AssistQ: Affordance-centric Question-driven Task Completion for Egocentric Assistant

Benita Wong[⋆], Joya Chen[*], You Wu[*], Stan Weixian Lei,
Dongxing Mao, Difei Gao, and Mike Zheng Shou[†]

Show Lab, National University of Singapore
benitawong@u.nus.edu {joyachen97,mike.zheng.shou}@gmail.com

Abstract. A long-standing goal of intelligent assistants such as AR glasses/robots has been to assist users in affordance-centric real-world scenarios, such as *"how can I run the microwave for 1 minute?"*. However, there is still no clear task definition and suitable benchmarks. In this paper, we define a new task called Affordance-centric Question-driven Task Completion, where the AI assistant should learn from instructional videos to provide step-by-step help in the user's view. To support the task, we constructed AssistQ, a new dataset comprising 531 question-answer samples from 100 newly filmed instructional videos. We also developed a novel Question-to-Actions (Q2A) model to address the AQTC task and validate it on the AssistQ dataset. The results show that our model significantly outperforms several VQA-related baselines while still having large room for improvement. We expect our task and dataset to advance Egocentric AI Assistant's development. Our project page is available at: https://showlab.github.io/assistq/.

Keywords: affordance-centric, egocentric AI, question-answering

1 Introduction

People often require assistance when dealing with new events. Consider the example in Figure 1: the user comes across an unfamiliar washing machine and wants to start a cotton wash, but he does not know how to operate it. He may search for the device's instructional video, and experiment with buttons on the machine. These actions are time-consuming and may not address the user's question effectively. This example highlights the need for an intelligent assistant to help us with affordance-centric queries. The intelligent assistant should: (1) understand the user's query and view, (2) learn from instructional video/manual, (3) guide the user to achieve his goal.

However, few works support the development of such an assistant. First, AI assistants should deal with affordance-centric queries, while current visual question answering (VQA) benchmarks [3,17,19,23,25,33] focus on fact-based questions and answers. Second, most ground-truth answers in VQA benchmarks

[⋆] Equal Contribution. [†] Corresponding Author.

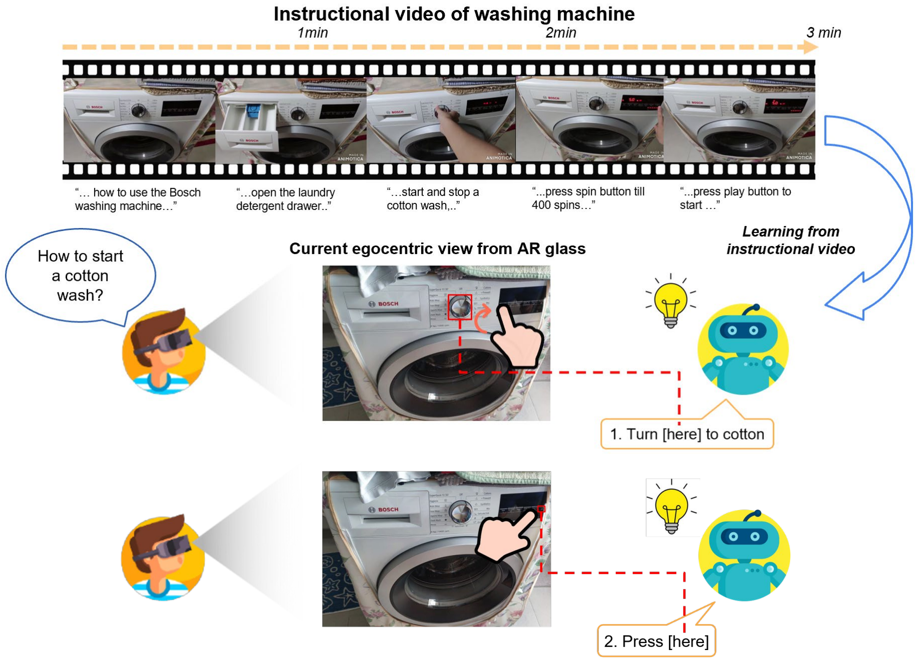

Instructional video of washing machine
1min
2min
3 min
"… how to use the Bosch washing machine…"
"…open the laundry detergent drawer.."
"…start and stop a cotton wash,.."
"...press spin button till 400 spins…"
"...press play button to start …"
Learning from instructional video
How to start a cotton wash?
Current egocentric view from AR glass
1. Turn [here] to cotton
2. Press [here]

Fig. 1. An illustration of an AI assistant for affordance-centric questions. The AI assistant on AR glass can guide the user to complete the intended task.

are single-step and textual, which fails to express detailed instructions to the user's complex question. Even though visual dialog [1,12] (VDialog) introduces sequence question answering, the answer of each dialog round is still single-step and textual. Finally, AI assistants should solve questions in users' egocentric perspective [20,28,31,37,40], which is more natural and meaningful for understanding users' situations. However, existing VQA and VDialog studies mainly comprise third-person web videos. Although embodied QA [9,41] (EQA) attempts to develop egocentric agents, they are mostly focused on virtual environments, which may be difficult to generalize to real-world scenarios, and they are also incapable of dealing with complex, affordance-centric user queries.

