Grounded Question-Answering in Long Egocentric Videos

Supplementary Material

A. Llama2 vs. ChatGPT on Data Generation

In the main paper, we default to using `Llama2-13B-chat` for generating QA data. Here, we experiment with `ChatGPT-3.5-turbo`[1]. To expedite the data generation and model training process, we reduce the amount of data relative to EGOTIMEQA. Specifically, we use both LLMs to generate QA data from the NLQ_{v2} training set, which includes 1.3K video clips and 221K narration sentences. Full prompts are detailed in Sec. D, where minor differences exist between ChatGPT's and Llams2's prompts. Consequently, Llama2 produces 92K and ChatGPT 97K data pairs.[2] Compared to QAEGO4D, the video clips are almost identical, but the QA pairs are denser in time. We then train $\text{GroundVQA}_{\text{S}}$ on each dataset to assess generation quality, referencing OpenQA and VLG performance. Note that for CloseQA, both training and testing data are generated by the LLMs. Thus, evaluating CloseQA on a test set produced by one LLM, such as Llama-2, would be unfair for ChatGPT because of the bias in the generation process. Therefore, we exclude CloseQA evaluation from this experiment.

As Tab. 1 (B-C) indicates, the model trained on data generated by `Llama2-13B-chat` slightly outperforms the one trained on `ChatGPT-3.5-turbo` data. That is to say, Llama2's capacity to generate QA pairs from narrations is comparable to, if not better than, ChatGPT. Additionally, from a cost perspective, Llama2 is more accessible for academic research labs or companies with certain computing resources compared to ChatGPT. In terms of data scaling, the data produced by both LLMs improves the model's performance in OpenQA and VLG (from A to B and A to C). Adding more data continues to enhance performance (from C to D).

	Training Data				OpenQA			VLG	
	Source	# Clip	# Sample	Cost	**Sim.**	ROUGE	METEOR	**Mean R@1**	Mean R@5
(A)	QAEGO4D	1.0K	11K	–	55.6	29.0	19.8	8.8	20.0
(B)	+ ChatGPT QA (NLQ_{v2})	1.3K	107K	$50	56.9	29.2	19.8	15.7	33.7
(C)	+ Llama2 QA (NLQ_{v2})	1.3K	103K	5 Gh	57.1	29.4	20.3	16.3	34.3
(D)	+ Llama2 QA (EM_{v2})	5.5K	314K	16 Gh	57.7	30.2	21.2	18.4	37.2

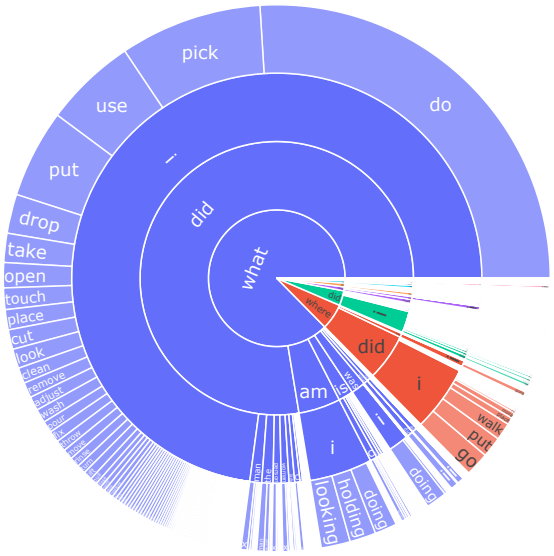
Table 1. **Effect of data scaling and using different LLMs for data generation.** We train $\text{GroundVQA}_{\text{S}}$ with different training data and report their OpenQA and VLG performance. "ChatGPT" denotes the `ChatGPT-3.5-turbo` model in OpenAI's API. "Llama2" is short for `Llama2-13B-chat`. Row D is our EGOTIMEQA. In the "Cost" column, we estimate the money or time spent on generating the corresponding data, where "Gh" stands for GPU hours tested on NVIDIA A100 (80GB) GPUs.

[1] Utilizing the OpenAI API: https://platform.openai.com

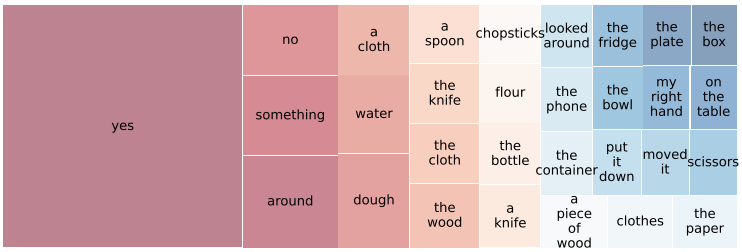
[2] The variation in numbers stems from differing rates of generation errors, *i.e.*, the generated string cannot be converted into a dictionary containing "Q" and "A" as the in-context examples.

