MoReVQA: Exploring Modular Reasoning Models for Video Question Answering

Juhong Min[1,2*] Shyamal Buch[1] Arsha Nagrani[1] Minsu Cho[2] Cordelia Schmid[1]
[1]Google Research [2]POSTECH[†]
http://juhongm999.github.io/morevqa

Abstract

This paper addresses the task of video question answering (videoQA) via a decomposed multi-stage, modular reasoning framework. Previous modular methods have shown promise with a single planning stage ungrounded in visual content. However, through a simple and effective baseline, we find that such systems can lead to brittle behavior in practice for challenging videoQA settings. Thus, unlike traditional single-stage planning methods, we propose a multi-stage system consisting of an event parser, a grounding stage, and a final reasoning stage in conjunction with an external memory. All stages are training-free, and performed using few-shot prompting of large models, creating interpretable intermediate outputs at each stage. By decomposing the underlying planning and task complexity, our method, MoReVQA, improves over prior work on standard videoQA benchmarks (NExT-QA, iVQA, EgoSchema, ActivityNet-QA) with state-of-the-art results, and extensions to related tasks (grounded videoQA, paragraph captioning).

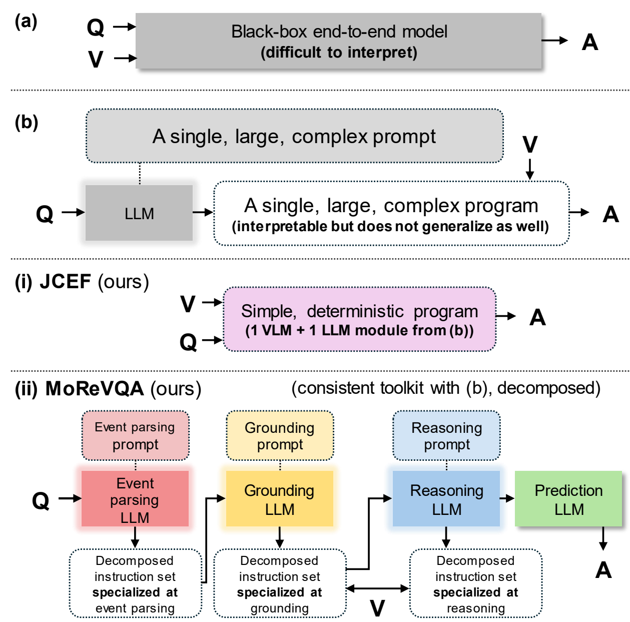

Figure 1. **MoReVQA: a new multi-stage, modular reasoning model for videoQA**. Prior work relies on either (a) black-box end-to-end models that are difficult to interpret, or (b) modular systems where an interpretable planning step (program generation) is done in a single, ungrounded stage. (i) In this work, we find that single-stage planning leads in practice to brittle behavior, underperforming a new simple baseline (JCEF) that captions frames and predicts an answer (with two modules from (b)). (ii) We then introduce our new MoReVQA method incorporating both *modularity* and *multi-stage planning*, providing interpretable, grounded planning and execution traces, while simultaneously delivering improvements in overall accuracy by effectively decomposing the underlying task complexity (still using consistent base models with (b)). Above: Q is question, V is video, A is answer.

1. Introduction

The predominant approach for solving video understanding tasks such as video question answering (videoQA) has long been end-to-end networks [1, 6, 7, 38, 42, 43]. A major challenge with such methods, however, is their black-box nature – leading to a lack of interpretability and compositional generalization. For videos in particular, an important desired capability is the ability to understand events at different temporal scales, which is challenging for existing end-to-end vision-language models (VLMs) that typically see only a few frames [6, 7, 42]. This has led to a recent interest in modular or programmatic approaches [22, 39, 40] to solve such problems, particularly leveraging the success of large language models (LLMs) [10, 41, 63] which have shown impressive reasoning and planning capabilities. These methods generate symbolic programs [39, 40] using an LLM capable of producing code. They are interpretable and can be executed directly (leveraging independent visual or language processing modules). Their advantages are that they are training-free, compositional, and achieve impressive performance on few-shot vision and language tasks.

[1]Work done while Student Researcher intern at Google Research.
[2]Pohang University of Science and Technology

In this paper, we analyze the performance of such methods in closer detail, particularly for the case of videoQA (across 4 datasets [33, 47, 51, 58], representative of a range of video domains, lengths, and question types) and for single stage modular frameworks (such as ViperGPT [40]). We find that, while recent modular approaches building on large, state-of-the-art end-to-end networks (LLMs and VLMs) as modules have shown significant promise [40], a simple Socratic [59] baseline, which we call Just Caption Every Frame (JCEF), based on the *same* underlying models, can actually *outperform* these prior approaches by a significant margin. As the name suggests, JCEF simply captions every frame in the video using a large vision-language model (VLM) [8], and then feeds all captions along with the question to an LLM to produce an answer (Fig. 1(i) and Fig. 2). We hypothesize that the reason this baseline outperforms prior work is that these modular frameworks (Fig. 1(b)) consist of a *single planning stage* which may be ungrounded in the video (*i.e.* the entire program or set of steps to be executed is determined in a single stage directly from the language prompt alone), and hence in practice the single-stage planner must be prompted with a large space of complex combinations required for answering diverse questions in video [27]. While the performance of JCEF is impressive, it is less interpretable than the previously mentioned modular approaches, as captions for each frame tend to be generic and not question-specific (Fig. 2).

In this work, we propose a decomposed, modular, and multi-stage approach for video question answering to address these limitations (Fig. 1(ii) and Fig. 3). Our method consists of three key planning and execution stages: (1) *event parsing* that explicitly decomposes the events in the question, (2) *grounding* that identifies corresponding temporal regions in the video that merit further tool use (so that every single frame does not have to be processed in detail), and (3) *reasoning* that gives the final answer after considering the outputs of composed modules/APIs and the shared memory. This decomposition of single-stage planning is motivated by natural sub-tasks for videoQA and related video-language reasoning tasks. All stages are training-free, and involve few-shot or zero-shot prompting of off-the-shelf modules (consistent with the API behavior in single-stage planning methods), in conjunction with an external read/write memory that maintains state and enables a more flexible design. We call our method Modular Reasoning for Video Question Answering (MoReVQA), and show that it outperforms JCEF and other key single-stage modular baselines, while providing a grounded, interpretable planning and execution trace.

