative language model calls into self-improving pipelines. In *International Conference on Learning Representations*, volume 2024, pp. 54928–54958, 2024.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L. Berg. TVQA: Localized, compositional video question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1369–1379, 2018.

Keliang Li, Yansong Li, Hongze Shen, Mengdi Liu, Hong Chang, and Shiguang Shan. LensWalk: Agentic video understanding by planning how you see in videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19518–19528, 2026.

Jingyang Lin, Jialian Wu, Ximeng Sun, Ze Wang, Jiang Liu, Yusheng Su, Xiaodong Yu, Hao Chen, Jiebo Luo, Zicheng Liu, and Emad Barsoum. Unleashing hour-scale video training for long video-language understanding. *Advances in Neural Information Processing Systems*, 38:17523–17552, 2026.

Ye Liu, Kevin Qinghong Lin, Chang-Wen Chen, and Mike Zheng Shou. VideoMind: A chain-of-LoRA agent for temporal-grounded video reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 57481–57506, 2026.

Lin Long, Yichen He, Wentao Ye, Yiyuan Pan, Yuan Lin, Hang Li, Junbo Zhao, and Wei Li. Seeing, listening, remembering, and reasoning: A multimodal agent with long-term memory. In *International Conference on Learning Representations*, volume 2026, pp. 146197–146246, 2026.

David Ma, Huaqing Yuan, Xingjian Wang, Qianbo Zang, Tianci Liu, Xinyang He, Yanbin Wei, Jiawei Guo, Jiahui Ni, Zhenzhu Yang, Meng Cao, Shanghaoran Quan, Yizhi Li, Wangchunshu Zhou, Jiaheng Liu, Wenhao Huang, Ge Zhang, Shiwen Ni, and Xiaojie Jin. ScaleLong: A multi-timescale benchmark for long video understanding. In *International Conference on Learning Representations*, volume 2026, pp. 154328–154347, 2026.

Ziyu Ma, Chenhui Gou, Hengcan Shi, Bin Sun, Shutao Li, Hamid Rezatofighi, and Jianfei Cai. DrVideo: Document retrieval based long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18936–18946, 2025.

Junwen Pan, Qizhe Zhang, Rui Zhang, Ming Lu, Xin Wan, Yuan Zhang, Chang Liu, and Qi She. TimeSearch-R: Adaptive temporal search for long-form video understanding via self-verification reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 8376–8405, 2026.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, Jie Tang, and Yuxiao Dong. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 79791–79821, 2025.

Qwen Team. Qwen3.7-Plus: Multimodal agent intelligence, May 2026a. URL `https://qwen.ai/blog?id=qwen3.7-plus`.

Qwen Team. Qwen-MM-Plugins: Make any agent harness multimodal-native, 2026b. URL `https://github.com/QwenLM/Qwen-MM-Plugins`.

Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. TimeChat: A time-sensitive multimodal large language model for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14313–14323, 2024.

Yan Shu, Zheng Liu, Peitian Zhang, Minghao Qin, Junjie Zhou, Zhengyang Liang, Tiejun Huang, and Bo Zhao. Video-XL: Extra-long vision language model for hour-scale video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26160–26169, 2025.

Haoqin Tu, Jianwen Chen, Zijun Wang, Siwei Han, Juncheng Wu, Hardy Chen, Haonian Ji, Kaiwen Xiong, Jiaqi Liu, Peng Xia, Jieru Mei, Hongliang Fei, Jason Eshraghian, Zeyu Zheng, Yuyin Zhou, Huaxiu Yao, and Cihang Xie. VisualClaw: A real-time, personalized agent for the physical world. *arXiv preprint arXiv:2606.16295*, 2026.

Qinsi Wang, Bo Liu, Tianyi Zhou, Jing Shi, Yueqian Lin, Yiran Chen, Hai Li, Kun Wan, and Wentian Zhao. Vision-Zero: Scalable VLM self-evolution via multi-agent self-play. In *International Conference on Learning Representations*, volume 2026, pp. 106716–106744, 2026.

Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. VideoAgent: Long-form video understanding with large language model as agent. In *European Conference on Computer Vision*, pp. 58–76. Springer, 2024.

Ziyang Wang, Shoubin Yu, Elias Stengel-Eskin, Jaehong Yoon, Feng Cheng, Gedas Bertasius, and Mohit Bansal. VideoTree: Adaptive tree-based video representation for LLM reasoning on long videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3272–3283, 2025.