Hence, we propose a novel **Affordance-centric Question-driven Task Completion (AQTC)** task. It presents a unique set-up: the user asks a question in his/her view, and then the AI assistant answers in a sequence of actions. Fiugre 2 shows the differences between our task and VQA, VDialog, and EQA. Unlike VQA and VDialog, where the question is fact-based and the answer is a short text, our question is task-oriented and our answer is multi-modal and multi-step, which is more challenging and closer to the intelligent assistant setting. Furthermore, our task makes use of new content – real-world, egocentric videos – for question answering, which is more suitable for real-world applications compared to virtual environments in the EQA.

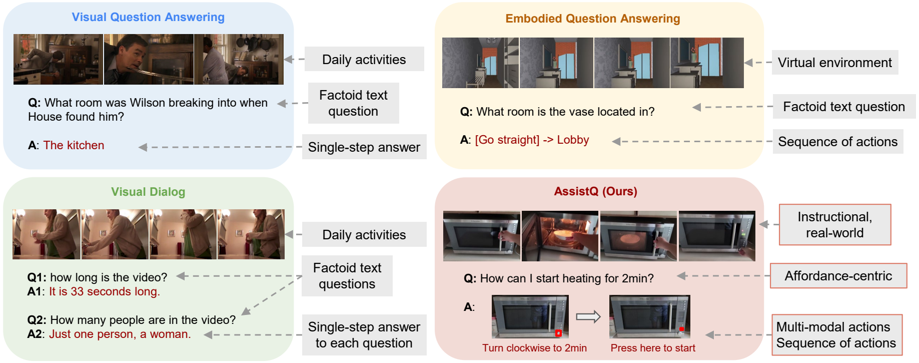

Visual Question Answering
Q: What room was Wilson breaking into when House found him?
A: The kitchen
Daily activities
Factoid text question
Single-step answer
Embodied Question Answering
Q: What room is the vase located in?
A: [Go straight] -> Lobby
Virtual environment
Factoid text question
Sequence of actions
Visual Dialog
Q1: how long is the video?
A1: It is 33 seconds long.
Q2: How many people are in the video?
A2: Just one person, a woman.
Daily activities
Factoid text questions
Single-step answer to each question
AssistQ (Ours)
Q: How can I start heating for 2min?
A:
Turn clockwise to 2min
Press here to start
Instructional, real-world
Affordance-centric
Multi-modal actions Sequence of actions

Fig. 2. The differences between VQA, VDialog, EQA, and our AQTC task.

To support the AQTC task, we collected the AssistQ dataset containing 531 multiple-choice QA samples on 100 newly filmed, with an average duration of 115s egocentric videos. Participants recorded instructional videos by themselves (*e.g.,* operating microwaves, washing machine), and provided scripts of their narration. These instructional videos were used to create affordance-centric, multi-step QA samples. We also developed a Question-to-Actions (Q2A) model to address the AQTC task. It uses a context grounding module to infer from multi-modal cues (video, script, user view), and introduces a steps network to generate step-by-step answers. Experimental results show that our Q2A model is especially suitable for the AQTC task. In summary, our contributions are:

• We proposed a new task, namely Affordance-centric Question-driven Task Completion (AQTC), whereby AI assistants should learn from instructional videos to guide users step-by-step in the users' view.

• We constructed a pilot dataset for the task, comprising 531 question-answer samples derived from 100 newly filmed instructional videos.

• We developed a Question-to-Actions (Q2A) model and performed extensive experiments to show its superiority in the AQTC task.

2 Related Work

Visual Question Answering. The goal of VQA is to answer questions based on visual cues. VQA v1 and v2 [3,19] are standard benchmarks for image QA. TGIF-QA [23] is one of the earlier large-scale video QA datasets and was built from short animated videos (GIFs) taken from Tumblr. In this task, the main goal is to recognize actions and their repetitions. Similar video QA datasets [17,25,26,33] mainly illustrate daily life or story. In contrast, our task focuses on instructional videos more suitable for affordance-centric tasks.

Visual Dialog. VDialog [12] is a task to generate natural responses based on the given image/video and the dialog context. An example is Audio Visual Scene-Aware Dialog (AVSD) [1], where the agent is shown a video of indoor activity and answers multiple questions about actions and events in the video. Compared to models developed for single-step VQA tasks, VDialog models should utilize the dialog history. Some models [1,12,27,32] simply splice historical dialogs or model using RNN, while others [13,24] introduce reinforcement learning methods. Like VDialog, historical actions in our AQTC task may affect the current answer step. But their differences are apparent: Visual Dialog is more like a single-step QA problem in each round, whereas AQTC requires a sequence of actions to answer. We believe the latter is more appropriate for affordance-centric questions.

Embodied Question Answering. If we consider the instructional video in our AQTC task as the environment, then our model should explore the environment to answer the question. This is similar to EQA [9], where the agent needs to explore the scene and answer the question. Recent works try to solve or extend EQA by improving the logical reasoning [11,41], applying multi-agent cooperation [10,22,41], or creating more challenging environments [18]. However, the current setting of EQA focuses on virtual environments, while our proposed AQTC task revolves around real-world images and videos. This allows AQTC to be more readily adopted for practical applications, such as intelligent agents built in portable devices (*e.g.,* mobile phones, AR glasses).

3 Affordance-centric Question-driven Task Completion

In this section, we propose Affordance-centric Question-driven Task Completion (AQTC), a novel task aimed at driving the evolution of egocentric AI assistants. We begin by outlining the abilities required of an egocentric, user-helping AI assistant. The AQTC work is then formalized in accordance with these abilities. Finally, we talk about how to generalize and apply the task.

