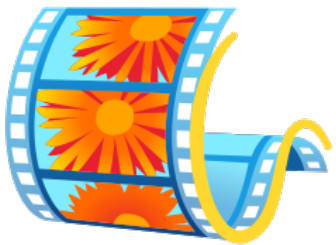
迈向高效视频理解智能体

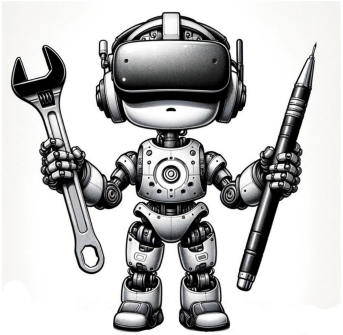
Torwards Efficient Video Understanding Agent

范孙奇

清华大学计算机系/新雅书院

2025 年 10 月 18 日





高效视频-语言理解的需求

打造能理解长视频、复杂视频，能用自然语言交互的 AI 助手一直是人工智能领域的研究热点。无论是在工业界还是学术界，视频-语言理解都有着巨大需求。

图 1: 豆包软件集成了“上传视频分析”的功能

高效视频-语言理解的需求

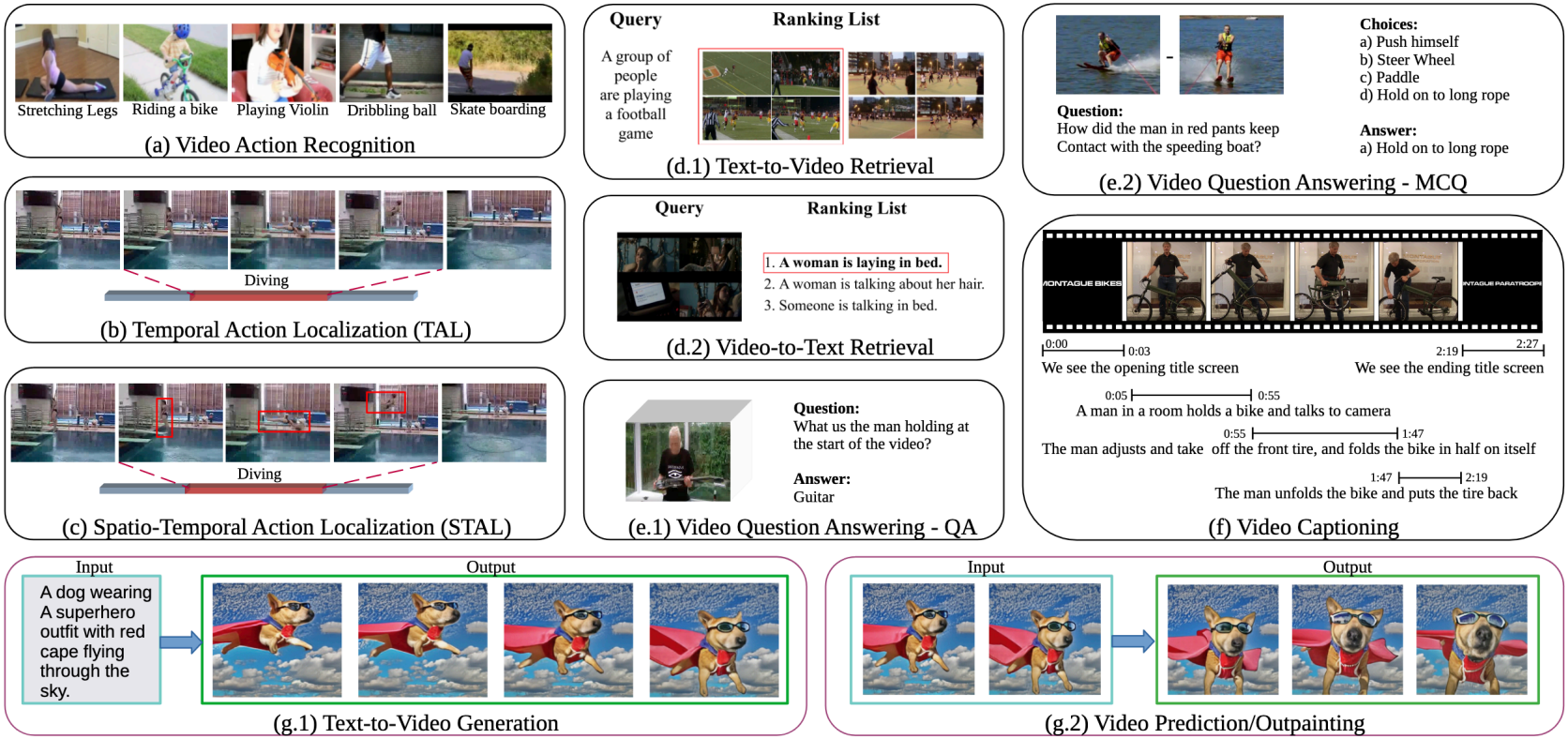


图 2: 多样的视频理解任务。其中视频问答是最关键的一个任务，也是随着大模型的发展进展最迅速的研究领域。

视频理解的难点

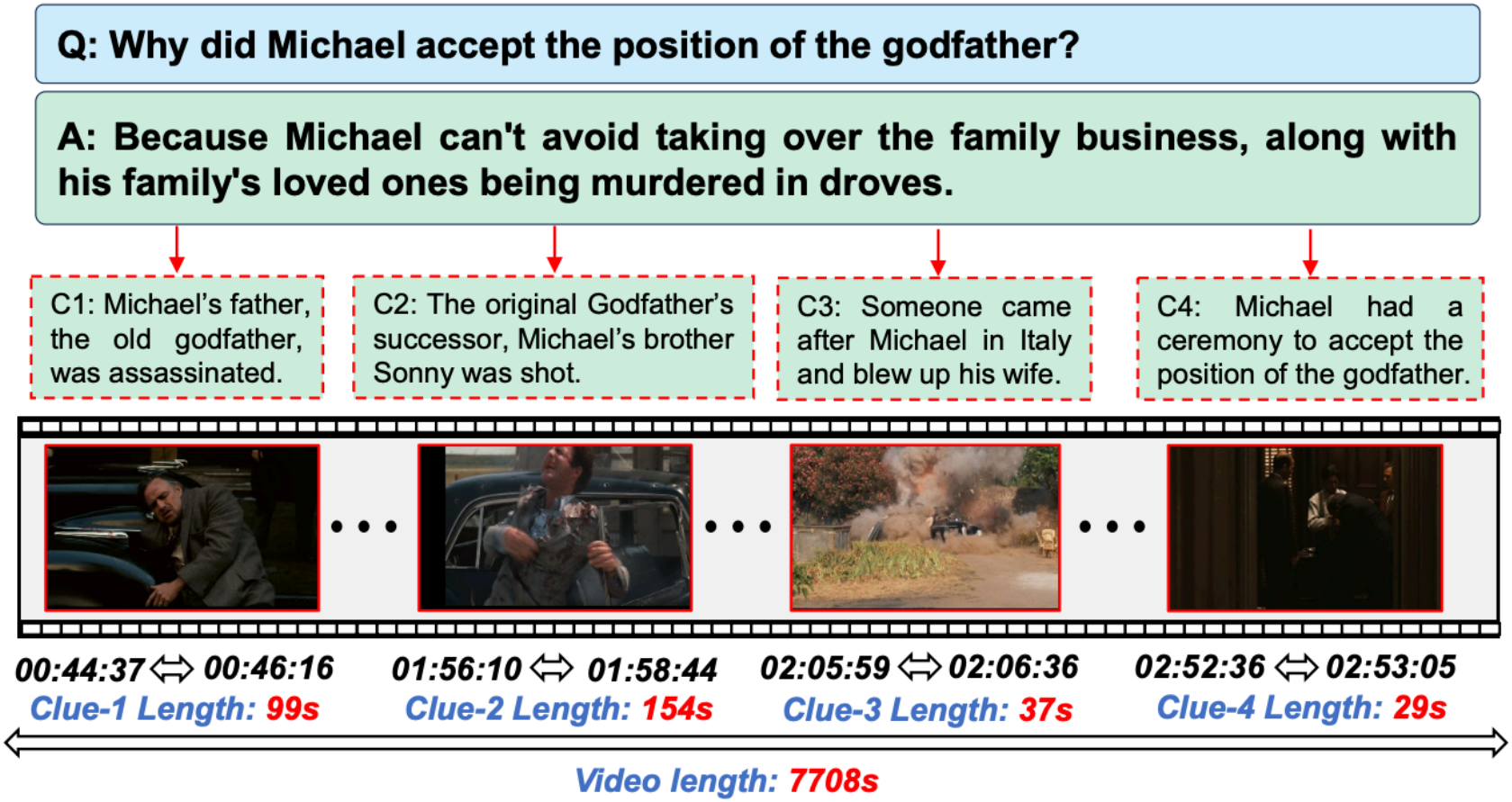


图 3: 视频理解的难点有：（1）电影级别的长视频（2）复杂问题，需要对长视频全局有很好的理解、有因果推断能力，才能正确回答。

为什么仅有 MLLM / Video-LLM 不够?

- 视频长度增加、日趋复杂，更像是一个需要主动探索的环境。
- 大模型的缺陷：例如在识别细小物体、OCR 等方面往往不如小模型，可以通过调用外部工具弥补。
- Scale towards multi-turn: 有些问题往往需要多轮对话解决，更适合 ReAct [19] 这种 Agent 范式（Observation-Reasoning-Action）。
- 由于 context 的限制，长视频需要维护 memory，优化输入的 context。
- ...

Agent 的概念

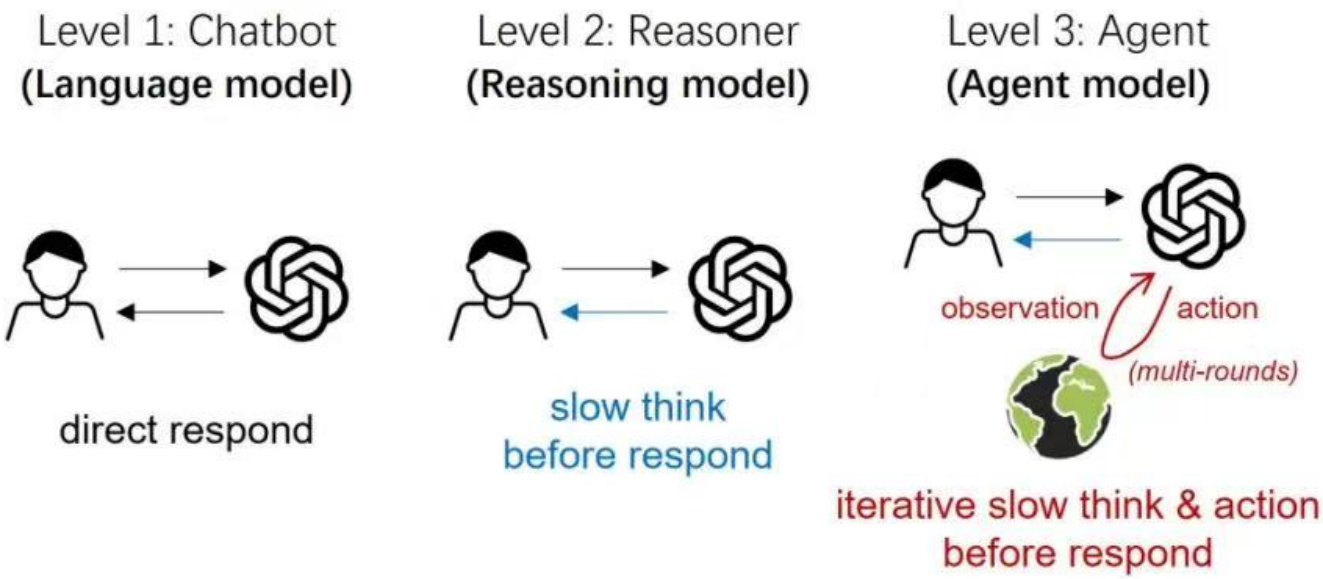


图 4: 大模型的演化之路：从 Chatbot 到 Reasoning Model（以 OpenAI o1 为代表），再到 Agentic Model（以 OpenAI o3 为代表）

Agent 的概念

什么是 Agent?