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.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Congzhi Zhang, Zhibin Wang, Yinchao Ma, Jiawei Peng, Yihan Wang, Qiang Zhou, Jun Song, and Bo Zheng. ReWatch-R1: Boosting complex video reasoning in large vision-language models through agentic data synthesis. In *International Conference on Learning Representations*, volume 2026, pp. 140958–140985, 2026a.

Hangfan Zhang, Shao Zhang, Kangcong Li, Chen Zhang, Yang Chen, Yiqun Zhang, Lei Bai, and Shuyue Hu. Self-Harness: Harnesses that improve themselves. *arXiv preprint arXiv:2606.09498*, 2026b.

Ruixu Zhang, Deyi Ji, Lanyun Zhu, Xuanyi Liu, Yuxin Meng, Ruihang Chu, and Yujiu Yang. Video-Zero: Self-evolution video understanding. *arXiv preprint arXiv:2605.14733*, 2026c.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Yilun Zhao, Lujing Xie, Haowei Zhang, Guo Gan, Yitao Long, Zhiyuan Hu, Tongyan Hu, Weiyuan Chen, Chuhan Li, Junyang Song, Zhijian Xu, Chengye Wang, Weifeng Pan, Ziyao Shangguan, Xiangru Tang, Zhenwen Liang, Yixin Liu, Chen Zhao, and Arman Cohan. MMVU: Measuring expert-level multi-discipline video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8475–8489, 2025.

Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Zhengyang Liang, Shitao Xiao, Minghao Qin, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. MLVU: Benchmarking multi-task long video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 13691–13701, 2025.

Orr Zohar, Xiaohan Wang, Yonatan Bitton, Idan Szpektor, and Serena Yeung-Levy. Video-STaR: Self-training enables video instruction tuning with any supervision. In *International Conference on Learning Representations*, volume 2025, pp. 92702–92716, 2025.

APPENDIX

A IMPLEMENTATION DETAILS

A.1 VIDEO MEMORY CONSTRUCTION

We construct video memory using the hierarchical graph pipeline provided by Qwen-MM-Plugins (Qwen Team, 2026b). First, scene boundaries are detected from frame differences in hue, lightness, and saturation, using frames sampled at 0.25 fps. Short segments are merged and long segments are split using minimum and maximum duration settings of 30 and 300 seconds. These memory segments are distinct from the fixed 300-second chunks used for Solver observation in Appendix A.5. In the frame-input construction path, each segment is sampled at 0.5 fps with a cap of 250 frames before being sent to the visual model. These construction settings are separate from the Solver's frame inspection budget.

For each segment, the model produces a local graph containing events, entities, visible text, and their relations. Event records include actions and time intervals; entity records include types, attributes, and descriptions; visible-text records retain the text and its time interval. Segment-relative timestamps are converted to positions in the full video, and each record retains its parent segment identifier. A text-based aggregation stage groups segments into higher-level events and constructs a video-level description. Node descriptions are embedded using Qwen3-Embedding-8B (Zhang et al., 2025) for semantic retrieval, while the graph retains identifiers and temporal links for direct lookup.

The saved graph and retrieval index are reused across questions. The Questioner computes references by accessing the recorded fields, counts, and timestamps, and retains the corresponding source segments for diagnosis. The long-video Solver instead retrieves memory using the question and inspects the associated video intervals without being told which segments were used to construct the question. Memory construction is a toolkit component on which our evolution framework builds, rather than a new representation learned by the evolution loop. Table 6 summarizes the construction defaults and the initial harness settings.

A.2 QUESTION QUALITY ESTIMATION

Let $\mathcal{G}$ denote the generated questions before filtering and $\mathcal{D}$ the retained subset. Blind accuracy b is measured on $\mathcal{G}$. For questions with K answer options, the excess Blind accuracy and excess rejection rate used in Equation 2 are

$$L_{\mathrm{b}} = \max\left\{0, \frac{b - 1/K}{1 - 1/K}\right\}, \qquad R = \max\left\{0, 1 - \frac{|\mathcal{D}|}{|\mathcal{G}|} - \frac{1}{K}\right\}. \tag{3}$$

Both terms account for correct Blind guesses at chance. Retained questions have failed the Blind probe, so their Blind accuracy is not used to estimate generation quality. Oracle accuracy A_{o} is the fraction of evaluated questions answered correctly with supporting memory. Questions answered incorrectly by this probe are excluded from the Solver's task set.

