A Multi-Agent Perception-Action Alliance for Efficient Long Video Reasoning

Yichang Xu[1], Gaowen Liu[2], Ramana Rao Kompella[2], Tiansheng Huang[1], Sihao Hu[1]
Fatih Ilhan[1], Selim Furkan Tekin[1], Zachary Yahn[1], Ling Liu[1]
[1]Georgia Institute of Technology, Atlanta, GA
[2]Cisco Systems, USA
xuyichang@gatech.edu, {gaoliu,rkompell}@cisco.com,
{sihaohu,thuang,filhan,stekin6,zachary.yahn}@gatech.edu, ling.liu@cc.gatech.edu

Abstract

This paper presents a multi-agent perception-action exploration alliance, dubbed A4VL, for efficient long-video reasoning. A4VL operates in a multi-round perception-action exploration loop with a selection of VLM agents. In each round, the team of agents performs video question-answer (VideoQA) via perception exploration followed by action exploration. During perception exploration, each agent learns to extract query-specific perception clue(s) from a few sampled frames and performs clue-based alignment to find the video block(s) that are most relevant to the query-specific event. During action exploration, A4VL performs video reasoning in three steps: (1) each agent produces its initial answer with rational, (2) all agents collaboratively scores one another through cross-reviews and relevance ranking, and (3) based on whether a satisfactory consensus is reached, the decision is made either to start a new round of perception-action deliberation by pruning (e.g., filtering out the lowest performing agent) and re-staging (e.g., new-clue and matching block based perception-action exploration), or to conclude by producing its final answer. The integration of the multi-agent alliance through multi-round perception-action exploration, coupled with event-driven partitioning and cue-guided block alignment, enables A4VL to effectively scale to real world long videos while preserving high quality video reasoning. Evaluation Results on five popular VideoQA benchmarks show that A4VL outperforms 18 existing representative VLMs and 11 recent methods optimized for long-video reasoning, while achieving significantly lower inference latency. Our code is released at https://github.com/git-disl/A4VL.

1. Introduction

Multimodal large language models (MLLMs) have been developed to support video understanding in recent years.

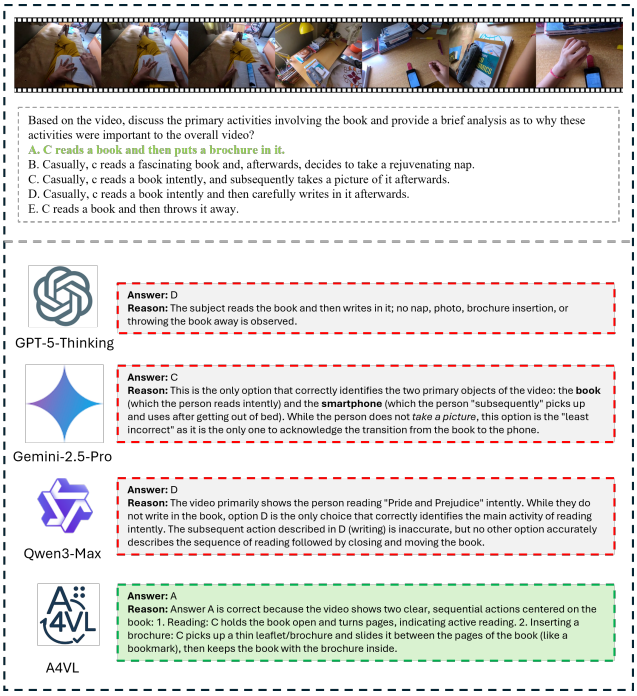

Figure 1. Responses from A4VL and three of the strongest closed-sourced models.

Compared to traditional approaches [11, 14, 46, 56, 57, 59], MLLM-based methods [15–17, 33, 37, 38, 41, 49, 51, 53, 54] exhibit strong ability for spatiotemporal visual reasoning, as well as human-level text understanding and generation. However, most models often fail to produce high performance reasoning on long videos, because when the number of frames is large, the memory and time costs scale quadratically. Even with enough computing resources and time, some recent works [20, 29] show that simply increasing the frame sampling density may even hurt performance, for example, due to redundant frames introducing noise and distracting attention away from truly informative key

frames. The problem is especially aggravated for video QA workloads that center on questioning events covered in only a small subset of frames, as the problem of identifying accurately where the most relevant frames reside is challenging. Eefficient and scalable solutions are on-demand to improve video understanding performance for complex vision–language reasoning about long and complex videos.

Our work is inspired by some pioneering efforts in this direction. For example, GPT-4o [25] is trained on a large multimodal corpus and, after such training, is able to process long videos by taking many more frames. Although this yields higher accuracy, it is much slower. On Video-MME [7], GPT-4o takes more than 150 s on average to answer a multiple-choice question. Therefore, the community has begun to explore optimization strategies. Some works focus on token merging [1, 55], sparsification [9, 13, 18, 20], or training-free agent approaches [2, 10, 35, 40, 45, 47, 52], while others design more efficient neural architectures [8, 44, 50] or memory-retrieval-based systems [22, 31, 34, 48]. For example, DYTO [55] proposes a dynamic bipartite token merging strategy, which supports a higher frame sampling rate while keeping the resource usage acceptable; AuroraLong [44] replaces the LLM component of MLLMs with a linear RNN layer to handle input sequences and also combines token merging techniques to improve efficiency. Among these methods, agent-based approaches and memory retrieval methods are orthogonal to token-based and architecture-focused optimizations. In this paper, we present an innovative multi-Agent Perception-Action Alliance based method for long video reasoning.

Problem Statement. We note some limitations of existing agent-based methods. First, their inference speed is low. For example, VideoAgent [5] takes over ten minutes on a long video that lasts for an hour. Second, most of them use a single MLLM to perform decision making and inference; they do not support effective collaboration among multiple agents. In addition, they often rely on strong video grounding models, and yet when the question is long or complex, they remain struggling to locate key frames. For example, MoReVQA [24] uses object detection and image QA models to find keyframes and achieves good performance on NeXT-QA [43]. However, its accuracy falls behind other approaches on the long and complex video datasets, e.g., EgoSchema [23] (see Table 1 in Section 3.2).

To address these limitations, we propose a multi-agent perception–action alliance framework (A4VL), featured with three core components: (i) an agent teaming strategy that leverages multiple diverse MLLMs to enhance video reasoning performance; (ii) an event-based partitioning and random sampling strategy to handle resource-intensive long-video processing; and (iii) an iterative agent coordination framework for perception exploration and action exploration that further boosts reasoning performance. Figure 1 shows the responses from three of the strongest closed-sourced models to date (GPT-5-Thinking [26], Gemini-2.5-Pro [4], and Qwen3-Max [39]), as well as our A4VL, on an example from the EgoSchema dataset. Although all these MLLM-based methods generate incorrect answers, A4VL remains correct. Figure 2 shows the general framework of A4VL. Figure 3 illustrates our agent teaming strategy, which selects the team that collaborates best from a pool of agents. Figure 4 in Section 2 visualizes the reasoning procedure of A4VL on this example.

Upon receiving a query, A4VL first goes through the perception exploration step. During this step, each agent samples a small number of frames (e.g., 4) to generate a perception clue. We then apply an event-based partition method to divide the video into several blocks and use models like CLIP [30] to obtain the similarity score of each block, measuring its consistency with the perception clue. Each agent samples from high-scoring blocks and forms a set of frames for the next step, e.g., 16 sampled frames. Next, agents move into the action exploration step: they first generate answers and reasons for the sampled frames relevant to a query. If the answers are consistent, a summarizer aggregates the reasons and returns the final answer to the user. Otherwise, each agent is asked to rate every other agent, and the agent team is pruned based on these scores. The surviving agents then revise their perception clues based on the current state and move back to Stage 2 of the perception exploration step, starting the next round of collaboration.

Our main contributions are summarized as follows:

- **A4VL framework.** We develop A4VL, a training-free multi-agent perception–action framework for long-video question answering under tight frame budgets.
- **Teaming and clue-guided perception.** We empower A4VL with an unsupervised agent teaming method and an event-based *sample→clue→block* pipeline, optimized by combining CLIP-similarity enhanced block selection with adaptive frame allocation, enabling A4VL agents to focus on key frames without retraining.
- **Accuracy–latency gains.** On five long-video benchmarks against 28 baselines, A4VL achieves the best accuracy while being substantially faster than strong closed- and open-source MLLMs that process many more frames.

2. Methodology

2.1. Design Overview

In this section, we provide an architecture overview of A4VL. As shown in Figure 2, A4VL consists of a perception exploration step and an action exploration step, where multiple agents work collaboratively to produce the final answer. The two steps run iteratively until agents reach a consensus. When we have multiple MLLMs available, we prefer to use a small subset as the agent pool. Hence, we de-

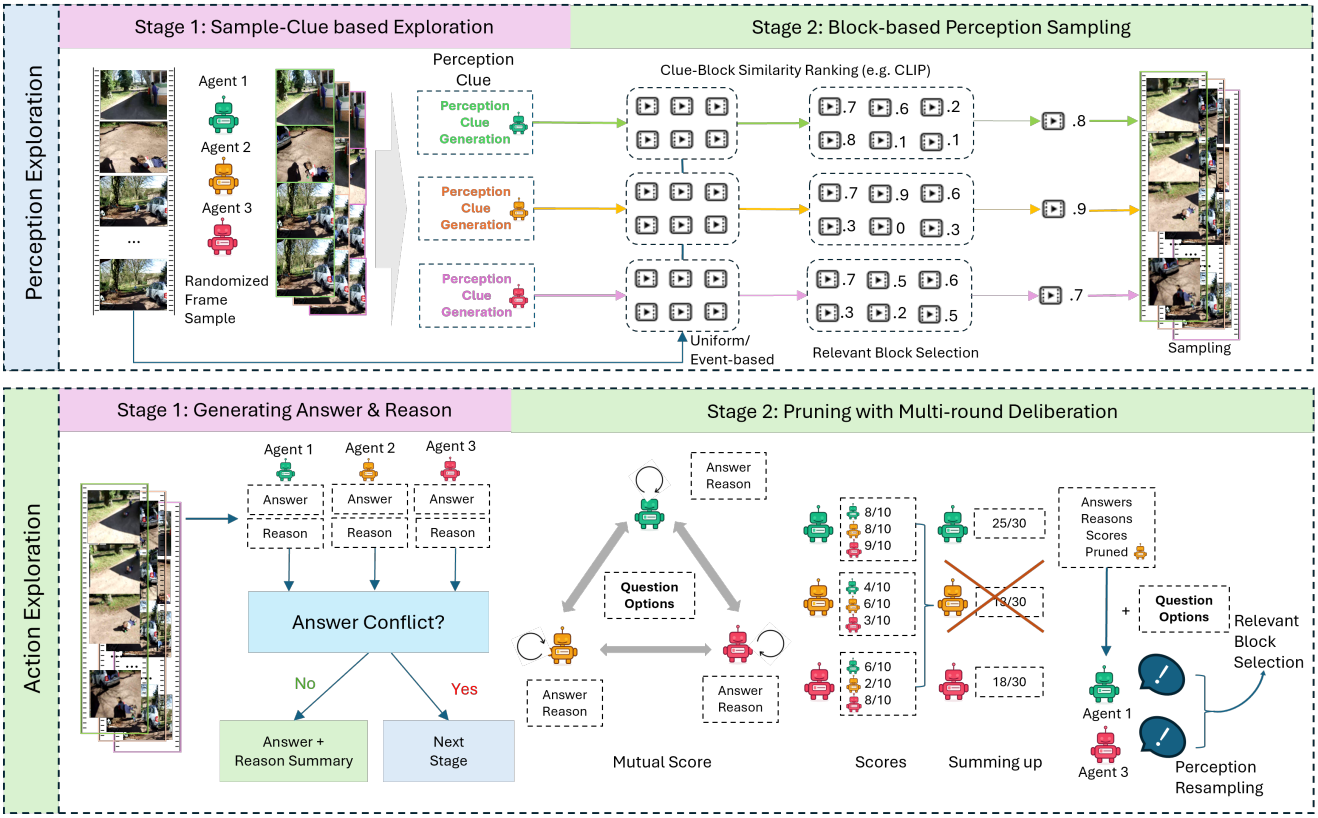

Figure 2. Overview of A4VL architecture.