B. Additional EGOTIMEQA Statistics

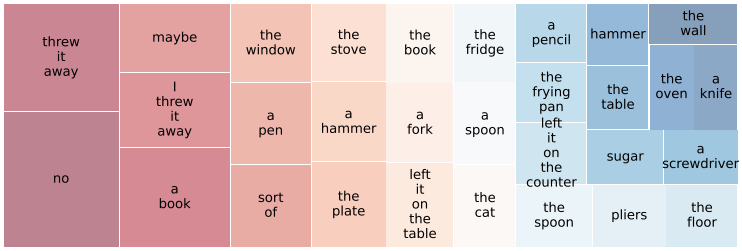
We offer additional statistical details about our EGOTIMEQA. In Fig. 1a, we present the question distribution based on their first four words. Most questions begin with "what", inquiring about objects or actions. Others start with "where", "did", "how", *etc.*, showcasing their diversity. In Fig. 1b and 1c, we present the distribution of the top 30 correct and incorrect answers, respectively. Fig. 1d presents a histogram of the duration of temporal windows in EGOTIMEQA, with the majority falling within the 0-3 second range. Fig. 1e, 1f, and 1g shows the word count distributions for questions, correct answers, and incorrect answers, respectively. Correct and incorrect answers have similar distributions, averaging 3.2 and 2.8 words respectively, while questions, averaging 6.6 words, indicate their greater complexity.



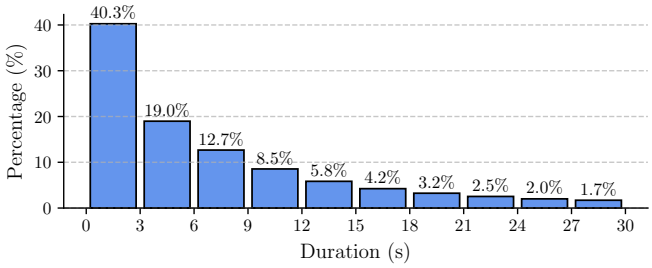
(a) **Question sunburst chart.** The word order starts at the center and extends outward. A larger area indicates a higher frequency of occurrence.



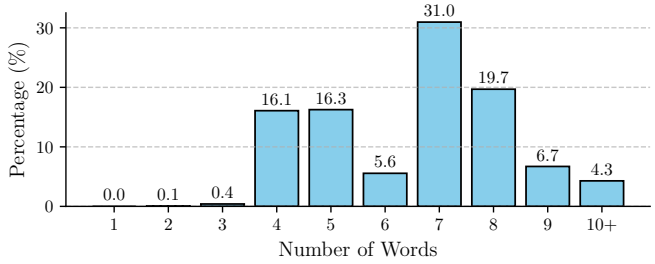
(b) **Correct answer treemap.** A larger area indicates a higher frequency of occurrence. Likewise for the figure below.



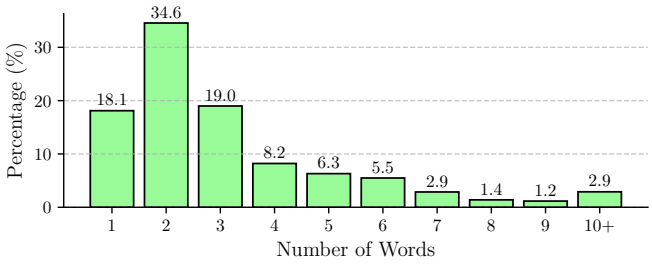
(c) **Wrong answer treemap.**



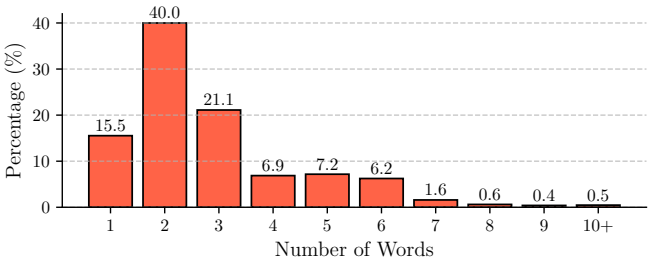
(d) **Histogram of temporal windows for VLG.**



