ICCV 2025 Submission #10033. CONFIDENTIAL REVIEW COPY. DO NOT DISTRIBUTE.

Agentic Keyframe Search for Video Question Answering

Anonymous ICCV submission

Paper ID 10033

Abstract

*Video question answering (VideoQA) enables machines to extract and comprehend key information from videos through natural language interaction, which is a critical step towards achieving intelligence. However, the demand for a thorough understanding of videos and high computational costs still limit the widespread applications of VideoQA. To address it, we propose Agentic Keyframe Search (*AKEYS*), a simple yet powerful algorithm for identifying keyframes in the VideoQA task. It can effectively distinguish key information from redundant, irrelevant content by leveraging modern language agents to direct classical search algorithms. Specifically, we first segment the video and organize it as a tree structure. Then,* AKEYS *uses a language agent to estimate heuristics and movement costs while dynamically expanding nodes. Finally, the agent determines if sufficient keyframes have been collected based on termination conditions and provides answers. Extensive experiments on the EgoSchema and NExT-QA datasets show that* AKEYS *outperforms all previous methods with the highest keyframe searching efficiency, which means it can accurately identify key information and conduct effective visual reasoning with minimal computational overhead. For example, on the EgoSchema subset, it achieves 1.8% higher accuracy while processing only 43.5% of the frames compared to VideoTree. We believe that* AKEYS *represents a significant step towards building intelligent agents for video understanding. Code will be publicly available.*

1. Introduction

The rapid advancement of image-based Multimodal Large Language Models (MLLMs) [24, 34] has significantly simplified image understanding tasks in daily life. Users can easily upload images to OpenAI's GPT-4V or Google Gemini, ask questions about them, and receive responses via natural language interaction. However, video understanding presents greater challenges, and the development of Video Large Language Models (Video-LLMs) has notably lagged behind image-based MLLMs. Existing Video-LLMs often struggle to capture details in videos and lack a holistic understanding of video content [19]. Moreover, the computational overhead of Video-LLMs is substantially higher than that of Large Language Models (LLMs) and image-based MLLMs, hindering their commercial deployment. To address video understanding tasks in daily life more effectively, This paper focuses on the efficient extraction of keyframes, and analyzing them using image-based MLLMs for video understanding.

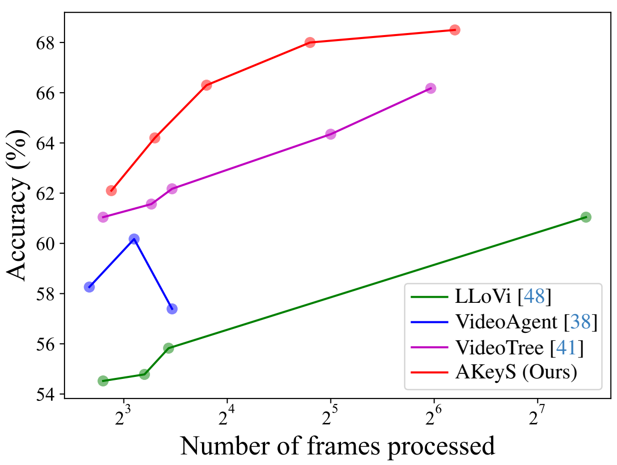


Figure 1. Demonstration of AKEYS's high frame efficiency. When processing the same number of video frames with the same (M)LLM, AKEYS achieves higher QA accuracy. **At the same accuracy level (66%), AKEYS uses only about 1/4 of the frames required by VideoTree.** Moreover, VideoTree clusters features of all frames during preprocessing, whereas AKEYS only has access to visible frames and does not utilize information from the rest. This experiment is conducted on EgoSchema [21] subset.

However, one key advantage of keyframe extraction is the ability to significantly reduce computational overhead while preserving essential information. Figure 2 presents three approaches to solving the VideoQA task. Among them, only the keyframe sampling based method achieves both accuracy and efficiency, highlighting the importance of keyframes in VideoQA task. However, a major challenge is how to effectively identify keyframes that contain the

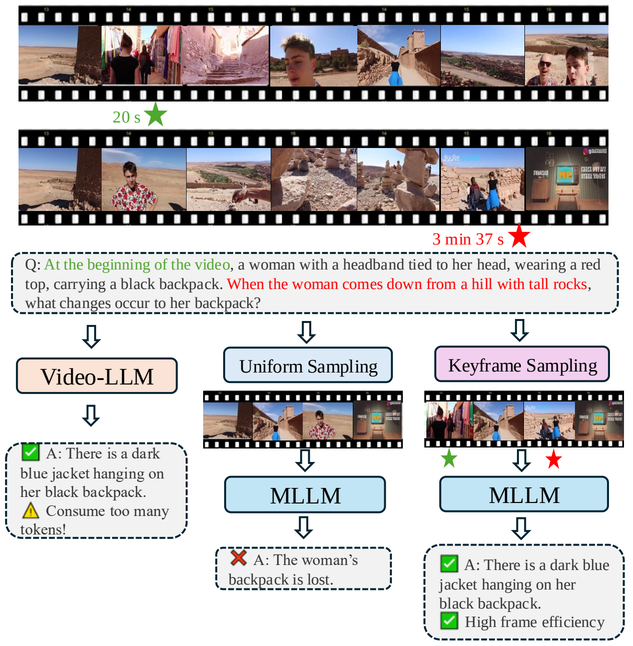

20 s
3 min 37 s
Q: At the beginning of the video, a woman with a headband tied to her head, wearing a red top, carrying a black backpack. When the woman comes down from a hill with tall rocks, what changes occur to her backpack?
Video-LLM
Uniform Sampling
Keyframe Sampling
MLLM
MLLM
A: There is a dark blue jacket hanging on her black backpack.
Consume too many tokens!
A: The woman's backpack is lost.
A: There is a dark blue jacket hanging on her black backpack.
High frame efficiency

Figure 2. Comparison of three methods for analyzing a travel vlog: (1) Video-LLM can generate correct answers but is highly token-intensive; (2) The method of uniform frame sampling may introduce irrelevant content, leading MLLM to incorrect predictions; (3) The method of keyframe sampling for MLLM achieves both accuracy and efficiency. The keyframes relevant to the given question are highlighted in the figure.

essential information needed to answer specific questions. This challenge becomes more pronounced in the context of long-form video understanding [21, 43], where the abundance of irrelevant information necessitates precise temporal localization of key content based on the question at hand. Addressing efficiency and accuracy keyframe location challenge is crucial in video understanding tasks.