We summarize our key contributions as follows: **(1)** We find that existing single-stage code-generation frameworks, while being modular and interpretable, are not necessarily well-suited for the complexity of generalizable VideoQA, and can be outperformed by a simple baseline we propose using a subset of its tool components (*e.g.* a large VLM and LLM), **(2)** we design a multi-stage modular reasoning system (MoReVQA) that alleviates this issue by decomposing the underlying planning sub-tasks effectively, and **(3)** we achieve state-of-the-art results on four standard videoQA benchmarks (NExT-QA, iVQA, EgoSchema, and ActivityNet-QA) across training-free (zero-shot/few-shot) methods, in some cases even outperforming fully-finetuned prior work. We also consider extensions to grounded videoQA (NExT-GQA [48]) and paragraph captioning (ActivityNet-Para [28]) with strong performance.

2. Related Work

VideoQA. Video Question-Answering (videoQA) is a key task for multimodal video understanding systems to assess their ability to reason about a video [33, 47, 49, 51, 64]. Recent benchmarks have pushed towards assessing reasoning for temporal questions [21, 46, 47], longer videos [33, 58], and on domains like instructional [51] and egocentric videos [16, 33]. We evaluate our modular approach on four diverse and representative videoQA tasks: NExT-QA [47], iVQA [51], EgoSchema [33], and ActivityNet-QA [58].

End-to-end Models for VideoQA. The recent success of LLMs [10, 19, 41, 63] has led to an explosion of multimodal models that jointly understand vision and text data. Many works map frozen image encoders [13, 14, 37] to the LLM textual embedding space: *e.g.*, Flamingo [1], via a Perceiver resampler [25], or BLIP2 [29] and Video-LLaMa [61], via Q-formers for audio/vision [14, 18]. GIT2 [42] and PALI [6–8] use simple encoder-decoder style architectures which are trained for image captioning, while MV-GPT [38] finetunes a native video backbone [3] for video captioning. Although trained with a generative (captioning) objective, such models achieve strong results for general vision-language tasks (cast as auto-regressive generation with question as prefix). More recent works such as Instruct-BLIP [11], MiniGPT-4 [66], and VideoBLIP [56] improve zero-shot results with strong instruction tuning. Generally, however, end-to-end methods can be difficult to interpret.

For videos in particular, memory limits in end-to-end models require significant downsampling: *e.g.* temporally sampling a few frames with large strides [7, 42], spatially subsampling each frame to a single token [43, 53, 65]. Such models also tend process each frame with equal importance. Unlike such works, our model has an explicit grounding stage which searches for the most relevant video frames to be processed in more detail. Other grounding works for videoQA include SeViLa [57], MIST [17], and NExT-GQA[48]; our model differentiates from these prior works by incorporating modular multi-stage reasoning.

Visual Programming and Modularity. Visual programming methods [2, 9, 22, 26, 39, 40] have shown promise towards addressing the limitations of end-to-end systems, by composing multiple sub-task specific modules into an executable program. Prior (earlier) work on neural modular networks [2, 26] made initial progress towards this goal, but were eventually outpaced by developments in large-scale end-to-end models. Recent work like CodeVQA [39], ViperGPT [40], and VisProg [22] demonstrated accuracies on par with some end-to-end systems [29], by replacing the event/language parsing with a code-finetuned LLM that can generate an entire python program (which invokes a number of provided APIs in the prompt). While these approaches are effective in terms of interpretability and flexibility in solving VQA, they share common limitations in that they heavily rely on a 'single-prompt' with large, complex code generation examples [27], which must generate the entire program without access to the image.

Multistage Planning Models. Recent methods have explored directly using natural language as the intermediate representation between large multimodal models. One emerging class of models are Socratic models [59], which use few-shot or zero-shot prompting of LLMs and VLMs to solve video tasks, *e.g.* VidIL [45] which feeds image captions, frame attributes and ASR to an LLM to perform video-language tasks. The closest to our work is AVIS [23], which also uses multistage LLMs with an external memory for the task of visual question answering. However unlike AVIS which works on knowledge focused QA for images, we focus on the more challenging domain of videos, where reasoning over multiple frames is required. A key difference therefore is our grounding stage, which determines which frames in a (potentially long) video contain the most relevant information to then deploy additional reasoning steps and tools over.

3. Technical Approach

In this section, we contextualize our technical approach for videoQA (Sec. 3.1) by discussing limitations in the standard single-stage (Sec. 3.2) paradigm before presenting our main multistage modular reasoning model MoReVQA(Sec. 3.3).

3.1. Preliminaries: Video Question-Answering

Task. We focus on the task of video question-answering (VideoQA) as it provides a good testbed for video reasoning for multimodal systems. Formally, we are given an input video $V = \{v_1, \ldots, v_l\}$ with l frames and a corresponding question Q in natural language with a groundtruth answer A, either directly from the question alone [51, 58], or from among a set of candidate options $A \in A_{cands}$ [33, 47]. The task is to develop a model M such that:

$$M(V, Q, [A_{cands}]) \mapsto A \tag{1}$$

where A_{cands} are present for closed-set VQA settings [33, 47] and not present for open-ended VQA [51, 58].

Design Approaches. The approaches for addressing this task can vary broadly (Sec. 2 and Fig. 1); here, we center our discussion around two key design principles in state-of-the-art systems for M: (1) *Modularity*, where individual, standalone modules focused on specific sub-tasks are leveraged, as opposed to a single monolithic black-box model; and (2) *Multi-stage planning*, where there are explicit intermediate outputs while the system determines *which* modules to leverage and *how* to use them most effectively, providing a more interpretable chain of execution. In this section, we focus on contrasting modular methods with single-stage (prior work) vs. multi-stage planning (our new model).

3.2. Single-Stage Planning

Overview. In Section 2, we discuss the broader space of visual programming and modular methods [26, 39, 40]. Here, we focus on a specific representative state-of-the-art model (ViperGPT [40]) and discuss key limitations with its single-stage planning approach for modular videoQA, using notation consistent with prior work [26, 40].