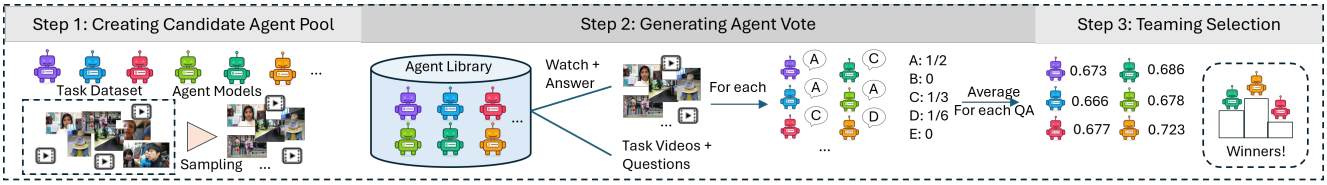

Figure 3. A4VL agent teaming workflow.

sign an optional agent teaming stage (Figure 3) that selects a task-specific subset of agents. Thereafter, all questions under the same task reuse the same agent subset. The agent teaming process is as follows. Suppose we have M agents and K unlabeled video-question pairs from the task, each with C answer choices. These K samples can be randomly drawn from the task dataset. Using labels would make the teaming algorithm more accurate, but we assume unlabeled data for practical deployment. In the agent teaming step, each agent independently runs the perception exploration and answer generation steps to produce predictions. We count the frequency of each choice in each sample and denote them as $\{f_{qr}\}_{q=1:K,r=1:C}$, where each f_{qr} is the fraction of agents that choose the rth option for question q. For each agent, let r_q denote the index of the option it selects for question q. We define its score as $\frac{1}{K}\sum_{q=1}^{K} f_{qr_q}$. We choose m ($m = 3$ in our implementation) agents with highest scores to form the task-specific agent pool. As illustrated in Figure 3, the teaming process has three steps.

- **Step 1**: we prepare the agent library and sample a small set of video-question pairs from the task.
- **Step 2**: each agent runs part of the A4VL workflow and independently generates answers, and we compute the frequency of each answer for each question. For the example in Figure 3, the frequencies of choices A–E are $1/2$, 0, $1/3$, $1/6$, and 0, respectively. Purple, blue, and light green agents therefore receive a score of 0.5 on that question, while the green and pink agents receive $1/3$, and so on.
- **Step 3**: we average each agent's score over all sampled questions, and the top 3 agents (green, yellow, and pink) are selected for this task.

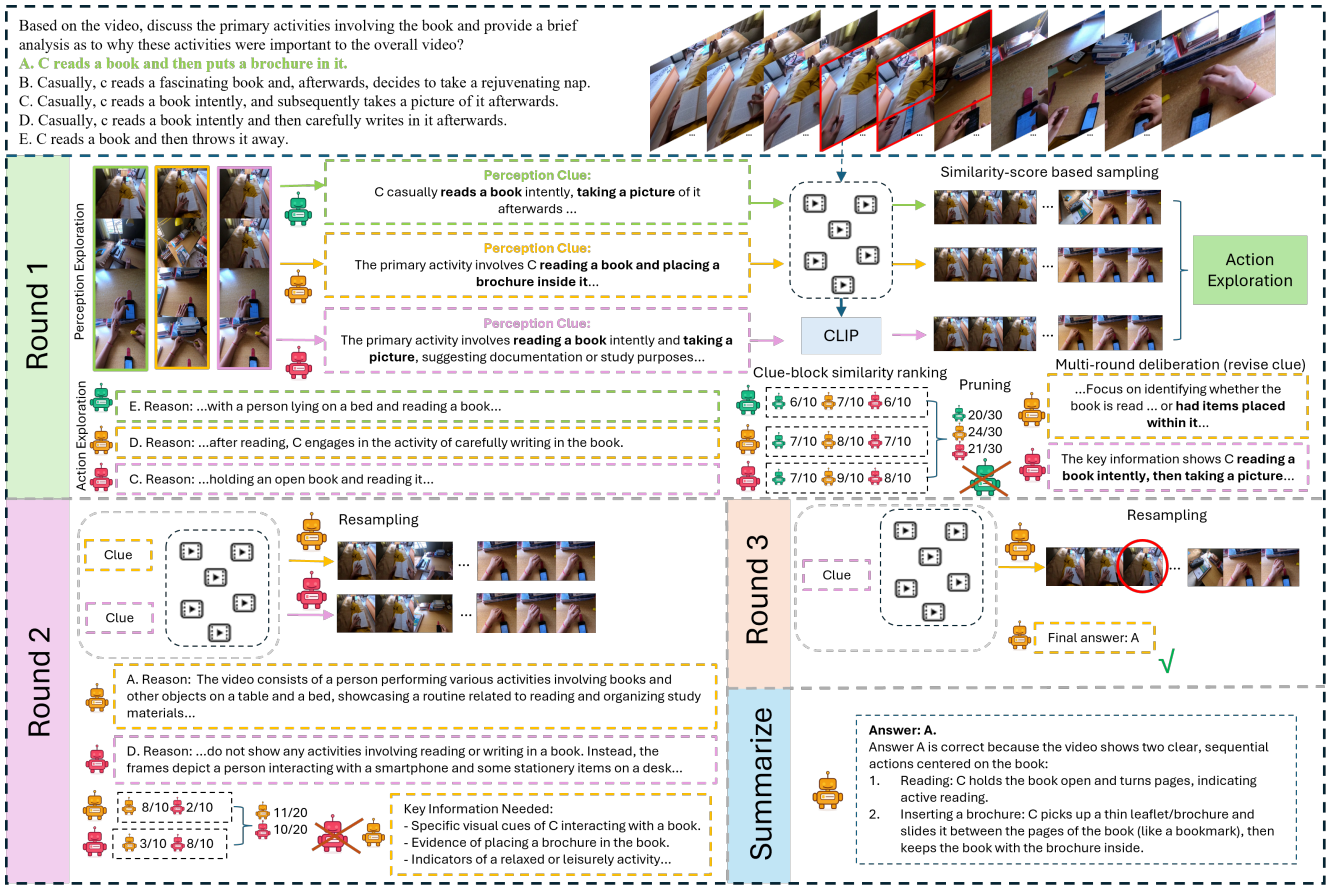

Figure 4. Example of A4VL on EgoSchema. Here all agents have wrong answers at the beginning, but later one of them turns to the correct answer and other agents are pruned.

Within the task, after A4VL receives a video and the user's question (including choices, and additional information like subtitles under some benchmarks), it first goes through the perception exploration step. As shown in Figure 2, the perception exploration step consists of two stages. In Stage 1, each agent randomly samples N_1 frames ($N_1 = 4$ in our settings) to preview the entire video, and then generates a perception clue to identify keyframes. In the illustrative example, the video is divided into six blocks using an event-based approach. For each agent, a CLIP model like [3] is used to compute similarity scores between its perception clue and each block. These scores indicate how relevant each block is to the current question. Then, N_2 frames (set to 16 in our experiments) are sampled from the blocks according to their scores, as described in the perception exploration subsection, and are used in the next action exploration step.

During the action exploration step, each agent is asked to provide an answer and a corresponding reason, given its N_2 frames obtained from the previous step. If they reach a consensus, then we give the answer to the user and summarize the reason to explain the answer. Otherwise, each agent is asked to evaluate all agents' answers, and the agent with the lowest total score (based on self- and peer evaluations) is pruned. For the example shown in Figure 2, the yellow agent is pruned and other agents are required to generate a new perception clue in order to better distinguish between conflicting answers in the next round. After that, all agents will go back to the stage 2 of perception exploration to start a new round, until finally all agents reach a consensus.

2.2. Perception Exploration

The top part of Figure 2 shows our perception exploration step workflow. We index rounds by j. In the first round, each agent goes through the entire perception exploration step; in subsequent rounds ($j > 1$), each agent starts directly from selecting relevant blocks. Suppose we have a video V which consists of n frames: $V = \{v_1, v_2, \cdots, v_n\}$, question Q, options O, and agent pool $\mathbb{A} = \{A_i : 1 \leq i \leq m\}$, where m is the number of selected agents after agent teaming. As shown in Figure 2, perception exploration step has two stages: sample-clue based exploration and block-

based perception sampling.

In the first stage, each agent generates a perception clue, which contains necessary information that can be used to identify keyframes in stage 2. For each agent $A_i \in \mathbb{A}$, we first sample N_1 frames from the video using a sampling strategy $p_1 : 2^V \times \mathbb{N}^+ \to 2^V$:

$$\hat{v}_{A_i} = p_1(V, N_1). \tag{1}$$

For p_1, we have two choices: (i) we can randomly select N_1 frames from the entire video, or (ii) we can divide the video into event-based blocks, and sample a few from each block. In experiments, we test both alternatives and find that (i) is better, so we use choice (i) as our final choice. The reason is that agents do not need fine-grained understanding of the video to generate a coarse perception clue, so uniform random sampling provides higher diversity and better temporal coverage. Given the sampled N_1 frames, question, and options, each agent is asked to generate a perception clue, which is later used to identify relevant blocks. We denote the perception clue for agent A_i at jth round as $P_{i,j}$. In the first round,

$$P_{i,1} = A_{i,clue}(\hat{v}_{A_i}, Q, O). \tag{2}$$

For subsequent rounds, $P_{i,j}$ is generated at the end of the action exploration step. For example, by the end of the first round, $P_{i,2}$ is generated for each $A_i \in \mathbb{A}$.

In the second stage, we first partition the video into at most B blocks, $B_1, B_2, \cdots, B_b$. We provide two partition strategies: (i) **uniform partition**, which means $B_i = V_{(i-1)\lfloor \frac{n}{B} \rfloor + 1 : i\lfloor \frac{n}{B} \rfloor}$; (ii) **event-based division**, where we detect scene changes using DINOv2 embeddings [27] and simple pixel cues (HSV/motion/sharpness), generate candidates with KTS [28], PELT [12], and SSM-based novelty [6], merge them and apply non-maximum suppression (NMS), and keep the top $B - 1$ boundaries (at most B blocks). This procedure is efficient, as most videos in our benchmarks can be processed within two seconds. Details are given in the appendix. We also test both strategies in our experiments, and finally choose (ii). After dividing the video into blocks, we compute the similarity between each block and each agent's perception clue. In this paper, we use a CLIP model to compute the similarity scores. Other approaches, such as computing the similarity between the caption of each block and each perception clue, are also possible, but since CLIP is the fastest, we leave the exploration of more advanced similarity computation as future work. We then sample the target frames under two scenarios. (i) If all blocks have similarity scores below ρ (set to 0.8 in our settings), we sample from the most relevant block for each agent: for agent A_i, we identify the block $B^*_{i,j}$ with the maximum score and use a sampling strategy $p_2 : 2^V \times \mathbb{N}^+ \to 2^V$ to sample N_2 frames from $B^*_{i,j}$:

$$V^{(i)}_{act} = p_2(B^*_{i,j}, N_2). \tag{3}$$

(ii) Otherwise, we retain blocks with similarity score $> \rho$. For each agent A_i, we obtain a score vector $\mathbf{s}^{(i)}$ over blocks. We first normalize it by $\mathbf{s}^{(i)} \leftarrow \mathbf{s}^{(i)} - \max(\mathbf{s}^{(i)})$, and then compute an integer allocation vector $\mathbf{c}^{(i)} := \lfloor N_2 \cdot SoftMax(\mathbf{s}^{(i)}) \rfloor$ for the blocks. If $\sum_k c^{(i)}_k < N_2$, we randomly distribute the remaining $N_2 - \sum_k c^{(i)}_k$ samples over the blocks to ensure a total of N_2 action frames. We then sample $c^{(i)}_k$ frames from each retained block B_k using p_2 and take their union to form $V^{(i)}_{act}$.