In this paper, we propose an efficient algorithm named AKEYS to tackle the video understanding and analysis problems, exemplified by VideoQA tasks. Drawing inspiration from both traditional search algorithms and modern language agents, our approach harnesses the cognitive capabilities of language agents, such as reasoning, planning, summarization and reflection, to guide and provide feedback to traditional search algorithms. This methodology effectively extracts key content from redundant information, similar to sifting wheat from chaff.

Specifically, given an input video, AKEYS divides it into segments and extracts textual information from the representative frame of each segment using a Vision-Language Model (VLM), such as an image captioner. Then, it employs language agents to perform temporal comparisons and identify key content in an iterative, deepening process until reaching the termination condition. This leads to a tree-like search through the video until sufficient key information is found to answer the question.

In our experiments, AKEYS achieves 63.1% accuracy on EgoSchema fullset [21] (surpassing the best baseline by 2.0%) and 77.4% average accuracy on NExT-QA [45] (surpassing the best baseline by 1.8%). In Figure 1, we compare the frame efficiency of AKEYS with two baselines, highlighting its ability to effectively identify key information. In summary, AKEYS not only achieves state-of-the-art accuracy, but also exhibits the highest frame searching efficiency, making it a highly promising approach for real-world video analysis tasks.

2. Related Work

Video Question Answering VideoQA is a typical subtask of video understanding, involving the comprehension, analysis, and responding to questions about video content. It comprehensively tests various capabilities of multimodal QA systems [23, 45, 51], while benchmarks and datasets for VideoQA have been progressively focused on longer videos and more complex reasoning scenarios [7, 21, 44, 52].

Early approaches for VideoQA typically employ neural networks (e.g., ResNet [11], 3D convolutional neural networks [2, 10, 35]) to extract visual features, while language models were used to process the questions. These two components are then aggregated to produce answers. With the advent of LLMs, the common practice is to use a pre-trained visual encoder to extract visual features, a projection layer to map visual representations into the text latent space of LLMs, and a pretrained LLM for response generation [17, 18, 39, 49]. These Video-LLMs, with their extensive parameters, can model long contexts [30, 32, 42] and, through instruction tuning and alignment, better address VideoQA tasks, serving as foundational models in the video domain. Another popular method is based on (M)LLMs or agents [33, 46]. Many works have achieved significant success by leveraging a range of agentic techniques, including prompting [48, 50], memory [6, 13, 40], tools [5, 9, 47], and planning [12, 20, 22, 27]. These advancements have progressively contributed to developing an intelligent and powerful video agent. A detailed discussion of these works can be found in the Appendix Section A.

Keyframe Extraction Another line of research focuses on extracting keyframes from videos, which is also highly relevant to our work. LVNet [25] selects keyframes using a small network that is specifically trained for this task. VCA [47] extracts key segments through selective attention. IG-VLM [14] performs uniform sampling across all frames, converting them into an image grid, which is then directly input into an off-the-shelf vision-language model. VideoAgent [38] mimics the human brain's process by recursively selecting key video segments. After each selection, it uses

LLM to evaluate whether there is enough confidence to answer the question based on the selected segments, terminating the recursion if sufficient confidence is reached.

VideoTree [41] is the most closely related work to ours. VideoTree computes image features and performs k-means clustering on video segments, constructing a static video tree. The LLM then searches along the tree until the key information is found and the question is answered. The key differences between our work and VideoTree are as follows: (1) VideoTree computes image features for all video frames using CLIP [26], whereas AKEYS only extracts information from the visible frames. Therefore, AKEYS has lower computational overhead. (2) VideoTree constructs a static video tree in advance, which remains unchanged regardless of the input question. In contrast, the video tree in AKEYS is dynamic and adaptive to the specific question, allowing us to identify the most suitable tree structure and optimal search path for each question.

3. Method

3.1. Background: Basic Searching Algorithms

Our AKEYS algorithm is built on basic search algorithm in Algorithm 1. Based on this fundamental process, the following search algorithms are distinguished by the method to determine priority for selecting nodes.

Algorithm 1 Basic Search Algorithm

```
1:  function SEARCH($\mathcal{N}_0$)
2:      Initialize open list $\mathcal{L} \leftarrow \{\mathcal{N}_0\}$
3:      while $\mathcal{L}$ is not empty do
4:          $\mathcal{N} \leftarrow$ Pop a node from $\mathcal{L}$ based on priority
5:          if $\mathcal{N}$ is the destination then
6:              return $\mathcal{N}$
7:          end if
8:          Expand $\mathcal{N}$ to obtain neighboring nodes
9:          Add neighboring nodes to $\mathcal{L}$
10:     end while
11:     return None
12: end function
```

Depth-First Search (DFS) prioritizes nodes with greater depth and explores as far as possible before backtracking.

Breadth-First Search (BFS) explores all neighbors at the current level before moving on to the next level.

Greedy Best First Search (GBFS) uses a heuristic evaluation function $h(n)$ as the cost function, i.e., $f(n) = h(n)$. Here, $h(n)$ represents the cost from the current node to the destination. It can guide the search algorithm towards the destination but does not guarantee an optimal path.

Dijkstra's Algorithm uses a movement cost function $g(n)$ as the cost function, i.e., $f(n) = g(n)$. Here, $g(n)$ represents the cost of moving from the starting point to the current node. It finds the shortest path from a starting node to all other nodes by considering the weights of edges.