ViperGPT. In the context of videoQA, ViperGPT is a system M that consists of a single-stage program generator π that takes as input the query Q and a specialized prompt P to directly output an intermediate executable program $z \in Z$, where Z represents the space of all programs (Python, natural language, etc.). This program z is then executed on the full input $(V, Q, [A_{cands}])$ to produce the final answer A. More formally, the full system can be described:

$$M_{\text{single-stage}} : \pi(Q, P) \mapsto z(V, Q, [A_{cands}]; L) \mapsto A \tag{2}$$

where L denotes the API module library used to construct the program z. The program generator π is instantiated as a code-finetuned LLM [5, 19] conditioned on a well-engineered prompt file P, consisting of two key components: (1) a custom API description with API usage examples, and (2) a set of dataset-specific program examples that illustrate how to translate the questions Q found in the dataset distribution into a full program z that composes these modules together effectively.

Modules for Modular Reasoning. ViperGPT and related models [22, 39, 40] leverage a specialized module library as described by their API to assemble executable programs z. We denote this library of API modules as $m \in L_{API}$: examples include open-vocabulary detection (OWL-ViT [35]), text-image scoring (CLIP [37], X-VLM [60]), and captioning (BLIP [29]). The overall program z then describes the modular reasoning of the single-stage code generation LLM for a given query.

Limitations. While a single-stage approach suggests an appealing promise of simplicity, in practice, we observe that this design leads to brittle programs that do not produce reliable outputs[1]. We show a representative example for videos

[1] Also noted by concurrent analysis [27] in the image-language domain.

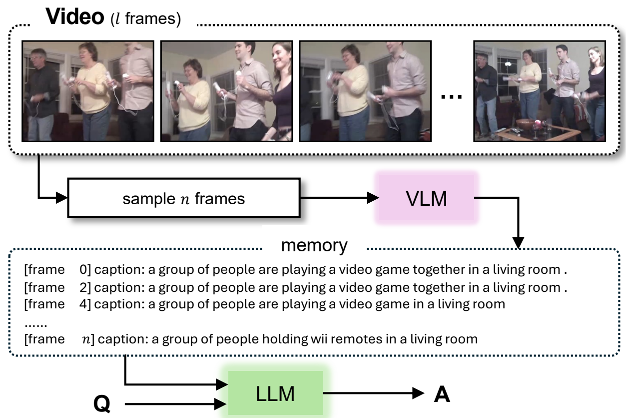

Figure 2. **A simple, strong baseline –** JCEF. Our proposed baseline consists of a zero-shot prompted vision-language model (VLM) which is used to caption n uniformly sampled frames from a video (n is all frames at 1FPS unless explicitly stated). These captions are then stored in an external memory, which is passed to a zero-shot prompted LLM that is used to answer a question about the video. We show that this baseline outperforms existing visual programming methods by a large margin and investigate ways to more effectively improve upon it in a modular, multistage manner.

in Fig. 4, with additional analysis in the supplement.

The core issue of the overall system lies in the difficult task given to its single-stage planner: before performing visual reasoning, the model must output a full program without any grounding in the actual video itself. Thus, natural language ambiguity in the question cannot be resolved by visual context, important for video / event reasoning tasks [24, 48]. Furthermore, by expecting the model to piece together full reasoning programs in one large LLM inference step, the necessary complexity of examples in the prompt grows accordingly. In practice, we observe this leads to the system overfitting on the specific examples provided (also noted by [27]), falling short of realizing its true potential for compositional modular generalization.

These limitations naturally beget *two key questions*: **(Q1)** to what extent is brittle single-stage planning impacting accuracy, and **(Q2)** how well can we overcome these limitations through a *multi-stage decomposition*? These motivate our proposed baseline and model in Sec. 3.3.

3.3. MoReVQA: Multi-stage, Modular Reasoning

Motivation: A Simple, Strong Baseline (JCEF). To empirically characterize the limitations of single-stage approaches (per Q1), we create a simple but effective Socratic baseline called Just Caption Every Frame (JCEF) (Figure 2), consisting of two strong modules m_{VLM} [8] and m_{LLM} [19]. Our baseline involves no training, directly prompting these large off-the-shelf models in what can be considered as a very simple, deterministic "program". For each video, we sample $n \leq l$ frames from the video V, captioning each frame with an image-language model m_{VLM}. These n captions are then combined with frame numbers (e.g., "[frame 5] caption: a person is throwing a baseball in a field") into a prompt P used to query the LLM m_{LLM} with the question Q and candidate answers A_{cand} for multiple choice questions (prompt details in supplement). By comparing to a state-of-the-art baseline (ViperGPT+), upgraded with the *same* modules m_*, we can observe the limitations of single-stage planning designs: surprisingly, JCEF *outperforms* this single-stage baseline (Sec. 4).

MoReVQA Overview. We address our second key question (Q2) by considering a *decomposition* of the single-stage pipeline into multiple stages, in order to effectively improve *beyond* our JCEF baseline. Our new proposed model, multi-stage modular reasoning for videoQA (MoReVQA), consists of three stages, rooted in key subtasks that are general to videoQA (and related video-language reasoning tasks) across benchmarks and domains: (1) *event parsing* (understanding what is relevant in the input language), (2) *grounding* (understanding what is relevant in the input video), and (3) *reasoning* (understanding the relevant events, their attributes, and their relationships).

An overview of the pipeline is provided in Fig. 3. Each stage is distinct yet interconnected, employing an LLM that generates a set of API calls tailored for the specific subtasks. Importantly, these APIs are backed by the same off-the-shelf pretrained models [8, 35, 37] considered in the single-stage setting (Sec. 3.2) for consistent comparison. Central to this process is a shared external memory, managing and storing information across stages, including natural language events, grounded regions of the video, video captions, and intermediate tool outputs (details in Sec. 4.3).

Through this decomposition, our MoReVQA model $M_{\text{multi-stage}} = \{M_1, M_2, M_3\}$ relies on smaller focused prompts $\{P_1, P_2, P_3\}$ for each stage[2]; furthermore, intermediate reasoning outputs $\{z_1, z_2, z_3\}$ are able to handle different aspects of the overall task, and incorporate grounding in the video itself to resolve ambiguities and inform new *intermediate* reasoning steps in a more effective manner than the ungrounded single-stage setting. We describe each stage M_i as follows:

Event parsing stage M_1. The first stage focuses on the initial analysis and processing of the input question Q. Different from traditional language parsing in early modular systems [2, 26], our M_1 stage parses at the *event*-level rather than word-level, focusing on higher-level video semantics while still decomposing relationships and attributes for later stages. Our event parsing prompt P_1 (see supplementary) conditions the LLM to examine the input question, perform parsing tasks such as detecting temporal hints and relationships ("in the beginning of the video", "before", "during"), sub-question types (location, description, explanation), and whether the language would suggest additional

[2]Please see supplement for prompts and API details.