(e) **Histogram of question word count.**



(f) **Histogram of correct answer word count.**



(g) **Histogram of wrong answer word count.**

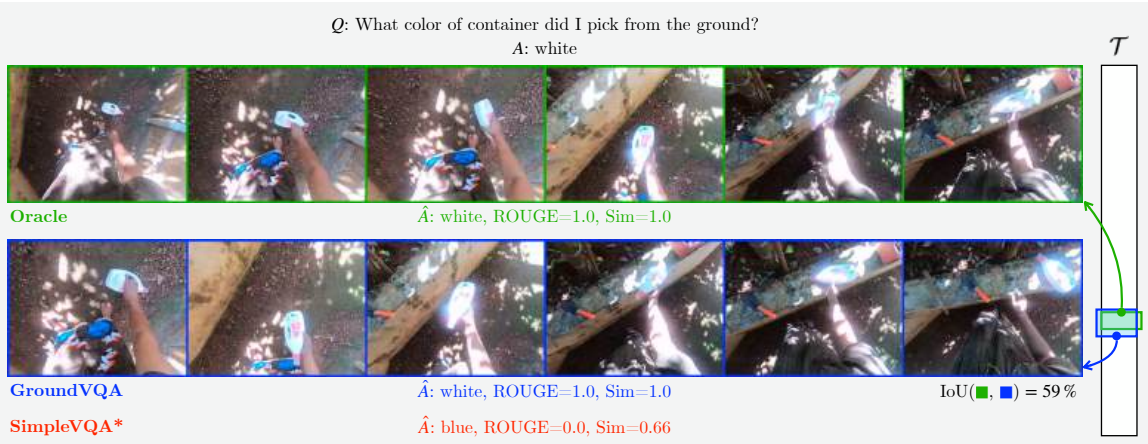
Figure 1. **Combined figure of various EgoTimeQA visualizations.**

C. Additional Qualitative Analysis

We conduct qualitative analysis on VLG, OpenQA, and CloseQA tasks, showcasing both success and failure scenarios. In all examples, we present results from our proposed GroundVQA, the Oracle baseline, and SimpleVQA. Each model is built upon the `Flan-T5-Base` language model. Specifically, GroundVQA is trained concurrently on VLG, OpenQA, and CloseQA tasks with QAEGO4D and EGOTIMEQA data. Oracle, a variant of GroundVQA, takes only the question-related video clips as input, eliminating the need for temporal grounding. SimpleVQA* is our reproduced SimpleVQA [1] model, trained on OpenQA and CloseQA tasks with QAEGO4D data. For a detailed examination, please zoom in on the figures.

C.1. Results on OpenQA

VLG & OpenQA succeed. In Fig. 2a, GroundVQA accurately localizes the video segment relevant to the query and correctly identifies *container*'s color. Conversely, SimpleVQA* predicts a wrong color, and its lack of temporal grounding hinders error analysis. Overall, integrating the VLG task not only boosts QA performance but also enhances the interpretability of our model by clarifying the sources of errors. In Fig. 2b, GroundVQA closely predicts the temporal window and provides an answer (*in the fridge*) that, while slightly different, conveys the same meaning as the ground truth (*inside the refrigerator*). This case illustrates the limitation of the ROUGE metric in distinguishing between correct and incorrect paraphrased answers. Therefore, we introduce the sentence similarity metric and an additional CloseQA task to address this evaluation challenge.