A* Algorithm combines the benefits of Dijkstra's Algorithm and GBFS. The cost function is defined as: $f(n) = g(n) + h(n)$. It balances efficiency and optimality, making it highly effective for path planning.

3.2. AKEYS Algorithm

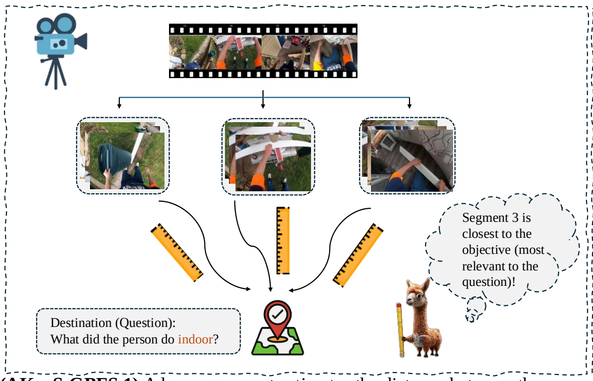
In the Method section, we first define the search objective, nodes, cost function, and termination conditions in our AKEYS algorithm, providing a comprehensive overview of the tree-structured keyframe search process. We also explain how the algorithm utilizes the retrieved information to answer questions. The key steps of AKEYS (leveraging language agents to evaluate the cost function and node expansion) are illustrated in Figure 3.

Search Objective In AKEYS, keyframes are defined as frames containing key information about the question. The search objective is to identify a sufficient set of keyframes whose combined information is sufficient to answer the question. For example, humans can view only these keyframes instead of watching the entire video, to answer the question. When using MLLMs for VideoQA, we can also discard non-keyframes and adopt one of the following two approaches: (1) directly input the keyframes into an image-based MLLM to generate an answer, or (2) apply a VLM such as BLIP [15] to caption the keyframes, and use the captions to derive an answer. The two approaches are essentially the same, as they rely on the information within the keyframes and the learned priors of models.

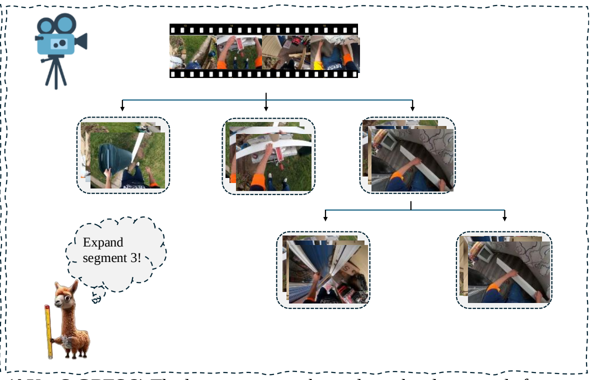
Nodes In AKEYS algorithm, we divide the video into multiple video segments, with each video segment representing a node. The initial node $\mathcal{N}_0$, is the entire video, which is first uniformly split into M segments, where M is a tunable hyper-parameter. These video segments are then put into an open list $\mathcal{L}$. The next node to be expanded, or the next video segment to be processed, is selected based on the cost function $f(n)$ we define. The expansion process means further subdividing the selected video segment. In this work, we perform a binary split on the segment for node expansion.

Answer Prediction We define the first and last frames of all current video segments as **Visible Frames** $\mathcal{F}_v$. They are connected to each other, meaning that the last frame of one video segment is the first frame of the next. We can fully utilize the information in the visible frames, while the information in the other framesis temporarily inaccessible. For the visible frames, we can employ either of the two approaches mentioned above: directly inputting the frames into the MLLM or first generating captions and then performing reasoning in the textual modality. In either way, we predict an answer based on the information from the visible frames. In this work, we choose the second approach. The predicted answer is a provisional guess during the in-

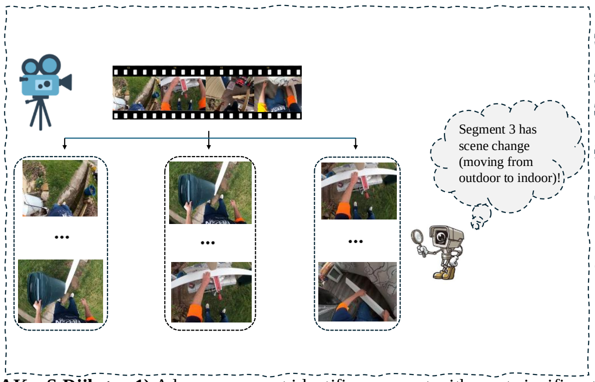
ICCV 2025 Submission #10033. CONFIDENTIAL REVIEW COPY. DO NOT DISTRIBUTE.


Segment 3 is closest to the objective (most relevant to the question)!
Destination (Question): What did the person do indoor?

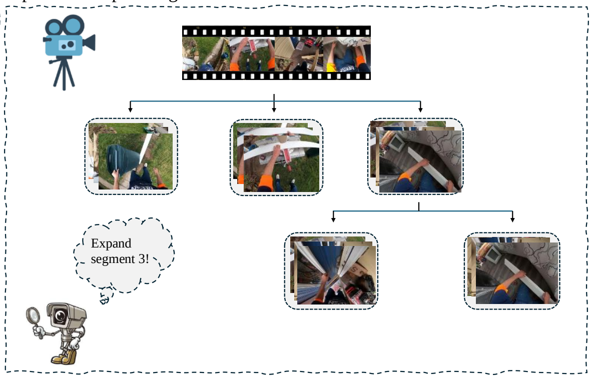
(AKeyS-GBFS.1) A language agent estimates the distance between the current node and the destination (i.e., the question).



(AKeyS-GBFS.2) The language agent then selects the closest node for expansion. Expand segment 3!