3.1 Required Abilities of Egocentric AI Assistants

Handling multi-modal inputs. In affordance-centric settings, the user often asks questions in both visual (*e.g.,* his/her current view) and linguistic (*e.g., "how to start the microwave?"*). As a result, the egocentric AI assistant is expected to handle multimodal inputs.

Learning from instructional sources. It is difficult for an AI assistant to answer questions without learning anything. An acceptable way is for the AI assistant to read the instructional manual first and then teach users. We notice that there are various instructional videos available on the internet, and it would be preferable if AI assistants could utilize these data, *i.e.,* learning from instructional videos and then guide users in their situation.

Grounding between modalities. The different input modalities are often not grounded with respect to each other. For example, a video guide might point to a button, whereas the corresponding textural guide may say “Press this”. The egocentric AI assistant should know how to link visual and language concepts.

Multi-step interaction with users. Interaction with the user is necessary for an egocentric AI assistant. For example, if a user inquires about *"how to start*

Tasks	Property	Model Input	Model Output
Video QA	Factoid	Reference Video	Single textual answer
Video Dialog	Factoid	Reference Video	Multi textual answers
Embodied QA	Factoid	User Situation	Multi actions, Single textual answer
AQTC	Affordance	Reference Video, User Situation	Multi <action, textual answer> pairs

Table 1. Comparisons with VQA-related tasks.

the microwave," the assistant should tell him/her the location of the power switch. When a user's inquiry is complex, the interaction between the user and the assistant may be held for several steps.

3.2 Task Definition of AQTC

Now we design a task for the abilities discussed in Section 3.1. To begin, the task should include multi-modal inputs, such as the user's egocentric view and textural question. Then, there should be a reference instructional video from which AI assistants can learn to address problems. Furthermore, rather than using different modalities independently, the grounding between them should be necessary to handle the task. Finally, the task should reflect multi-step interactions between users and assistants, such as a sequence of actions taught to users by the assistant.

With these considerations, we propose affordance-centric question-driven task completion (AQTC) task. Given an instructional video (V), the corresponding video's textural script (S_V), the user's view (U), and the question (Q_U) asked from the view, the model should answer the question by a sequence of answers ($A^1, A^2, ..., A^I$) in I ($I >= 1$) steps. Note that the question Q_U is proposed under the user view U rather than the instructional video V, so the model should ground between U and V (with S_V) to answer Q_U. Currently, we set candidate answers (A^i_j) to lower the difficulties of answering, where A^i_j denotes the j-th potential answer in the i-th step.

Comparing with existing VQA-related tasks (*e.g.,* VQA [3,25,33], VDialog [1,12], EQA [9]), our proposed AQTC task is more suitable in supporting the affordance-centric requirements of AI assistants. We list the reasons in Table 1:

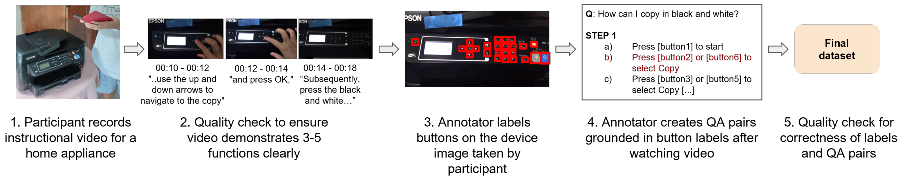
(1) Property: existing VQA-related datasets are mostly factoid, but some applications go beyond just visual facts and focus on affordance.

(2) Model Input: our work is the first to consider both reference instructional video and the video of user's current situation.

(3) Model Output: to complete the user's task, our model needs to operate across multiple steps, necessitating the model to perform an action described by textual answer at each step.

3.3 Task Application of AQTC

Our AQTC task actually proposes a typical and general AI process to assisting humans, *i.e.,* model solves a user's query depicted in text and user view image


00:10 - 00:12
"..use the up and down arrows to navigate to the copy"
00:12 - 00:14
"and press OK,"
00:14 - 00:18
"Subsequently, press the black and white…"
Final dataset
1. Participant records instructional video for a home appliance
2. Quality check to ensure video demonstrates 3-5 functions clearly
3. Annotator labels buttons on the device image taken by participant
4. Annotator creates QA pairs grounded in button labels after watching video
5. Quality check for correctness of labels and QA pairs

Q: How can I copy in black and white?

STEP 1
a) Press [button1] to start
b) Press [button2] or [button6] to select Copy
c) Press [button3] or [button5] to select Copy [...]

Fig. 3. Data Collection Pipeline for our AssistQ benchmark.

by seeking answers from the instructional video. Therefore, the task can support a wide range of applications:

(1) Life assistant: supporting AR glass to guide users in daily life, like operating home appliances, cooking dishes.

(2) Skills training: reducing the labor costs, *e.g.,*, replacing human instructors to teach bicycle repair/PC assembly.

(3) Working partner: improving people's working efficiency, such as recent DARPA's Virtual Partners program. In research, our task is also not limited in a specific scope. It can be a typical task in egocentric AI, which is a new trend in computer vision (*e.g.,* Ego4D dataset).

4 AssistQ Benchmark

We now describe our new benchmark for the AQTC task. The data collection and annotation pipeline is shown in Figure 3. Participants would record videos of themselves operating home appliances, and then annotators would create QA pairs after watching the instructional video.