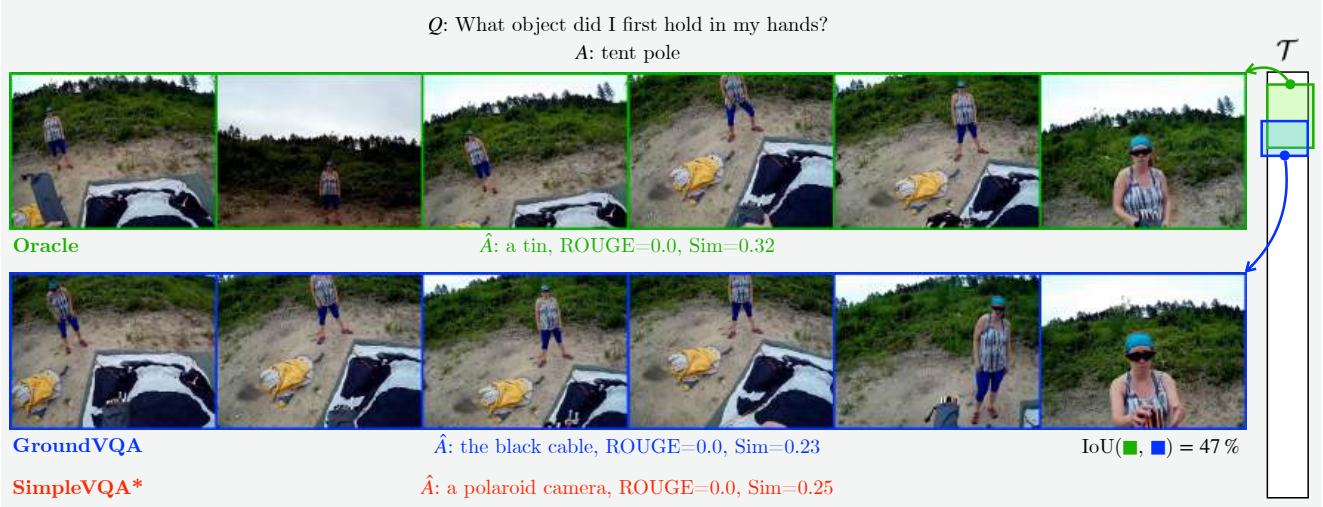
(a) GroundVQA has correct VLG and OpenQA predictions.

(b) GroundVQA has correct VLG and valid OpenQA predictions.

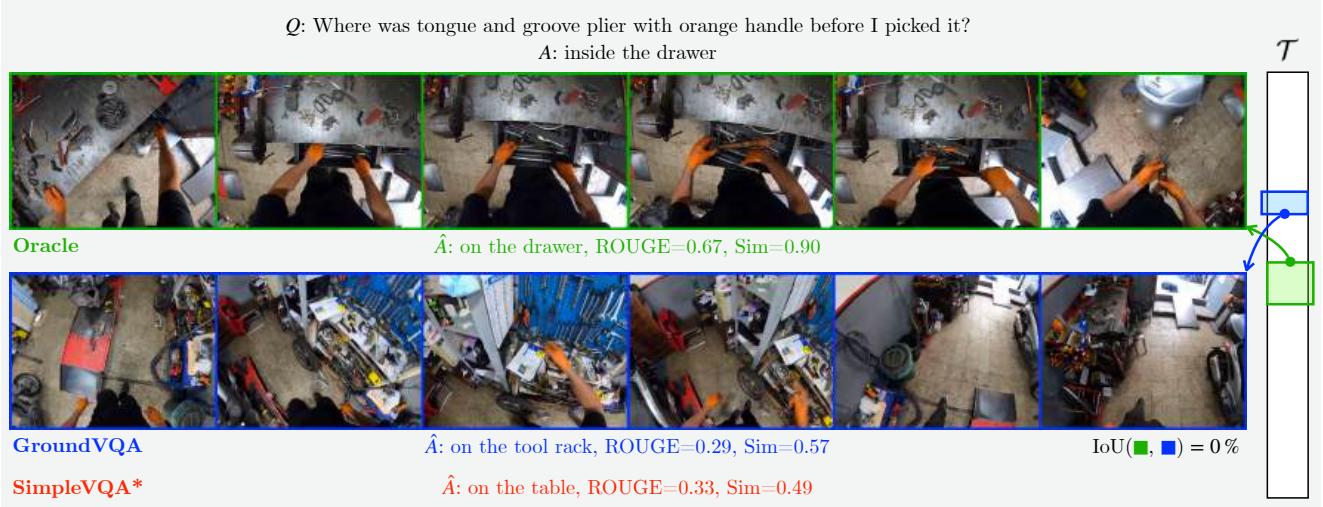
Figure 2. **OpenQA success cases.** we compare three models: Oracle, our GroundVQA, and SimpleVQA*. From top to bottom are the query $\mathcal{Q}$, answer $\mathcal{A}$, six frames uniformly sampled from the grounded video segment, and the predicted answer $\hat{\mathcal{A}}$ with metrics. Additionally, the right side illustrates the video's time span and the predicted temporal window $\mathcal{T}$, with Oracle's temporal window serving as the ground truth. Note that SimpleVQA* is incapable of temporal grounding.