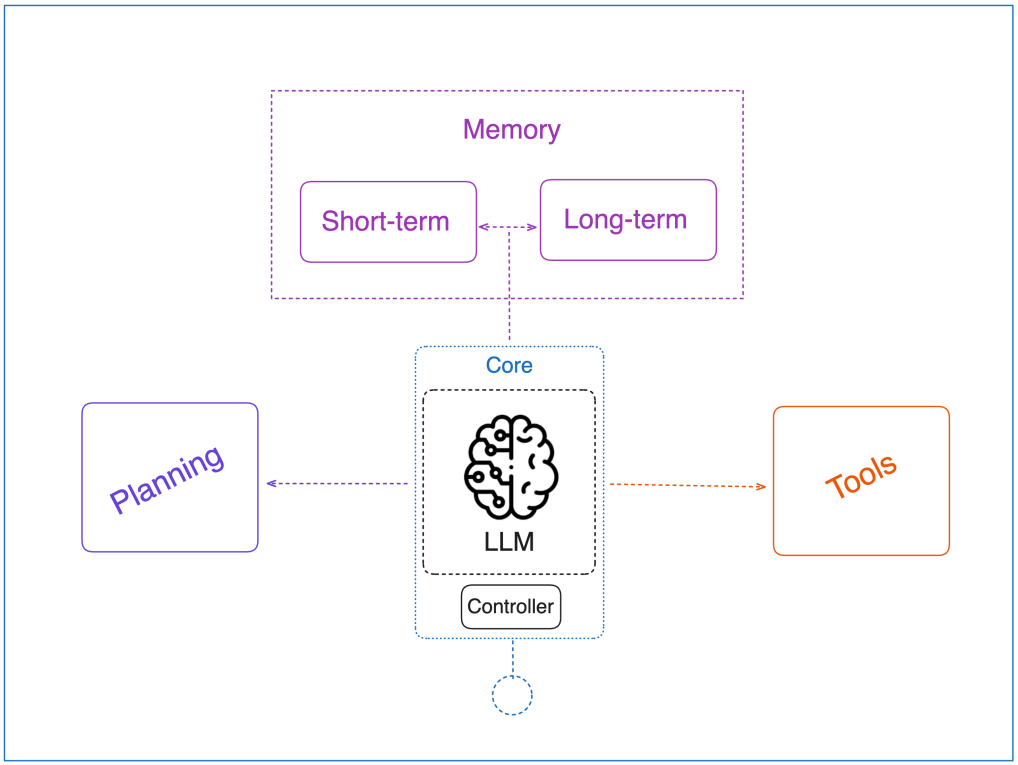
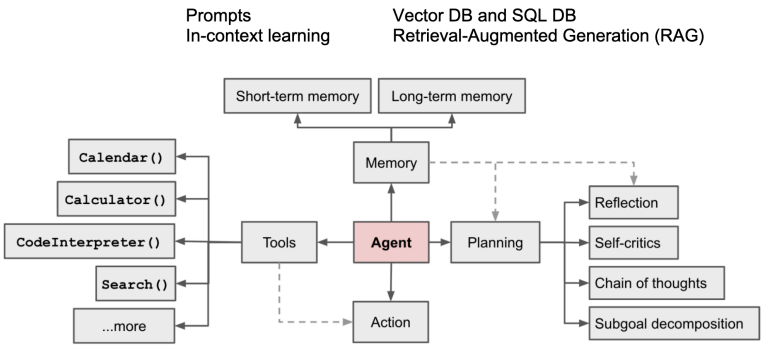


图 5: 前 OpenAI safety 负责人，Thinking Machines Lab 联创 Lilian Weng 提出了 Agent 的四要素.

构建 Video Agent 的概念



https://lilianweng.github.io/posts/2023-06-23-agent/

图 6: Agent 四要素：Planning, Tools, Memory, Action

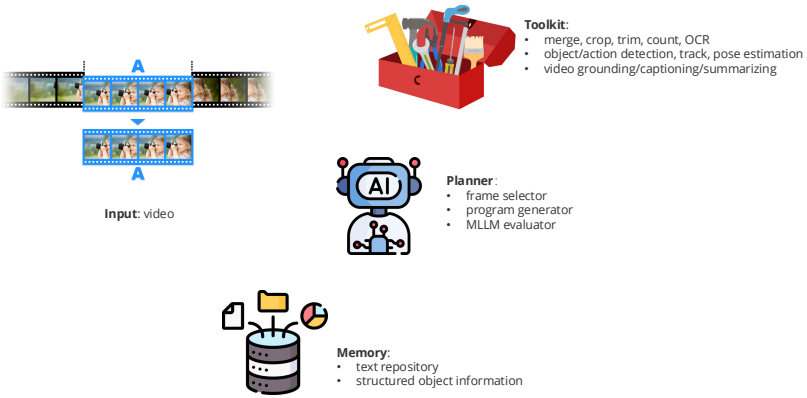


图 7: Video Agent 的对应要素

在本篇工作中，Action := Video Question Answering;
Planning 是前半部分工作；Tools & Memory 是后半部分工作。