2.3. Action Exploration

Now each agent has its own set of N_2 sampled frames, denoted by $V^{(i)}_{act}$. The action exploration step uses these frames to generate answers and, when necessary, to prune agents. It has two stages: answer and reason generation, and pruning with multi-round deliberation, as shown in the bottom part of Figure 2.

In the first stage, each agent is asked to generate an answer and corresponding reason given its own sampled frames and the question information. For agent A_i, it generates the answer $a_{i,j}$ and reason $R_{i,j}$ as follows:

$$a_{i,j} = A_{i,act}(V^{(i)}_{act}, Q, O), \tag{4}$$

$$R_{i,j} = A_{i,reason}(V^{(i)}_{act}, a_{i,j}, Q). \tag{5}$$

Then for all agents, we have the answer set $S_{a,j}$ and reason set $S_{r,j}$:

$$S_{a,j} = \{a_{1,j}, a_{2,j}, \cdots, a_{|\mathbb{A}|,j}\}, \tag{6}$$

$$S_{r,j} = \{R_{1,j}, R_{2,j}, \cdots, R_{|\mathbb{A}|,j}\}. \tag{7}$$

If all agents reach a consensus, we send all $P_{i,j}$ values in round j, $S_{a,j}$, and $S_{r,j}$ to our reason summarizer to give the user the final answer and an explanation. We consider two ways to check for consensus: (i) **Majority consensus**: the most frequent answer appears more than $|\mathbb{A}|/2$ times; (ii) **Full consensus**: all agents output the same answer. We test both strategies and ultimately adopt (ii).

If the agents do not reach a consensus, then we step into the second stage. Given $S_{a,j}, S_{r,j}, Q$ and O, each agent evaluates the soundness of its answer and every other agent's answer. It is asked to give a rating from 1 to 10:

$$\{s_{i,\mathbb{A}_1}, \cdots, s_{i,\mathbb{A}_{|\mathbb{A}|}}\} \leftarrow A_{i,eval}(Q, S_{a,j}, S_{r,j}), \forall A_i \in \mathbb{A}. \tag{8}$$

In Figure 2, the yellow agent has a rating of 8/10, 6/10 and 2/10 given by three agents. After that, for each agent, we

sum up the ratings given by all agents:

$$s_{A_i} := \sum_{A_k \in \mathbb{A}} s_{k,\mathbb{A}_i} \tag{9}$$

For example, the yellow agent now has an overall score of 16/30, which is the lowest. Suppose it's A_2, then we have $s_{A_2} = 16$. Now we do agent pruning, where the agent with the lowest score is removed from the agent pool:

$$\mathbb{A} \leftarrow \mathbb{A} \setminus \{A_{min}\}, \tag{10}$$

$$A_{min} := \arg\min_{A \in \mathbb{A}} s_A. \tag{11}$$

In the Figure 2 case, $A_{min} = A_2$, and the yellow agent is removed. For each of the remaining agent A_i, we generate $P_{i,j+1}$ using $P_{i,j}, S_{a,j}, S_{r,j}, A_{min}, Q, O$ to generate the refined perception clue, which is used in the next round:

$$P_{i,j+1} = A_{i,refine}(P_{i,j}, S_{a,j}, S_{r,j}, A_{min}, Q, O). \tag{12}$$

Figure 4 shows an example from EgoSchema. We can see at the beginning, all agents generate different and incorrect answers. However, in the second round, the orange agent begins to realize that choices B, C, D, and E do not have clear visual evidence, and it revises the perception clue to focus on the interaction between the person and the book. In the third round, the keyframe where the person is putting a brochure into the book is sampled (highlighted by the red circle), and therefore the orange agent continues to generate the correct answer. Finally, our summarizer reads the entire multi-round process and returns the final answer together with a concise explanation to the user. The pseudocode of A4VL is provided in the appendix.

3. Experiment

3.1. Experimental Settings

We evaluate A4VL on five benchmarks that cover both short and long videos, with an emphasis on long-video reasoning.

To test our method on short videos with strong temporal/causal reasoning requirements, we evaluate A4VL on NeXT-QA [43]. We randomly sample 200 QA pairs per question type from its test split, resulting in a subset of 1,493 QA pairs. EgoSchema [23] is a medium-length video QA dataset consisting of videos lasting about 3 minutes. We evaluate on its test split with ground-truth labels, which contains 500 QA pairs.

For long videos, we evaluate on LongVideoBench [42], MLVU [58] (test split), and Video-MME [7]. LongVideoBench consists of 1,337 QA pairs, and each video is accompanied by subtitles; many questions are strongly subtitle-related. The videos range from about 3 minutes to 2 hours in length. MLVU-Test is the harder test split of MLVU, containing roughly 500 videos with durations between 3 minutes and 2 hours, and the task is to select the correct answer out of six options. Video-MME consists of 2,700 QA pairs and is divided into short, medium, and long groups according to video duration, with 900 QA pairs in each group. Videos in the short group last about 10 seconds, whereas videos in the long group can last for over an hour.

In the agent teaming step, we select $m = 3$ agents from a pool of 8 models: LLaVA-Video-7B-Qwen2 [54], QwenVL-2.5-7B [38], InternVL3.5-8B [41], QwenVL-2.5-32B [38], InternVL3.5-38B [41], InternVL3-78B [41], QwenVL-2.5-72B [38], and LLaVA-Video-72B-Qwen2 [54]. Selecting $m > 3$ agents may further boost the accuracy score, but it will harm the efficiency. Unless otherwise specified, all experiments involving A4VL are conducted on six H200 GPUs.

3.2. Results

We compare A4VL against two closed-source MLLMs, GPT-4o [25] and Gemini 1.5 Pro [37], 16 open-source MLLMs, and 10 agent-based or long-video-oriented methods. As shown in Table 1, A4VL outperforms all baselines on every benchmark. On NeXT-QA, accuracy saturates near 80% as the backbone size increases; the best baseline, InternVL3-78B, reaches 84.0%, while A4VL further improves to 85.1%. On EgoSchema, InternVL3-78B achieves 76.8%, and A4VL is the only method exceeding 80%, with 82.2% (+3.2). LongVideoBench is where we observe the largest gain: compared to GPT-4o at 66.7%, A4VL (built solely from open-source models) reaches 72.2% (+5.5). MLVU-Test is substantially harder, with more answer choices and more complex question types (e.g., tracking specific players in long sports videos), yet A4VL still improves over prior methods by 1.9–19.6 points. On Video-MME, A4VL surpasses all baselines across all duration groups and on the overall average. Notably, although the strongest single MLLM backbone reaches at most 67.1% without subtitles, A4VL attains 77.2% on average, 2.2 points higher than the best closed-source model, Gemini 1.5 Pro (75.0%). The models selected by our agent teaming step for each benchmark are listed in Table 2.

Table 3 reports the average inference time per sample for four representative methods—the closed-source MLLM GPT-4o, the strong open-source MLLM InternVL3-78B, and two well-known agent-based systems, VideoAgent and TraveLER—as well as our A4VL. We measure efficiency on NeXT-QA, EgoSchema, and MLVU, which correspond to short-, medium-, and long-video settings. On short videos (NeXT-QA), all methods except TraveLER exhibit similar and acceptable latency. As video length increases, however, the inference time of other methods grows rapidly; for example, on MLVU, even GPT-4o takes over two minutes to answer a single question, whereas A4VL scales

Table 1. A4VL compared with other methods on five benchmarks. For Video-MME, the result is in format w. subtitle/wo. subtitle. For Video-MME, the improvement row shows the improvement without subtitle, since A4VL is mainly designed to optimize visual reasoning. * near the model names mean we take the results from the model paper or benchmark leaderboard.

Model	NeXT-QA	EgoSchema	LongVideoBench	MLVU	Video-MME			
					Short	Medium	Long	Average
GPT-4o* [25]	-	72.2	66.7	54.9	80.0/82.8	70.3/76.6	65.3/72.1	71.9/77.2
Gemini 1.5Pro* [37]	-	71.1	64.0	-	81.7/84.5	74.3/81.0	67.4/77.4	75.0/81.3
LLaVA-Video-7B-Qwen2 [54]	81.2	61.8	61.1	51.7	72.4/75.6	58.1/63.3	50.7/60.4	60.4/66.4
VideoLlama3-7B [49]	67.4	52.6	52.5	43.9	80.1/80.2	63.7/69.6	54.9/61.0	66.2/70.3
InternVL3.5-8B [41]	76.2	62.0	53.6	47.1	69.6/74.0	56.3/63.9	47.7/61.9	57.9/66.6
LLaVA-NeXT-Video-7B [53]	39.7	39.6	43.5	26.3	36.4/40.1	30.8/38.0	27.9/17.2	31.7/31.8
QwenVL-2.5-7B [38]	76.8	60.7	54.7	47.7	72.1/73.6	55.7/63.9	48.2/60.9	58.7/66.1
LongVA-7B* [51]	69.3	-	51.3	41.1	61.1/61.6	50.4/53.6	46.2/47.6	52.6/54.3
LongVU-7B* [33]	-	37.6	-	-	-	-	-/59.5	-/60.6
VideoChat2-7B* [17]	61.7	56.7	39.3	30.1	48.3/52.8	37.0/39.4	33.2/39.2	39.5/43.8
LLaVA-Video-72B-Qwen2 [54]	83.7	69.4	60.7	51.5	77.8/80.0	64.9/70.8	58.6/73.0	67.1/74.6
InternVL3.5-38B [41]	78.9	75.4	57.0	56.1	76.1/79.1	58.6/68.9	57.4/68.3	64.0/72.1
InternVL3-78B [41]	84.0	76.8	56.4	55.3	78.6/80.4	65.4/73.4	56.7/67.8	66.9/73.9
LLaVA-NeXT-Video-34B [53]	70.1	46.6	50.5	39.5	63.1/66.4	51.1/53.2	44.6/48.7	52.5/56.0
QwenVL-2.5-32B [38]	76.8	64.8	53.3	48.5	73.2/77.1	61.6/68.2	54.1/68.6	63.0/71.3
QwenVL-2.5-72B [38]	79.6	70.7	56.1	53.7	75.1/77.0	66.3/73.3	59.6/70.8	67.0/73.7
LLaVA-OneVision-72B* [15]	80.2	62.0	63.2	-	76.7/79.3	62.2/66.9	66.0/62.4	66.3/69.6
Aria-25.3B* [16]	-	-	63.0	-	67.9/78.3	67.0/71.7	58.8/66.3	67.6/72.1
ViperGPT [36]	56.9	24.8	-	24.9	-	-	-	-
SeViLA [19]	64.0	28.4	-	31.7	-	-	-	-
VideoAgent [5]	66.1	55.0	-	34.3	-	-		-
TraveLER [32]	66.0	55.4	-	36.3	-	-	-	-
MoReVQA* [24]	69.2	51.7	-	-	-	-	-	-
VideoRAG-72B* [21]	-	-	65.4	-	81.1/-	72.9/-	73.1/-	75.7/-
AuroraLong* [44]	-	-	-	52.7	-	-	-	-
DYTO-34B* [55]	72.9	56.8	-	-	-	-	-	-/53.4
BOLT* [20]	79.5	64.0	59.6	-	70.1/-	66.0/-	49.6/-	59.9/-
DynFocus* [8]	-	-	31.8	49.6	50.9/53.9	43.7/46.0	37.7/43.6	44.1/47.8
LVAgent [2]	83.0	78.4	66.9	50.0	82.4/83.6	72.8/78.6	66.3/76.4	73.9/79.5
A4VL	**85.1**	**82.2**	**72.2**	**58.0**	**86.6/87.3**	**76.8/83.2**	**68.3/77.9**	**77.2/82.8**
Improvement↑	**1.9 ∼ 45.4**	**3.8 ∼ 57.4**	**5.3 ∼ 41.6**	**1.9 ∼ 19.6**	**4.2 ∼ 20.2**	**2.5 ∼ 46.0**	**0.9 ∼ 40.4**	**2.2 ∼ 45.5**