We use the total variation distance between the retained question type distribution and the target proportions as the coverage penalty C. Target proportions are specified within the question generation configuration rather than estimated from benchmark statistics. Difficulty D is the fraction of retained questions whose type has Solver accuracy within a prescribed interval, with default bounds of 0.3 and 0.7. These accuracies are estimated from the tasks used for diagnosis. The quality profile supplies feedback for generation policy proposals. Questioner revisions take effect when tasks are generated again, rather than when the Solver is rerun on an unchanged task set.

A.3 SOLVER CANDIDATE VALIDATION

In each round, about 600 usable generated questions are divided into an evolution subset of 200 questions and a validation subset of 400 questions. The evolution subset supports failure diagnosis and candidate proposals, while Solver comparisons use the same questions and references from the validation subset. For N validation questions, let a_i be the reference answer and let $\hat{a}_i$ and $\hat{a}'_i$ be the

current and candidate answers. We define the paired correctness difference and its mean as

$$d_i = \mathbf{1}\{\hat{a}'_i = a_i\} - \mathbf{1}\{\hat{a}_i = a_i\}, \qquad \widehat{\Delta} = \frac{1}{N}\sum_{i=1}^{N} d_i, \tag{4}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. A candidate passes the accuracy criterion when $\widehat{\Delta} > 0$. Among candidates passing this criterion and the resource checks, the one with the highest validation accuracy is retained. The current harness is preserved if none qualifies. This generated validation subset is distinct from the dedicated benchmark question set in Appendix A.7.

A.4 Generated Tool Verification

The model proposes each tool as a Python function, together with its name, input signature, purpose, supported question types, and test cases. Each test specifies input arguments and an expected output or condition. For example, an entity-counting tool accepts a time interval and an entity type, retrieves a matching memory segment, and counts its entity entries. A supplied test can request `PERSON` entries between 10 and 20 seconds and require a nonnegative numeric result.

We first inspect the code's syntax tree for disallowed imports, calls, and attribute accesses. We then execute the supplied tests against a selected video's memory through the toolkit. Tests check equality to a supplied value, a specified minimum, a nonempty result, or an output type. All supplied assertions must pass. Errors and failed assertions are returned to the model for revision, with up to three generation attempts.

These tests are supplied by the same model and check basic execution and output properties rather than independently establishing correctness. Passing them makes the tool available during candidate Solver evaluation, not part of the retained harness. The tool is added to the persistent library only if its candidate harness is selected under the validation procedure in Appendix A.3.

A.5 Video Evidence Acquisition

Evidence acquisition depends on video duration. We describe the initial harness configuration and the sampling defaults below; parameters exposed to evolution can change in later rounds. The Questioner uses memory to construct tasks on all paths, but reference answers and designated source intervals are withheld from the Solver during ordinary execution.

Sampling path. Videos of at most 5 minutes receive a single overview at a requested rate of 1 frame per second (fps), followed by one answer call. Videos longer than 5 minutes but shorter than 30 minutes are divided into consecutive 300-second chunks. Each chunk is inspected at 1 fps to produce question-relevant observations and candidate intervals for closer inspection. Candidate intervals receive 5 seconds of context on each side; overlapping intervals or those separated by at most 2 seconds are merged. Up to six intervals, each capped at 60 seconds, are inspected at 2 fps. Their textual observations are then combined in one final answer call. Overview and detail inspection use the toolkit's `normal` and `large` visual budgets, respectively (Qwen Team, 2026b). The default limit is 192 frames per visual model call, not per question; frame and payload limits can reduce the effective sampling rate.

Memory path. Videos of at least 30 minutes use memory to locate intervals before inspecting their frames. Initially, segment-level and node-level searches each request up to six results. Their parent segments are merged and deduplicated, and up to two segments supply the initial text evidence within a 6,000-character budget. The Solver can then use memory tools for up to three rounds and select up to three intervals for frame inspection. The initial configuration adds 24 seconds of context on each side of a matched node's time interval, subject to a 60-second window cap. Frames are requested at 1 fps, with a shared budget of 192 frames across the selected intervals. These are starting values rather than fixed settings for all evolved harnesses.

Failure analysis. We assess evidence coverage using execution trajectories and, where applicable, additional observations. These may include denser frame sampling or memory records from source segments and broader video context. Diagnostic comparisons preserve the evidence representation

of each path. Together, the evidence comparisons and trajectories help identify possible localization, coverage, and reasoning limitations.