构建 Video Agent 的概念

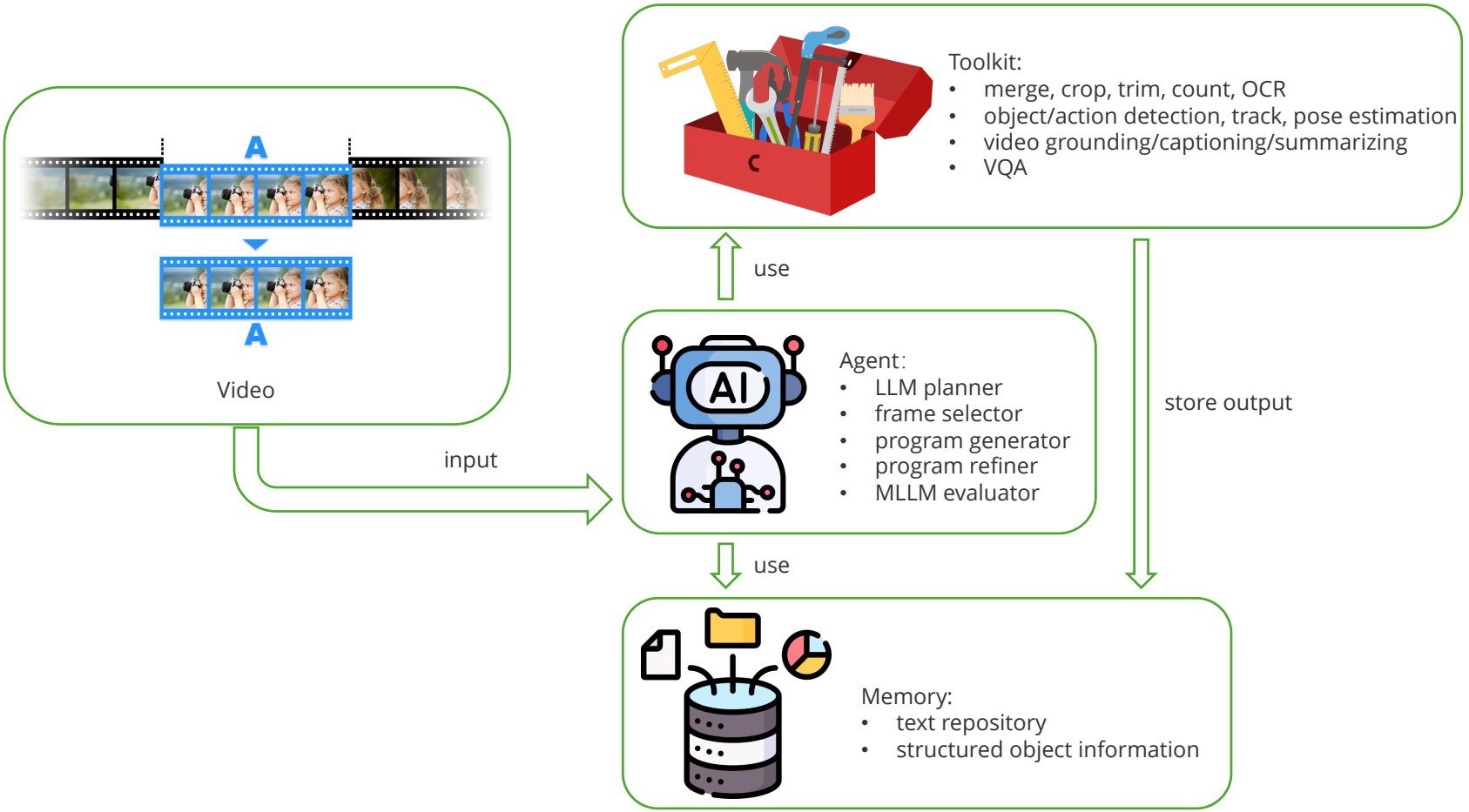


图 8: Video Agent 的示例

关键帧对于视频理解的重要性

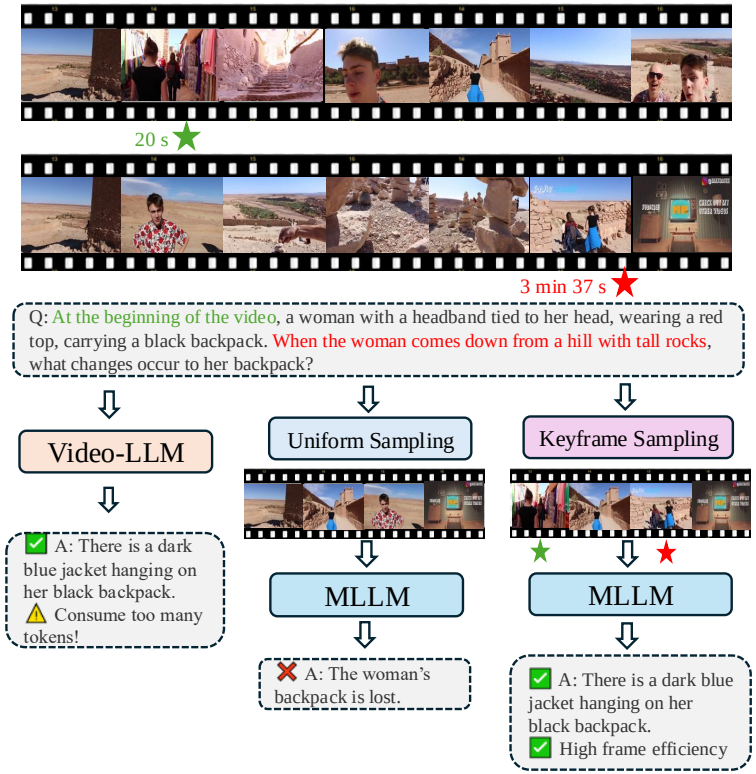


图 9: 三种使用大模型的方法分析一个旅行 vlog

Video Agent 首先需要规划一条寻找关键信息的路径。因此，我们提出了一种智能化关键帧搜索算法 Agentic Keyframe Search。

构造节点

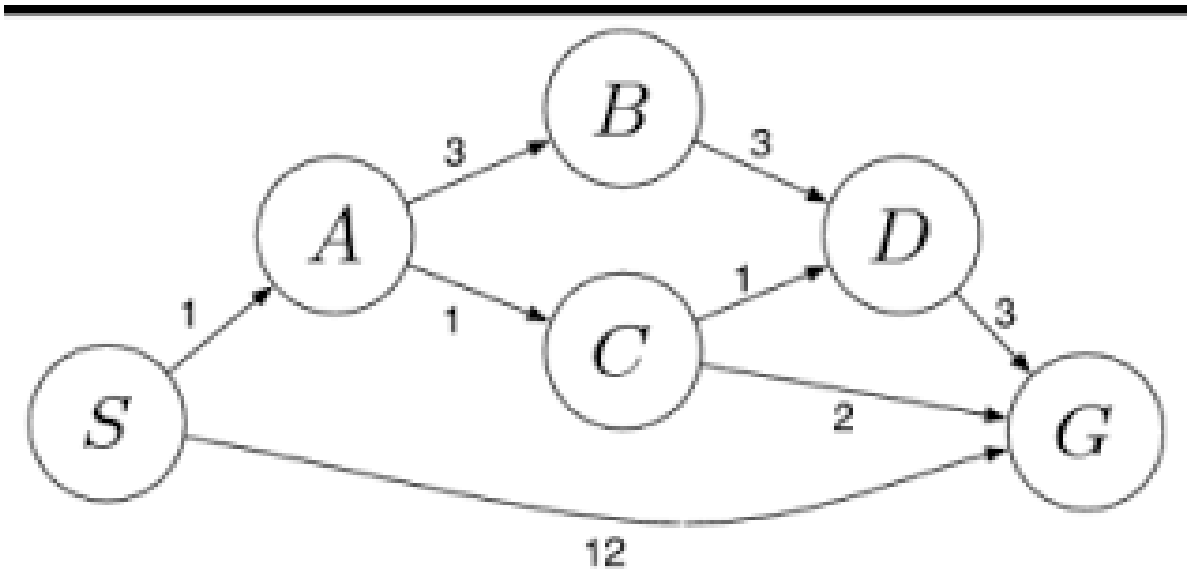
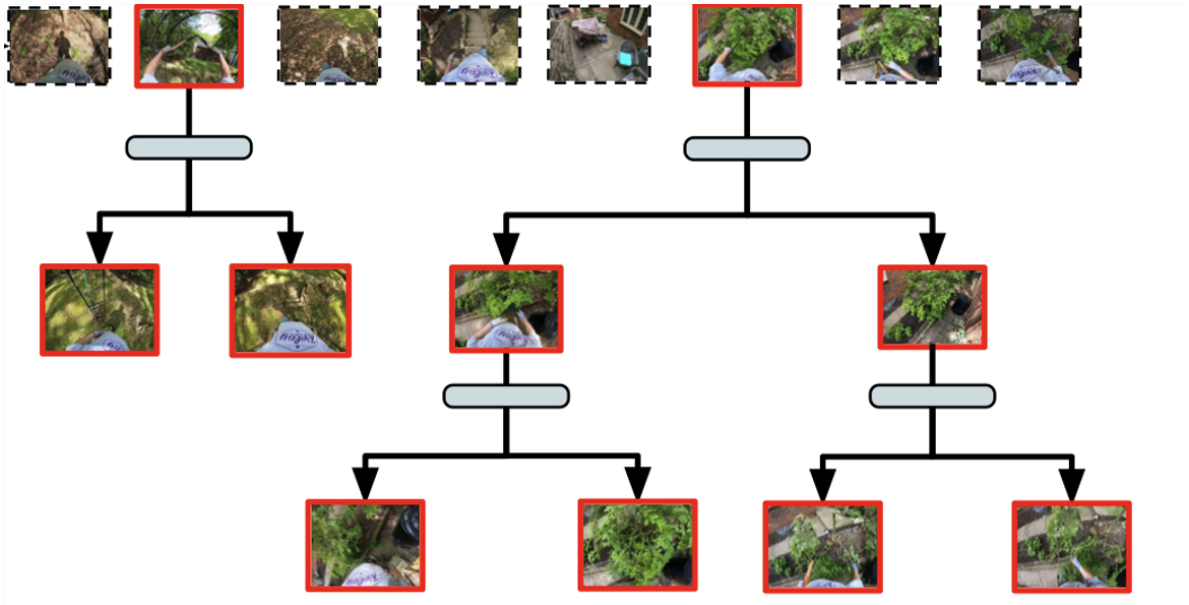


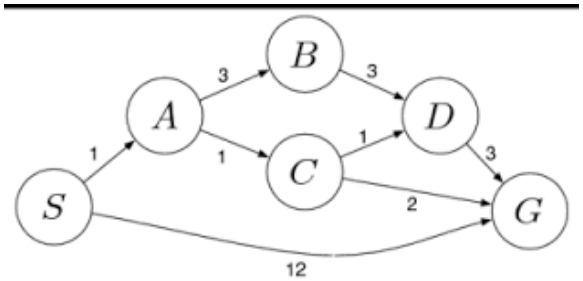
图 10: 从传统的搜索算法中汲取灵感，首先是需要构造节点

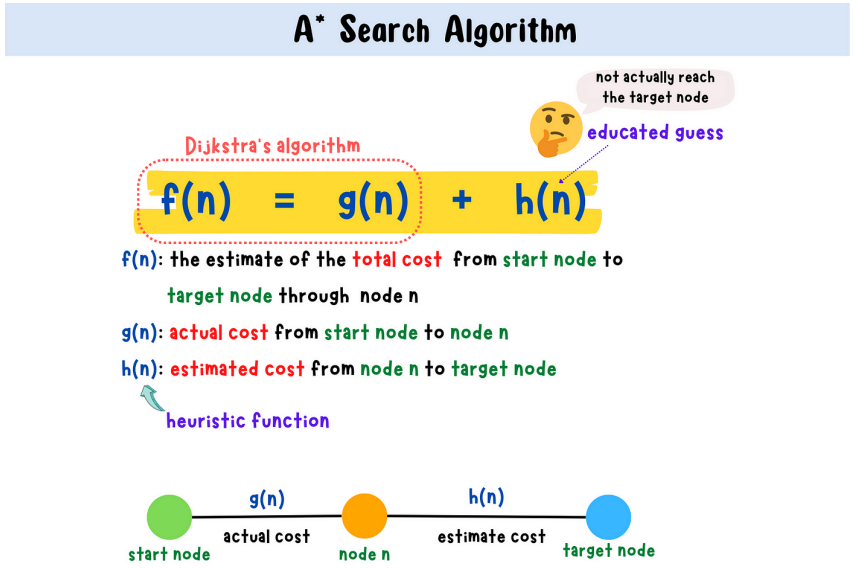
构造节点



- **搜索目标**：找到足够多的关键信息，足以用来回答问题。
- **节点**：视频切片（Video Segment）。
- **可见帧**：每个切片的首帧和尾帧。
- **预测答案**：把可见帧输入大模型预测答案。
- **搜索终止**：LLM 自我评估是否有足够信心回答问题。
- **关键帧**：搜索终止后，所有的可见帧集合。

构造代价

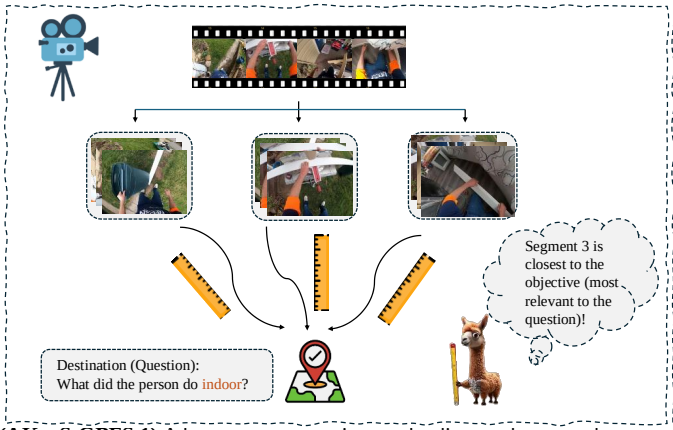




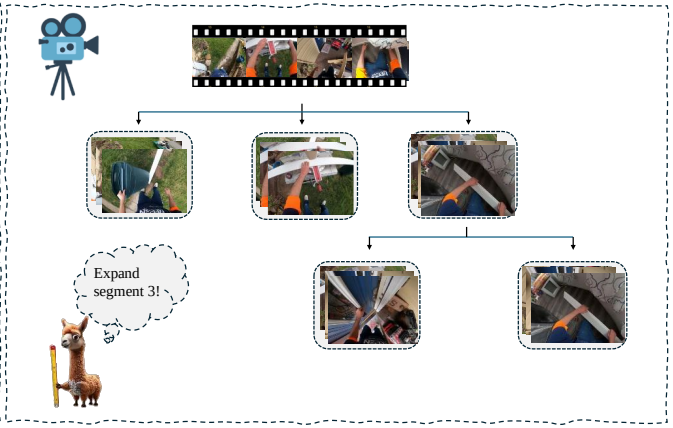
传统的搜索算法用代价函数 $f(n)$ 来挑选待扩展节点 n。$g(n)$ 是移动代价，$h(n)$ 是启发式函数。

- Dijkstra Algorithm: $f(n) = g(n)$
- Greedy-Best First Search, GBFS: $f(n) = h(n)$
- A* Algorithm: $f(n) = g(n) + h(n)$

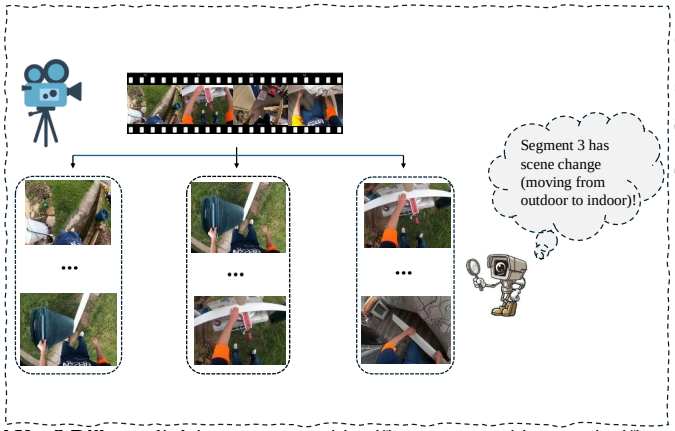
代价函数（Cost Function）的定义



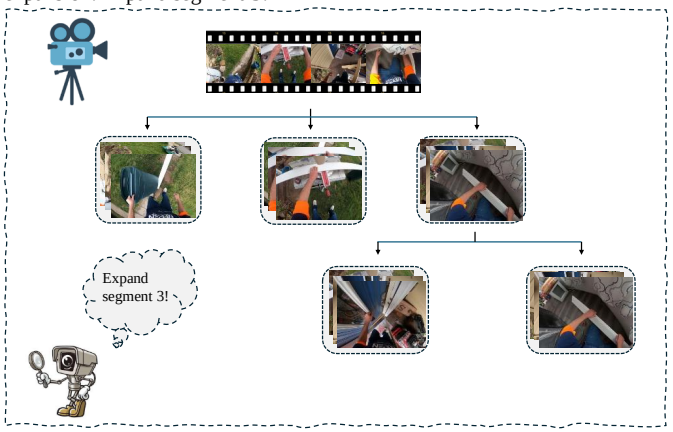
(AKeyS-GBFS.1) A language agent estimates the distance between the current node and the destination (i.e., the question).