Table 2. Models used for each benchmark. Intern-78B refers to InternVL3-78B [41], Intern-38B refers to InternVL3.5-38B [41], QwenVL-72B refers to QwenVL-2.5-72B [38], and LLaVA-72B refers to LLaVA-Video-72B-Qwen2 [54].

Benchmark	Models		
NeXT-QA	Intern-78B	Intern-38B	QwenVL-72B
EgoSchema	Intern-78B	Intern-38B	QwenVL-72B
LongVideoBench	Intern-78B	Intern-38B	QwenVL-72B
MLVU	Intern-78B	Intern-38B	LLaVA-72B
Video-MME	Intern-78B	LLaVA-72B	QwenVL-72B

Table 3. Average inference time per sample on four representative methods and A4VL.

Model	NeXT-QA	EgoSchema	MLVU
GPT-4o	23s	54s	127s
InternVL3-78B	15s	50s	204s
VideoAgent	20s	83s	175s
TraveLER	101s	94s	450s
A4VL	18s	37s	74s

much more gracefully. For TraveLER, the inference time on EgoSchema is lower because its official implementation uses image captions only to extract visual information on this dataset, yet A4VL still requires about 60% less time than TraveLER.

3.3. Ablation Study

Since multi-round collaboration is a core component of A4VL, we first study how the maximum number of rounds affects accuracy. In our default implementation, A4VL can run at most three rounds because there are three agents in total. In this ablation, we stop the procedure after a fixed maximum round and take the majority answer at that round as the final prediction; if there are multiple majorities, we choose the answer with the highest score. Figure 5 shows how accuracy changes as the maximum number of rounds increases. On all benchmarks, accuracy consistently improves when more rounds are allowed. Table 4 reports the actual number of rounds used in the default A4VL setting. As the dataset becomes harder, agents tend to collaborate for more rounds. Even within Video-MME, when subtitles are available the task becomes easier and thus requires fewer rounds.

We also test several design choices discussed in Section 2, using EgoSchema as the testbed. We first vary the sampling strategies. For perception-frame sampling p_1, we

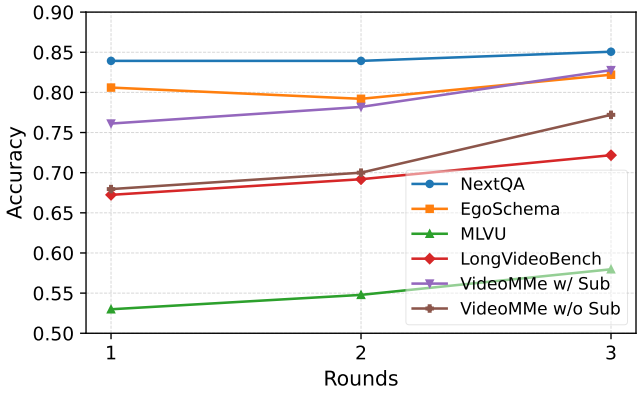

Figure 5. Relationship between accuracy and maximum number of rounds.

Table 4. Number of actual running rounds in A4VL. The data of Video-MME is shown in w/o subtitle / w. subtitle format.

#Rounds	1	2	3
NeXT-QA	1072	230	191
EgoSchema	303	101	96
LongVideoBench	566	341	430
MLVU	183	116	203
Video-MME	1289/1667	557/421	854/612

either perform **r**andom sampling over the entire video (R) or sample one frame from each **e**vent-based block (E). For action-frame sampling p_2, we either sample from uniformly divided blocks (R) or from event-based blocks (E). We evaluate three variants: RRSampling, RESampling, and ERSampling, where the first character denotes p_1 and the second denotes p_2. A4VL corresponds to RESampling. The accuracy and latency of each variant are shown in Table 5. Among all strategies, RESampling achieves the best accuracy with similar latency, so we adopt it as our default. Intuitively, perception exploration benefits from coarse, uniformly distributed frames to cover the whole video, while action exploration benefits from frames concentrated on active events. From the table, we can also know the event-based partitioning only takes about 2 seconds.

Table 5. Accuracy and latency of different sampling strategies on EgoSchema. R means random sampling and E means sampling from event-based blocks. The first character represents the sampling strategy for N_1 perception frames, and the second character represents the sampling strategy for N_2 action frames.

Method	RRSampling	RESampling	ERSampling
Accuracy (%)	80.2	82.2	79.6
Time/Sample	35s	37s	37s

Next, we compare different consensus criteria. Table 6 shows results on EgoSchema under majority consensus (MConsens) and full consensus (FConsens), as defined in Section 2.3. Both standards yield strong accuracy; FConsens has an accuracy of 82.2%, which is better than MConsensus (81.4%) but incurs higher (still moderate) latency. The reason is FConsens benefits from increasing collaboration rounds to make sure the final answer is strongly confident. Since the additional latency is acceptable, our default method uses FConsens.

Table 6. Results of different consensus conditions on EgoSchema. M means majority and F means full.

Method	MConsens	FConsens
Accuracy (%)	81.4	82.2
Time/Sample	26s	37s

Finally, we test whether agent pruning is necessary. We design two variants without pruning. **NoPruneSum** selects the answer with the highest total score, and **NoPruneMaj** selects the majority answer as the final prediction (breaking ties by total score). For both variants, the early stop condition is either (i) all agents in one round generates the same answer (FConsens) or (ii) in two subsequent rounds, the highest score corresponds to the same answer. As shown in Table 7, no matter how we choose the final answer, removing pruning leads to lower accuracy and substantially higher latency, indicating that pruning is important for both effectiveness and efficiency.

Table 7. Results of different pruning strategies on EgoSchema.

Method	NoPruneSum	NoPruneMaj	A4VL
Accuracy (%)	80.8	79.4	82.2
Time/Sample	60s	60s	37s

4. Conclusion

We presented A4VL, a training-free multi-agent framework that tightly couples perception exploration with action exploration for long video question answering. By combining event-aware block partitioning, clue-guided frame selection, and multi-round agent alliance with consensus and pruning, A4VL turns a pool of heterogeneous MLLMs into a coordinated reasoner over long videos under fixed compute budgets. Across five different benchmarks, A4VL consistently improves over strong baselines and prior agent-based systems, and our round-level ablations show stable gains from each collaboration round.

Looking ahead, we believe that perception–action coordination at the agent level is a promising direction for long-video understanding. Extending A4VL to audio–text–video tri-modal inputs, richer task-conditioned similarity functions, and neuro-symbolic techniques are natural next steps for enhanced perception exploration toward high performance general long-horizon video agents.

Acknowledgement. This research is partially sponsored by NSF CISE grants 2302720 and 2312758, a grant from CISCO Edge AI programs, and PACE at the Georgia Institute of Technology. The first and the last author are the primary contacts for this work.

References

[1] Xiaoyi Bao, Chenwei Xie, Hao Tang, Tingyu Weng, Xiaofeng Wang, Yun Zheng, and Xingang Wang. Dynimg: Key frames with visual prompts are good representation for multi-modal video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23678–23688, 2025. 2, 18

[2] Boyu Chen, Zhengrong Yue, Siran Chen, Zikang Wang, Yang Liu, Peng Li, and Yali Wang. Lvagent: Long video understanding by multi-round dynamical collaboration of mllm agents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20237–20246, 2025. 2, 7

[3] Siran Chen, Qinglin Xu, Yue Ma, Yu Qiao, and Yali Wang. Attentive snippet prompting for video retrieval. *IEEE Transactions on Multimedia*, 26:4348–4359, 2023. 4

[4] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 2, 18

[5] 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*, pages 75–92. Springer, 2024. 2, 7, 18

[6] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *2000 ieee international conference on multimedia and expo. icme2000. proceedings. latest advances in the fast changing world of multimedia (cat. no. 00th8532)*, pages 452–455. IEEE, 2000. 5, 16

[7] Chaoyou Fu, Yuhan Dai, Yondong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*, 2024. 2, 6, 18

[8] Yudong Han, Qingpei Guo, Liyuan Pan, Liu Liu, Yu Guan, and Ming Yang. Dynfocus: Dynamic cooperative network empowers llms with video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 8512–8522, 2025. 2, 7

[9] Kai Hu, Feng Gao, Xiaohan Nie, Peng Zhou, Son Tran, Tal Neiman, Lingyun Wang, Mubarak Shah, Raffay Hamid, Bing Yin, et al. M-llm based video frame selection for efficient video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13702–13712, 2025. 2, 18

[10] Sihao Hu, Tiansheng Huang, Gaowen Liu, Ramana Rao Kompella, Fatih Ilhan, Selim Furkan Tekin, Yichang Xu, Zachary Yahn, and Ling Liu. A survey on large language model-based game agents. *arXiv preprint arXiv:2404.02039*, 2024. 2

[11] Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2758–2766, 2017. 1, 17

[12] Rebecca Killick, Paul Fearnhead, and Idris A Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107 (500):1590–1598, 2012. 5, 16

[13] Bruno Korbar, Yongqin Xian, Alessio Tonioni, Andrew Zisserman, and Federico Tombari. Text-conditioned resampler for long form video understanding. In *European Conference on Computer Vision*, pages 271–288. Springer, 2024. 2, 18

[14] Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L Berg. Tvqa: Localized, compositional video question answering. *arXiv preprint arXiv:1809.01696*, 2018. 1, 17

[15] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer, 2024. 1, 7, 18

[16] Dongxu Li, Yudong Liu, Haoning Wu, Yue Wang, Zhiqi Shen, Bowen Qu, Xinyao Niu, Fan Zhou, Chengen Huang, Yanpeng Li, et al. Aria: An open multimodal native mixture-of-experts model. *arXiv preprint arXiv:2410.05993*, 2024. 7, 18

[17] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206, 2024. 1, 7, 18

[18] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *European Conference on Computer Vision*, pages 323–340. Springer, 2024. 2, 18