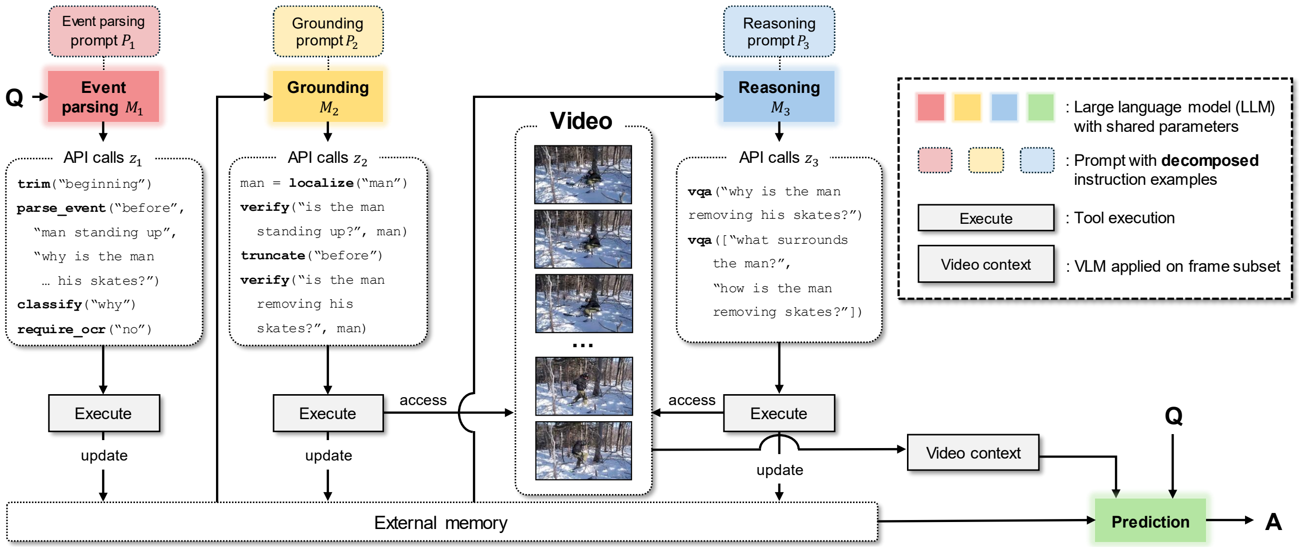

Figure 3. **Modular Reasoning for Video Question-Answering (MoReVQA).** To address the limitations of single-stage planning LLMs, we propose a new multi-stage, modular method $M_{\text{multi-stage}}$ that decomposes planning and execution into three key steps, motivated by sub-tasks inherent to videoQA: (i) event parsing M_1, (ii) grounding M_2, and (iii) reasoning M_3. See Section 3.3 for additional details.

tool types (*e.g.* OCR). The LLM then produces a set of API calls based on these parsing tasks, expressed as z_1, which when executed populates the external memory with relevant language-only data for later stages.

Grounding stage M_2**.** In this stage, the focus shifts to *grounding* identified events, a critical process to help resolve ambiguities and direct tool-use in the final reasoning stage to the temporal regions where they can be most effective. Here, the prompt P_2 is constructed with the external memory state with outputs from M_1 (*e.g.*, parsed events), and conditions the LLM to identify candidate frames and temporal regions in the video with vision-language modules m for entity detection and image-text alignment. The resulting z_2 is then executed on the video, and the output grounding (spatially and temporally) is appended to the external memory. Importantly, this process includes API calls designed to help verify and resolve event ambiguity through visual grounding, as illustrated in Fig. 3.

Reasoning stage M_3**.** The final stage of our system performs grounded reasoning before the final prediction. The LLM prompt P_3 is based on the memory state after the previous two stages, and constructs a final z_3 executable with API calls (Fig. 3) designed around reasoning sub-questions to unravel different aspects of the original question, and focusing vision-language modules on the specific grounded regions of the video identified previously. This localized, context-specific information is subsequently combined with a more general $n \leq l$ captions from frames sampled uniformly (*general video context*, in Fig. 3) across the video to form a comprehensive (temporally-sorted) basis for a final *prediction* LLM to output the final answer A (in general, n here is significantly less than with JCEF). This final API call here corresponds to the standard `llm_query` module found in prior work [40], typically at the end of the program to ensure correct formatting and candidate answer selection.

Flexibility and Memory. The modular architecture of MoReVQA allows it to be dynamically tailored to a wide range of datasets, question types, and tasks by selectively engaging different APIs and reasoning strategies based on the task at hand. In particular, simple questions beget a "simpler" execution pipeline (stages are equipped with "no-op" like behavior, if necessary), while more complex questions are processed with a complex instruction set. This adaptability is facilitated by the external memory component, which not only serves as a repository of information across stages but also enables the system to iteratively refine its understanding and approach based on the evolving context. We highlight that each stage (planning and execution) are informed by previous stages through this memory, which leads to more robust reasoning behavior.

4. Experiments

Here, we describe the VideoQA datasets and evaluation metrics used (Sec. 4.1), our key baselines (Sec. 4.2) and implementation details (Sec. 4.3), and our discussion of results and analysis (Sec. 4.4).

4.1. Datasets and Evaluation Metrics

We consider four standard videoQA benchmarks to assess the efficacy of our proposed method, across a range of representative video domains, lengths, and question types.

NExT-QA [47] is focused on understanding the ability of videoQA systems to effectively answer questions across three types: causal (C), temporal (T) and descriptive (D). We focus on the same multiple choice (MC) setting reported in prior single-stage modular reasoning work [40], where each video clip (avg length, 43s) contains one question and 5 candidate answers; we use 4996 val video-question pairs.

iVQA [51] consists of 7-30s instructional video clips sampled from the HowTo100M dataset [34], with 5615 training and 1879 testing clips (after removing clips no longer online). Each clip has a question and annotated set of ground truth answers. We note that iVQA is open-ended (OE) videoQA, and no candidate answers are provided as input.