(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.

- 启发函数 $h(n)$：和问题比较，当前缺失了多少视觉信息？
- 移动代价 $g(n)$：哪个切片场景/主要视觉元素变化最剧烈？

代价函数（Cost Function）的定义

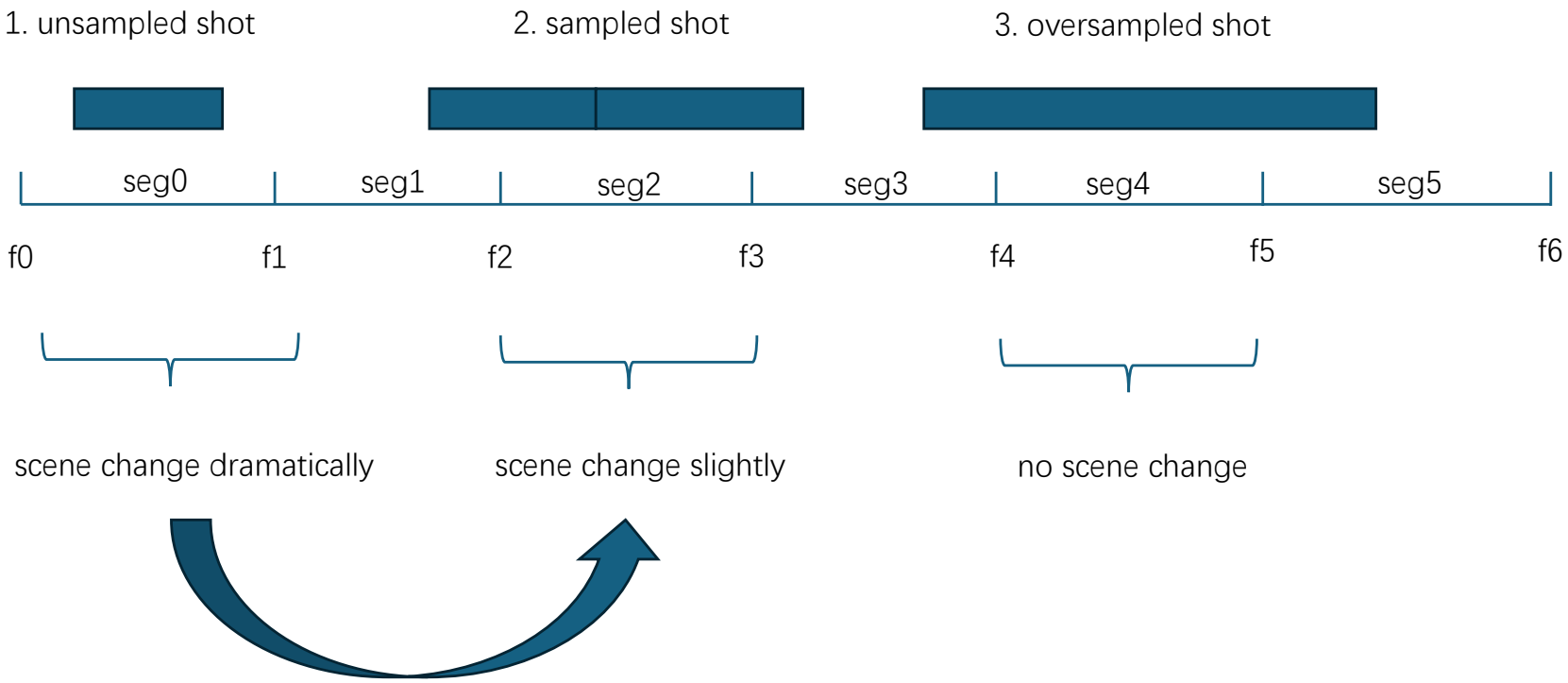


图 11: 将移动代价定义为场景变化的“信息熵”

算法可视化

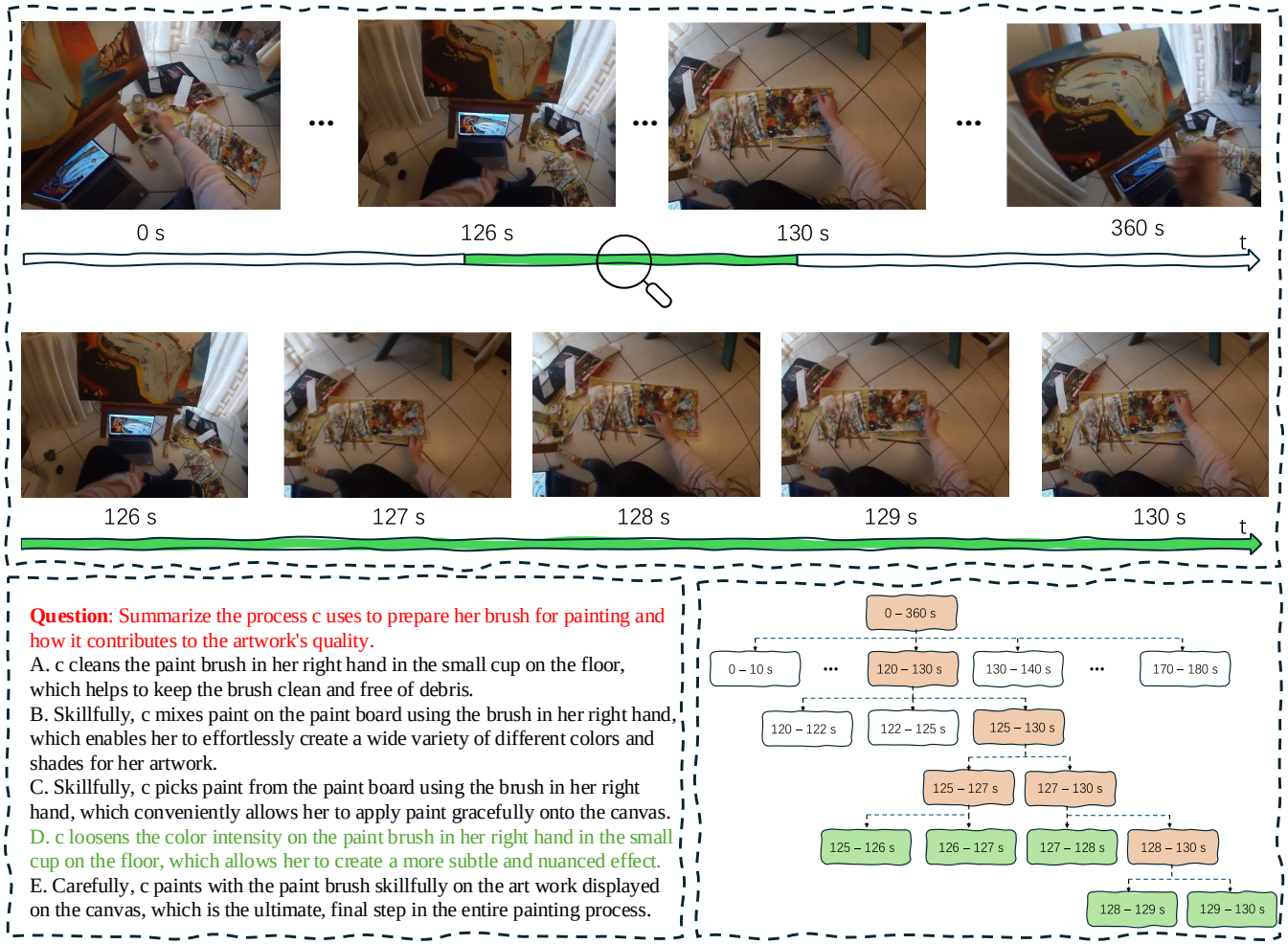


图 12: 一个可视化实例

主结果

表 1: **主实验结果**，标蓝了相对于 VideoTree [17] 的提升.

Model	(M)LLM	EgoSchema [7]		NExT-QA [18]			
		Sub.	Full	Tem.	Cau.	Des.	Avg.
Based on Open-source Captioners and LLMs							
MVU [12]	Mistral-13B	60.3	37.6	55.4	48.1	64.1	55.2
LangRepo [5]	Mixtral-8x7B	66.2	41.2	51.4	64.4	69.1	60.9
Video-LLA+INTP [13]	Vicuna-7B v1.5	-	38.6	58.6	61.9	72.2	62.7
Based on Proprietary MLLMs							
IG-VLM [6]	GPT-4V	59.8	-	63.6	69.8	74.7	68.6
LVNet [10]	GPT-4o	68.2	61.1	65.5	75.0	81.5	72.9
Based on Open-source Captioners and Proprietary LLMs							
ProViQ [1]	GPT-3.5	57.1	-	-	-	-	64.6
MoReVQA [8]	PaLM-2	-	51.7	64.6	70.2	-	69.2
Vamos [14]	GPT-4	51.2	48.3	-	-	-	-
LLoVi [20]	GPT-4	61.2	-	61.0	69.5	75.6	67.7
VideoAgent [15]	GPT-4	60.2	54.1	64.5	72.7	81.1	71.3
VideoAgent [3]	GPT-4	62.8	60.2	-	-	-	-
LifelongMemory [16]	GPT-4	64.1	58.6	-	-	-	-
VideoTree [17]	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 ↑)