(AKeyS-Dijkstra.1) A language agent identifies segment with most significant changes. Segment 3 has scene change (moving from outdoor to indoor).



(AKeyS-Dijkstra.2) Segment 3 is considered the node that minimizes the movement cost function and is subsequently expanded.

Figure 3. Illustration of AKEYS's cost function evaluation and node expansion steps.

termediate stages and may change as the search progresses and more visible frames are revealed. When the termination condition is met, the search process concludes, and the predicted answer becomes the final answer. The total number of visible frames serves as a measure of the frame efficiency of the QA system: fewer visible frames mean fewer images for the MLLM to process, which results in higher efficiency. The final visible frames represent the keyframes obtained through our search process.

Cost Function Leveraging the evaluation capability of the language agents, we design different cost function based on different basic search algorithms.

- **AKEYS-GBFS** In Greedy Best-First Search (GBFS), the cost function $h(n)$ represents the distance from the current node to the destination. Accordingly, we let the language agent evaluate the current visible frame's information and identify what visual information is missing for answering the question. The missing information can be seen as the distance between the current node and the destination. GBFS algorithm selects the node with the smallest $h(n)$ to expand. In our adaptation, the language agent attempts to identify **the missing visual information is likely located between which two specific invisible frames**, determining which video segments should be expanded. This leads to a variant of the AKEYS algorithm named AKEYS-GBFS.
- **AKEYS-DIJKSTRA** For Dijkstra's Algorithm, the cost function $g(n)$ represents the cost of moving from the start point to the current node. In our adaption, we have the language agent assess the current visible frame's information to identify **which video segment exhibits the most significant scene change** (e.g., the primary scenes, figures, or activities in the first and last frame of a video segment differ, indicating a transition or an important visual element's introduction). Note that in Dijkstra's algorithm, the cost function does not consider the destination location, and similarly, in AKEYS-DIJKSTRA, the question is invisible to the language agent.
We would like to further discuss why the video segment with the most significant scene change is considered the node closest to the start point. When segmenting and extracting keyframes from a long-form video with multiple scene transitions, the ideal scenario would be to treat each scene as a separate segment. This ensures that the video's visual elements are non-overlapping and non-missing in the visible frames, leading to the fewest visible frames

needed and highest frame efficiency. Meanwhile, adjacent visible frames would contain information from two distinct scenes, allowing for visual comparison. This comparison would help QA system infer major changes in the video, leading to a better overall understanding of the video content. Therefore, intuitively, regardless of the specific question, treating each scene as an individual video segment is an optimal segmentation strategy. It is the most efficient segmentation method that achieves the same level of accuracy, which can be view as an abstraction of the shortest distance from the start point that achieves the same result. Hence, we consider the video segment with the most significant scene change to be the node closest to the start point.

- **AKEYS-A*** For the A* Algorithm, the cost function is the sum of the heuristic evaluation function and the movement cost function, i.e., $f(n) = h(n) + g(n)$, which means that A* Algorithm takes into account both the distance from the current node to the destination and the distance from the start point to the current node. Correspondingly, in our AKEYS-A* variant, the language agent must simultaneously consider two factors: (1) which video segment is likely to contain the missing information, and (2) which video segment exhibits the most significant scene change. Only video segments that satisfy both are prioritized for expansion.
- **AKEYS-BFS** We also propose a naive algorithm variant, AKEYS-BFS which does not rely on a language agent to evaluate the cost function. Instead, it performs a breadth-first expansion, continually splitting all the existing video segments (in the case of no pruning). Like BFS, AKEYS-BFS advances in a wave-like manner, steadily progressing. This variant is suitable for situations where a language agent cannot be accessed, or where the overhead introduced by the LLM is less of a concern, with a greater emphasis on ensuring no information is overlooked.

We do not intend to introduce an AKEYS-DFS variant, as its depth-first expansion focuses on a single initial segment. Without strict termination conditions, it is prone to falling into local optima.

Termination Condition Traditional search algorithms typically have a deterministic termination condition: whether the search objective has been reached. However, in keyframe search algorithms for VideoQA, the termination condition is much more vague and difficult to define. It is challenging to determine whether sufficient information has been gathered, or key information is missing or over-inference. Inspired by the reflection, summarization, and self-evaluation abilities of language agents, we use the base LLM to evaluate the confidence in the predicted answer and determines whether to terminate the search accordingly. In this way, the AKEYS will terminate when a sufficiently confident prediction is made. Specifically, we combine two

Algorithm 2 Agentic KeyFrame Search (AKEYS)

```
Require: Video $v$, question $q$, MLLM $F$, confidence
    threshold $C$, max iteration $T$, uniform sampling size
    $M$, beam size $B$
Ensure: Answer $\hat{y}$, keyframes $\{\mathcal{F}_k\}$
 1: $\mathcal{N}_0 \leftarrow v$
 2: $\mathcal{N}_1, \mathcal{N}_2, ..., \mathcal{N}_M \leftarrow$ UniformSegment$(\mathcal{N}_0, M)$
 3: $\mathcal{L} \leftarrow \{\mathcal{N}_0, \mathcal{N}_1, ..., \mathcal{N}_M\}$
 4: $t \leftarrow 1$
 5: while $t \leq T$ do
 6:     $\{F_v\} \leftarrow$ ExtractVisibleFrames$(\mathcal{L})$
 7:     $\hat{y} \leftarrow$ PredictAnswer$(F, \{\mathcal{F}_v\}, q)$
 8:     $c_1, c_2 \leftarrow$ EvaluateConfidence$(F, \hat{y}, \{\mathcal{F}_v\}, q)$
 9:     if $c_1 \geq C$ and $c_2 \geq C$ then
10:         break
11:     else
12:         $\{p\} \leftarrow$ EvaluateCostFunction$(F, \{\mathcal{F}_v\}, \mathcal{L})$
13:         $\mathcal{L} \leftarrow$ SelectAndExpandNodes$(\mathcal{L}, \{p\}, B)$
14:     end if
15:     $t \leftarrow t + 1$
16: end while
17: $\{\mathcal{F}_k\} \leftarrow \{\mathcal{F}_v\}$
18: return $\hat{y}$, $\{\mathcal{F}_k\}$
```

methods of confidence evaluation by a voting mechanism, as outlined below.