EgoSchema [33] is a recent dataset of *long* egocentric videos (180s) based on the Ego4D [20] benchmark with multiple-choice (MC) questions, designed specifically to assess long video understanding. EgoSchema is focused entirely on evaluation: the hidden test set consists of 5000 videos via an evaluation server, of which 500 were publicly released for validation. We report results (accuracy) on the main (full, 5k) test set for comparison with prior work.

ActivityNet-QA [58] has 5800 videos, each accompanied by 10 annotated question-answer pairs to characterize model comprehension of actions, objects, locations, and events. ActivityNet-QA is open-ended (like iVQA) with long videos (180s avg., like EgoSchema). We report test set results using GPT-based evaluation following [30, 32, 62].

4.2. Baselines

We compare our method against a key set of baselines:

Single-stage Planning (ViperGPT+). As a representative state-of-the-art baseline for single-stage planning and modular reasoning, we reimplement ViperGPT[40], as described in Sec. 3.2. We extend the open-source implementation and upgrade some of the modules/APIs to ensure consistent comparisons with our method and to replace prior module components that are not available (eg. GPT-3 Codex [5]); full description in the supplement. We evaluate this baseline on video datasets that were not used in the original paper (iVQA, EgoSchema, ActivityNet-QA) to better characterize single-stage planning for videoQA.

Just Caption Every Frame (JCEF). We also consider our JCEF baseline described in Sec. 3.3 as a simple but powerful Socratic model that is a step up in interpretability to a purely end-to-end system, but lacks the kind of modular compositionality that is present in more fully fledged modular reasoning systems. The VLM and LLM models used here are the same as with ViperGPT+ and our full system, for consistent comparison (details in Sec. 4.3).

Language-only baseline. We also compare our model with a language-only baseline, which is an LLM [19] prompted to answer questions without any visual inputs, as a way to quantify the amount of non-visual language and/or common sense bias in each dataset. For consistent comparison, this language model is used across all modular methods.

Method	Accuracy (%)			
	NExT-QA	iVQA	EgoSchema	ActivityNet-QA
Random (for MC)	20.0	-	20.0	-
LLM-only [19]	48.5	15.0	41.0	-
ViperGPT [40]	60.0	-	-	-
ViperGPT+	64.0	46.6	49.3	37.1
JCEF	<u>66.7</u>	<u>56.9</u>	<u>49.9</u>	<u>43.3</u>
MoReVQA	**69.2**	**60.9**	**51.7**	**45.3**

Table 1. **Comparison to single-stage modular methods.** ViperGPT [40] represents the state-of-the-art single-stage modular question answering system, and ViperGPT+ is our upgraded reimplementation for consistent comparison. Our JCEF strong performance highlights the relative weakness in single-stage planning models, which can lead to brittle programs and outputs. We find that our MoReVQA model outperforms all key baselines.

4.3. Implementation Details

Across all of our baselines and proposed models (*e.g.*, MoReVQA, JCEF, ViperGPT+), our core VLM is PALI-3 (5B) [6] for image captioning and related APIs, and our core LLM is PaLM-2[19] (*e.g.*, every LLM stage in MoReVQA, JCEF, the language-only baseline, and for the `llm_query` module in ViperGPT+), unless otherwise specified. Our video context / captioner component for MoReVQA considers $n = 16$ uniformly sampled video frames as a default. For JCEF, the default is set to $n = l$, the number of frames in the video (at 1 frame per second); we provide additional JCEF ablations for different values of n in the supplement. We set decoding temperature to 0 to match prior work [40]; other base models and settings (*e.g.*, OWL-VIT [35], CLIP [37], etc.) for MoReVQA and ViperGPT+ are in the supplement and are also consistent wherever applicable. Our implementation relies on JAX/Scenic [4, 12].

MoReVQA stores outputs of each stages in an external memory system, backed by global variables for tracking and updating information through the model's processing stages. These stages execute different API calls, *e.g.*, event parsing reduces frame data for efficiency, the grounding stage focuses on object localization and action verification, and the reasoning stage decomposes and addresses the question with VQA on selected frames. Additional API, memory, and LLM prompt details for MoReVQA and other models are provided in supplementary.

4.4. Results and Discussion

4.4.1 Comparison to Baselines and Analysis

We compare our method to the baselines outlined in Sec. 4.2 in Table 1. The LLM-only baseline performance serves as a measure of language and commonsense bias: our results for NExT-QA and iVQA align with prior expectations (*e.g.*, iVQA was explicitly designed to mitigate language bias). However, this baseline also does surprisingly well on EgoSchema, in spite of its explicit emphasis on testing