帧效率研究

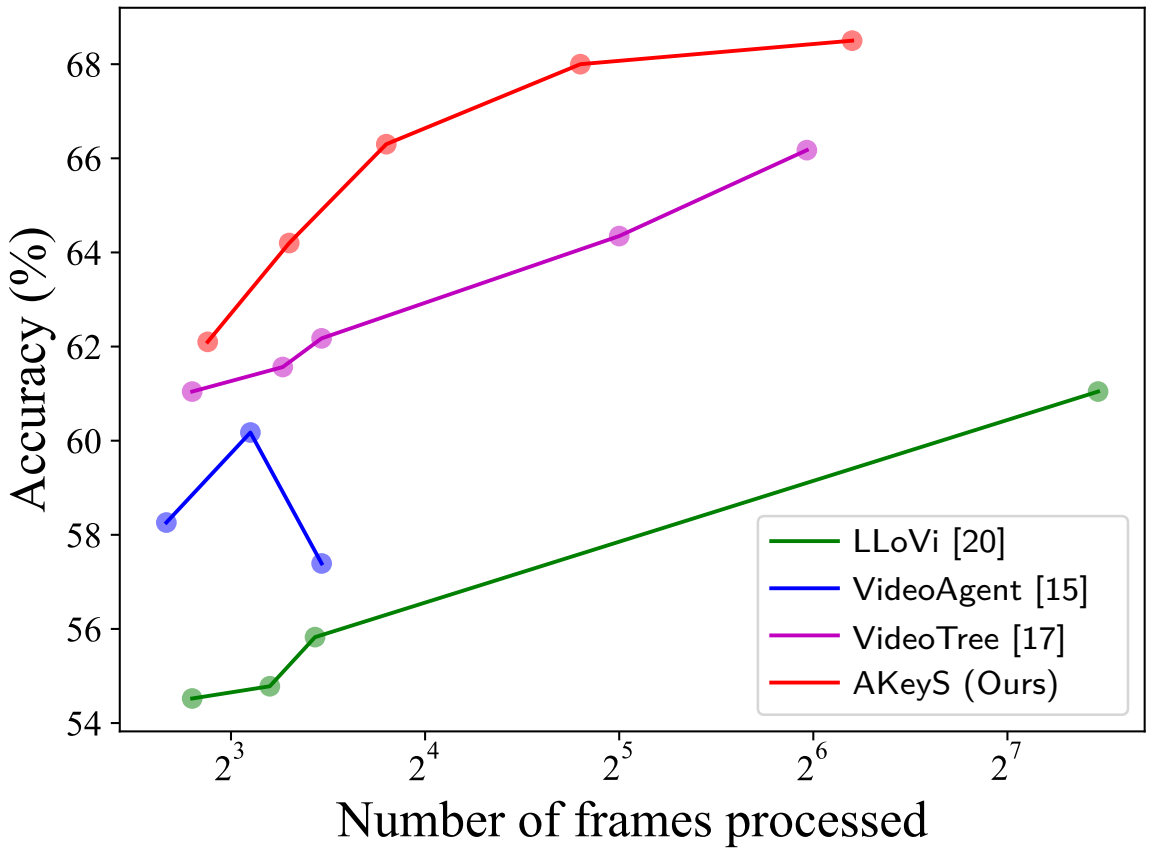


图 13: 我们的方法具有最高的帧效率。和 VideoTree [17] 相比，达到相同的准确率（66%），我们的方法仅用了 1/4 可见帧。

不同基底算法比较

表 2: A* 算法效果最好。图中标蓝了 A* 算法相对于 BFS 算法的提升，这是代价函数 $f(n)$ 的功劳。

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

不同基底 LLM 比较

表 3: GPT-4o 效果最好

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

Motivation：让 LLM 调用工具，解决视频问答问题

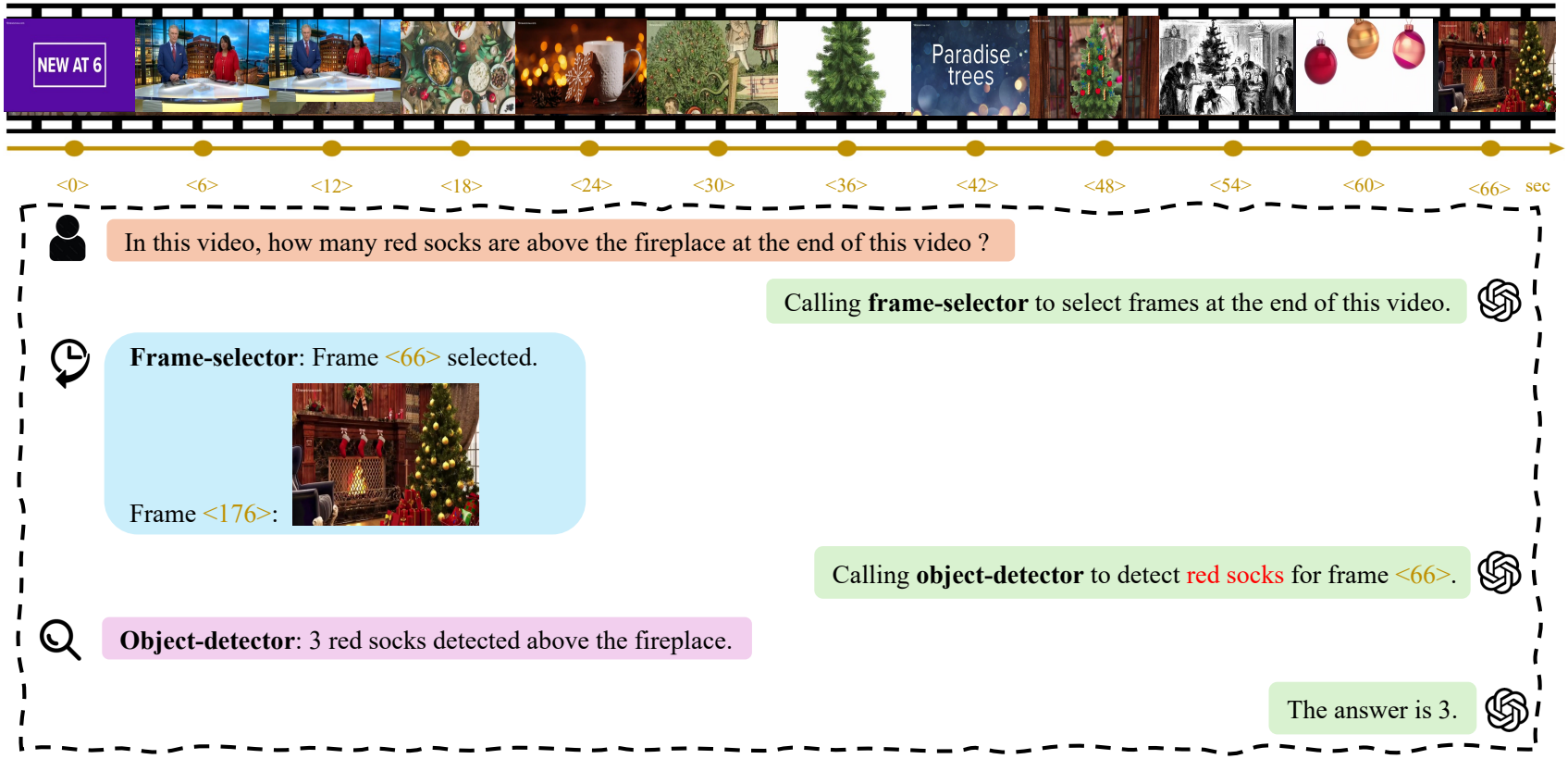
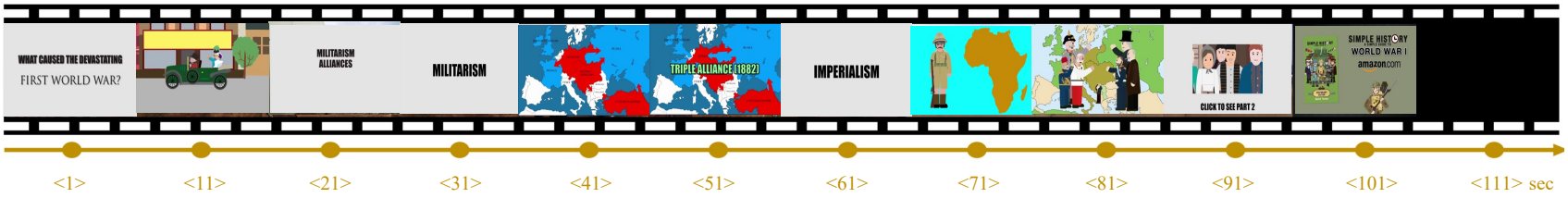


图 14: 面对大模型数不清楚数目的问题，调用目标检测（object detection）工具解决。

Motivation: 让 LLM 调用工具，解决视频问答问题



What's the trigger that set off the war mentioned in the video? Select from the below options:
A. Militarisim. B. The assasination of Archduke Franze Fredinand. C. King Edward III of England's claim to the French throne.
D. Germany's blitzkrieg against Poland.

Calling **Image-grid-QA-tool**. Question: Which war is mentioned in the image sequence?

Image-grid-QA-tool: Use frame <1><11><21><31><41><51><61><71><81> to form a 3*3 image grid.
Answer: This image sequence mention the First World War in frame <1>.

Calling **google-search** for the trigger of World War I.

Google-search: the trigger of World War I is the assasination of Archduke Franze Fredinand.

The answer is B.