- **Self-Evaluation and Self-Reflection** LLMs can be instructed to self-evaluate their responses, reflecting on potential shortcomings in their responses [28, 31]. Therefore, after generating an answer, we input the question, information of visible frames, and the LLM's previous reasoning chain and predicted answer back into the model. The LLM then assesses the accuracy and reliability of its previous answer and output a confidence score (c_1).
- **Temporal Summarization** The captions of the sampled frames are discrete. To integrate the sampled frames along the temporal dimension, we instruct the LLM to summarize their captions to form a cohesive overview of the video. We use few-shot examples [1] to generate a more accurate and detailed video summary. Then we prompt the LLM to predict the answer and output a confidence score (c_2) based on the summary. The advantage of this approach is to consider the sampled frames in a complete temporal context rather than in isolation.

We employ a voting mechanism to ensemble the above two methods. The search process only terminates when both methods independently determine that they have sufficient confidence ($c_1 \geq C$ and $c_2 \geq C$, C is the threshold).

Other search techniques can also be integrated into AKEYS framework, for instance, using beam search to expand multiple nodes each step. It not only reduces the computational overhead of evaluating the cost function by lan-

Table 1. **Comparison between AKEYS and other methods.** We highlight the gain of our method over VideoTree [41] in blue.

Model	**(M)LLM**	**EgoSchema**		**NExT-QA**			
		Sub.	**Full**	**Tem.**	**Cau.**	**Des.**	**Avg.**
Based on Open-source Captioners and LLMs							
MVU [27]	Mistral-13B	60.3	37.6	55.4	48.1	64.1	55.2
LangRepo [13]	Mixtral-8x7B	66.2	41.2	51.4	64.4	69.1	60.9
Video-LLA+INTP [29]	Vicuna-7B v1.5	-	38.6	58.6	61.9	72.2	62.7
Based on Proprietary MLLMs							
IG-VLM [14]	GPT-4V	59.8	-	63.6	69.8	74.7	68.6
LVNet [25]	GPT-4o	68.2	61.1	65.5	75.0	81.5	72.9
Based on Open-source Captioners and Proprietary LLMs							
ProViQ [5]	GPT-3.5	57.1	-	-	-	-	64.6
MoReVQA [22]	PaLM-2	-	51.7	64.6	70.2	-	69.2
Vamos [36]	GPT-4	51.2	48.3	-	-	-	-
LLoVi [48]	GPT-4	61.2	-	61.0	69.5	75.6	67.7
VideoAgent [38]	GPT-4	60.2	54.1	64.5	72.7	81.1	71.3
VideoAgent [6]	GPT-4	62.8	60.2	-	-	-	-
LifelongMemory [40]	GPT-4	64.1	58.6	-	-	-	-
VideoTree [41]	GPT-4	66.2	61.1	70.6	76.5	83.9	75.6
AKEYS (Ours)	GPT-4	**68.0 (1.8 ↑)**	**63.1 (2.0 ↑)**	**72.3 (1.7 ↑)**	**78.2 (1.7 ↑)**	**85.4 (1.5 ↑)**	**77.4 (1.8 ↑)**
AKEYS (Ours)	GPT-4o	**68.6 (2.4 ↑)**	**63.6 (2.5 ↑)**	**72.9 (2.3 ↑)**	**79.0 (2.5 ↑)**	**86.1 (2.2 ↑)**	**78.1 (2.5 ↑)**

guage agents, but also helps prevent the search algorithm from getting stuck in local optima, as shown in Algorithm 2.

Finally, we would like to comment that the term **agentic** in our algorithm's name reflects in two aspects: (1) the use of an LLM to evaluate the cost function, which enables the LLM to engage in path planning while interacting with the environment (i.e., the video and the question); (2) the use of the LLM to assess the termination condition, which enables the LLM to engage in decision-making. These two aspects together endow the base LLM with agentic properties, making it clear that AKEYS is essentially powered by a language agent.

4. Experiments

4.1. Datasets

EgoSchema [21] dataset comprises over 5,000 human-curated multiple-choice question-answer pairs, making it one of the most widely used datasets for long-form video question and answering. Its subset contains 500 video and QA pairs. Each video in the datset is three minutes in length. A notable feature of EgoSchema is its high difficulty level: humans can only achieve 76% accuracy, and current Video-LLMs perform below 70%. The extended video length and increased complexity underscore the importance of keyframe search and key information retrieval.

NExT-QA [45] dataset consists of 5,440 videos and approximately 52K manually annotated question-answer pairs. Its primary focus is to assess whether QA models truly understand the causal and temporal structures of actions within a video. We use the multiple-choice QA part of NExT-QA. Based on the types, the questions are divided into casual questions, temporal questions and descriptive questions.

4.2. Main Results

Following VideoTree [41], we compare the performance of AKEYS with various related approaches on LLM-driven VideoQA using AKEYS-A* variant. Most of the baselines are mentioned in the related work and Appendix Section A. Implementation details are provided in Appendix Section B. Prompts we use are listed in Appendix Section C. Table 1 demonstrates that AKEYS significantly outperforms all these baselines. Specifically, AKEYS (with GPT-4 as base LLM) achieves 63.1% accuracy on EgoSchema fullset (surpassing the best baseline by 2.0%) and 77.4% accuracy on NExT-QA (surpassing the best baseline by 1.8%). Moreover, AKEYS operates in a training-free, zero-shot setting, while it still outperforms training-based methods such as LVNet [25] and Vamos [36]. Meanwhile, AKEYS processes only visible frames, for instance, achieving the reported performance requires only about 15% of the total frames. In contrast, methods like LangRepo [13] and LifeLongMemory [40] process all frames without selection.