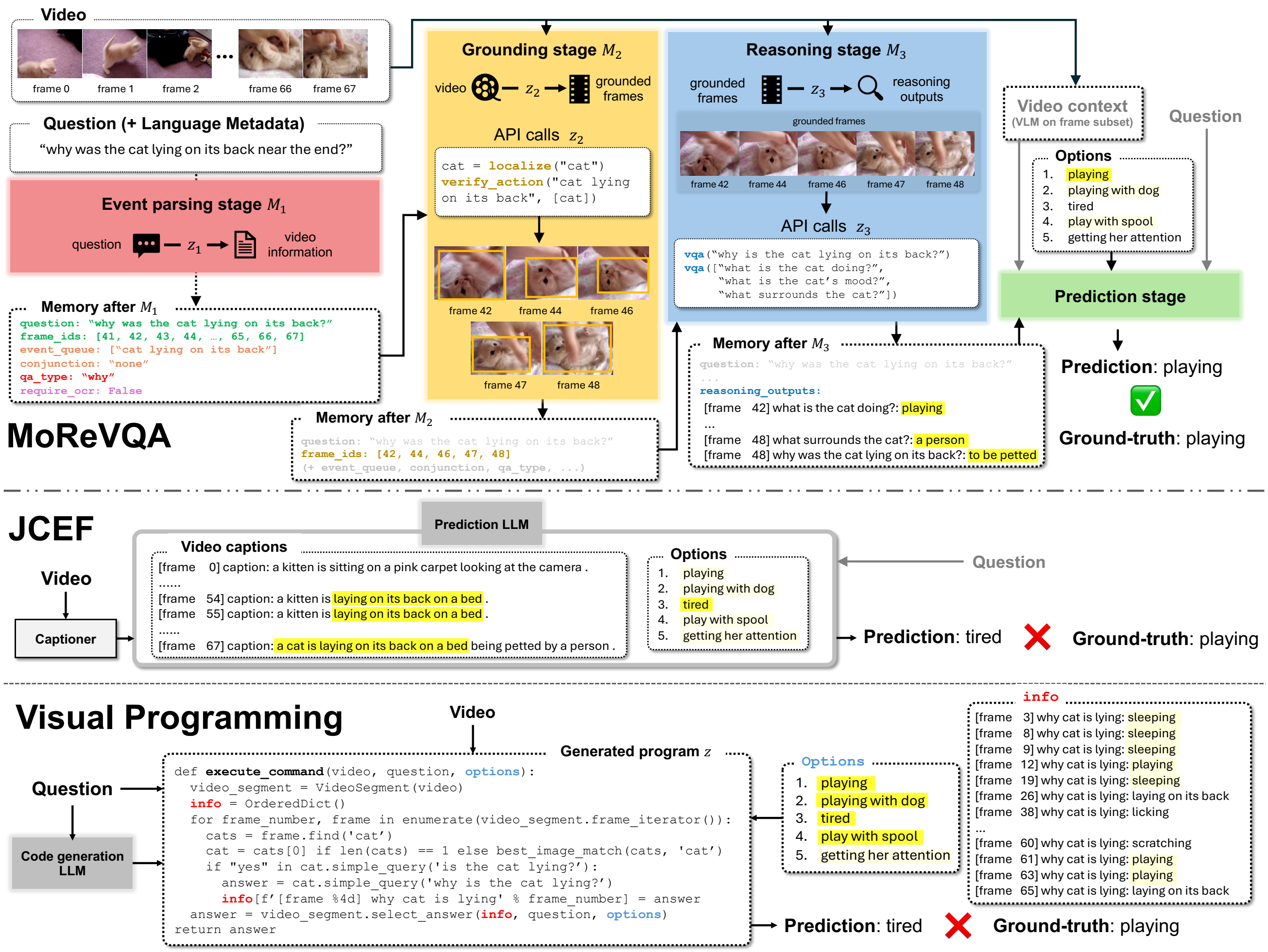

Figure 4. **Example qualitative result of MoReVQA** on NExT-QA. We observe that the intermediate outputs from our MoReVQA model are interpretable: event parsing stage parses key events from language, and other tool-use metadata. The grounding stage then determines which frames contain the 'cat lying on its back', and the reasoning stage reasons about relevant sub-questions for the final answer, which when combined with general video-level context (subset of frame captions), gives us the final correct answer. We observe that JCEF and ViperGPT+ fail to predict correct answer for the same example (Sec. 4.4.1); we provide additional examples and analysis in the supplement.

Stages			NExT-QA	iVQA
Event parsing	Grounding	Reasoning	Val	Test
✗	✗	✗	66.7	56.9
✓	✗	✓	68.3	56.9
✓	✓	✗	68.7	57.5
✓	✓	✓	**69.2**	**60.9**

Table 2. **Ablation study of the various stages in MoReVQA.** We show the impact of each of the key stages of our proposed design, highlighting the improvements between a system without our proposed stages (top row; defaults to the JCEF baseline) and our multi-stage reasoning setting (bottom row; all 3 stages). We observe stages provide complementary (*e.g.*, NExT-QA) and synergistic (*e.g.*, iVQA) gains (additional ablations in supplement).

long form video understanding. We believe this could potentially be an artifact of its dataset construction process, which leveraged automatic LLM generations to form the question/candidate answer language inputs [33] and may have introduced unintended language bias.

Next, we highlight our simple Socratic baseline JCEF outperforms state-of-the-art single-stage programming methods such as ViperGPT+ (our upgraded implementation) across all datasets, even though both baselines have access to the *same* VLM and LLM modules. This gives us a quantitative assessment of the impact of brittle program generations and tool executions for the single-stage model, as observed in our qualitative analysis (Figure 4, bottom + additional examples in supplement); we also highlight concurrent analysis [27] which found similar failure modes in image-language settings.

Finally, we note that our model MoReVQA outperforms all previous training-free baselines across all datasets, while

Method	Val	FT
MIST-CLIP [17]	57.2	
HiTeA [54]	63.1	✓
SeViLa [57]	**73.8**	
ViperGPT [40]	60.0	
BLIP-2$^{\text{concat}}$ [29]	62.4	
BLIP-2$^{\text{voting}}$ [29]	62.7	✗
SeViLA [57]	63.6	
JCEF	66.7	
MoReVQA	**69.2**	

(a) NExT-QA [47]

Method	Test	FT
VideoCoCa [50]	39.0	
FrozenBiLM [52]	39.7	✓
Text+Text [31]	**40.2**	
FrozenBiLM [52]	27.3	
BLIP-2$_{\text{(FlanT5XXL)}}$ [29]	45.8	
InstructBLIP$_{\text{(FlanT5XL)}}$ [11]	53.1	✗
InstructBLIP$_{\text{(FlanT5XXL)}}$ [11]	53.8	
JCEF	56.9	
MoReVQA	**60.9**	

(b) iVQA [51]

Method	Test	FT
VIOLET [15]	19.9	
SeViLA [57]	22.7	
FrozenBiLM [52]	26.9	
mPLUG-Owl [55]	31.1	
InternVideo [44]	32.1	✗
*ShortViViT [36]	31.0	
*LongViViT [36]	33.3	
JCEF	50.0	
MoReVQA	**51.7**	

(c) EgoSchema [33]

Method	Test	FT
Video-LLaMA [61]	12.4	
VideoChat [30]	26.5	
*LLaMa adapter [62]	34.2	
*Video-ChatGPT [32]	35.2	✗
ViperGPT+	37.1	
JCEF	43.3	
MoReVQA	**45.3**	

(d) ActivityNet-QA [58]

Table 3. **Comparison to SOTA on the standard video question-answering datasets:** (a) NExT-QA, (b) iVQA, (c) EgoSchema, and (d) ActivityNet-QA. Our method MoReVQA outperforms all training-free prior work or exceeds prior state-of-the-art fine-tuned systems (in grey), on the main validation datasets [33, 47, 51, 58]. FT indicates fine-tuned methods. Methods with asterisk * indicate concurrent work.

at the same time providing intermediate interpretable outputs – in Fig. 4, we show a representative example on NExT-QA. We also show limitations in both JCEF and ViperGPT given the same example: while JCEF is a strong baseline, the general-level captions are not always informative enough to answer the question, while the program generated by ViperGPT+ does not focus on the frames in the correct part of the video (specifically, the "if" statement condition erroneously triggers on irrelevant early frames, resulting in misleading "info"). See supplement for more.