[19] Runze Liu, Yaqun Fang, Fan Yu, Ruiqi Tian, Tongwei Ren, and Gangshan Wu. Deep video understanding with video-language model. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 9551–9555, 2023. 7

[20] Shuming Liu, Chen Zhao, Tianqi Xu, and Bernard Ghanem. Bolt: Boost large vision-language model without training for long-form video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3318–3327, 2025. 1, 2, 7, 18

[21] Yongdong Luo, Xiawu Zheng, Xiao Yang, Guilin Li, Haojia Lin, Jinfa Huang, Jiayi Ji, Fei Chao, Jiebo Luo, and Rongrong Ji. Video-rag: Visually-aligned retrieval-augmented long video comprehension. *arXiv preprint arXiv:2411.13093*, 2024. 7, 18

[22] 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 Computer Vision and Pattern Recognition Conference*, pages 18936–18946, 2025. 2, 18

[23] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. *Advances in Neural Information Processing Systems*, 36:46212–46244, 2023. 2, 6, 18

[24] Juhong Min, Shyamal Buch, Arsha Nagrani, Minsu Cho, and Cordelia Schmid. Morevqa: Exploring modular reasoning models for video question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13235–13245, 2024. 2, 7, 18

[25] OpenAI. Gpt-4o system card, 2024. Accessed: 2025-11-04. 2, 6, 7, 18

[26] OpenAI. Gpt-5 system card. OpenAI System Card, 2025. 2

[27] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 5, 15

[28] Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. Category-specific video summarization. In *European conference on computer vision*, pages 540–555. Springer, 2014. 5, 17

[29] Tianyuan Qu, Longxiang Tang, Bohao Peng, Senqiao Yang, Bei Yu, and Jiaya Jia. Does your vision-language model get lost in the long video sampling dilemma? *arXiv preprint arXiv:2503.12496*, 2025. 1

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 2

[31] Saul Santos, António Farinhas, Daniel C McNamee, and Andre Martins. ∞-video: A training-free approach to long video understanding via continuous-time memory consolidation. In *International Conference on Machine Learning*, pages 52877–52893. PMLR, 2025. 2, 18

[32] Chuyi Shang, Amos You, Sanjay Subramanian, Trevor Darrell, and Roei Herzig. Traveler: A modular multi-lmm agent framework for video question-answering. *arXiv preprint arXiv:2404.01476*, 2024. 7, 18

[33] Xiaoqian Shen, Yunyang Xiong, Changsheng Zhao, Lemeng Wu, Jun Chen, Chenchen Zhu, Zechun Liu, Fanyi Xiao, Balakrishnan Varadarajan, Florian Bordes, Zhuang Liu, Hu Xu, Hyunwoo J. Kim, Bilge Soran, Raghuraman Krishnamoorthi, Mohamed Elhoseiny, and Vikas Chandra. Longvu: Spatiotemporal adaptive compression for long video-language understanding. *arXiv:2410.17434*, 2024. 1, 7, 18

[34] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, et al. Moviechat: From dense token to sparse memory for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18221–18232, 2024. 2, 18

[35] Yucheng Suo, Fan Ma, Linchao Zhu, Tianyi Wang, Fengyun Rao, and Yi Yang. From trial to triumph: Advancing long video understanding via visual context sample scaling and self-reward alignment. *arXiv preprint arXiv:2503.20472*, 2025. 2, 18

[36] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11888–11898, 2023. 7, 18

[37] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. 1, 6, 7, 18

[38] Qwen Team. Qwen2.5-vl, 2025. 1, 6, 7, 13, 18

[39] Qwen Team. Qwen3-max: Just scale it, 2025. 2, 18

[40] Lucas Ventura, Antoine Yang, Cordelia Schmid, and Gül Varol. Chapter-llama: Efficient chaptering in hour-long videos with llms. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18947–18958, 2025. 2, 18

[41] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025. 1, 6, 7, 12, 13, 18

[42] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding, 2024. 6, 18

[43] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9777–9786, 2021. 2, 6

[44] Weili Xu, Enxin Song, Wenhao Chai, Xuexiang Wen, Tian Ye, and Gaoang Wang. Auroralong: Bringing rnns back to efficient open-ended video understanding. *arXiv preprint arXiv:2507.02591*, 2025. 2, 7, 18

[45] Yichang Xu, Gaowen Liu, Ramana Rao Kompella, Sihao Hu, Fatih Ilhan, Selim Furkan Tekin, Zachary Yahn, and Ling Liu. A neurosymbolic agent system for compositional visual reasoning, 2025. 2

[46] Hongyang Xue, Zhou Zhao, and Deng Cai. Unifying the video and question attentions for open-ended video question answering. *IEEE Transactions on Image Processing*, 26(12): 5656–5666, 2017. 1, 17

[47] Zeyuan Yang, Delin Chen, Xueyang Yu, Maohao Shen, and Chuang Gan. Vca: Video curious agent for long video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20168–20179, 2025. 2, 18

[48] Jinhui Ye, Zihan Wang, Haosen Sun, Keshigeyan Chandrasegaran, Zane Durante, Cristobal Eyzaguirre, Yonatan Bisk, Juan Carlos Niebles, Ehsan Adeli, Li Fei-Fei, et al. Rethinking temporal search for long-form video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 8579–8591, 2025. 2

[49] Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang,

Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025. 1, 7

[50] Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025. 2, 18

[51] Peiyuan Zhang, Kaichen Zhang, Bo Li, Guangtao Zeng, Jingkang Yang, Yuanhan Zhang, Ziyue Wang, Haoran Tan, Chunyuan Li, and Ziwei Liu. Long context transfer from language to vision. *arXiv preprint arXiv:2406.16852*, 2024. 1, 7, 18

[52] Xiaoyi Zhang, Zhaoyang Jia, Zongyu Guo, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Deep video discovery: Agentic search with tool use for long-form video understanding. *arXiv preprint arXiv:2505.18079*, 2025. 2, 18

[53] Yuanhan Zhang, Bo Li, haotian Liu, Yong jae Lee, Liangke Gui, Di Fu, Jiashi Feng, Ziwei Liu, and Chunyuan Li. Llava-next: A strong zero-shot video understanding model, 2024. 1, 7, 18

[54] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data, 2024. 1, 6, 7, 13

[55] Yiming Zhang, Zhuokai Zhao, Zhaorun Chen, Zenghui Ding, Xianjun Yang, and Yining Sun. Beyond training: Dynamic token merging for zero-shot video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22046–22055, 2025. 2, 7, 18

[56] Zhou Zhao, Jinghao Lin, Xinghua Jiang, Deng Cai, Xiaofei He, and Yueting Zhuang. Video question answering via hierarchical dual-level attention network learning. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1050–1058, 2017. 1, 17

[57] Zhou Zhao, Qifan Yang, Deng Cai, Xiaofei He, Yueting Zhuang, Zhou Zhao, Qifan Yang, Deng Cai, Xiaofei He, and Yueting Zhuang. Video question answering via hierarchical spatio-temporal attention networks. In *IJCAI*, page 8, 2017. 1, 17

[58] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Shitao Xiao, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. Mlvu: A comprehensive benchmark for multi-task long video understanding. *arXiv preprint arXiv:2406.04264*, 2024. 6, 18

[59] Linchao Zhu, Zhongwen Xu, Yi Yang, and Alexander G Hauptmann. Uncovering the temporal context for video question answering. *International Journal of Computer Vision*, 124(3):409–421, 2017. 1, 17

A. A4VL Algorithm

The detailed algorithm is shown in Algorithm 1. First, the video is partitioned to at most B blocks. Suppose the event-based partitioning algorithm finally divides the video into b ($\leq B$) blocks. In case it divides the video into more than B blocks, we simply merge similar blocks until there are B blocks in total. Line 4-9 implements sample-clue based exploration, where the perception clues are generated. Line 10-19 performs block-based perception sampling. Then action exploration step starts from line 20. From line 20-26, each agent generates the answer and the reason, and a consensus check is made to decide whether we early exit. Line 27-34 performs the agent pruning with multi-round deliberation. The worst agent is pruned and remaining agents revise their perception clue for the next round.

B. Additional Visualizations

Figure 6 shows the result of event-based partitioning on two cases. Segments are annotated by colored blocks below the video frames. The top case is a video from NeXT-QA, which lasts for 25 seconds. Our event-based partitioning algorithm accurately divides the video into two blocks: in the first block, the girl keeps talking, and in the second block, the girl takes out a paper. The bottom example is from EgoSchema, which lasts for three minutes. Our partitioning algorithm divides the video into six blocks, where the person alternates between reading the book, using the phone, taking out a pen, and looking around. The entire procedure is training free and query agnostic. Figure 7

Figure 6. Visualization of event-based video partitioning.

presents a case in which the majority answer (B) in the first round is incorrect, yet A4VL ultimately returns the correct answer in the final round, supported by a sound explanation from our summarizer. In this example, the green agent corresponds to InternVL3.5-38B [41], the orange agent to InternVL3-78B [41], and the pink agent to QwenVL-2.5-

Algorithm 1: A4VL Algorithm.

Input: Video $V = \{v_1, \ldots, v_n\}$, question Q, options O, agent pool $\mathbb{A} = \{A_i : 1 \leq i \leq m\}$, preview frames N_1, inference frames N_2, max number of blocks B.

Output: Answer $\mathcal{A}$.

1 $\{B_1, \ldots, B_b\} \leftarrow \text{EventPartition}(V, B)$
2 $j \leftarrow 1$
3 **while** $|\mathbb{A}| \geq 1$ **do**
4 **foreach** $A_i \in \mathbb{A}$ **do** Step 1.1: Sample-Clue Based Exploration
5 **if** $j = 1$ **then**
6 $\hat{v}_{A_i} \leftarrow p_1(V, N_1)$
7 $P_{i,1} \leftarrow A_{i,\text{clue}}(\hat{v}_{A_i}, Q, O)$
8 **else**
9 $P_{i,j}$ given
10 **foreach** $A_i \in \mathbb{A}$ **do** Step 1.2: Block-Based Perception Sampling
11 $\mathbf{s}^{(i)} \leftarrow \{Sim(B_k, P_{i,j}) | 1 \leq k \leq b\}$
12 **if** $\max(\mathbf{s}^{(i)}) > \rho$ **then**
13 $\mathbf{s}^{(i)}[\mathbf{s}^{(i)} \leq \rho] = -\infty$
14 $\mathbf{s}^{(i)} \leftarrow \mathbf{s}^{(i)} - \max(\mathbf{s}^{(i)})$
15 $\mathbf{c}^{(i)} \leftarrow N_2 \lfloor SoftMax(\mathbf{s}^{(i)}) \rfloor$
16 $V_{\text{act}}^{(i)} \leftarrow \bigcup_{k=1}^{b} p_2(B_k, c_k^{(i)})$
17 **else**
18 $k^* \leftarrow \arg\max_{k \in \{1,\ldots,b\}} s_k^{(i)}$
19 $V_{\text{act}}^{(i)} \leftarrow p_2(B_{k^*}, N_2)$
20 **foreach** $A_i \in \mathbb{A}$ **do** Step 2.1: Generating Answer & Reason
21 $a_{i,j} \leftarrow A_{i,\text{act}}(V_{\text{act}}^{(i)}, Q, O)$
22 $R_{i,j} \leftarrow A_{i,\text{reason}}(V_{\text{act}}^{(i)}, a_{i,j}, Q)$
23 $S_{a,j} \leftarrow \{a_{i,j} \mid A_i \in \mathbb{A}\}$
24 $S_{r,j} \leftarrow \{R_{i,j} \mid A_i \in \mathbb{A}\}$
25 **if** $\forall i, i' : a_{i,j} = a_{i',j}$ **then**
26 **return** $\mathcal{A} \leftarrow a_{1,j}$
27 **foreach** $A_i \in \mathbb{A}$ **do** Step 2.2: Pruning with Deliberation
28 $(s_{i,\mathbb{A}_1}, \ldots, s_{i,\mathbb{A}_{|\mathbb{A}|}}) \leftarrow A_{i,\text{eval}}(Q, S_{a,j}, S_{r,j})$
29 **foreach** $A_i \in \mathbb{A}$ **do**
30 $s_{A_i} \leftarrow \sum_{A_k \in \mathbb{A}} s_{k,\mathbb{A}_i}$
31 $A_{\min} \leftarrow \arg\min_{A \in \mathbb{A}} s_A$
32 $\mathbb{A} \leftarrow \mathbb{A} \setminus \{A_{\min}\}$
33 **foreach** $A_i \in \mathbb{A}$ **do**
34 $P_{i,j+1} \leftarrow A_{i,\text{refine}}(P_{i,j}, S_{a,j}, S_{r,j}, A_{\min}, Q, O)$
35 $j \leftarrow j + 1$
36 **return** $\mathcal{A} \leftarrow a_{i^\star,j-1}$ // $A_{i^\star}$ is the last remaining agent