Table 2. **Ablation on basic search algorithms**. We highlight the improvement of AKEYS-A* over the naive AKEYS-BFS in the table, emphasizing the role of the cost function evaluation.

Algorithm	Accuracy	# Visible Frames
AKEYS-BFS	64.7	31.2
AKEYS-GBFS	67.0	**27.3**
AKEYS-DIJKSTRA	66.8	27.6
AKEYS-A*	**68.0** (**3.3 ↑**)	27.9

Additionally, as shown in Figure 1, we compare AKEYS's frame efficiency with other keyframe extraction methods in the same condition. The results show that AKEYS utilizes frames more efficiently than LLoVi [48], VideoAgent [38] and VideoTree [41], demonstrating its superior ability to identify key information.

4.3. Ablation Studies

4.3.1. Basic Search Algorithms

In Table 2, We investigate performance and frame efficiency of several AKEYS algorithm variants with different base search algorithms on the EgoSchema subset. The frame efficiency is measured by the number of visible frames[1]. We observe that AKEYS-A* achieves the highest accuracy. AKEYS-BFS ranks second in accuracy but has lower frame efficiency, as BFS Algorithm exhaustively explores all branches, leading to higher exploration costs. AKEYS-GBFS slightly outperforms AKEYS-DIJKSTRA on both metrics, while AKEYS-A* combines the strengths of them, significantly improving accuracy with only a slight compromise in frame efficiency. This demonstrates that efficient keyframe localization requires both the heuristic search function and the movement cost function.

4.3.2. Termination Condition

In Table 3, we conduct ablation experiments on the termination condition of the search process, using AKEYS-A* on the EgoSchema subset. The results show that self-evaluation & self-reflection and temporal summarization assess information sufficiency from different perspectives. When combined, they enhance the reliability of confidence estimation, leading to improved algorithm performance.

4.3.3. Base LLM

We also conducted an ablation study on the base LLM of the AKEYS algorithm in Table 4. We find that GPT-4o achieves the best performance as the base LLM. In contrast, reasoning models such as o3-mini and Deepseek-R1 perform slightly worse than GPT-4o, likely due to the relatively straightforward nature of visual reasoning in our tasks.

[1]On EgoSchema dataset, all videos are three minutes long, and we set the frame rate $fps = 1$, which means the overall frame number is 180.

Table 3. **Ablation on termination condition**

Termination Condition	Accuracy	# Visible Frames
Self-Evaluation	67.4	**27.4**
Summarization	67.3	28.2
Vote	**68.0**	27.9

Table 4. **Ablation on different base LLMs**

Base LLM	Accuracy	# Visible Frames
GPT-4	68.0	27.9
GPT-4O	**68.6**	**26.7**
O3-MINI	67.3	28.3
DEEPSEEK-R1	67.6	26.9
LLAMA-3.3-70B	65.2	27.4