图 15: 针对 knowledge-intensive Video QA 任务，在大模型世界知识欠缺的情况，调用检索工具（google search）来解决。

Motivation：让 LLM 调用工具，解决视频问答问题

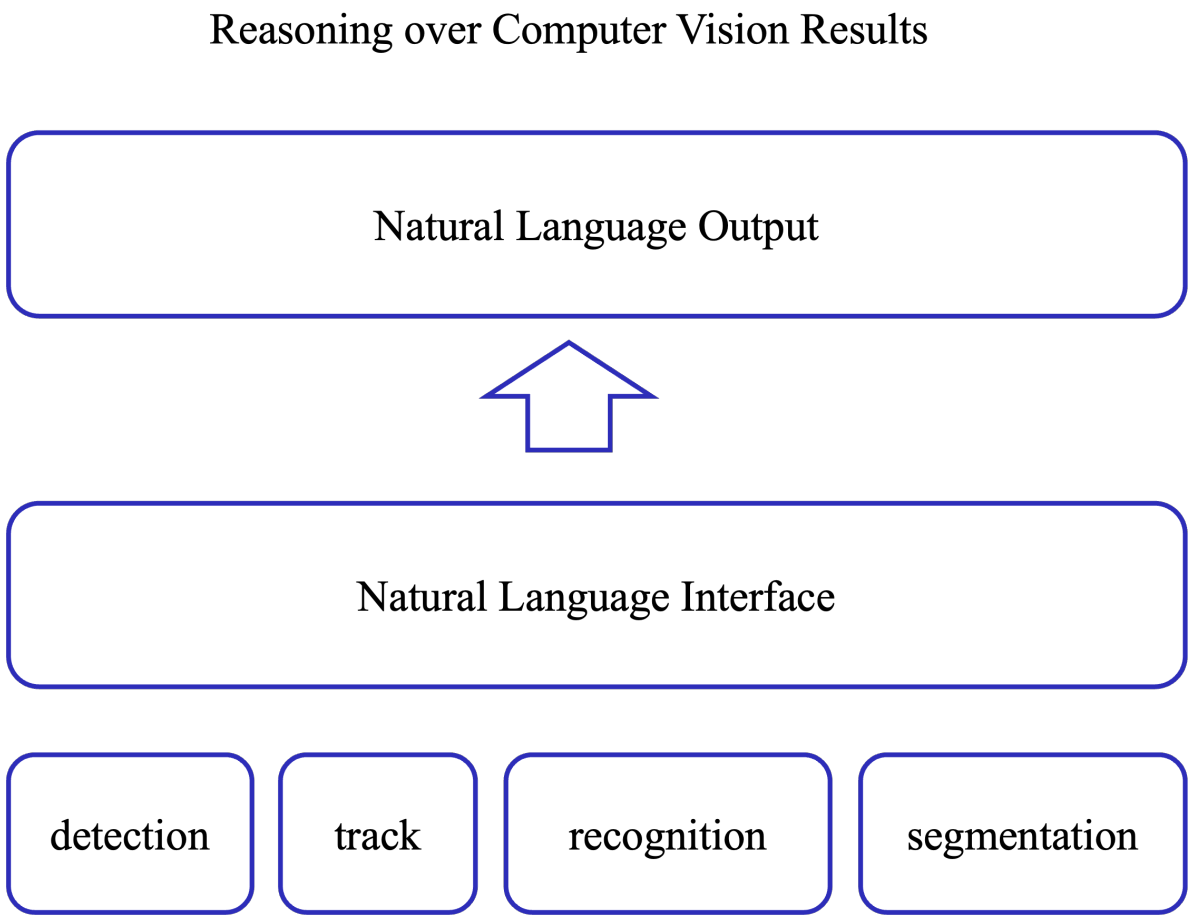


图 16: 总体思路：通过自然语言总结像素空间的结果，并进行推理

第一步：建立 Video Toolkit

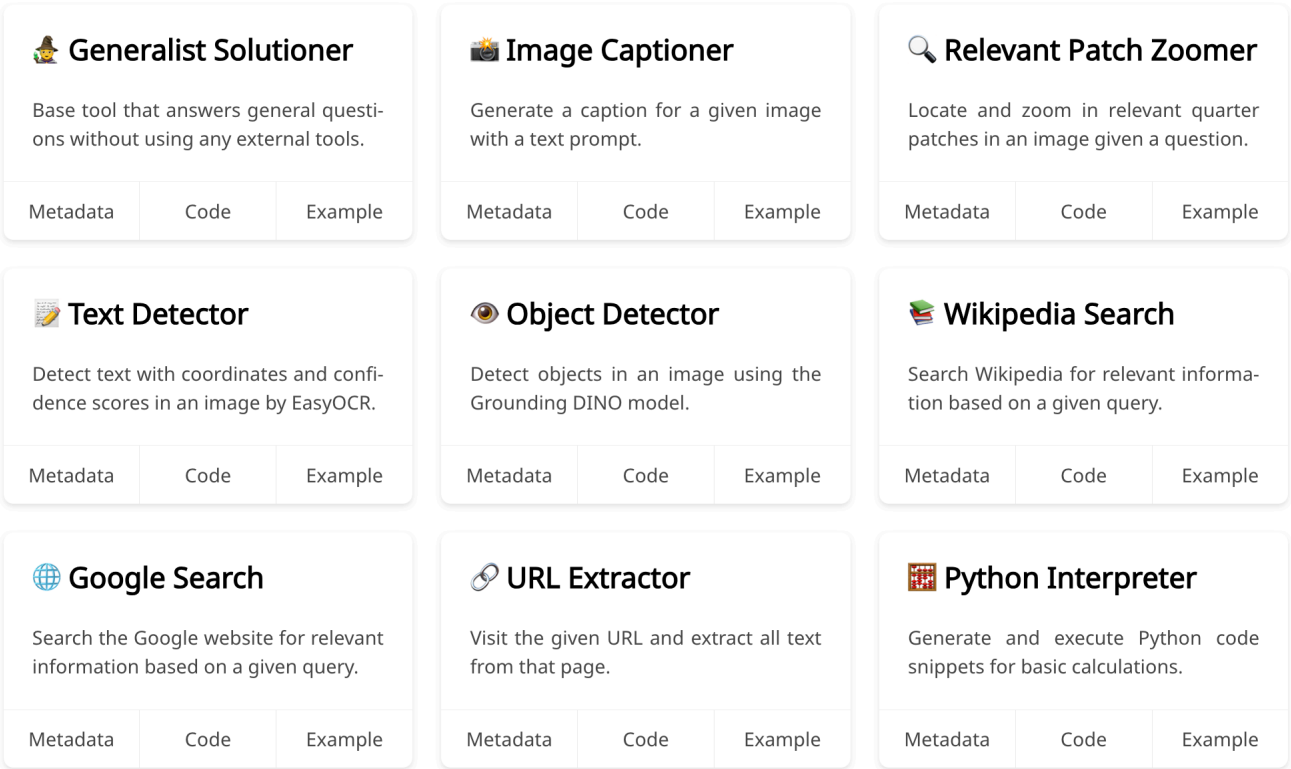


图 17: 部分工具卡片展示。工具对于 Video Agent 来说是即插即用的，完成工具卡片的填写，Video Agent 就可以使用这个工具。

第一步：建立 Video Toolkit

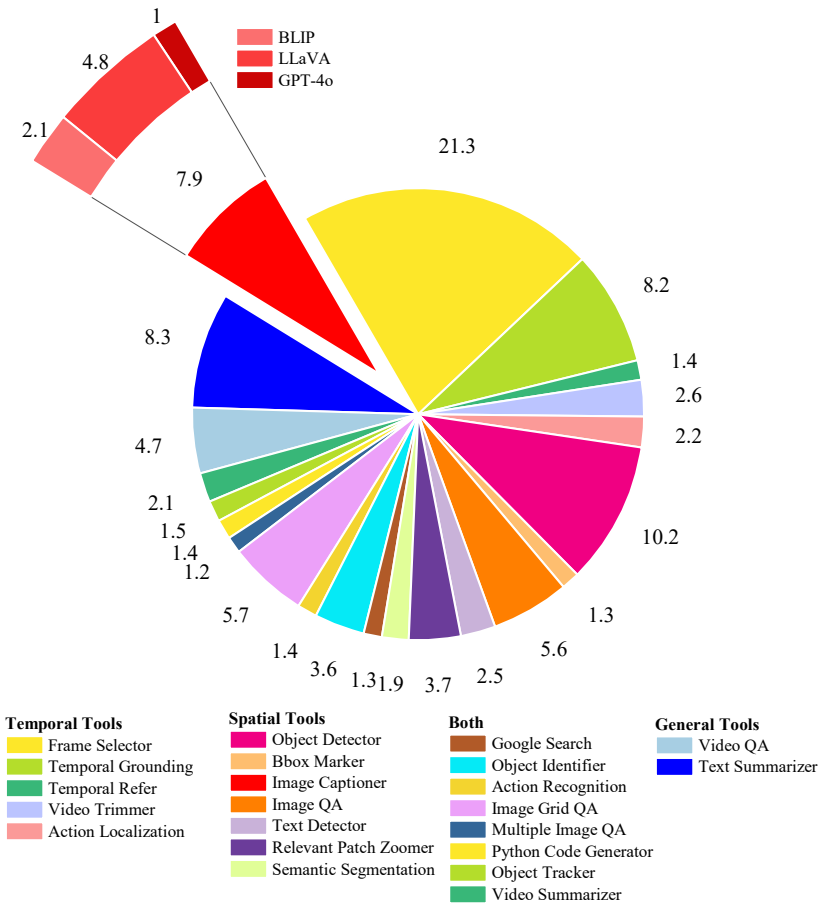


图 18: Video Toolkit 中包含了多样全面的视频工具，可以分为时间工具、空间工具与通用工具。

第二步：优化调用工具的策略

建立了工具集之后，如何更好地使用这些工具?

衡量工具“用得好不好”的指标：

- 是否多样化?
- 是否均衡?
- 工具链的长度 -> 是否能充分利用工具进行多步推理?

Tool Learning 的研究，优化工具链（即工具使用的先后顺序）往往能提升上面的指标，最终带来性能的提升。

第二步：优化调用工具的策略

Methods	Accuracy ↑	Num of Frames ↓	Toolchain Length ↑	Num of Tools ↑
No Constraints	61.2	112.6	2.9	1.3
Prompting	60.4	98.7	3.6	1.9
In-Context Learning	63.2	50.1	5.4	3.2
Spatiotemporal Disentanglement	**68.6**	**40.6**	**5.6**	**3.4**
STAR	**70.0** (1.4 ↑)	**30.2** (10.4 ↓)	**8.7** (3.1 ↑)	**6.3** (2.9 ↑)

考察了多种工具调用策略：

- No Constraints：无限制调用，常导致 Toolchain Shortcut
- Prompting：在 prompt 中“提要求”
- In-Context Learning：通过例子示范
- Spatiotemporal Disentanglement：要求模型交替调时间工具和空间工具，获得了最好的效果

第二步：优化调用工具的策略

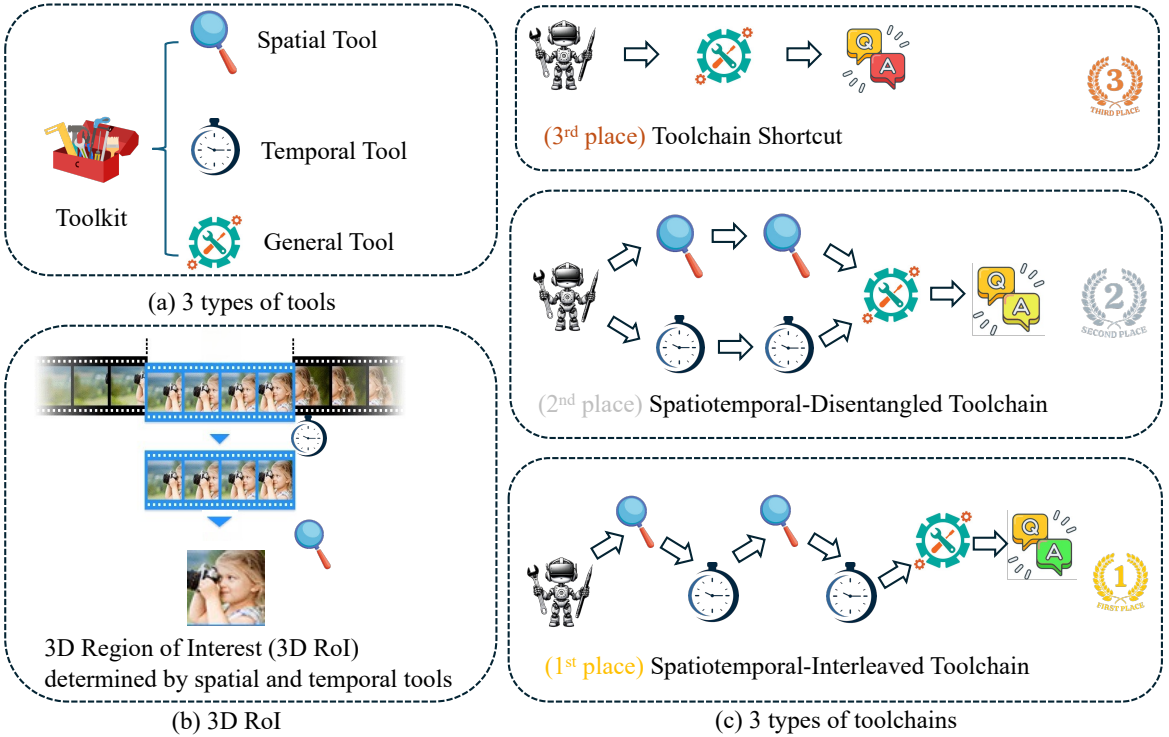
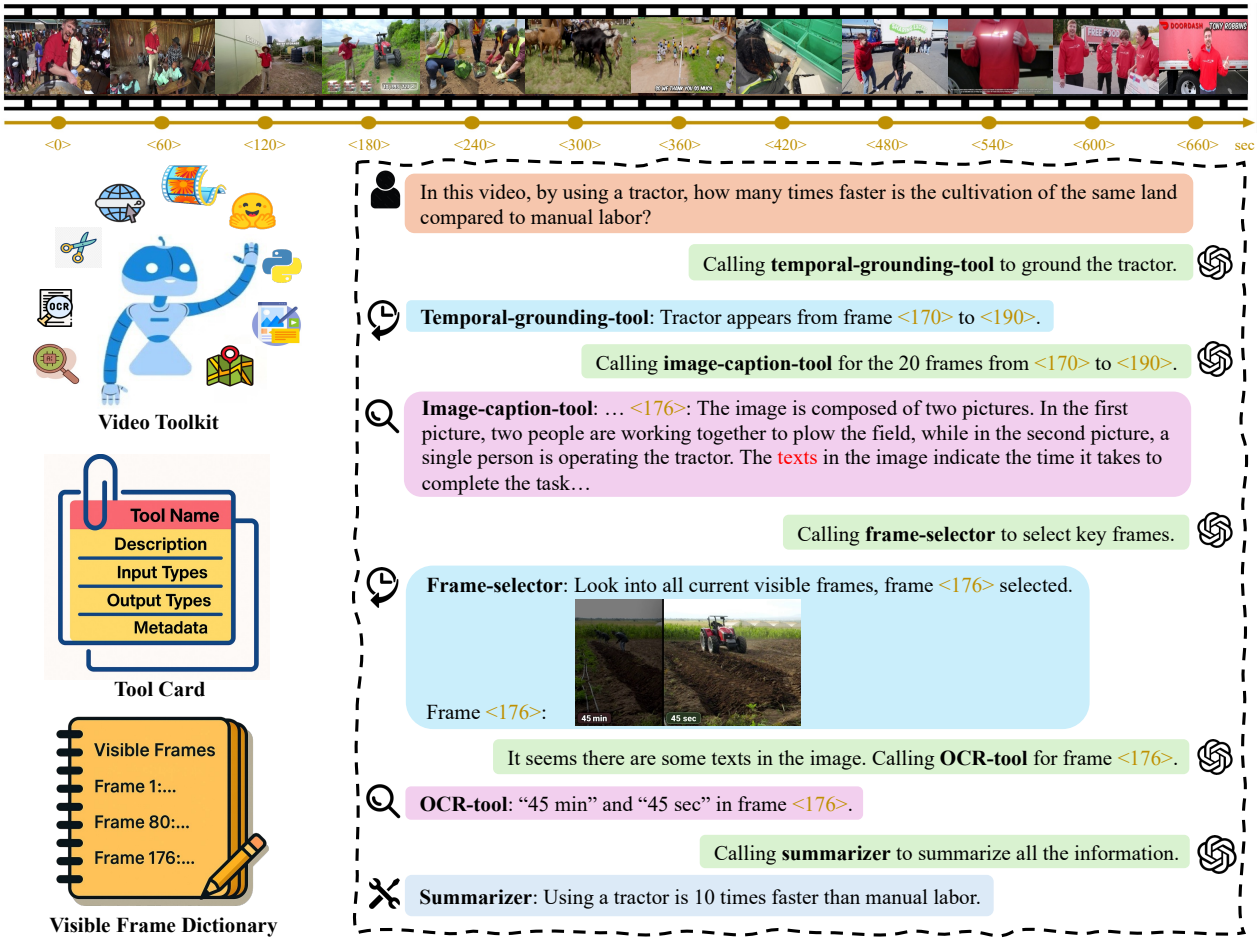