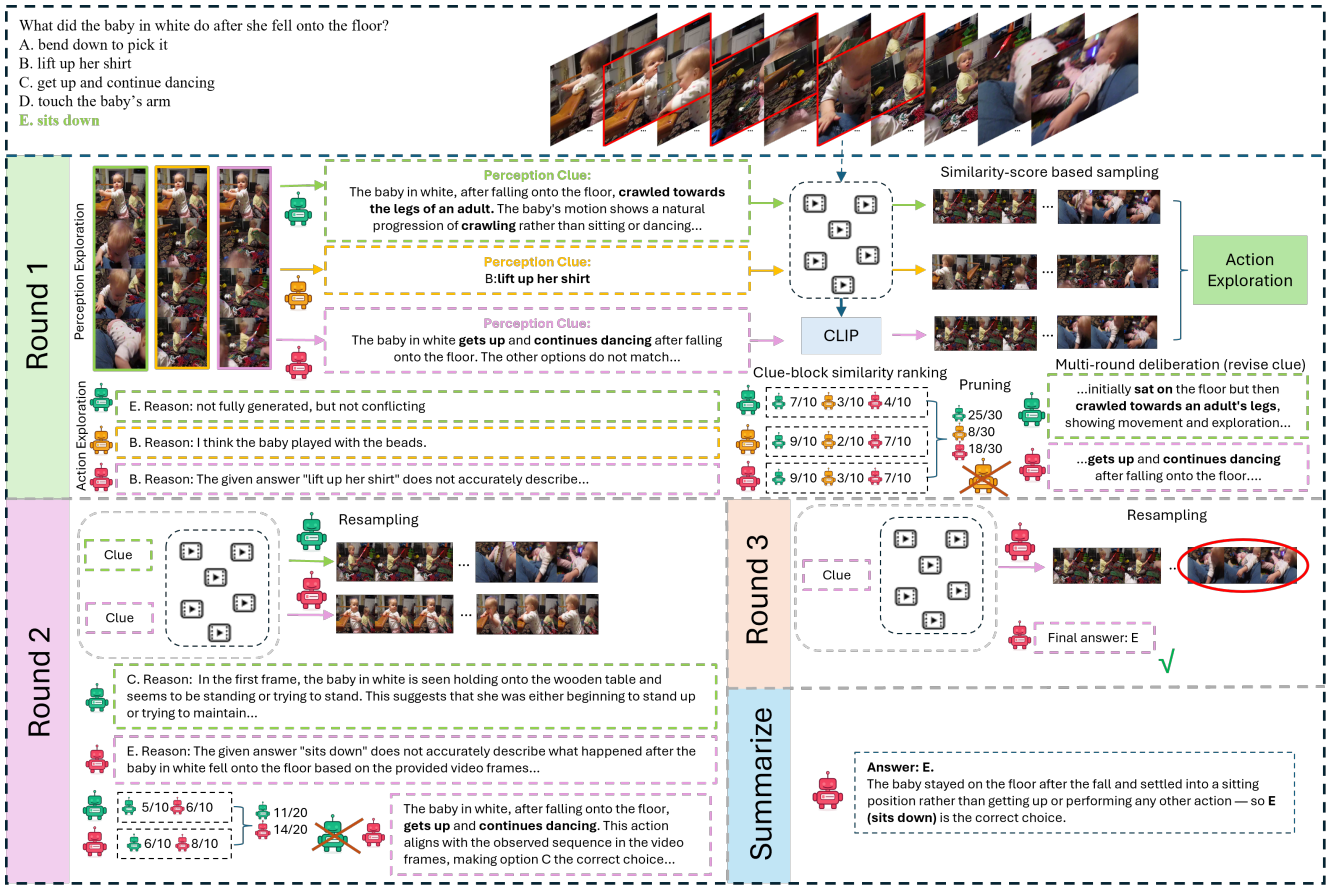

Figure 7. Example where the majority is wrong in the first round, but finally A4VL produces the correct answer. The example is taken from NeXT-QA.

72B [38]. We observe that, in the first round, only the green agent produces the correct answer, while the other two receive significantly lower scores. Notably, the green agent is not the model with the largest number of parameters. However, it exhibits instability and becomes incorrect in the second round, resulting in its removal. In contrast, the pink agent becomes correct in the second round and remains correct through the third round, leading to the correct final answer. These visualizations complement Figure 2 to 4 in the main paper by illustrating how event-based partitioning and multi-round collaboration behave on real videos.

C. Failure Cases

Although A4VL demonstrates superior performance over existing models, it can still fail in certain cases. For example, when all agents make an incorrect prediction in the first round, the early-exit mechanism forces the final answer to be wrong. Figure 8 shows such a failure case: the final answer is incorrectly chosen as option D because all agents in the first round output D. This error arises from several factors: (i) it is difficult for the perception frames to accurately capture the moments after the baby smiles; (ii) although the green agent generates the perception clue "lay still," CLIP is an image–text similarity model and thus struggles to capture temporal movements such as "still," leading to the selection of an incorrect block, while the other two agents, whose perception clues are themselves incorrect, also choose wrong blocks; and (iii) the question references the video position "at the end," but CLIP cannot measure the temporal alignment between frames and text. Addressing such cases is a challenging but important direction for future work, and potential remedies include integrating more advanced video grounding models and routing complex cases to these models when detected, as well as incorporating neuro-symbolic techniques for precise video processing, such as explicitly cropping the end segment of the video.

It is also possible that the majority vote in the first round is correct, while the final answer is wrong. As shown in Figure 9, we illustrate such a case from Video-MME (without subtitle). The green, orange, and pink agents represent InternVL3-78B [41], QwenVL-2.5-72B [38], and LLaVA-Video-72B-Qwen2 [54], respectively. This is a challenging

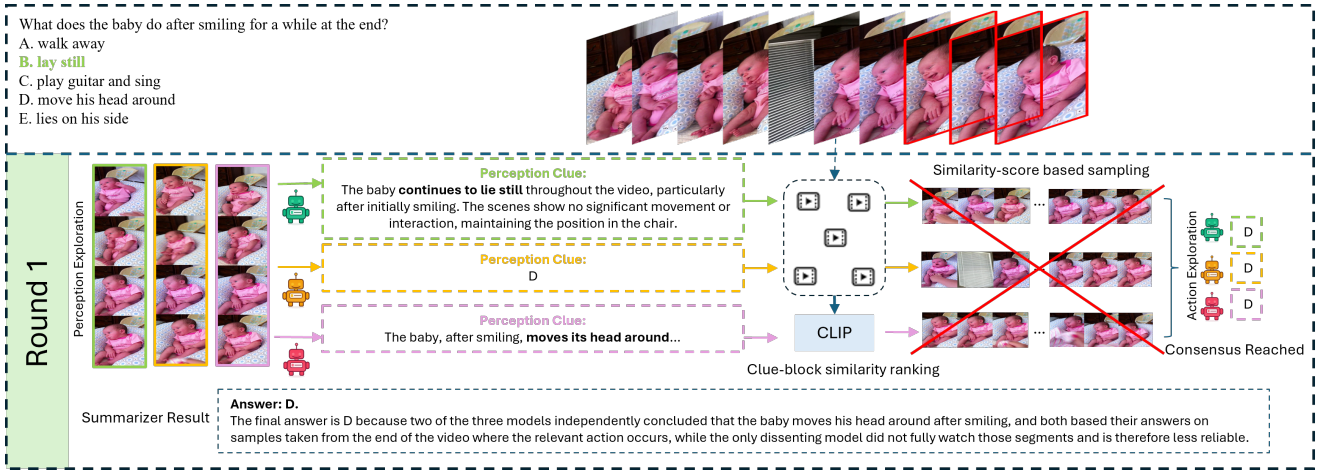

Figure 8. Failure case where all agents achieve a wrong consensus in the first round. The example is taken from NeXT-QA.

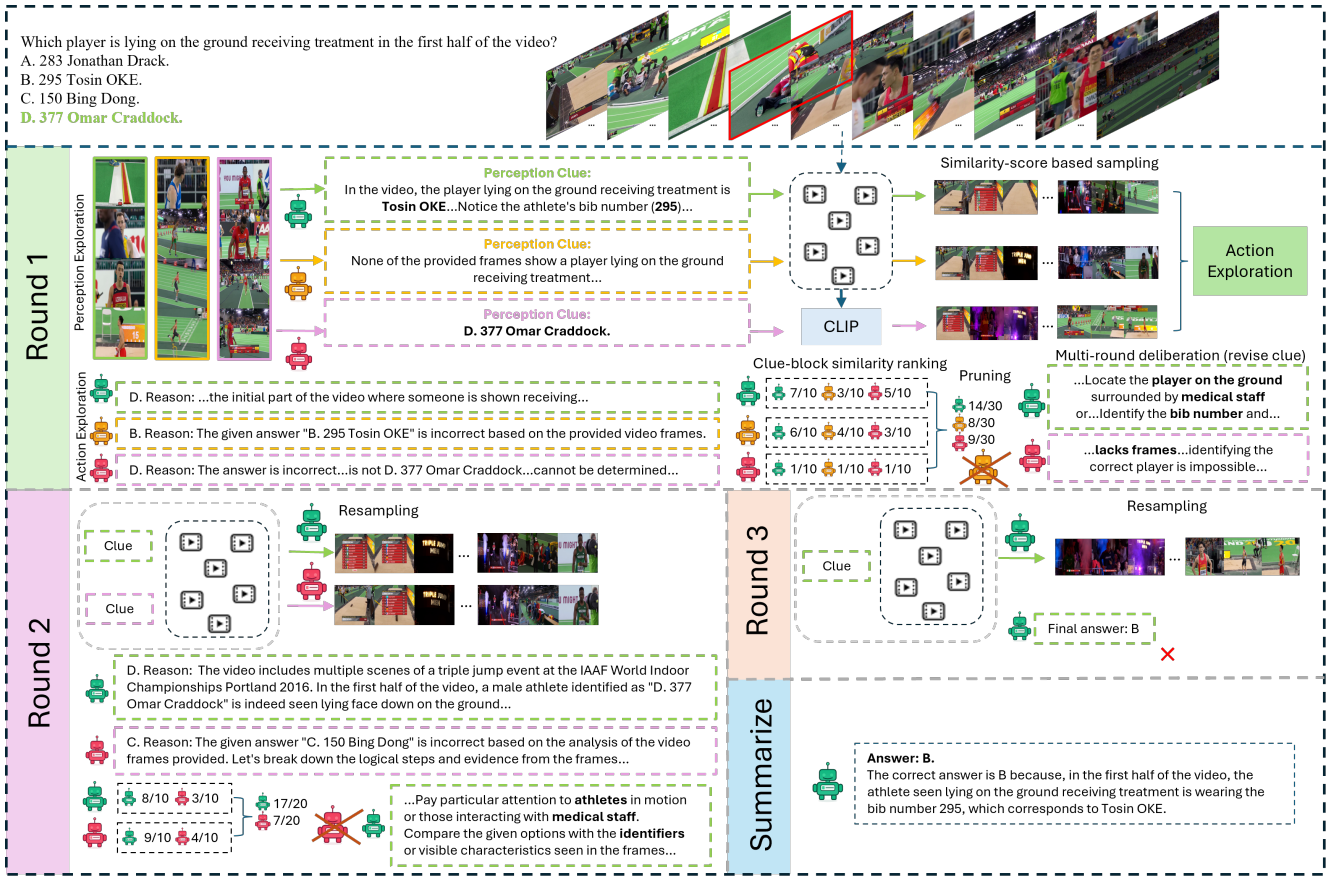

Figure 9. Failure case where the majority is correct in the first round, but finally the final answer gets wrong. This example is taken from Video-MME (w/o sub).