5. Analysis

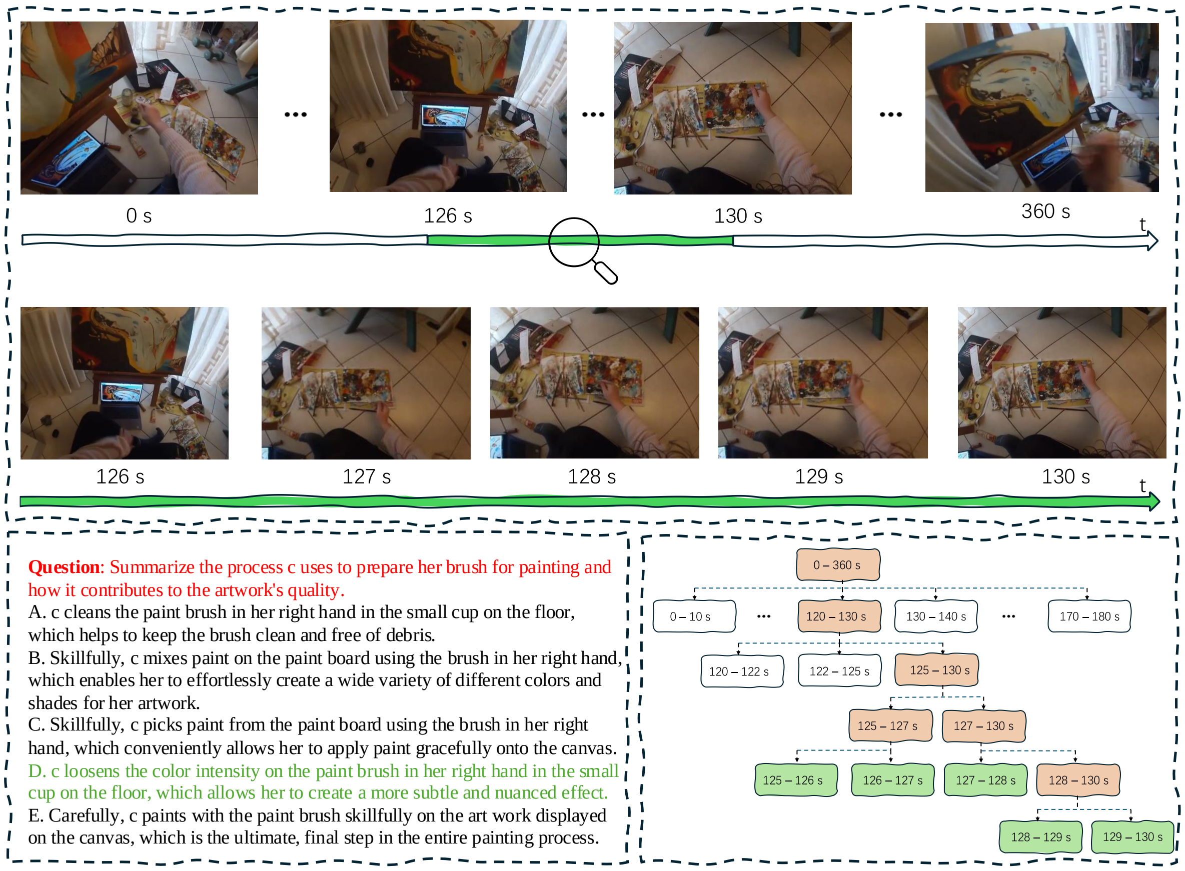
5.1. Comparison with Video-LLMs

As previously discussed, there are two primary method for VideoQA task: (I) utilizing Video-LLMs for end-to-end computation; (II) employing (M)LLM-driven, keyframe-based, training-free method, like AKEYS. We argue that both methods have their respective advantages. The key strength of Method I is that state-of-the-art Video-LLMs [3, 8] outperform Method II. It is suitable for scenarios where high accuracy is required, and computational cost is not a concern. In contrast, the primary advantage of Method II is its practical value for daily video analysis tasks, as it offers a more favorable balance between performance and computational cost. In the following, we use AKEYS as an example to illustrate the relative advantages of Method II.

- **Training-Free**. The training-free nature of Method II significantly reduces the overall cost. In Table 5, we present the training costs and resource requirements of Video-LLMs that achieve comparable performance with AKEYS on EgoSchema [21] and NExT-QA [45] benchmarks. The table highlights the complexity and high cost of training Video-LLMs, underscoring the training-free advantage of Method II.
- **Lower Inference Overhead**. Method II still relies on large model inference. However, AKEYS significantly reduces inference overhead by efficient keyframe selection instead of processing the whole video.
- **Better Interpretability**. AKEYS provides greater interpretability by generating intermediate results, such as the keyframe selection and textual reasoning process. In contrast to the end-to-end nature of Method I, this enhances transparency and interpretability.

Table 5. **Comparison of computation costs between Video-LLMs and AKEYS**

Model	Train Model Size	Tain Data Size	Computational Resources	EgoSchema	NExT-QA
ViLA [37]	4B	36.4K	8×40 GB A100s	-	75.6
VideoChat2 [16]	7B	4M	32×80 GB A100s	54.4	78.6
VideoLLaMA2 [4]	72B	13.6M	32×80 GB A100s	63.9	75.6
AKEYS (ours)		Training-free		63.1	77.4



Question: Summarize the process c uses to prepare her brush for painting and how it contributes to the artwork's quality.
A. c cleans the paint brush in her right hand in the small cup on the floor, which helps to keep the brush clean and free of debris.
B. Skillfully, c mixes paint on the paint board using the brush in her right hand, which enables her to effortlessly create a wide variety of different colors and shades for her artwork.
C. Skillfully, c picks paint from the paint board using the brush in her right hand, which conveniently allows her to apply paint gracefully onto the canvas.
D. c loosens the color intensity on the paint brush in her right hand in the small cup on the floor, which allows her to create a more subtle and nuanced effect.
E. Carefully, c paints with the paint brush skillfully on the art work displayed on the canvas, which is the ultimate, final step in the entire painting process.

Figure 4. Visualization of tree-search process of a case from EgoSchema [21].

5.2. Visualization

Figure 4 presents a visualized case study. In this 3-minute video, the key information for answering the question is located between 126s and 130s. Our AKEYS algorithm precisely identifies this critical video segment by searching along the video tree and expanding relevant nodes. And it retrieves all frames within the 125s–130s range as keyframes, successfully answering the question. In the video tree, we mark the nodes traversed by the key search path in yellow and mark the final leaf node obtained from the key search path in green. The nodes outside the key search path are barely expanded.

6. Conclusion

In this paper, we introduce AKEYS, a novel keyframe search algorithm tailored for efficient video analysis. AKEYS leverages the language agent to guide the search process. Like separating the wheat from the chaff, it effectively distinguishes key information from redundancy in videos. We evaluate AKEYS on the EgoSchema and NExT-QA datasets, where it achieves higher accuracy and frame efficiency than all baseline methods. We believe that AKEYS represents a significant step towards building more powerful video agents and tackling various video understanding challenges.

References

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. 5

[2] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset, 2018. 2

[3] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yimin Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Huipeng Deng, Jiaye Ge, Kai Chen, Kaipeng Zhang, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling, 2025. 7

[4] Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, and Lidong Bing. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms, 2024. 8

[5] Rohan Choudhury, Koichiro Niinuma, Kris M. Kitani, and László A. Jeni. Zero-shot video question answering with procedural programs, 2023. 2, 6

[6] Yue Fan, Xiaojian Ma, Rujie Wu, Yuntao Du, Jiaqi Li, Zhi Gao, and Qing Li. Videoagent: A memory-augmented multimodal agent for video understanding, 2024. 2, 6

[7] 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, Rongrong Ji, and Xing Sun. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis, 2024. 2

[8] Lishuai Gao, Yujie Zhong, Yingsen Zeng, Haoxian Tan, Dengjie Li, and Zheng Zhao. Linvt: Empower your image-level large language model to understand videos, 2024. 7

[9] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training, 2022. 2

[10] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?, 2018. 2

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

[12] Sullam Jeoung, Goeric Huybrechts, Bhavana Ganesh, Aram Galstyan, and Sravan Bodapati. Adaptive video understanding agent: Enhancing efficiency with dynamic frame sampling and feedback-driven reasoning, 2024. 2

[13] Kumara Kahatapitiya, Kanchana Ranasinghe, Jongwoo Park, and Michael S. Ryoo. Language repository for long video understanding, 2024. 2, 6

[14] Wonkyun Kim, Changin Choi, Wonseok Lee, and Wonjong Rhee. An image grid can be worth a video: Zero-shot video question answering using a vlm, 2024. 2, 6

[15] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation, 2022. 3

[16] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, Limin Wang, and Yu Qiao. Mvbench: A comprehensive multi-modal video understanding benchmark, 2024. 8