4.1 Video Collection

Videos. The recruited participants were asked to record instructional videos for home appliances. In each video, the participant demonstrated how to operate a device, such as a washing machine or oven. Participants were asked to propose the functions that they were to record to ensure that the video was sufficiently informative for meaningful question-answering. For example, filming the steps needed to change the temperature of a microwave would result in an information-rich video of interest to AssistQ, whereas filming a video about merely turning the device on would be inadequate. Participants were also encouraged to perform at least 3-5 functions in each video and were given video samples on the recommended placement of their camera to ensure that the videos recorded are of a baseline quality. In addition, participants were required to narrate their actions in the video similar to the audio instructions that accompany product videos which were then submitted.

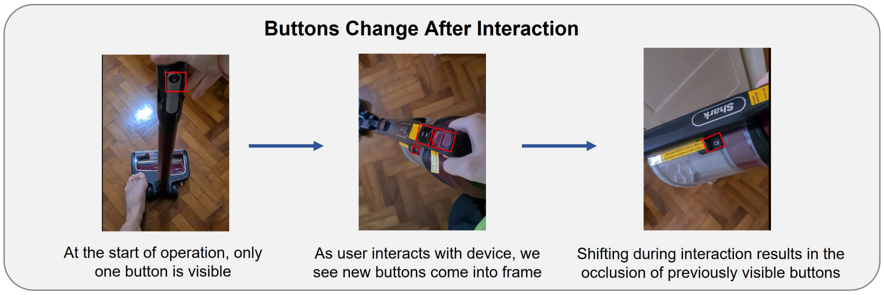

Buttons Change After Interaction
At the start of operation, only one button is visible
As user interacts with device, we see new buttons come into frame
Shifting during interaction results in the occlusion of previously visible buttons

Fig. 4. Sample Device (vacuum cleaner) with new buttons in the video frame.

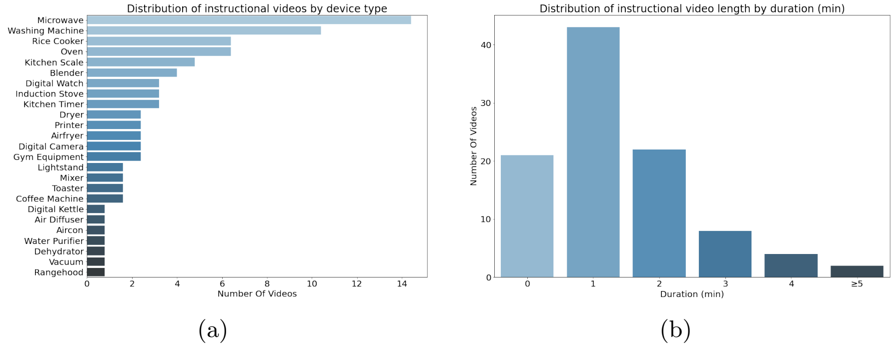
4.2 Query Collection

User images and Button Annotation. For visual query, participants took front-facing pictures of their appliances. In most cases, one image was sufficient to capture all device buttons. But when there is huge change of the device state (Figure 4), participants were asked to provide pictures of the other states to ensure that the agent has sufficient visual information about available choices at each query stage. Annotators were tasked to place bounding boxes over all visible buttons in the image(s), using Makesense.AI annotation tool.

QA pairs. Annotators were asked to create at least 3 questions that were answerable from the video. For greater language diversity in the question set, annotators were encouraged to vary the question stems (*"How do I"*, *"How to"*, *"What to do to"*), action words (*"cook"*, *"bake"*, *"turn"*) and question type. Questions were categories as either specific functionality or pure functionality. Specific functionality questions include specific end-stage values, whereas pure functionality type questions only reference the intended functionality. For example, queries such as *"How to change timer on rice cooker to 3 min"* are specific functionality in nature as they contain a desired end-stage value (3 min), whereas queries such as *"How do I access the timer on the rice cooker"* are pure functionality.

Annotators were asked to write MCQ options grounded to labelled buttons on user image(s), creating a vision-language option space. There were no restrictions on the number of MCQ options, though there should be at least as many MCQ options as there are buttons, with each button referenced in at least one option. The structure of the MCQ options varied by question type. MCQ options for pure functionality questions were typically written as `action <button>`, such as *Press* `<button1>`. For specific functionality questions, relevant function details were appended to the option space *i.e., Press* `<button1>` *to select Defrost*. This evaluated the agent's ability to reason about the user's desired outcome. To push the model to understand the underlying semantics of QA pairs, at least one wrong MCQ option had similar instructions to the correct one.

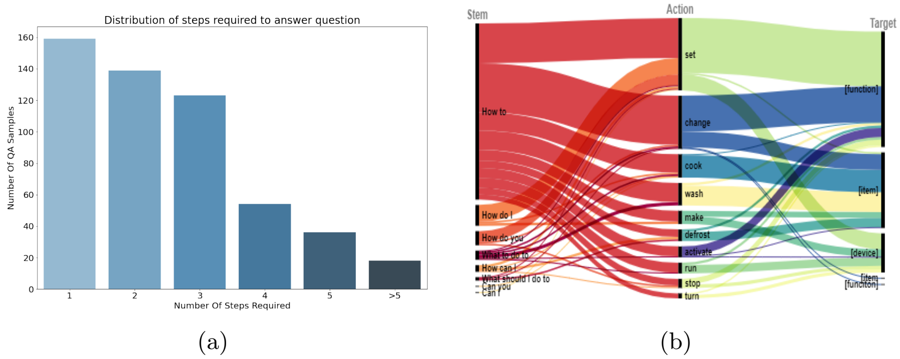
In summary, 18 participants were engaged in video collection; over 350 man-hours were involved in recruitment, filming and liaison over a span of 6 months.