VLG succeeds, OpenQA fails. In Fig. 3a, although GroundVQA successfully identifies the relevant video segment, it incorrectly answers the query. The *tent pole* visible in the sampled frames occupies a minor portion of the frame and is easily mistaken for similar objects, such as a *tin* or *black cable*, leading to errors in both Oracle and GroundVQA. In contrast, SimpleVQA*'s response (*a polaroid camera*) is entirely off-topic, indicating a misdirected focus.

VLG & OpenQA fail. In Fig. 3b, GroundVQA fails to correctly ground the query and answer the question. SimpleVQA* also errs in its response. However, the Oracle model, with access to the ground truth video segment, provides a valid answer. A closer look at the frames grounded by GroundVQA reveals its attention to the *tongue and groove plier with an orange handle* on the tool rack, but it overlooks the action of *picking* mentioned in the query. This indicates that GroundVQA still has limitations in comprehending questions and reasoning about video content.



(a) GroundVQA has correct VLG but wrong OpenQA predictions.



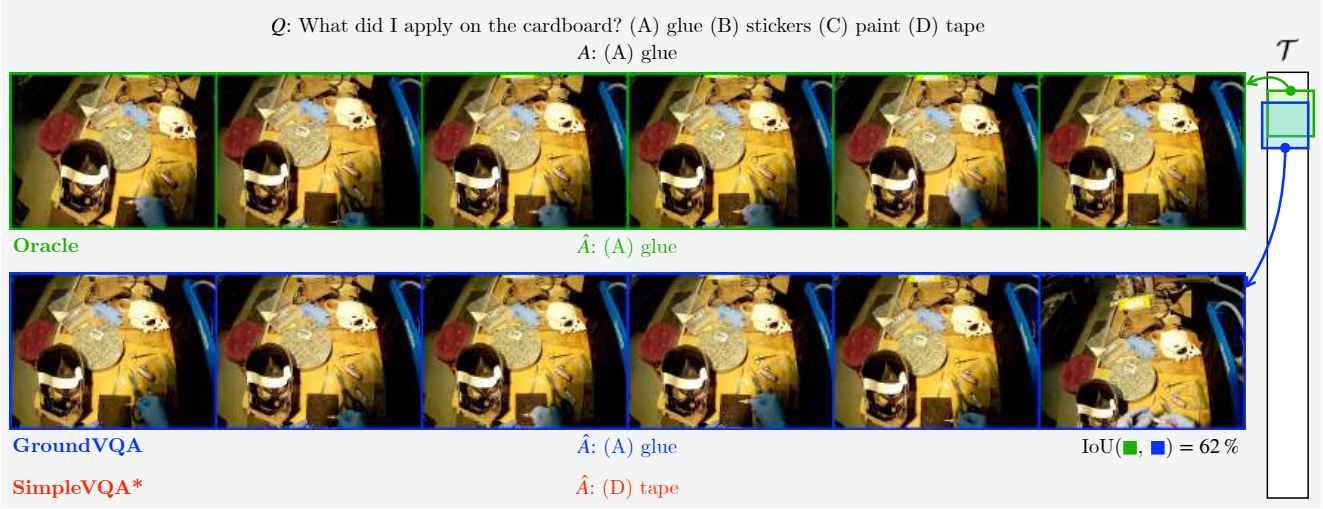
(b) GroundVQA has wrong VLG and OpenQA predictions.

Figure 3. **OpenQA failure cases.** Refer to Fig. 2 for the descriptions of the figures.