[17] Rui Li, Xiaohan Wang, Yuhui Zhang, Zeyu Wang, and Serena Yeung-Levy. Temporal preference optimization for long-form video understanding, 2025. 2

[18] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection, 2024. 2

[19] Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. Tempcompass: Do video llms really understand videos?, 2024. 1

[20] Ahmad Mahmood, Ashmal Vayani, Muzammal Naseer, Salman Khan, and Fahad Shahbaz Khan. Vurf: A general-purpose reasoning and self-refinement framework for video understanding, 2024. 2

[21] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding, 2023. 1, 2, 6, 7, 8

[22] Juhong Min, Shyamal Buch, Arsha Nagrani, Minsu Cho, and Cordelia Schmid. Morevqa: Exploring modular reasoning models for video question answering, 2024. 2, 6

[23] Thong Nguyen, Yi Bin, Junbin Xiao, Leigang Qu, Yicong Li, Jay Zhangjie Wu, Cong-Duy Nguyen, See-Kiong Ng, and Luu Anh Tuan. Video-language understanding: A survey from model architecture, model training, and data perspectives, 2024. 2

[24] OpenAI. Gpt-4 technical report, 2024. 1

[25] Jongwoo Park, Kanchana Ranasinghe, Kumara Kahatapitiya, Wonjeong Ryoo, Donghyun Kim, and Michael S. Ryoo. Too many frames, not all useful: Efficient strategies for long-form video qa, 2024. 2, 6

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 3

[27] Kanchana Ranasinghe, Xiang Li, Kumara Kahatapitiya, and Michael S. Ryoo. Understanding long videos with multimodal language models, 2025. 2, 6

[28] Jie Ren, Yao Zhao, Tu Vu, Peter J. Liu, and Balaji Lakshminarayanan. Self-evaluation improves selective generation in large language models, 2023. 5

[29] Yuzhang Shang, Bingxin Xu, Weitai Kang, Mu Cai, Yuheng Li, Zehao Wen, Zhen Dong, Kurt Keutzer, Yong Jae Lee, and

Yan Yan. Interpolating video-llms: Toward longer-sequence lmms in a training-free manner, 2024. 6

[30] 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, 2024. 2

[31] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. 5

[32] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, Yan Lu, Jenq-Neng Hwang, and Gaoang Wang. Moviechat: From dense token to sparse memory for long video understanding, 2024. 2

[33] Yunlong Tang, Jing Bi, Siting Xu, Luchuan Song, Susan Liang, Teng Wang, Daoan Zhang, Jie An, Jingyang Lin, Rongyi Zhu, Ali Vosoughi, Chao Huang, Zeliang Zhang, Pinxin Liu, Mingqian Feng, Feng Zheng, Jianguo Zhang, Ping Luo, Jiebo Luo, and Chenliang Xu. Video understanding with large language models: A survey, 2024. 2

[34] Gemini Team. Gemini: A family of highly capable multimodal models, 2024. 1

[35] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks, 2015. 2

[36] Shijie Wang, Qi Zhao, Minh Quan Do, Nakul Agarwal, Kwonjoon Lee, and Chen Sun. Vamos: Versatile action models for video understanding, 2024. 6

[37] Xijun Wang, Junbang Liang, Chun-Kai Wang, Kenan Deng, Yu Lou, Ming Lin, and Shan Yang. Vila: Efficient video-language alignment for video question answering, 2024. 8

[38] Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. Videoagent: Long-form video understanding with large language model as agent, 2024. 1, 2, 6, 7

[39] Yi Wang, Kunchang Li, Yizhuo Li, Yinan He, Bingkun Huang, Zhiyu Zhao, Hongjie Zhang, Jilan Xu, Yi Liu, Zun Wang, Sen Xing, Guo Chen, Junting Pan, Jiashuo Yu, Yali Wang, Limin Wang, and Yu Qiao. Internvideo: General video foundation models via generative and discriminative learning, 2022. 2

[40] Ying Wang, Yanlai Yang, and Mengye Ren. Lifelongmemory: Leveraging llms for answering queries in long-form egocentric videos, 2024. 2, 6

[41] 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, 2024. 1, 3, 6, 7

[42] Yuetian Weng, Mingfei Han, Haoyu He, Xiaojun Chang, and Bohan Zhuang. Longvlm: Efficient long video understanding via large language models, 2024. 2

[43] Chao-Yuan Wu and Philipp Krähenbühl. Towards long-form video understanding, 2021. 2

[44] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding, 2024. 2

[45] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa:next phase of question-answering to explaining temporal actions, 2021. 2, 6, 7

[46] Junbin Xiao, Nanxin Huang, Hangyu Qin, Dongyang Li, Yicong Li, Fengbin Zhu, Zhulin Tao, Jianxing Yu, Liang Lin, Tat-Seng Chua, and Angela Yao. Videoqa in the era of llms: An empirical study, 2024. 2

[47] Zongxin Yang, Guikun Chen, Xiaodi Li, Wenguan Wang, and Yi Yang. Doraemongpt: Toward understanding dynamic scenes with large language models (exemplified as a video agent), 2024. 2

[48] Ce Zhang, Taixi Lu, Md Mohaiminul Islam, Ziyang Wang, Shoubin Yu, Mohit Bansal, and Gedas Bertasius. A simple llm framework for long-range video question-answering, 2024. 1, 2, 6, 7

[49] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding, 2023. 2

[50] Haoyu Zhang, Yuquan Xie, Yisen Feng, Zaijing Li, Meng Liu, and Liqiang Nie. Hcqa @ ego4d egoschema challenge 2024, 2024. 2

[51] Yaoyao Zhong, Junbin Xiao, Wei Ji, Yicong Li, Weihong Deng, and Tat-Seng Chua. Video question answering: Datasets, algorithms and challenges, 2022. 2

[52] 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, 2025. 2