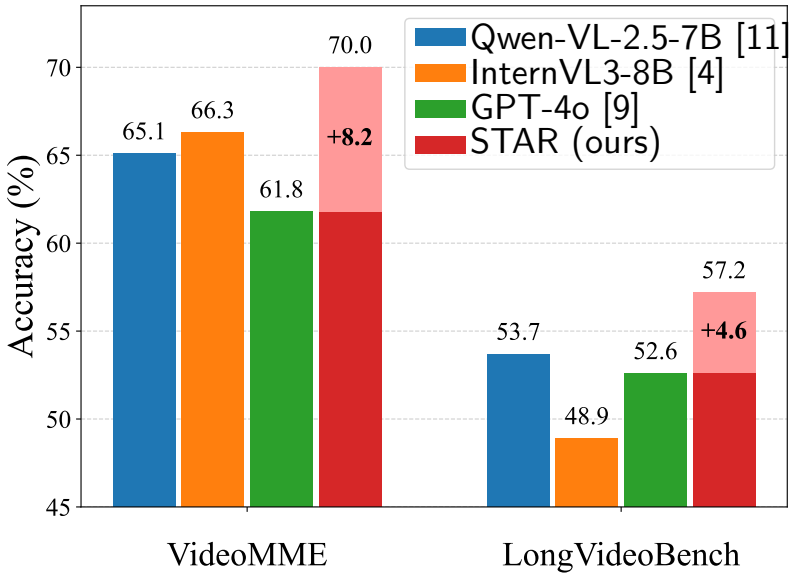
图 19: 在以上实验的基础上，从视频有时间和空间两个维度的角度切入，总结了 3 种类型的工具链：(1) 过短、走捷径的工具链 (2) 时间和空间互相解耦的工具链 (3) 时间和空间互相交替、互相利用的工具链，例如时间定位片段再做图像处理。获得的效果往往是 (3) > (2) > (1)

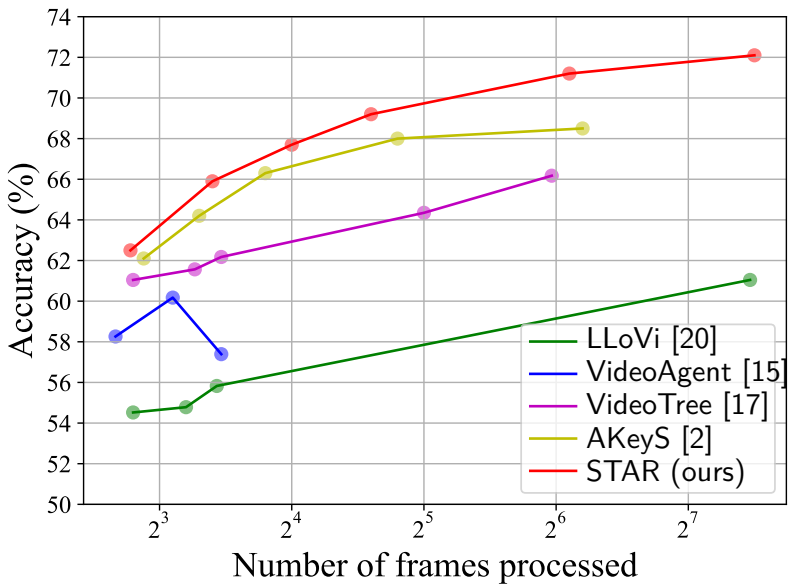
工具增强的时空组合推理框架（STAR）



要求模型交替调时间工具和空间工具，最后再使用 Video-Language Model 作为 fallback。

实验结果





左图：STAR 通过给 GPT-4o 添加轻量级工具，大幅提升准确率。
右图：STAR 具有最高的帧效率和很强的可扩展性（scalability）。

实验结果

Model	Size	Frames ↓	Runtime ↓	Short ↑	Medium ↑	Long ↑	All ↑
Proprietary Image-based MLLMs							
GPT-4o [41]	-	1 fps / 384	> 10 min	80.0	70.3	65.3	71.9
GPT-4o	-	32	**< 30 s**	68.3	60.7	56.3	61.8
GPT-4o + T^* [69]	-	32	30 s - 1 min	69.5	63.5	59.3	64.1
GPT-4v [40]	-	**10**	**< 30 s**	70.5	55.8	53.5	60.0
Gemini 1.5 Pro [15]	-	0.5 / 1 fps	> 10 min	**81.7**	**74.3**	**67.4**	**75.0**
Claude 3.5 Sonnet [1]	-	20	**< 30 s**	71.0	57.4	51.2	60.9
Open-source Video-LLMs							
Qwen2-VL [55]	72B	2 fps / 768	6 min - 8 min	**80.1**	**71.3**	**62.2**	**71.2**
Qwen2-VL	7B	-	-	-	-	-	63.3
Qwen2.5-VL [2]	7B	-	-	-	-	-	65.1
InternVL2.5 [6]	8B	**64**	**< 30 s**	-	-	-	64.2
InternVL3 [75]	8B	**64**	**< 30 s**	-	-	-	66.3
VideoLLaMA3 [71]	7B	180	30 s - 60 s	**80.1**	63.7	54.9	66.2
mPLUG-Owl3 [68]	7B	128	30 s - 60 s	70.0	57.7	50.1	59.3
STAR (ours)	-	**30.2**	**15.8 s**	**78.9**	**68.3**	**62.9**	**70.0 (8.2 ↑)**

图 20: 在 VideoMME benchmark 上取得了不错的效果：GPT-4o 在掌握工具之后，有 8.2% 准确率的提升。

实验结果

Model	Params	Frames ↓	Runtime ↓	8s-15s ↑	15s-60s ↑	180s-600s ↑	900s-3600s ↑	All ↑
Proprietary Image-based MLLMs								
GPT-4o [41]	-	256	> 10 min	**71.6**	**76.8**	**66.7**	**61.6**	**66.7**
GPT-4o	-	32	**< 30 s**	60.7	62.4	50.1	49.0	52.6
Gemini 1.5 Pro [15]	-	256	> 10 min	-	-	-	-	64.0
Open-source Video-LLMs (~7B)								
Qwen2.5-VL	7B	1 fps / 512	1 min - 3 min	**59.0**	65.6	**52.9**	**48.8**	**53.7**
InternVL3	8B	256	1 min - 3 min	54.7	**66.9**	46.1	44.0	48.9
STAR	-	**29.6**	**15.3 s**	**63.6**	**68.0**	**56.8**	**52.3**	**57.2 (4.6 ↑)**

图 21: 在 LongVideoBench 上取得了不错的效果：GPT-4o 在掌握工具之后，有 4.6% 准确率的提升。

实验结果

Method	Base Model	Frames ↓	Cau. ↑	Tem. ↑	Des. ↑	All ↑
Open-source Video-LLMs (~7B)						
InternVL3-8B [75]	-	**8**	77.5	72.1	77.1	75.7
Qwen2.5-VL-7B [2]	-	2 fps	**80.6**	**79.3**	**85.5**	**80.9**
(M)LLM-based Frame Selection Methods						
VideoAgent [56]	GPT-4	8.2	64.5	72.7	81.1	71.3
VideoTree [60]	GPT-4	63.2	70.6	76.5	83.9	75.6
LVNet [42]	GPT-4o	12	65.5	75.0	81.5	72.9
VidF4 [29]	-	8	69.6	74.2	83.3	74.1
AKeyS [9]	GPT-4o	**7.6**	**72.9**	**79.0**	**86.1**	78.1
Detector-based T^* [69]	GPT-4o	8	-	-	-	**80.4**
STAR (ours)	GPT-4o	**7.2**	**81.1**	**81.5**	**86.3**	**82.1** (**1.2** ↑)

图 22: 在 NExT-QA benchmark 上，与其它关键帧抽取算法作比较

实验结果

Method	Base Model	Frames ↓	#LLM calls ↓	Cau. ↑	Tem. ↑	Des. ↑	All ↑
LLM-driven Tool Learning Methods							
ViperGPT [49]	GPT-3 Codex	-	-	43.2	41.0	62.3	45.5
VideoChat [24]	-	-	-	50.2	47.0	65.7	51.8
DoraemonGPT [66]	GPT-3.5-turbo	28.7 fps / 1144.4	8.5	**54.7**	**50.4**	**70.3**	**55.7**
STAR-MINI (ours)	GPT-3.5-turbo	**0.6 fps / 22.6**	**5.4 (3.1 ↓)**	**62.8**	**55.1**	**73.7**	**62.0 (6.3 ↑)**