A.6 HYPERPARAMETERS

Table 6 summarizes key hyperparameters, including those for memory construction, question generation, evidence acquisition, and model calls. For evolvable parameters, we report the initial values, which may change across rounds. Frame rates are requested sampling rates, subject to frame-count and payload limits. Output-token limits apply to individual model calls, not entire question-solving runs.

Table 6: Key hyperparameters. Sampling settings for memory construction and Solver execution are listed separately. Model-call entries specify temperature / maximum output tokens.

Parameter	Value
Memory construction	
Scene detection rate	0.25 fps
Minimum / maximum segment duration	30 / 300 s
Subgraph sampling rate / frame cap per segment	0.5 fps / 250
Question generation and quality	
Number of answer options K	4
Configured weights $(w_{\mathrm{b}}, w_{\mathrm{o}}, w_{\mathrm{c}}, w_{\mathrm{d}}, w_{\mathrm{r}})$	$(2, 1, 1, 1, 0.5)$
Intermediate type-accuracy interval	$[0.3, 0.7]$
Solver evidence acquisition	
Direct / memory routing cutoffs	300 / 1,800 s
Coarse observation chunk duration	300 s
Overview / detail sampling rates	1 / 2 fps
Maximum detail intervals / interval duration	6 / 60 s
Detail padding per side / merging gap	5 / 2 s
Sampling-path frame cap per visual call	192
Memory search results per entry / retained segments	6 / 2
Initial memory-text budget	6,000 characters
Memory-tool rounds / selected replay intervals	3 / 3
Replay sampling rate / shared frame cap	1 fps / 192
Replay padding per side / window cap	24 / 60 s
Model calls and tool generation	
Solver answering	0.1 / 2,048
Memory-tool decisions	0.0 / 3,000
Hyperparameter proposals	0.3 / 1,200
Tool-routing policy proposals	0.4 / 1,600
Memory-strategy code generation	0.4 / 2,600
Memory-tool code generation	0.3 / 2,000
Maximum tool-generation attempts	3

A.7 DEDICATED BENCHMARK QUESTION SET

The dedicated benchmark question set contains 400 questions and is used for Figure 3 and Section 4.4. We first define ten question types, then sample from each benchmark according to the types it covers, so that no single benchmark determines the overall composition. Each question retains its original answer options and reference answer, and evaluation follows the protocol of Section 4.1. The set remains fixed across rounds and analyses; missing answers count as incorrect. For reporting, the ten types are grouped into perception, temporal grounding, and reasoning and synthesis, which contain 200, 110, and 90 questions. Analyses report accuracy within each group and over the full set, since an individual benchmark contributes too few questions for a separate comparison. Table 7 reports the composition together with this grouping.

Table 7: Composition of the dedicated benchmark question set used in Figure 3 and Section 4.4. Question types are divided into three reporting groups, with the group size given in parentheses. Each column reports the number of questions contributed by one benchmark, and Share is the proportion of the full set. LongVB, V-Holmes, and V-MME denote LongVideoBench, Video-Holmes, and Video-MME.

Group	Question type	LVBench	MLVU	LongVB	V-Holmes	MMVU	V-MME	Total	Share
Perception (200)	Counting	3	8	2	0	1	6	20	5.00
	Entities and attributes	15	13	25	4	6	17	80	20.00
	Actions and events	10	15	12	6	5	12	60	15.00
	Spatial context	7	6	13	3	4	7	40	10.00
Temporal (110)	Temporal order	10	15	14	7	5	9	60	15.00
	Retrieval and localization	10	12	8	5	3	12	50	12.50
Reasoning (90)	Causal and multi step	7	7	4	14	10	3	45	11.25
	Topic and knowledge	3	6	0	5	9	2	25	6.25
	Anomaly and egocentric	1	6	0	1	1	1	10	2.50
	Summarization	4	2	2	0	1	1	10	2.50
Total		70	90	80	45	45	70	400	100.00

A.8 Evolution Video Collection

The evolution pool contains 130 videos drawn from six source collections. Tables 8 and 9 report their source counts and duration distribution, respectively, using the source identifiers in the video collection. Only raw videos are used; accompanying questions, answers, and temporal annotations are excluded.

Table 8: Source counts of the evolution videos.

Source	Videos
`mlvu_dev`	33
`vript_long`	30
`videomarathon`	29
`tvqa_long`	23
`scalelong`	8
`video_holmes_train`	7
Total	**130**

Table 9: Duration distribution of the evolution videos.