(a)

(b)

Fig. 5. Statistics of Videos in AssistQ. (a) Distribution of devices by appliance types. (b) Distribution of instructional videos by length.



(a)

(b)

Fig. 6. Statistics of Questions in AssistQ. (a) Distribution of steps required to answer generated queries. (b) Analysis of question structure by stem-action-target.

4 annotators were then engaged over another 300 man-hours to label images and create multiple-choice, multi-step QA pairs.

4.3 Dataset Statistics

100 videos with duration averaging 115 seconds were collected. Figure 5(b) shows the distribution of the video duration, with instructional videos spanning 1-2 min having the highest frequency in AssistQ. The videos spanned 25 common household appliances, with microwave, washing machine and oven having the highest frequencies due to their ubiquity in households (Figure 5(a)). The devices came from a diverse pool of 53 distinct appliance brands such as Panasonic (5%), Cornell (4%), Sharp (4%), LG (4%) and Bosch (4%). Instructional videos of more complex devices such as printers (240s) and gym equipment (216s) have longer average duration to capture their larger variety of functions.

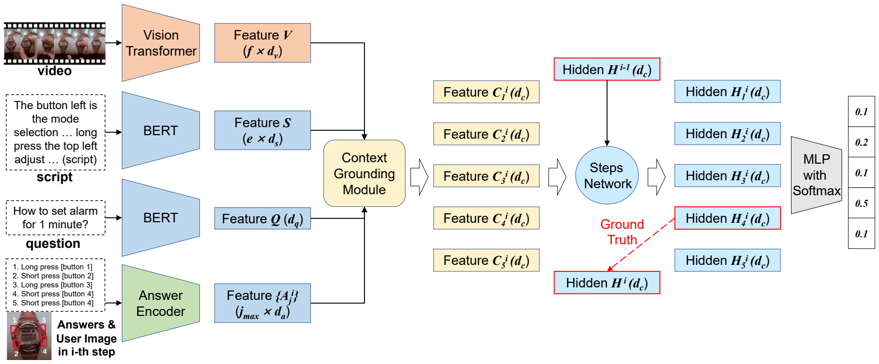


Fig. 7. Architecture of our question-to-answer (Q2A) model. It can be viewed as an encoder-decoder architecture, where the input encoders obtain feature representations for different inputs (video, script, question, answer), and the decoder modules (context grounding, steps network) estimate the answer from encodings. Best viewed in colors.

From the 100 instructional videos, we collected 531 multiple-choice QA pairs (Figure 6) Of which, 251 are pure functionality questions and 278 are specific functionality questions. 70% of QA pairs require more than 2 steps in the answer sequence (Figure 6(a)), rendering a majority of our questions multi-step. Figure 6(b) shows the alluvial chart of question stem-action-target words obtained from the generated queries. Only the top 10 most common action words were included in the alluvial chart for greater clarity. From the chart, we can see that while distributed, questions beginning with *"How to"* and *"How do I"*, actions words *set* and *change* as well as target words *[function]* and *[item]* having the highest occurrence in our dataset. Apart from diversity, this dataset also poses a large number of interesting challenges. See supplementary material for details.

5 Question-to-Actions (Q2A) Model

In this section, we propose a question-to-actions (Q2A) model for the proposed AQTC task. As shown in Figure 7, our Q2A model is an encoder-decoder architecture including: (1) Input Encoder extracts feature from video, script, question, and answers; (2) Context Grounding Module generates context-aware feature for each question-answer pair; (3) Steps Network answers the question in the current step. Next, we describe them in detail.

5.1 Input Encoders

Video. Video encoding aims to obtain the visual location and of buttons. This not only requires the video embedding [16,34,35], but also requires some object information in video embedding [36,38,39]. For simplicity, we just use a pretrained vision transformer (ViT) [15] to encode frames instead of a video

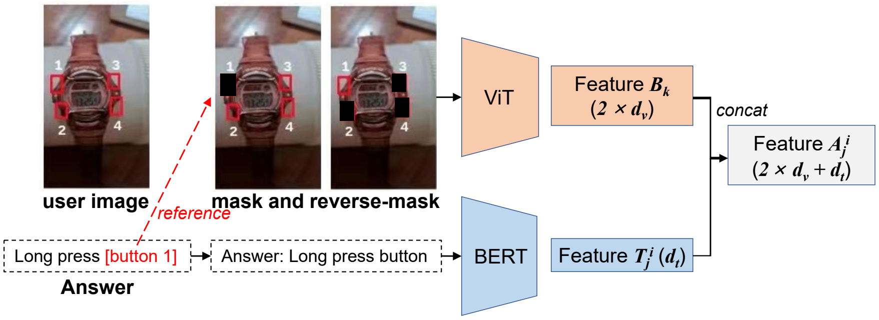


Fig. 8. The architecture of the answer encoder. This figure presents the case of visual button encoding with both mask and reverse-mask images. Note that ViT and BERT used here are the same networks shown in Figure 7. Best viewed in colors.

transformer [4,6], but use a special button encoding method for user image (described latter). We encode one frame per second for the video. Suppose the video has f frames and the encoding dimension is d_v, we can obtain the feature representation V of shape $f \times d_v$.