example because the entire video is about 20 minutes long, while the keyframes last only a few seconds. The perception frames provide no useful clues, and none of the sampled action frames contain the keyframe where a player is lying on the ground receiving treatment. As a result, each agent's behavior is highly unstable when reasoning over these irrelevant frames. In the first round, two agents choose answer D, likely because that player happens to appear most fre-

quently and is often centered in the view. However, their predictions shift in rounds 2 and 3, eventually converging on the wrong answer B. Another difficulty is that the question explicitly restricts the focus to the first half of the video. If this constraint were reliably detected and only the first half were used in the multi-round reasoning process (e.g., via the neural-symbolic approach mentioned earlier), the outcome might improve, though sampling the true keyframes would remain non-trivial. For such questions, an efficient framework based on fine-grained watching is therefore needed.

D. Event-Based Partitioning

Given a video V, our goal is to obtain a small set of temporal boundaries that partition V into at most B visually coherent blocks $\{B_1, \ldots, B_b\}$, where $b \leq B$. We follow a training-free, query-agnostic pipeline: (i) sampled decoding and feature extraction, (ii) construction of a fused novelty signal on time, (iii) multi-scale pooling and PELT-based segmentation on the 1D signal, and (iv) embedding-based heads (SSM and KTS) whose outputs are combined with non-maximum suppression (NMS) and a cap of at most $B-1$ cuts.

Sampled frames and notation. Let V have T frames $\{I_1, \ldots, I_T\}$ with timestamps $0 \leq t_1 < \cdots < t_T = L$. We uniformly sample at most F frames (in practice $F = 200$) and denote their indices by

$$\mathcal{S} = \{i_1 < i_2 < \cdots < i_N\}, \quad N \leq F,$$

with corresponding images I_{i_r} and timestamps t_{i_r}. For readability, we abbreviate $I_r := I_{i_r}$ and $t_r := t_{i_r}$ when referring to the sampled frames.

Color, sharpness, motion, and embeddings. For each sampled frame I_r, we compute four types of features.

HSV color histograms. we detect color changes across frames using histogram distances in HSV space. Specifically, we convert I_r from BGR to HSV, obtaining $H_r^{\text{img}}(x)$, $S_r^{\text{img}}(x)$, and $V_r^{\text{img}}(x)$ for pixel locations $x \in \Omega$ (image grid). We discretize each channel into K bins (we use $K = 32$) and compute histograms

$$h_r^H \in \mathbb{R}^K, \quad h_r^S \in \mathbb{R}^K, \quad h_r^V \in \mathbb{R}^K,$$

followed by a joint ℓ_1 normalization:

$$\tilde{h}_r^H = \frac{h_r^H}{S_r}, \quad \tilde{h}_r^S = \frac{h_r^S}{S_r}, \quad \tilde{h}_r^V = \frac{h_r^V}{S_r},$$

with

$$S_r = \sum_{k=1}^{K} h_r^H[k] + \sum_{k=1}^{K} h_r^S[k] + \sum_{k=1}^{K} h_r^V[k] + \varepsilon,$$

where $\varepsilon > 0$ is a small constant for numerical stability. The concatenated color descriptor is

$$c_r = \left[\tilde{h}_r^H; \tilde{h}_r^S; \tilde{h}_r^V\right] \in \mathbb{R}^{3K}.$$

Sharpness. we detect focus changes using the variance of a Laplacian filter. We convert I_r to gray-scale $G_r(x)$ and apply a discrete Laplacian filter $\Delta G_r(x)$ (e.g., the standard 3×3 kernel). The sharpness score is the variance of the Laplacian:

$$\mu_r^{\Delta} = \frac{1}{|\Omega|} \sum_{x \in \Omega} \Delta G_r(x), \qquad s_r = \frac{1}{|\Omega|} \sum_{x \in \Omega} \left(\Delta G_r(x) - \mu_r^{\Delta}\right)^2.$$

Motion. we measure frame-to-frame intensity differences as a proxy for motion magnitude. For $r \geq 2$, we define a robust frame-to-frame motion magnitude using the median absolute difference between consecutive gray-scale images:

$$d_r^{\text{mot}} = \text{median}_{x \in \Omega} \left|G_r(x) - G_{r-1}(x)\right|, \qquad m_r = \frac{1}{255}\, d_r^{\text{mot}},$$

and set $m_1 = 0$.

DINOv2 embeddings. we use cosine distance in feature space to capture semantic changes: we pass a resized RGB version of I_r through a DINOv2 encoder [27] and obtain a feature vector

$$e_r \in \mathbb{R}^D.$$

We use ℓ_2-normalized embeddings $\hat{e}_r = e_r / (\|e_r\|_2 + \varepsilon)$ when computing cosine similarities.

Per-step novelty from consecutive frames. We now define novelty cues on the temporal midpoints

$$\tau_r = \frac{t_{r-1} + t_r}{2}, \qquad r = 2, \ldots, N.$$

Color novelty. For each channel we use cosine distance between histograms; for hue, for example:

$$d_r^H = 1 - \cos\left(\tilde{h}_r^H, \tilde{h}_{r-1}^H\right) = 1 - \frac{\langle \tilde{h}_r^H, \tilde{h}_{r-1}^H \rangle}{\|\tilde{h}_r^H\|_2 \, \|\tilde{h}_{r-1}^H\|_2 + \varepsilon},$$

and analogously d_r^S and d_r^V. We combine them as

$$c_r^{\text{nov}} = \alpha\, d_r^H + \beta\, d_r^S + \gamma\, d_r^V.$$

where $\alpha = 0.55, \beta = 0.35$ and $\gamma = 0.10$ in our settings.

Sharpness and motion novelty. We use absolute temporal differences:

$$u_r^{\text{shp}} = \left|s_r - s_{r-1}\right|, \qquad u_r^{\text{mot}} = \left|m_r - m_{r-1}\right|.$$

Embedding novelty with EMA. We first compute a forward exponential moving average (EMA) of embeddings:

$$\tilde{e}_1 = e_1, \qquad \tilde{e}_r = \delta \tilde{e}_{r-1} + (1 - \delta) e_r,$$

with $\delta = 0.9$. For each $r \geq 2$ we define two cosine distances:

$$d_r^{\text{prev}} = 1 - \cos(\hat{e}_r, \hat{e}_{r-1}), \qquad d_r^{\text{ema}} = 1 - \cos(\hat{e}_r, \widehat{\bar{e}}_{r-1}),$$

where $\widehat{\bar{e}}_{r-1}$ is the ℓ_2-normalized EMA. The embedding novelty is

$$u_r^{\text{emb}} = \tfrac{1}{2} d_r^{\text{prev}} + \tfrac{1}{2} d_r^{\text{ema}}.$$

Robust normalization and fusion. For any scalar sequence x_r ($r = 2, \ldots, N$) we denote its *robust z-score* by

$$\text{med}(x) = \text{median}_r\, x_r,$$
$$\text{MAD}(x) = \text{median}_r\, \big|x_r - \text{med}(x)\big|,$$
$$z_r(x) = \frac{x_r - \text{med}(x)}{1.4826\ \text{MAD}(x) + \varepsilon}.$$

The constant $1.4826 = 1/\Phi^{-1}(0.75)$, where Φ is the standard normal CDF, so that $1.4826\,\text{MAD}$ is a consistent estimator of the standard deviation for Gaussian data.

We additionally apply a short moving average of window w (we use $w = 5$) to reduce noise:

$$\bar{z}_r(x) = \frac{1}{|\mathcal{W}_r|} \sum_{u \in \mathcal{W}_r} z_u(x),$$

where $\mathcal{W}_r = \{u : |u - r| \leq \lfloor w/2 \rfloor\}$ truncated at the sequence boundaries. We finally clip values to $[-4, 4]$.

We apply this procedure separately to the color, motion, sharpness, and embedding novelties:

$$z_r^{\text{col}} = \bar{z}_r\big(c^{\text{nov}}\big), \quad z_r^{\text{mot}} = \bar{z}_r\big(u^{\text{mot}}\big),$$
$$z_r^{\text{shp}} = \bar{z}_r\big(u^{\text{shp}}\big), \quad z_r^{\text{emb}} = \bar{z}_r\big(u^{\text{emb}}\big).$$

We then compute cue-specific empirical standard deviations

$$\sigma_{\text{col}} = \text{std}_r\left(z_r^{\text{col}}\right), \quad \sigma_{\text{mot}} = \text{std}_r\left(z_r^{\text{mot}}\right)$$
$$\sigma_{\text{emb}} = \text{std}_r\left(z_r^{\text{emb}}\right), \quad \sigma_{\text{shp}} = \text{std}_r\left(z_r^{\text{shp}}\right)$$

and combine them with fixed base weights

$$\beta_{\text{col}} = 0.20, \quad \beta_{\text{mot}} = 0.30, \quad \beta_{\text{emb}} = 0.35, \quad \beta_{\text{shp}} = 0.05.$$

The final fusion weights are

$$w_c = \frac{\beta_c \sigma_c}{\sum_{c' \in \{\text{col,mot,emb,shp}\}} \beta_{c'} \sigma_{c'}}.$$

The fused novelty signal on midpoints τ_r is

$$s(\tau_r) = w_{\text{col}}\, z_r^{\text{col}} + w_{\text{mot}}\, z_r^{\text{mot}} + w_{\text{emb}}\, z_r^{\text{emb}} + w_{\text{shp}}\, z_r^{\text{shp}}.$$

We apply another short moving average in time (again with window 5) to obtain a smooth 1D signal, which we still denote by $s(\tau_r)$.