Duration (min)	Videos
$d \leq 5$	30
$5 < d \leq 30$	70
$30 < d \leq 60$	25
$d > 60$	5
Total	**130**

B Overall Evolution Algorithm

Algorithm 1 summarizes the evolution loop using the notation of Equation 1. Memory is constructed once and reused. Feedback $\mathcal{F}_t$ combines question quality reports, Solver trajectories, and failure diagnosis. Reference answers and source labels support evaluation and diagnosis but are withheld during ordinary solving. Candidates may modify any enabled component of $(P_t, C_t, \lambda_t, \mathcal{T}_t)$; unchanged components are retained.

Algorithm 1 VideoEvoHarness Evolution

Input: Videos $\mathcal{V}$, frozen model f_θ, initial policies (ϕ_0, H_0), rounds T
Output: Updated Questioner ϕ_T and Solver harness H_T

$\mathcal{M} \leftarrow$ *BuildMemory*$(\mathcal{V})$
$\mathcal{B} \leftarrow$ *GenerateAndFilter*$(\phi_0, \mathcal{M})$
$r \leftarrow$ false
for $t = 0, \ldots, T-1$ **do**
  $H_{t+1} \leftarrow H_t$, $\phi_{t+1} \leftarrow \phi_t$
  if r or a scheduled question refresh is due **then**
    $\mathcal{B} \leftarrow$ *GenerateAndFilter*$(\phi_t, \mathcal{M})$; $r \leftarrow$ false
  end if
  $(\mathcal{D}_t^{\text{evo}}, \mathcal{D}_t^{\text{val}}) \leftarrow$ *SelectTasks*$(\mathcal{B}, t)$
  $\mathcal{F}_t \leftarrow$ *CollectFeedback*$(H_t, \mathcal{D}_t^{\text{evo}}, \mathcal{B}, \mathcal{M})$
  $\mathcal{H}_t \leftarrow$ *ProposeHarnesses*$(H_t, \mathcal{F}_t)$
  $\mathcal{H}_t \leftarrow$ *Precheck*$(\mathcal{H}_t)$
  Compute $\widehat{\Delta}(H')$ for each $H' \in \mathcal{H}_t$ on the same $\mathcal{D}_t^{\text{val}}$
  $\mathcal{H}_t^+ \leftarrow \{H' \in \mathcal{H}_t : \widehat{\Delta}(H') > 0$ and *ResourceCheck*$(H')\}$
  if $\mathcal{H}_t^+ \neq \emptyset$ **then**
    $H_{t+1} \leftarrow \arg\max_{H' \in \mathcal{H}_t^+} \widehat{\Delta}(H')$
    Register any newly selected strategy or tool code
    $\mathcal{Q}_t \leftarrow$ *ProposeQuestioners*$(\phi_t, \mathcal{F}_t)$
    $J_t \leftarrow$ *PairedQuality*$(\{\phi_t\} \cup \mathcal{Q}_t, \mathcal{M}, \mathcal{F}_t)$
    $\mathcal{Q}_t^+ \leftarrow \{\phi' \in \mathcal{Q}_t : J_t(\phi') > J_t(\phi_t)\}$
    if $\mathcal{Q}_t^+ \neq \emptyset$ **then**
      $\phi_{t+1} \leftarrow \arg\max_{\phi' \in \mathcal{Q}_t^+} J_t(\phi')$
      $r \leftarrow$ true
    end if
  end if
end for
return (ϕ_T, H_T)

Precheck applies code restrictions and supplied tests before candidate evaluation (Appendix A.4). The gain $\widehat{\Delta}(H')$ is measured relative to H_t using Equation 4, and *ResourceCheck* enforces the enabled budget limits. *PairedQuality* regenerates and filters questions for each policy using the same video scope, generation budget, and random seed. It returns the scores $J_t(\phi) = \mathcal{J}_Q(\phi)$ from Equation 2, with difficulty assessed using the shared feedback $\mathcal{F}_t$. Accepted Questioner revisions trigger a question-bank refresh in the next round, leaving the current validation questions unchanged. Benchmark evaluations are separate from this loop.

C Case Study

C.1 Generated Question Examples

Figures 5 and 6 illustrate two memory-grounded questions produced by the Questioner. The displayed memory notes identify the records supporting each reference answer.

Action recognition (perception). Figure 5 asks which craft action is performed. The micro-event at 2132–2138 seconds records a person spraying textured paint onto a pink plastic plate, supporting option A. The question tests recognition of a visible action, with its reference grounded in the event record rather than an interpretation of the subsequent painting steps.