Script. The video script describes the button's operation and function, which are the crucial cues to answer the question. We use BERT [14] for text embedding. Because the answer in a step usually corresponds to a specific sentence in the script, we transform the script into e sentences, and use the pooled outputs from BERT as sentence embeddings S $(e \times d_s)$, where d_s is the feature dimension.

Question. Following the embedding way of the script, we also use the pooled output from BERT [14] as the question feature Q[1], with the dimension d_q. To keep the consistency, we leverage the same BERT used in the script (*i.e.*, $d_q = d_s$). By this way, the question feature and the script feature may have similarities, which can help answer the question. To distinguish the question from the script, we add the prefix "Question:" to each question before encoding.

The j-th answer of the i-th step. An answer contains both text and visual button (bounding-box in the user image), making it difficult to encode. For answer text, we still use BERT to extract the feature. Like the prefix in question, we also prefix each answer with "Answer:". The challenging part is the button representation. As shown in Figure 8, we encode the user image U with the masked referenced button. Meanwhile, we also encode U with all buttons masked except the referenced button. The visual button feature B_k, and text feature T_j^i are concatenated to the answer's feature A_j^i. The masked button image allows our model to utilize the appearance cues when buttons differ in their appearance, and another reverse-masked image allows our model to focus on the mutual cues (like the relative location) when buttons' appearances are similar.

Context Grounding Module. In video QA models [7,25] there are usually modules that utilize context for better question and answer representation. Fol-

[1] Q_U in Section 3.2 denotes the question under user view. To simplify, we use Q here.

lowing this idea, we also design a context grounding module. Consider the input feature representations V, S, Q, A_j^i (T_j^i as the text encoding, B_j^i as the button encoding), we generate context-aware question-answer pair feature C_j^i as:

$$C_j^i = MLP([T_j^i, B_j^i, At_{QA\rightarrow S}(T_j^i, S, S), At_{QA\dashrightarrow S\dashrightarrow V}(T_j^i, S, V)]) \tag{1}$$

where $At(query, key, value)$ denotes the attention operation [5]. MLP denotes a 2-layer MLP, of which dimension is carefully adjusted to satisfy the calculation requirements. $[\cdot]$ is the concatenate operation. $QA \rightarrow S$ denotes the attention from question-answer (query) to script (key), and produces the QA-aware script feature. We also introduce a "transfer attention" ($QA \dashrightarrow S \dashrightarrow V$) here to simulate the process of finding visual cues based on text:

$$Mask_{QA\dashrightarrow S\dashrightarrow V}(T_j^i, S, V) = Mask_{QA\rightarrow S}(T_j^i, S, S) \times Mask_{S\rightarrow V}(S, V, V), \tag{2}$$

where $Mask$ denotes the attention mask. With the attention mask, we can obtain the QA-aware video feature by $At_{QA\dashrightarrow S\dashrightarrow V} = Mask_{QA\dashrightarrow S\dashrightarrow V} \times V$.

Steps Network. Our Q2A leverages the historical steps via a steps network, which is a 2-layer MLP or a GRU. In i-th step, the steps network would use the state from $i-1$-th step to produce the state H_j^i for j-th answer. Then, the state of the ground-truth answer would be reused as the hidden state of the $i+1$ step. The initial state is a random-initialized vector with standard normal distribution if the step just begins.

Prediction Head. In each step, the hidden states generated by the steps indicate the answers' feature representations. We use a two-layer MLP followed by softmax activation to predict a score for each answer.

Training and Inference. Following common practice in Visual QA/Dialog [1,2], BERT and ViT in the input encoder are frozen since they have a large number of parameters. Other parts are trainable. We use cross-entropy (CE) with softmax activation to calculate the loss. During inference, our Q2A model chooses the state that can yield the highest predicted score after the prediction head. In this way, the model can finish the whole multi-step answering procedure by itself.

6 Experiments

6.1 Experimental Setting

Data Splits. We randomly split 100 instructional videos of AssistQ into the training set and the validation set with a ratio of 8:2. In statistics, the training set has 80 instructional videos with 425 QA instances, and the testing set has 20 instructional videos with 106 QA instances. About 25% device types in the testing set are not present in the training set, making our AssistQ benchmark more challenging.

Evaluation Metrics. As illustrated in Section 4, the "multi-step" characteristic is similar to "multi-round" characteristic in visual dialog [1,12]. Therefore, it is natural to use their evaluation metrics for the proposed AssistQ task:

(a) Ablation on button encoding

Mask	Reverse-mask	R@1 ↑	R@3 ↑	MR ↓	MRR ↑
×	×	17.9	54.8	3.8	2.5
✓	×	19.4	53.6	3.7	2.5
×	✓	**30.2**	62.3	3.2	3.2
✓	✓	21.4	59.1	3.7	2.6

(b) Ablation on input modalities

Video	Script	R@1 ↑	R@3 ↑	MR ↓	MRR ↑
×	×	23.4	57.9	3.6	2.8
✓	×	29.4	59.5	3.3	3.1
×	✓	24.6	58.7	3.6	2.9
✓	✓	**30.2**	62.3	3.2	3.2

(c) Ablation on context grounding

QA→S	S→V	QA⇢S⇢V	R@1 ↑	R@3 ↑	MR ↓	MRR ↑
×	×	×	19.8	52.8	3.8	2.5
✓	×	×	28.6	69.8	3.1	3.2
×	✓	×	22.6	56.0	3.6	2.6
✓	✓	×	29.0	62.7	3.2	3.2
✓	✓	✓	**30.2**	62.3	3.2	3.2

(d) Ablation on historical steps

Network	History	R@1 ↑	R@3 ↑	MR ↓	MRR ↑
MLP	×	21.4	60.7	3.5	2.7
	✓	21.8	56.8	3.7	2.6
GRU	×	25.0	62.3	3.4	2.9
	✓	**30.2**	62.3	3.2	3.2

Table 2. Ablation studies of our proposed Q2A model on the AssistQ benchmark. (a) and (b) are for the Q2A encoder, where (a) explores the necessary input modalities and (b) studies the ways of button encoding. (c) and (d) are for the Q2A decoder, where (c) illustrates the importance of using historical steps and (d) shows different combinations for context grounding. Note → and ⇢ denote attention methods we described in Section 5.1.