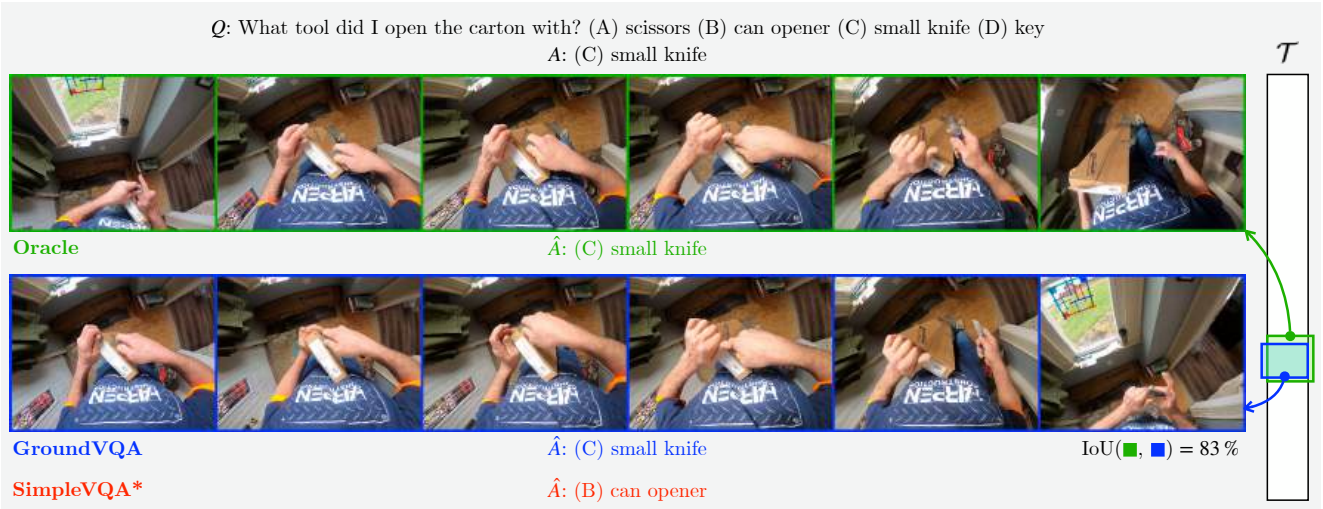
C.2. Results on CloseQA

VLG & CloseQA succeed. In Fig. 4a, GroundVQA successfully localizes the video clip corresponding to the question and selects the correct answer. Notice how tiny the *glue* is in the grounded video frames, which demonstrates our method's object recognition capability. On the contrary, SimpleVQA* chooses an incorrect option, and the reason for this is unclear. This example also highlights the advantage of the CloseQA task, which eliminates ambiguities and paraphrasing dilemmas in evaluating open-ended answers.

In Fig. 4b, GroundVQA excels in both temporal grounding and question-answering, whereas SimpleVQA* fails. This example underscores the importance and challenge of temporal grounding, as the model needs to identify the *carton* target and recognize the *open* action. Once the precise temporal window is grounded, identifying the *small knife* used to *open the carton* becomes significantly easier.



(a) GroundVQA has correct VLG and CloseQA predictions for a video with complex environments.

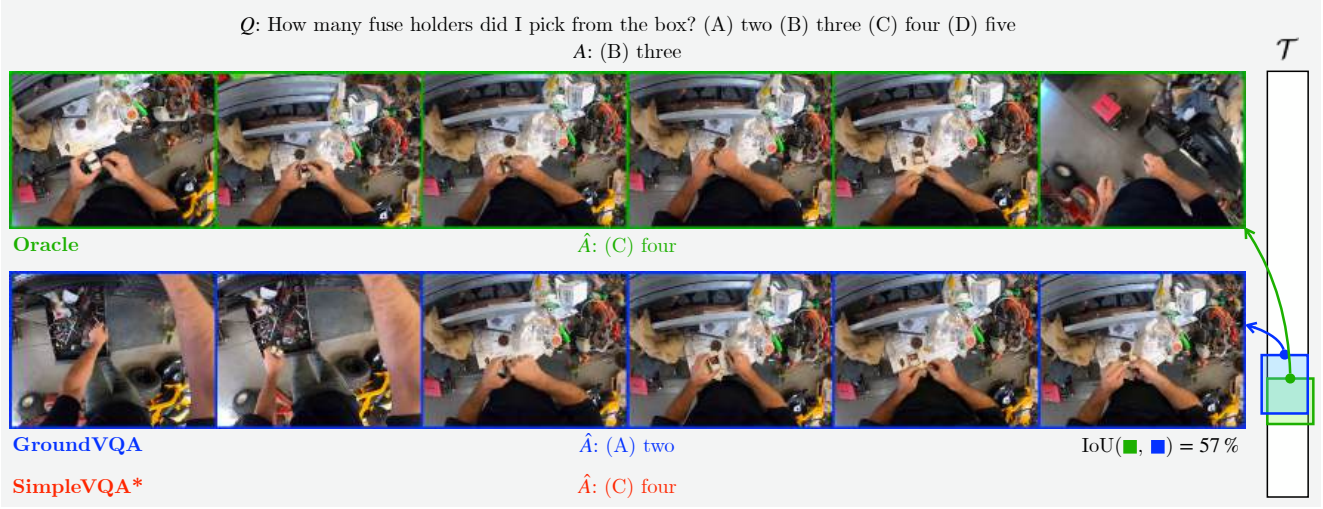


(b) GroundVQA has correct VLG and CloseQA predictions.