We also ablate the stages in MoReVQA (Table 2). Our ablation without any stages (top row) effectively defaults to the JCEF baseline with only frame-level captions and a final LLM prediction stage. The absence of grounding indicates that we simply return a single middle frame for this stage. For all ablations without the final reasoning stage, we retain a final LLM prediction on top of the shared memory state, *i.e.*, the reasoning performs the final VQA only with the input question (without supporting questions). We observe all three stages of our model (event parsing, grounding, and reasoning) provide complementary (*e.g.* on NExT-QA) and synergistic (*e.g.* on iVQA) gains, and meaningfully improve over the JCEF baseline. The added benefit is the interpretability of the intermediate outputs stored in the external memory. Further, we ablate the impact of key components in the memory; when the original question is provided to the reasoning stage, as opposed to a revised version (*e.g.*, `question` in Fig. 4), we note accuracy drops of 1.3% and 3.9% on NExT-QA and iVQA respectively. When grounded frame locations are only given to prediction stage, instead of reasoning stage, we observe drops of 1.2% and 3.9% on the same datasets. Additional examples and analysis are in the supplement, including specific API usage statistics.

4.4.2 Comparison to State-of-the-Art

Finally, we compare our method to the state-of-the-art methods on four datasets – NExT-QA, iVQA, EgoSchema, and ActivityNet-QA (Table 3) – in which the numbers in bold and underline respectively indicate the best and second best. On all datasets we outperform previous zero-shot and few-shot methods by large margins – on NExT-QA we outperform SeViLA by almost 6%, making progress towards closing the gap to fully finetuned performance. On iVQA we outperform the nearest method InstructBLIP [11] by almost 7%, while on EgoSchema the gaps are the largest (approx. 20%). For EgoSchema, we report results using $n = 30$ video frames; we provide results with other values of n in the supplement. We also show strong results on ActivityNet-QA, outperforming concurrent work [32, 62] under consistent evaluation protocols.

Extensions to related tasks. In our supplement, we describe extensions of our MoReVQA system to other tasks. We consider grounded videoQA (localizing the relevant video segment while providing the answer) on the recent NExT-GQA [48] dataset, and highlight our *training-free* MoReVQA achieves strong performance (37.8 mIoP / 39.6 Acc@GQA) vs. prior state-of-the-art SeViLa (29.5 mIoP / 16.6 Acc@GQA; trained with grounding annotations). We also show on ActivityNet-Para [28] strong performance for video paragraph captioning (MoReVQA 28.2 CIDEr vs. finetuned SOTA Vid2Seq [53] with 28.0), even though our method is *training-free*. We observe our system's reasoning enables diverse long captions of human-centric events.

5. Conclusion

In this work, we have presented a baseline (JCEF) to help characterize limitations with single-stage planning models, along with MoReVQA, a new, modular, and decomposed multi-stage pipeline for video question answering. Our framework consists of 3 stages – event parsing, grounding, and reasoning with an external memory. MoReVQA achieves state-of-the-art results on popular VideoQA benchmarks, while producing interpretable intermediate outputs. We discuss limitations and broader impacts in supplement.

Acknowledgements. We sincerely thank ViperGPT [40] authors for sharing additional details helpful for the development of ViperGPT+, and grateful to Chen S., Jasper U., and Lluis C. for discussions. Minsu Cho acknowledges IITP grant (2022-0-00959: "Few-shot learning of causal inference in vision and language") support by Korea (MSIT).

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022. 1, 2

[2] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48, 2016. 2, 3, 4

[3] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. ViViT: A Video Vision Transformer. In *ICCV*, 2021. 2

[4] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. 6

[5] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 3, 6

[6] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022. 1, 2, 6

[7] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, et al. Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*, 2023. 1, 2

[8] Xi Chen, Xiao Wang, Lucas Beyer, Alexander Kolesnikov, Jialin Wu, Paul Voigtlaender, Basil Mustafa, Sebastian Goodman, Ibrahim Alabdulmohsin, Piotr Padlewski, et al. Pali-3 vision language models: Smaller, faster, stronger. *arXiv preprint arXiv:2310.09199*, 2023. 2, 4

[9] Jaemin Cho, Abhay Zala, and Mohit Bansal. Visual programming for text-to-image generation and evaluation. *arXiv preprint arXiv:2305.15328*, 2023. 2

[10] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 1, 2

[11] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *arXiv preprint arXiv:2305.06500*, 2023. 2, 8

[12] Mostafa Dehghani, Alexey Gritsenko, Anurag Arnab, Matthias Minderer, and Yi Tay. Scenic: A jax library for computer vision research and beyond. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21393–21398, 2022. 6

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[14] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. EVA: Exploring the limits of masked visual representation learning at scale. In *CVPR*, 2023. 2

[15] Tsu-Jui Fu, Linjie Li, Zhe Gan, Kevin Lin, William Yang Wang, Lijuan Wang, and Zicheng Liu. Violet : End-to-end video-language transformers with masked visual-token modeling. *arXiv:2111.12681*, 2021. 8

[16] Difei Gao, Ruiping Wang, Ziyi Bai, and Xilin Chen. Env-qa: A video question answering benchmark for comprehensive understanding of dynamic environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1675–1685, 2021. 2

[17] Difei Gao, Luowei Zhou, Lei Ji, Linchao Zhu, Yi Yang, and Mike Zheng Shou. Mist: Multi-modal iterative spatial-temporal transformer for long-form video question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14773–14783, 2023. 2, 8

[18] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. ImageBind: One embedding space to bind them all. In *CVPR*, 2023. 2

[19] Google, Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, et al. Palm 2 technical report, 2023. 2, 3, 4, 6

[20] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, et al. Ego4d: Around the world in 3,000 hours of egocentric video, 2022. 6

[21] Madeleine Grunde-McLaughlin, Ranjay Krishna, and Maneesh Agrawala. Agqa: A benchmark for compositional spatio-temporal reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11287–11297, 2021. 2

[22] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14953–14962, 2023. 1, 2, 3

[23] Ziniu Hu, Ahmet Iscen, Chen Sun, Kai-Wei Chang, Yizhou Sun, David A Ross, Cordelia Schmid, and Alireza Fathi. Avis: Autonomous visual information seeking with large language model agent. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3