- **Recall@k** measures how often the ground-truth answer selected in top k choices. We report Recall@1 and Recall@3 in our experiments.
- **Mean rank (MR)** refers to mean value of the predicted ranking position of the correct answer. The model should pursue lower MR.
- **Mean reciprocal rank (MRR)** is the mean value of the reciprocal predicted ranking position of the correct answer. Higher MRR is better.

Implementation Details. We use PyTorch [30] to perform experiments. For a fair comparison, all models use the same optimizer and learning rate scheduler. Specifically, we use a momentum SGD optimizer with a batch size of 16, and a cosine annealing scheduler [29] with the learning rate 2×10^{-3}, 1 epoch for warmup, and maximum 6 training epochs.

6.2 Ablation Study

As illustrated in Section 5, we design an encoder-decoder architecture Q2A to solve the AQTC task. In its encoder, we encode the video, script, question, and candidate answers, where a special visual button encoding method is designed for the answer encoding. Meanwhile, in the Q2A decoder, we also specifically develop the context grounding module and utilize historical steps for better prediction. A default setting of our model is (a) encode both video and script, (b) double-mask button encoding, (c) $QA \rightarrow S \rightarrow V$ attention, (d) MLP for historical steps. Next, we perform ablation studies on the points mentioned above.

Visual Button Encoding. See Table 2(a), we analyze the encoding ways of visual button encoding. Interestingly, the model achieves the best performance when only using reverse-mask image encoding (mask × and reverse-mask ✓), with a clear margin compared to other configurations. These results demonstrate that it is better to mask the other buttons when encoding a visual button. In our opinion, both mask and reverse-mask schemes can help the model to infer the button location, but the reverse-mask scheme can help the model use the button appearance.

Input Modalities. As shown in Table 2(b), when the model ignores some contexts (video × or script ×), it achieves much lower performance (23.4 v.s. 30.2) than the model with the full contexts (video ✓ and script ✓). We also found that the independent video (video ✓ or script ×) can lead to much better results (29.4 v.s. 24.6) than the independent script (video × or script ✓). This is because when using only script w/o video, some information in scripts might be misleading. For example, say we have two candidate answers: (a) "press here [refer to <button1> in video, but not accessible when using only script] on the microwave"; (b) "press here [refer to <button2>] to heat it up". If the script has a language context (*e.g.,* "press here to heat it up") which is similar to (b), it will mislead the model, making it more inclined to choose (b); but in fact, (a) is the ground truth option, which we can clearly tell by watching the instructional video. These results suggest that the model should fully utilize both video and script to answer questions, rather than only one of them.

Context Grounding.[2] According to Table 2(c), all parts in our proposed grounding schemes (see Section 5.1) can improve the performances. Among them, the model benefits less from $S \rightarrow V$ but obtains a significant gain from $QA \rightarrow S$. This is because $S \rightarrow V$ is unrelated to the question, suggesting the importance of finding the corresponding context according to the question. Moreover, based on $QA \rightarrow S$ and $S \rightarrow V$, the model would benefit from the "transfer attention" $QA \dashrightarrow S \dashrightarrow V$. We believe the "transfer attention" could link QA to V, which simulates finding the related script based on the question, and then locating the related video moment by the related script. Compared with direct $QA \rightarrow V$, the calculation cost of our "transfer attention" is much smaller since the script is shared for all candidate QA pairs.

Steps Network. Table 2(d) presents the ablation on whether to use cues from historical steps (✓ or ×) and how to use them (MLP or GRU [8]). We can find that historical information can stably improve the performances of GRU on all metrics, especially on Recall@1 (25.0 to 30.2). But for MLP with historical information, only a few gains (21.4 to 21.8) can be observed on Recall@1, while other metrics even declined. Compared to MLP, GRU has advantages in temporal modeling, which is more suitable as the steps network.

6.3 Comparison

As shown in Table 3, we compare our Q2A model with baselines in other domains on the AssistQ benchmark. Since the answers encoded in other methods are plain text (without placeholders for buttons), we simply introduce our double-mask answer encoding scheme. For all models, we use the same backbone network in the encoder for a fair comparison. We also provide the results of random guess for improvement comparison.

[2] We use the same notation of Section 5. V: video, S: script, Q: question, A: answer.