Object reasoning (entity co-occurrence). Figure 6 asks which background element appears alongside the bookshelves. The split-screen scene contains bookshelves behind the participant in the bottom panel and a window in the top-left panel, supporting option D. This question belongs to the entity-co-occurrence form of object reasoning: the answer is grounded in the entities recorded within the same segment, not physical adjacency in one room.

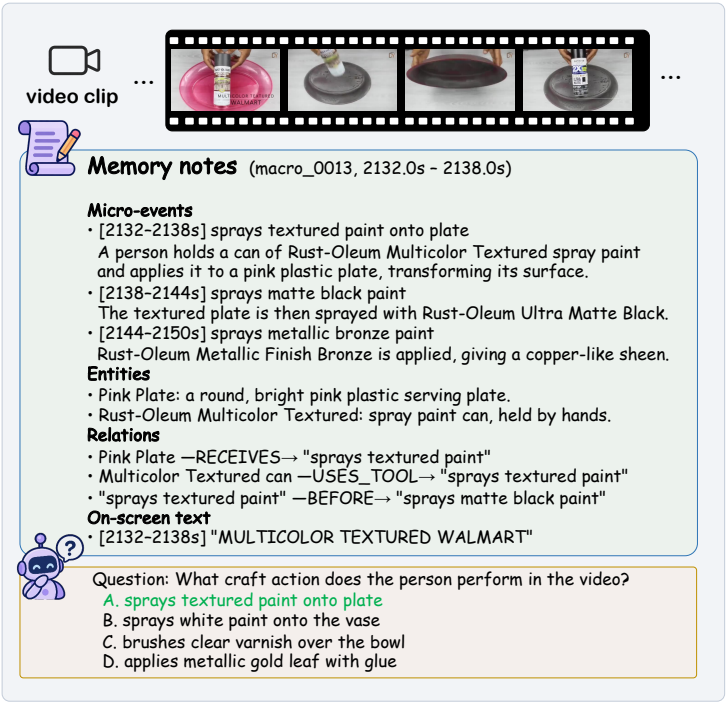


Figure 5: Generated action-recognition question. The micro-event record supports answer A: spraying textured paint onto a plate.

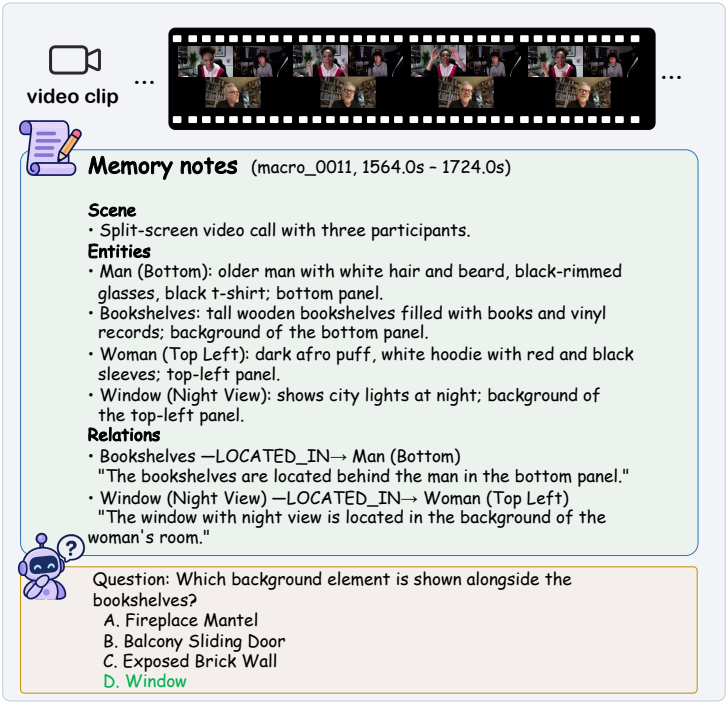


Figure 6: Generated object-reasoning question based on entity co-occurrence. Bookshelves and a window appear in different panels of the same scene, supporting answer D.

C.2 Solver Harness Evolution

Figure 7 compares the initial and R3 harnesses on a Video-Holmes question using Qwen3.7 Plus with frozen model parameters. The question asks what a black-hat clown's clapping after another performer's failed act implies. The initial harness targets the clapping interval at 01:31–01:34 for frame inspection. It interprets the gesture as conventional acknowledgment between performers and selects A, "Respect for performers." This interpretation focuses on the local action without establishing its meaning in the broader interaction.