[24] De-An Huang*, Shyamal Buch*, Lucio Dery, Animesh Garg, Li Fei-Fei, and Juan Carlos Niebles. Finding "it": Weakly-supervised, reference-aware visual grounding in in-

structional videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4

[25] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021. 2

[26] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE international conference on computer vision*, pages 2989–2998, 2017. 2, 3, 4

[27] Apoorva Khandelwal, Ellie Pavlick, and Chen Sun. Analyzing modular approaches for visual question decomposition. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023. 2, 3, 4, 7

[28] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715, 2017. 2, 8

[29] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023. 2, 3, 8

[30] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding, 2024. 6, 8

[31] Xudong Lin, Simran Tiwari, Shiyuan Huang, Manling Li, Mike Zheng Shou, Heng Ji, and Shih-Fu Chang. Towards fast adaptation of pretrained contrastive models for multi-channel video-language retrieval. *arXiv preprint arXiv:2206.02082*, 2023. 8

[32] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models, 2023. 6, 8

[33] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. *arXiv preprint arXiv:2308.09126*, 2023. 2, 3, 6, 7, 8

[34] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In *ICCV*, 2019. 6

[35] M Minderer, A Gritsenko, A Stone, M Neumann, D Weissenborn, A Dosovitskiy, A Mahendran, A Arnab, M Dehghani, Z Shen, et al. Simple open-vocabulary object detection with vision transformers. *European Conference on Computer Vision (ECCV)*, 2022. 3, 4, 6

[36] Pinelopi Papalampidi, Skanda Koppula, Shreya Pathak, Justin Chiu, Joe Heyward, Viorica Patraucean, Jiajun Shen, Antoine Miech, Andrew Zisserman, and Aida Nematzdeh. A simple recipe for contrastively pre-training video-first encoders beyond 16 frames, 2023. 8

[37] 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. In *icml*, 2021. 2, 3, 4, 6

[38] Paul Hongsuck Seo, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. End-to-end generative pretraining for multimodal video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17959–17968, 2022. 1, 2

[39] Sanjay Subramanian, Medhini Narasimhan, Kushal Khangaonkar, Kevin Yang, Arsha Nagrani, Cordelia Schmid, Andy Zeng, Trevor Darrell, and Dan Klein. Modular visual question answering via code generation. *arXiv preprint arXiv:2306.05392*, 2023. 1, 2, 3

[40] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. *arXiv preprint arXiv:2303.08128*, 2023. 1, 2, 3, 5, 6, 8

[41] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1, 2

[42] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*, 2022. 1, 2

[43] Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. End-to-end dense video captioning with parallel decoding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6847–6857, 2021. 1, 2

[44] Yi Wang, Kunchang Li, Yizhuo Li, Yinan He, Bingkun Huang, Zhiyu Zhao, Hongjie Zhang, Jilan Xu, Yi Liu, Zun Wang, et al. Internvideo: General video foundation models via generative and discriminative learning. *arXiv preprint arXiv:2212.03191*, 2022. 8

[45] Zhenhailong Wang, Manling Li, Ruochen Xu, Luowei Zhou, Jie Lei, Xudong Lin, Shuohang Wang, Ziyi Yang, Chenguang Zhu, Derek Hoiem, et al. Language models with image descriptors are strong few-shot video-language learners. *Advances in Neural Information Processing Systems*, 35: 8483–8497, 2022. 3

[46] Bo Wu, Shoubin Yu, Zhenfang Chen, Joshua B Tenenbaum, and Chuang Gan. Star: A benchmark for situated reasoning in real-world videos. In *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS)*, 2021. 2

[47] 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*, pages 9777–9786, 2021. 2, 3, 5, 8

[48] Junbin Xiao, Angela Yao, Yicong Li, and Tat Seng Chua. Can i trust your answer? visually grounded video question answering. *arXiv preprint arXiv:2309.01327*, 2023. 2, 4, 8

[49] Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. Video question answering via gradually refined attention over appearance and motion. In *ACM Multimedia*, 2017. 2

[50] Shen Yan, Tao Zhu, Zirui Wang, Yuan Cao, Mi Zhang, Soham Ghosh, Yonghui Wu, and Jiahui Yu. Videococa: Video-text modeling with zero-shot transfer from contrastive captioners. *arXiv preprint arXiv:2212.04979*, 2022. 8

[51] Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Just ask: Learning to answer questions from millions of narrated videos. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1686–1697, 2021. 2, 3, 6, 8

[52] Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Zero-shot video question answering via frozen bidirectional language models. *Advances in Neural Information Processing Systems*, 35:124–141, 2022. 8

[53] Antoine Yang, Arsha Nagrani, Paul Hongsuck Seo, Antoine Miech, Jordi Pont-Tuset, Ivan Laptev, Josef Sivic, and Cordelia Schmid. Vid2seq: Large-scale pretraining of a visual language model for dense video captioning. In *CVPR*, 2023. 2, 8

[54] Qinghao Ye, Guohai Xu, Ming Yan, Haiyang Xu, Qi Qian, Ji Zhang, and Fei Huang. Hitea: Hierarchical temporal-aware video-language pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15405–15416, 2023. 8

[55] Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*, 2023. 8

[56] Keunwoo Peter Yu. VideoBLIP, 2023. 2

[57] Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. Self-chained image-language model for video localization and question answering. In *NeurIPS*, 2023. 2, 8

[58] Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9127–9134, 2019. 2, 3, 6, 8

[59] Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, et al. Socratic models: Composing zero-shot multimodal reasoning with language. *ICLR*, 2023. 2, 3

[60] Yan Zeng, Xinsong Zhang, , and Hang Li. Aligning texts with visual concepts. *arXiv preprint arXiv:2111.08276*, 2021. 3

[61] Hang Zhang, Xin Li, and Lidong Bing. Video-LLaMA: An instruction-tuned audio-visual language model for video understanding. In *EMNLP 2023 Demo*, 2023. 2, 8

[62] Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *ICLR*, 2024. 6, 8

[63] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022. 1, 2

[64] Yaoyao Zhong, Junbin Xiao, Wei Ji, Yicong Li, Weihong Deng, and Tat-Seng Chua. Video question answering: Datasets, algorithms and challenges. *arXiv preprint arXiv:2203.01225*, 2022. 2

[65] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *CVPR*, 2018. 2

[66] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023. 2