图 23: 在 NExT-QA benchmark 上，与其它工具调用算法作比较

可视化

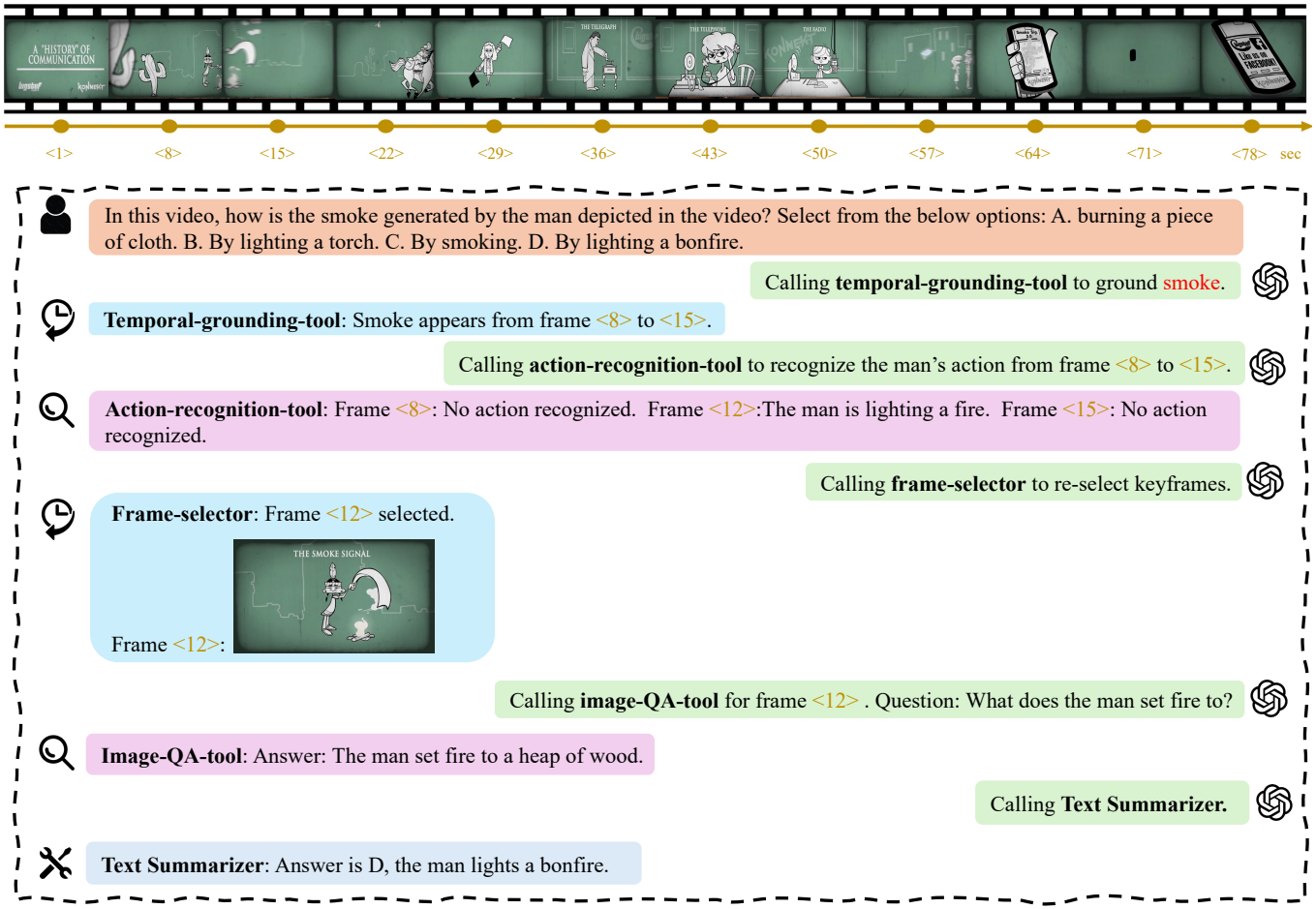


图 24: 体现了基于工具的推理（Tool-integrated Reasoning），在时间和空间上逐步逼近关键内容，从而解决问题。

STAR 是否让工具使用更加均衡?

Tool Type & Name	No Constraints (%)	STAR framework (%)
Temporal Tools All	**32.1**	**35.7**
Frame Selector	27.1	21.3
Temporal Grounding	2.1	8.2
Temporal Referring	0.0	1.4
Video Trimmer	1.3	2.6
Action Localization	1.6	2.2
Spatial Tools All	**23.6**	**33.1**
Object Detector	2.3	10.2
Bbox Marker	0.2	1.3
Image Captioner	2.1	7.9
Image QA	17.8	5.6
Text Detector	1.0	2.5
Relevant Patch Zoomer	0.2	3.7
Semantic Segmentation	0.0	1.9
Both (Temporal and Spatial Tools) All	**5.4**	**16.1**
Google Search	0.3	1.3
Object Identifier	1.0	3.6
Action Recognition	0.5	1.4
Image Grid QA	0.1	5.7
Multiple Image QA	2.8	1.2
Python_Code_Generator	0.0	1.4
Object Tracker	0.7	1.5
General Tools All	**38.9**	**15.1**
Text Summarizer	2.2	8.3
Video Summarizer	3.5	2.1
Video QA	33.2	4.7

Tool Type	Var. *of* No Constraints	Var. *of* STAR framework
Temporal Tools	134.26	69.90 (64.36 ↓)
Spatial Tools	41.34	11.09 (30.25 ↓)
Both	0.92	2.95 (2.03 ↑)
General Tools	307.45	9.69 (297.76 ↓)

图 25: 左图：各种工具使用频率百分比；右图：工具类内方差减小

scalability

Model	LLM Planner	Average Frames	Short (%)	Medium (%)	Long (%)	All (%)
GPT-4o	-	32	68.6	60.6	56.0	61.5
GPT-4o	-	100	72.8	63.2	59.5	64.9
GPT-4o	-	1 fps / 384	79.2	70.4	66.5	71.8
STAR	GPT-4o	31.3	78.2	68.7	62.7	69.6 (8.1% ↑)
STAR	GPT-4o	100.2	80.1	71.0	66.4	72.4 (7.5% ↑)
STAR	GPT-4o	0.98 fps / 384	**85.3**	**78.0**	**68.5**	**77.0** (5.2% ↑)

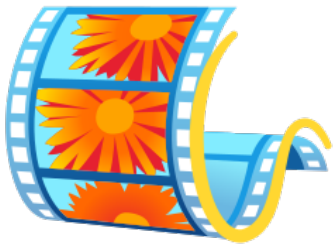
图 26: 性能随着帧数 scale

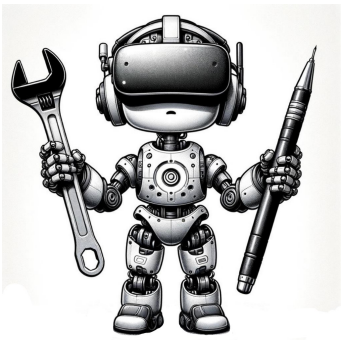
generalizability

Model	LLM Planner	Average Frames	Short (%)	Medium (%)	Long (%)	All (%)
GPT-4o	-	32	68.6	60.6	56.0	61.5
Gemini-2.5-pro	-	32	77.6	62.6	57.1	65.4
Qwen2.5-VL-72B	-	32	68.9	58.8	55.4	60.8
STAR	GPT-4o	31.3	78.2	68.7	62.7	69.6 (8.1 ↑)
STAR	Gemini-2.5-pro	**31.0**	**79.8**	**70.7**	**69.1**	**72.9** (7.5 ↑)
STAR	Qwen2.5-VL-72B	31.5	77.9	66.1	62.4	68.5 (7.7 ↑)
STAR	DeepSeek-R1	31.2	77.2	67.2	63.0	68.9

图 27: 能泛化到不同的 LLM Planner

总结





Video Agent 目标：提升 VideoQA 的准确率和效率。

- Planning：使用树状关键帧搜索算法 AKeyS，化整为零
- Tools & Memory：建造了 Video Toolkit，使用 STAR 组合推理框架，化繁为简

两者之间的联系：STAR 框架中的时间工具 Frame Selector 是依据 AKeyS 的算法逻辑。

未来工作：更复杂的视频-语言理解任务

- 更长、更复杂的视频理解（例如电影级别）
- 能否做到在线（online）、即时（real-time）推理?
- 机器如何从视频 (例如课程录像、使用教程) 中获取知识?
- 视觉智能体（visual agent ）能否跟随视频完成相应任务?
- 面对文本数据的匮乏，能否从视频教程中提取训练数据或轨迹（trajectory），来训练提升模型或智能体的表现?

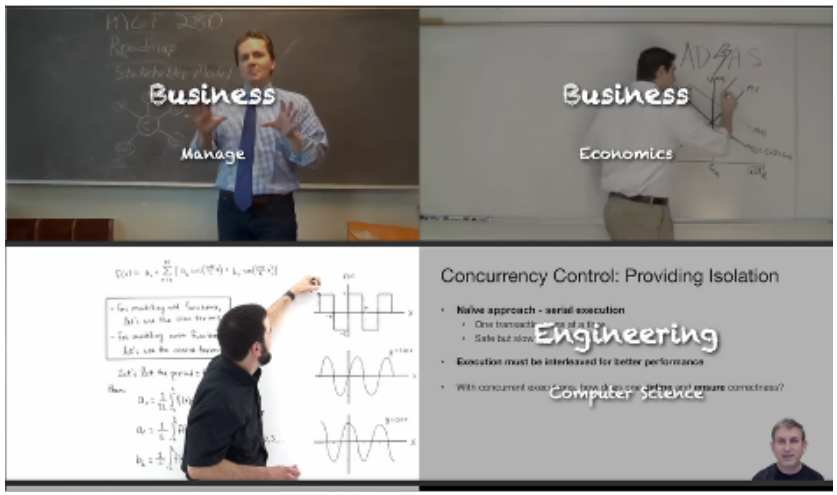


图 28: 人类大量的视频教程可作为 AI 数据来源

图 29: 可能的研究场景：GUI Agent

[1] R. Choudhury, K. Niinuma, K. M. Kitani, and L. A. Jeni.
Zero-shot video question answering with procedural programs, 2023.

[2] S. Fan, M.-H. Guo, and S. Yang.
Agentic keyframe search for video question answering, 2025.

[3] Y. Fan, X. Ma, R. Wu, Y. Du, J. Li, Z. Gao, and Q. Li.
Videoagent: A memory-augmented multimodal agent for video understanding, 2024.

[4] InternVL Team.
Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models, 2025.

[5] K. Kahatapitiya, K. Ranasinghe, J. Park, and M. S. Ryoo.
Language repository for long video understanding, 2024.

[6] W. Kim, C. Choi, W. Lee, and W. Rhee.
An image grid can be worth a video: Zero-shot video question answering using a vlm, 2024.

[7] K. Mangalam, R. Akshulakov, and J. Malik.
Egoschema: A diagnostic benchmark for very long-form video language understanding, 2023.

[8] J. Min, S. Buch, A. Nagrani, M. Cho, and C. Schmid.
Morevqa: Exploring modular reasoning models for video question answering, 2024.

[9] OpenAI.
Gpt-4o system card.
https://openai.com/index/gpt-4o-system-card/, 2024.

[10] J. Park, K. Ranasinghe, K. Kahatapitiya, W. Ryoo, D. Kim, and M. S. Ryoo.
Too many frames, not all useful: Efficient strategies for long-form video qa, 2024.

[11] Qwen Team.
Qwen2.5-vl technical report, 2025.

[12] K. Ranasinghe, X. Li, K. Kahatapitiya, and M. S. Ryoo.
Understanding long videos with multimodal language models, 2025.

[13] Y. Shang, B. Xu, W. Kang, M. Cai, Y. Li, Z. Wen, Z. Dong, K. Keutzer, Y. J. Lee, and Y. Yan.
Interpolating video-llms: Toward longer-sequence lmms in a training-free manner, 2024.

[14] S. Wang, Q. Zhao, M. Q. Do, N. Agarwal, K. Lee, and C. Sun.
Vamos: Versatile action models for video understanding, 2024.

[15] X. Wang, Y. Zhang, O. Zohar, and S. Yeung-Levy.
Videoagent: Long-form video understanding with large language model as agent, 2024.

[16] Y. Wang, Y. Yang, and M. Ren.
Lifelongmemory: Leveraging llms for answering queries in long-form egocentric videos, 2024.

[17] Z. Wang, S. Yu, E. Stengel-Eskin, J. Yoon, F. Cheng, G. Bertasius, and M. Bansal.
Videotree: Adaptive tree-based video representation for llm reasoning on long videos, 2024.

[18] J. Xiao, X. Shang, A. Yao, and T.-S. Chua.
Next-qa:next phase of question-answering to explaining temporal actions, 2021.

[19] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao.
React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

[20] C. Zhang, T. Lu, M. M. Islam, Z. Wang, S. Yu, M. Bansal, and G. Bertasius.
A simple llm framework for long-range video question-answering, 2024.

Thanks!