The R3 harness inspects the same interval and additionally requests frames from 03:22–03:55. These later observations provide context about the clown's attitude toward the other performer. The Solver interprets the subsequent expressions as mocking rather than respectful and selects E, "The personification of a vindictive mentality." By connecting the focal gesture with evidence from a later interval, the evolved harness supports a more context-sensitive answer. This case illustrates how harness evolution improves answering through broader temporal evidence acquisition, rather than through changes to the question or model parameters.

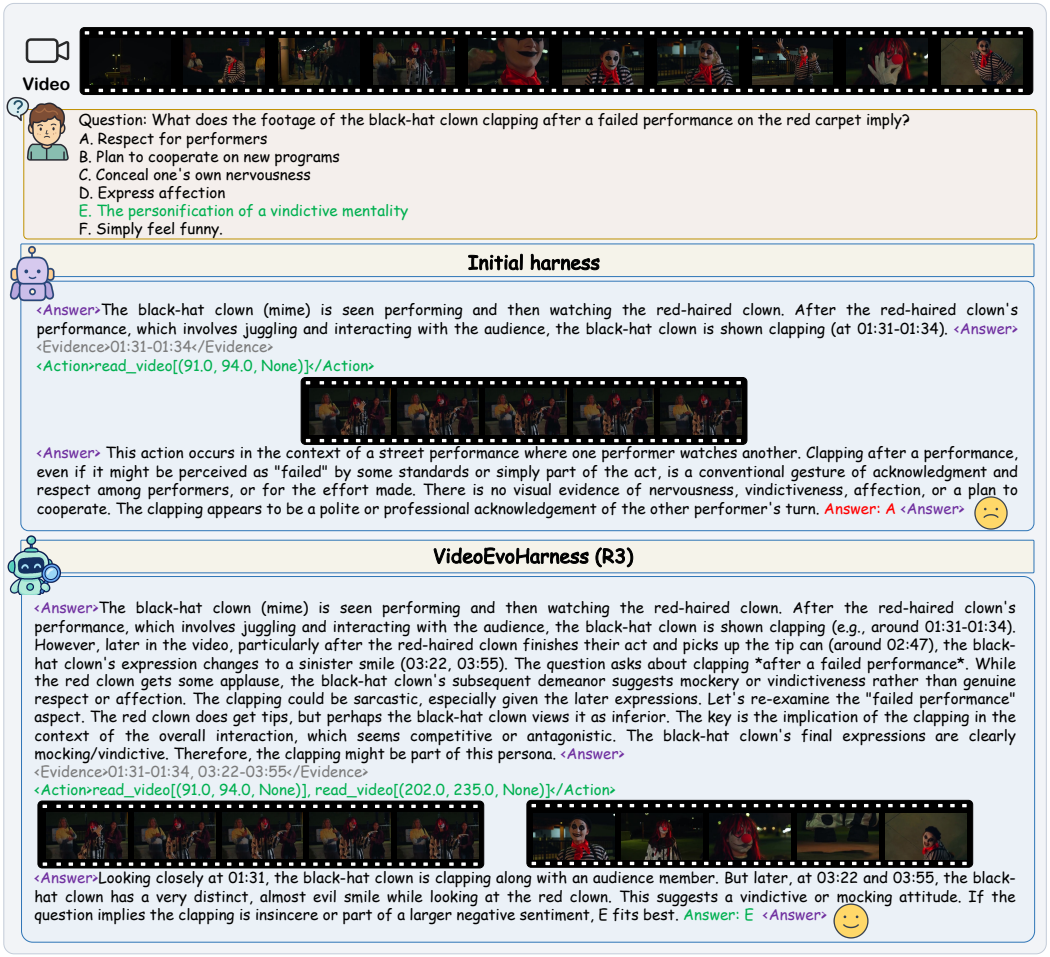


Figure 7: Comparison of the initial and R3 Solver harnesses on Video-Holmes with Qwen3.7 Plus. Compared with the initial harness, the R3 harness supplements inspection of the clapping interval with later evidence, leading to answer E rather than A. The comparison illustrates a change in evidence acquisition while the backbone remains frozen.

C.3 Strategy and Hyperparameter Evolution

Figures 8 and 9 illustrate changes to retrieval logic and evidence allocation during evolution.