Multi-scale pooling and PELT heads. To handle varying frame rates and video lengths, we pool $s(\tau_r)$ onto several temporal grids. For a grid resolution $\Delta > 0$ (we use $\Delta \in \{0.25, 0.5, 1.0\}$ seconds), we define bin edges

$$u_0 = \tau_2, \quad u_J = \tau_N, \quad u_j = u_0 + j\Delta,$$

and bins $[u_j, u_{j+1})$ for $j = 0, \ldots, J-1$. The pooled signal is

$$s_j^{(\Delta)} = \max\big\{s(\tau_r)\ :\ \tau_r \in [u_j, u_{j+1})\big\},$$

with bin centers $t_j^{(\Delta)} = (u_j + u_{j+1})/2$.

On each pooled sequence $\{s_j^{(\Delta)}\}_{j=0}^{J-1}$ we apply PELT [12] with a standard ℓ_2 cost and a penalty λ swept over a small range. Denoting by $\mathcal{K}^{(\Delta)}$ the predicted change-point indices for a given λ, the breakpoints correspond to bin centers $\{t_k^{(\Delta)} : k \in \mathcal{K}^{(\Delta)}\}$. We only keep candidates that yield segments of length at least

$$\ell_{\min} = \max\left(\ell_0,\ \frac{L}{15}\right),$$

where ℓ_0 is a user-defined minimum (e.g., 2 s). For each candidate index k we estimate a simple boundary strength via the difference of local averages:

$$\gamma_k^{(\Delta)} = \left| \frac{1}{W} \sum_{j=k}^{k+W-1} s_j^{(\Delta)} - \frac{1}{W} \sum_{j=k-W}^{k-1} s_j^{(\Delta)} \right|,$$

with a small window W (e.g., $W \approx \ell_{\min}/\Delta$). We discard boundaries with $\gamma_k^{(\Delta)}$ below a quantile threshold and collect the remaining breakpoints from all grid levels into a set $\mathcal{T}_{\text{PELT}}$.

Embedding-based SSM and KTS heads. We additionally operate directly on the embedding sequence. We first resample embeddings $\{e_r\}$ to a grid $\{\tilde{t}_q\}$ at roughly 1 Hz and obtain averaged embeddings $\tilde{e}_q$ in each bin.

SSM-based novelty. We L_2-normalize $\tilde{e}_q$ to $\hat{e}_q$ and build the Gram matrix

$$S_{pq} = \langle \hat{e}_p, \hat{e}_q \rangle.$$

We then follow Foote's self-similarity matrix (SSM) novelty detector [6]. For each center index c we extract a $(2w) \times (2w)$ block around (c, c), partition it into four $w \times w$ quadrants, and apply a Gaussian-weighted checkerboard kernel K with $+1$ on the top-left and bottom-right quadrants and -1 on the others:

$$n_c = \sum_{i,j} K_{ij}\, S_{(c+i-w),(c+j-w)}.$$

We robustly normalize the sequence $\{n_c\}$ via the same MAD-based z-score and keep local maxima separated by at

least $\ell_{\min}$ and above a quantile threshold. The corresponding times form $\mathcal{T}_{\text{SSM}}$. We set $\ell_{\min} = \max\{l, 4\}$, where l is the video length.

KTS-style segmentation. We also run a KTS-like greedy procedure [28]. Let E be the matrix whose rows are $\hat{e}_q$. We form the Gram matrix $G = EE^\top$ and its integral image to enable $O(1)$ evaluation of segment similarity. For a segment $[a, b]$, we define the cost

$$C(a,b) = -\frac{1}{(b-a+1)^2} \sum_{p=a}^{b} \sum_{q=a}^{b} G_{pq}.$$

Starting from the full interval $[0, Q-1]$, we recursively search for a split index k that maximizes the gain

$$\text{gain}(a,b,k) = C(a,b) - \big(C(a,k) + C(k+1,b)\big) - \lambda_{\text{KTS}},$$

with a data-dependent penalty λ_{KTS} (set to 1.4 in our experiment, under all benchmarks). If the best gain is positive and both resulting segments respect $\ell_{\min}$, we split and recurse. The resulting cut times form $\mathcal{T}_{\text{KTS}}$.

Merging, NMS, and truncation. We merge all candidate boundaries

$$\mathcal{T}_{\text{all}} = \mathcal{T}_{\text{PELT}} \cup \mathcal{T}_{\text{SSM}} \cup \mathcal{T}_{\text{KTS}},$$

sort them increasingly, and apply a one-dimensional non-maximum suppression with minimum separation $\ell_{\min}$. Concretely, we scan $\mathcal{T}_{\text{all}}$ in order and keep a candidate τ if $\tau - \tau_{\text{last}} \geq \ell_{\min}$, where τ_{last} is the time of the last kept boundary; otherwise we discard it. This yields a filtered set $\tilde{\mathcal{T}}$ and ensures that every resulting segment has duration at least $\ell_{\min}$.

If $|\tilde{\mathcal{T}}| > B-1$, we compute a boundary strength measure (e.g., $\gamma_k^{(\Delta)}$ on a fixed grid) for each $\tau \in \tilde{\mathcal{T}}$, rank boundaries by strength, and keep only the top $B-1$. Adding the implicit boundaries at 0 and L, we obtain

$$0 = \tau_0 < \tau_1 < \cdots < \tau_{b-1} < \tau_b = L,$$

which define the final blocks $B_k = V[\tau_{k-1}, \tau_k)$ used by A4VL.

The overall algorithm is shown in Algorithm 2.

E. Related Works

Video understanding. After the development of image QA, researchers began to explore video QA as a more complex multimodal understanding task. Pioneering works [11, 14, 46, 56, 57, 59] mainly adopt simple CNN–RNN style pipelines that encode pre-extracted frame features with LSTMs/GRUs over short clips, offering only limited modeling of long-range temporal structure and multimodal context. Modern methods begin to use LLM backbones for

Algorithm 2: Event-based video partitioning (EVENTPARTITION).

Input: Video V, max blocks B, max feature frames F, min segment length $\ell_{\min}$.
Output: Blocks $\{B_1, \ldots, B_b\}$ with $b \leq B$.

`// Step 1: sampled decode & feature extraction`

Uniformly sample at most F frames from V and record $\{I_r, t_r\}_{r=1}^{N}$;
for $r = 1$ **to** N **do**
  Compute $c_r, G_r, s_r, m_r, e_r, \hat{e}_r$ as in Appendix. D;

`// Step 2: per-step cues and fused novelty`

for $r = 2$ **to** N **do**
  Compute color novelty c_r^{nov}, $u_r^{\text{shp}} = |s_r - s_{r-1}|$, $u_r^{\text{mot}} = |m_r - m_{r-1}|$, and u_r^{emb} from embeddings (previous & EMA);
  Set midpoint $\tau_r = (t_{r-1} + t_r)/2$;
Apply MAD-based robust z-scoring and short moving average to obtain $z_r^{\text{col}}, z_r^{\text{mot}}, z_r^{\text{shp}}, z_r^{\text{emb}}$;
Compute fusion weights w_c and fused novelty $s(\tau_r) = \sum_c w_c z_r^{(c)}$, then smooth in time;

`// Step 3: PELT heads on pooled grids`

$\mathcal{T}_{\text{PELT}} \leftarrow \emptyset$;
foreach $\Delta \in \{0.25, 0.5, 1.0\}$ **do**
  Pool $s(\tau_r)$ into bins of width Δ (max pooling) to get $s_j^{(\Delta)}$;
  Run PELT on $\{s_j^{(\Delta)}\}_j$ with segment length $\geq \ell_{\min}$ and keep strong boundaries (via local strength $\gamma_k^{(\Delta)}$);
  Add resulting times to $\mathcal{T}_{\text{PELT}}$;

`// Step 4: embedding-based SSM and KTS heads`

Resample $\{e_r\}$ to a ~ 1 Hz grid $\{\tilde{e}_q, \tilde{t}_q\}$;
Compute SSM-based novelty on the Gram matrix of $\{\tilde{e}_q\}$ and keep strong peaks (gap $\geq \ell_{\min}$) to form $\mathcal{T}_{\text{SSM}}$;
Run KTS-style greedy segmentation on $\{\tilde{e}_q\}$ with penalty λ_{KTS} to obtain $\mathcal{T}_{\text{KTS}}$;

`// Step 5: merge, NMS, and truncation`

$\mathcal{T}_{\text{all}} \leftarrow \mathcal{T}_{\text{PELT}} \cup \mathcal{T}_{\text{SSM}} \cup \mathcal{T}_{\text{KTS}}$;
Sort $\mathcal{T}_{\text{all}}$ and apply 1D NMS with minimum gap $\ell_{\min}$ to obtain $\tilde{\mathcal{T}}$;
if $|\tilde{\mathcal{T}}| > B - 1$ **then**
  Rank boundaries by strength and keep the top $B-1$;
Let $\tau_0 = 0$, $\tau_b = L$, and $\{\tau_1, \ldots, \tau_{b-1}\} = \tilde{\mathcal{T}}$ in order;
return $B_k = V[\tau_{k-1}, \tau_k)$, $k = 1, \ldots, b$;

more powerful multimodal understanding. Representative multimodal LLMs such as GPT-4o [25], Gemini [4, 37], LLaVA-OneVision / LLaVA-NeXT-Video [15, 53], InternVL3.5 [41], Qwen2.5-VL / Qwen3-Max [38, 39], VideoLLaMA3 [50], and ARIA [16] couple strong language backbones with unified visual encoders, long-context modeling, and instruction tuning, substantially advancing open-ended image and long-form video understanding. Also, agent-based methods like VideoAgent [5], TraveLER [32] or MoReVQA [24] use multi-step reasoning for tasks with strong temporal or causal relationship.

Long video understanding. Recent benchmarks such as EgoSchema [23], LongVideoBench [42], MLVU [58], and Video-MME [7] push models to reason over minutes- to hour-long videos with complex queries and interleaved modalities, exposing the limitations of naive uniform sampling and single-pass decoding [7, 17, 23, 58]. To improve scalability, a line of work compresses or sparsifies the visual stream via token merging, adaptive compression, and content-aware resampling [1, 9, 13, 16, 18, 18, 51, 55], while others design specialized architectures or training objectives, e.g., RNN-based MLLMs and self-reward–aligned long-video learners [33, 35, 44]. Retrieval- and memory-based methods instead treat long videos as external knowledge to be indexed, using document retrieval, video-RAG, or sparse memory to focus reasoning on a small subset of salient clips [21, 22, 34, 42, 52]. Closest to ours are agentic or search-style systems that iteratively query long videos, such as VCA, MovieChat, Chapter-LLaMA, ∞-Video, and related multi-agent frameworks [5, 24, 31, 32, 34, 40, 47]. In contrast, A4VL is a training-free multi-agent alliance that combines event-based partitioning with clue-guided block selection and multi-round agent collaboration, and can be plugged on top of arbitrary MLLM backbones to improve long-video reasoning under strict frame budgets.

Agents in video understanding. Agent-based frameworks have recently been introduced for video understanding, where multiple specialized modules ("agents") collaborate via planning, tool use, and iterative querying. For short videos, systems such as ViperGPT [36], MoREVQA [24], and Traveler [32] decompose questions into sub-tasks and route them to dedicated reasoning or perception components, often improving compositionality and interpretability over clip-level inputs. For long video understanding, BOLT [20], VCA [47], VideoAgent [5], Deep Video Discovery [52], MovieChat [34], and related agentic pipelines employ global controllers, external memory, and iterative search over the temporal axis to locate relevant segments under strict token budgets. Our A4VL follows this agentic perspective but remains training-free, combining event-based partitioning, clue-guided block selection, and multi-agent consensus to robustly discover key evidence in very long videos.