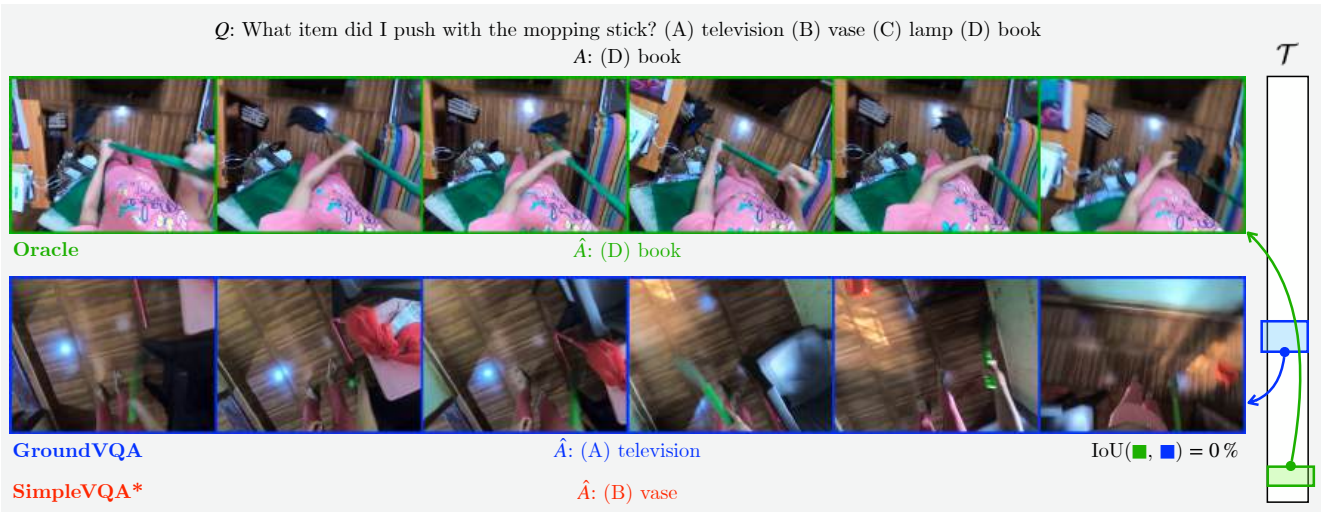
Figure 4. **CloseQA success cases.** Refer to Fig. 2 for the descriptions of the figures.

VLG succeeds, CloseQA fails. In Fig. 5a, GroundVQA succeeds in temporal grounding. However, all three models choose the wrong answer. This example, which involves a counting task, highlights the models' limitations in counting objects across sequential frames. Future improvements could include training with more data, incorporating object-centric representations [2, 4, 6], or adopting object detection/tracking techniques [3, 5].

VLG & CloseQA fail. In Fig. 5b, GroundVQA fails in both query grounding and answering. SimpleVQA* also fails. By contrast, the Oracle model, with access to relevant video content, chooses the correct answer. The frames grounded by GroundVQA contains the *mopping stick* but miss the *pushing* action and the *television* it erroneously selects.



(a) GroundVQA has correct VLG but wrong CloseQA predictions.



(b) GroundVQA has wrong VLG and CloseQA predictions.

Figure 5. **CloseQA failure cases.** Refer to Fig. 2 for the descriptions of the figures.

D. Full Prompts for Data Generation

Here, we list the full prompts utilized for generating OpenQA and CloseQA data. Text in red indicates variable inputs.

D.1. Generate OpenQA Data Using Llama-2

```
<s>[INST] <<SYS>>
You are an AI Assistant and always write the output of your response in JSON. I will provide
    you with a series of narrations that depict my behavior. You should generate one QA
    pair based on the narrations in the format of {"Q": <question>, "A": <answer>}. In the
    narrations, "C" represents me, and "O" represents someone else. Use as much information
    as possible from narrations to generate the question, and the question you generate
    should be able to be answered using the information provided in the narrations. The
    question should be in the past tense. The question should be within 10 words, and the
    answer should be within 5 words. <</SYS>>

C pours hot water from the frying pan in his left hand into the bowl in his right hand. [/
    INST] {"Q": "What did I pour in the bowl?", "A": "boiling water"} </s>
<s>[INST] C searches through the cabinet. C closes the cabinet. C picks the tin from the
    cabinet. C places the tin on the counter. [/INST] {"Q": "Where was the tin before I took
    it?", "A": "at the cabinet"} </s>
<s>[INST] C turns on sink knob. C washes the cucumber on the sink. C turns off sink knob. [/
    INST] {"Q": "Did I wash the cucumber?", "A": "yes"} </s>
<s>[INST] <narrations> [/INST]
```