Strategy code. The original procedure in Figure 8 uses a configured set of retrieval entry points for every question. The generated strategy instead uses question keywords to activate OCR search for text-related queries and event enumeration for counting or non-OCR queries, while retaining semantic node search. It ranks the retrieved segments and assembles their records according to the question: visible text is included for OCR queries, and event descriptions for counting and temporal queries. If retrieval yields no candidates, a summary-based fallback supplies candidate segments. This example shows how code evolution introduces question-dependent tool use and evidence assembly, rather than simply increasing the number of retrieved records.

Hyperparameters. Figure 9 shows two configuration changes: `use_learned_strategy` switches from `false` to `true`, enabling the generated retrieval routine, and `max_window_sec` decreases from 60 to 15 seconds. The latter limits the duration of each replay window on the memory path, restricting local inspection to a shorter interval rather than increasing the frame budget. Together, these examples illustrate how evolution adapts both the evidence-gathering procedure and its temporal scope within a frozen-backbone harness.

```python
# Before evolution: retrieval entries are fixed by config, same for every question
for entry in r["entries"]:
    if entry == "nodes":
        _collect(tk.search_nodes(question, top_k=topk * mult,
                                 node_types=r.get("node_types")))
    elif entry == "macro":
        _collect(tk.search_nodes(question, top_k=topk * mult, node_types=["macro"]))
    elif entry == "ocr":
        _collect(tk.search_ocr_text(question, top_k=topk * mult))
    elif entry == "enumerate":
        _collect(tk.enumerate_events(question, min_cosine=r.get("enumerate_min_cosine",
0.35),
                                     max_results=topk * mult))
    elif entry == "time":
        _collect(tk.search_by_time(0, 10**9))

ranked = [(mid, scores[mid]) for mid in order]
...
return ranked[:base_topk]
```

Strategy Code Evolution

```python
# After evolution: route retrieval by question type
q_lower = question.lower()
is_ocr      = any(k in q_lower for k in ['text', 'read', 'write', 'sign', 'word', ...])
is_counting = any(k in q_lower for k in ['how many', 'count', 'number of', 'total'])
is_temporal = any(k in q_lower for k in ['when', 'before', 'after', 'during', ...])

if is_ocr:
    hits = tk.search_ocr_text(question, top_k=topk_hint * 2)
if is_counting or not is_ocr:
    hits = tk.enumerate_events(question, min_cosine=0.5, max_results=topk_hint * 3)
hits = tk.search_nodes(question, top_k=topk_hint,
                       node_types=['Event', 'Entity', 'OnScreenText', 'macro'])

if not candidate_macros:                          # fallback
    candidate_macros = {m['macro_id']: 0.1 for m in summary['macros'][:max_macros]}
```

Figure 8: Illustrative strategy code change during evolution. The generated routine combines keyword-conditioned retrieval with semantic search and a summary-based fallback.

```
# Hyperparameter Evolution

  retrieval:
-   use_learned_strategy: false
+   use_learned_strategy: true        # model-written retrieval strategy

  policy.read_video:
-   max_window_sec: 60
+   max_window_sec: 15.0              # narrower replay window

```

Figure 9: Illustrative configuration changes during evolution: enabling the generated retrieval strategy and reducing the replay-window cap from 60 to 15 seconds.

D LIMITATIONS AND FUTURE PLANS

Despite the gains observed across benchmarks and backbones, VideoEvoHarness has several limitations that motivate further research. First, improvements are not uniform across capabilities. Some categories decline as evolution progresses, even when overall accuracy increases. Future work could explore capability-aware validation and update selection to address weaker abilities while preserving existing strengths.

Second, although the framework supports evolution of skills, strategies, hyperparameters, and tools, the accepted changes in our current experiments do not cover all four components. Under the three-round budget, the retained Solver updates concern strategies and hyperparameters, while skills and tools remain unchanged. The cost of candidate generation and evaluation limits our ability to extend evolution and systematically characterize its longer-term behavior. Longer runs under controlled budgets, together with more efficient candidate evaluation, could clarify when different components change and how their contributions interact.

Finally, our evaluation relies on conventional video understanding benchmarks. These measure downstream task performance but are not designed to assess the evolution process itself. Dedicated benchmarks and protocols could evaluate sustained improvement, capability retention, generalization to changing tasks, and the cost of acquiring these gains. Such evaluation would provide a more direct account of an agent's ability to evolve beyond endpoint accuracy.

Taken together, our experiments support self evolving video agent harnesses as a promising research direction. More balanced capability development, broader exploration of component evolution, and dedicated evaluation offer concrete opportunities to advance this direction.