D.2. Generate OpenQA Data Using ChatGPT

```
You're an AI Assistant, outputting responses in JSON. I'll give behavior narrations, in
    which "C" is me, "O" is someone else. Generate a QA pair like {"Q": <question>, "A": <
    answer>} based on them. The question should use the narration info, be in the past tense
    , <= 10 words, and the answer <= 5 words.

User: C pours hot water from the frying pan in his left hand into the bowl in his right hand
Assistant: {"Q": "What did I pour in the bowl?", "A": "boiling water"}

User: C searches through the cabinet. C closes the cabinet. C picks the tin from the cabinet
    . C places the tin on the counter.
Assistant: {"Q": "Where was the tin before I took it?", "A": "at the cabinet"}

User: C turns on sink knob. C washes the cucumber on the sink. C turns off sink knob.
Assistant: {"Q": "Did I wash the cucumber?", "A": "yes"}

User: <narrations>
```

D.3. Generate CloseQA Data Using Llama-2

```
<s>[INST] <<SYS>>
I'll provide a question and its correct answer. Generate three plausible, but incorrect,
    answers that closely resemble the correct one Make it challenging to identify the right
    answer. No preamble, get right to the three wrong answers and present them in a list
    format. <</SYS>>

Question: How many frying pans can i see on the shelf? Correct Answer: two pieces. Wrong
    Answers: [/INST] ["one piece", "three piece", "five pieces"] </s>
<s>[INST] Question: What colour bowl did i carry from the plate stand? Correct Answer: green
    . Wrong Answers: [/INST] ["blue", "black", "white"] </s>
<s>[INST] Question: What did i pour in the bowl? Correct Answer: boiling water. Wrong
    Answers: [/INST] ["hot oil", "steamed milk", "warm broth"] </s>
<s>[INST] Question: <question> Correct Answer: <answer>. Wrong Answers: [/INST]
```

E. Limitations and Future Work

From the experiments and analysis presented in the main paper and this supplementary material, we can draw the following observations. *First*, the performance of our method is closely tied to the quality of video features and training data. Enhancements in video features, particularly through improved visual-language alignment, and the inclusion of more training data, could lead to future improvements. *Second*, despite our efforts in designing appropriate prompts and rigorously filtering data, biases and inaccuracies still exist in the LLM-generated data. *Third*, our method faces challenges in fine-grained perception tasks, for example, object recognition and counting, particularly in complex environments. The adoption of object-centric features, for example, those for tracking and counting techniques could enhance performance in these areas. *Fourth*, processing long egocentric videos demands significant computational resources. Future research should explore the use of memory networks for compressing video features and developing more efficient models that maintain accuracy. *Finally*, a query may relate to multiple video segments, but Ego4D-NLQ and QA-Ego4D assume only one is relevant per question. We advocate for loosening this assumption, as it is typical for multiple personal experiences to be triggered by a single query. Considering the difficulty of labeling multiple temporal windows for one query, a practical solution is to employ a well-trained Vision-Language Model (VLM) to identify potential candidates for further confirmation via human review. Subsequently, our method can be applied to generate QA pairs for each confirmed candidate.

References

[1] Leonard Bärmann and Alex Waibel. Where did i leave my keys? - episodic-memory-based question answering on egocentric videos. In *CVPR Workshop*, 2022. 3

[2] Gamaleldin Elsayed, Aravindh Mahendran, Sjoerd van Steenkiste, Klaus Greff, Michael C Mozer, and Thomas Kipf. Savi++: Towards end-to-end object-centric learning from real-world videos. In *NeurIPS*, 2022. 6

[3] Yongdong Li, Liang Qu, Guiyan Cai, Guoan Cheng, Long Qian, Yuling Dou, Fengqin Yao, and Shengke Wang. Video object counting with scene-aware multi-object tracking. *Journal of Database Management*, 2023. 6

[4] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In *NeurIPS*, 2020. 6

[5] Pavel Tokmakov, Jie Li, Wolfram Burgard, and Adrien Gaidon. Learning to track with object permanence. In *ICCV*, 2021. 6

[6] Chuhan Zhang, Ankush Gupta, and Andrew Zisserman. Helping hands: An object-aware ego-centric video recognition model. In *ICCV*, 2023. 6