Baseline Method	Recall@1 ↑	Recall@3 ↑	MR ↓	MRR ↑
Random Guess	18.3	50.4	3.9	2.3
Multi-stream [25] (Video QA)	22.8 (+4.5)	54.8 (+4.4)	3.7 (-0.2)	3.0 (+0.7)
LateFusion [1] (Video Dialog)	19.9 (+1.6)	49.3 (-1.1)	4.1 (+0.2)	2.6 (+0.3)
PACMAN [9] (EQA)	24.6 (+6.3)	54.0 (+3.6)	3.9 (-0.0)	2.9 (+0.6)
Q2A (Ours)	**30.2 (+11.9)**	**62.3 (+11.9)**	**3.2 (-0.7)**	**3.2 (+0.9)**

Table 3. Baseline comparison on the AssistQ benchmark. Benefiting from the specific design for the benchmark, Q2A achieves the highest performance on all metrics.

Video QA. Multi-stream is proposed as a baseline model for the TVQA [25] task. Like our model, it has modules similar to context grounding. However, It cannot take advantage of cues from historical steps (as video QA is a single-step task), limiting its performance on the AssistQ benchmark.

Video Dialog. [1] proposes LateFusion to solve the multi-round video dialog, which simply concatenates the historical dialog and introduces an LSTM [21] to model them. However, its performance is even worse than Multi-stream. We believe that simple late fusion leads to inferior performance as there is no module similar to context grounding. Also, its LSTM models a very long sequence of the entire script and historical dialogue, which may lose useful information.

EQA. PACMAN [9] uses an LSTM-based planner to update the agent state, with an MLP-based control planner to predict answers. This procedure is similar to our GRU-based decoder that utilizes historical steps, but PACMAN only considers the visual information for navigation. As shown in Table 3, PACMAN achieves better performance than Multi-stream and LateFusion, but still lags behind our Q2A model.

7 Conclusion

In this paper, we proposed Affordance-centric Question-driven Task Completion (AQTC) task, which enables AI assistants to guide users by learning from instructional videos. To support the task, we collected the AssistQ benchmark, consisting of 531 multiple-step QA samples derived from 100 newly filmed instructional videos, with efforts underway to continue scaling this dataset. We also present a new baseline called Question-to-Actions (Q2A) for our AQTC task, and experimental results demonstrate the effectiveness of the Q2A model on the AQTC task. We hope that our proposed AQTC and AssistQ can advance the development of AI assistants that see the world through our eyes, assisting humans in daily, real-world scenarios.

Acknowledgements. This project is supported by the National Research Foundation, Singapore under its NRFF Award NRF-NRFF13-2021-0008, and Mike Zheng Shou's Start-Up Grant from NUS. The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore.

References

1. AlAmri, H., Cartillier, V., Das, A., Wang, J., Cherian, A., Essa, I., Batra, D., Marks, T.K., Hori, C., Anderson, P., Lee, S., Parikh, D.: Audio visual scene-aware dialog. In: CVPR. pp. 7558–7567 (2019)
2. Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., Zhang, L.: Bottom-up and top-down attention for image captioning and visual question answering. In: CVPR. pp. 6077–6086 (2018)
3. Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C.L., Parikh, D.: VQA: visual question answering. In: ICCV. pp. 2425–2433 (2015)
4. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: ICCV. pp. 6836–6846 (2021)
5. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: ICLR (2015)
6. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: ICML (2021)
7. Chadha, A., Arora, G., Kaloty, N.: iperceive: Applying common-sense reasoning to multi-modal dense video captioning and video question answering. arXiv:2011.07735 (2020)
8. Chung, J., Gülçehre, Ç., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555 (2014)
9. Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., Batra, D.: Embodied question answering. In: CVPR. pp. 1–10 (2018)
10. Das, A., Gervet, T., Romoff, J., Batra, D., Parikh, D., Rabbat, M., Pineau, J.: Tarmac: Targeted multi-agent communication. In: ICML. pp. 1538–1546 (2019)
11. Das, A., Gkioxari, G., Lee, S., Parikh, D., Batra, D.: Neural modular control for embodied question answering. In: CoRL. pp. 53–62 (2018)
12. Das, A., Kottur, S., Gupta, K., Singh, A., Yadav, D., Moura, J.M.F., Parikh, D., Batra, D.: Visual dialog. In: CVPR. pp. 1080–1089 (2017)
13. Das, A., Kottur, S., Moura, J.M.F., Lee, S., Batra, D.: Learning cooperative visual dialog agents with deep reinforcement learning. In: ICCV. pp. 2970–2979 (2017)
14. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL. pp. 4171–4186 (2019)
15. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)
16. Feichtenhofer, C., Fan, H., Xiong, B., Girshick, R.B., He, K.: A large-scale study on unsupervised spatiotemporal representation learning. In: CVPR (2021)
17. Gao, D., Wang, R., Bai, Z., Chen, X.: Env-qa: A video question answering benchmark for comprehensive understanding of dynamic environments. In: CVPR. pp. 1675–1685 (2021)
18. Gordon, D., Kembhavi, A., Rastegari, M., Redmon, J., Fox, D., Farhadi, A.: IQA: visual question answering in interactive environments. In: CVPR. pp. 4089–4098 (2018)
19. Goyal, Y., Khot, T., Agrawal, A., Summers-Stay, D., Batra, D., Parikh, D.: Making the V in VQA matter: Elevating the role of image understanding in visual question answering. International Journal of Computer Vision **127**(4), 398–414 (2019)
20. Grauman, K., Westbury, A., Byrne, E., Chavis, Z., Furnari, A., Girdhar, R., Hamburger, J., Jiang, H., Liu, M., Liu, X., Martin, M., Nagarajan, T., Radosavovic, I.,

Ramakrishnan, S.K., Ryan, F., Sharma, J., Wray, M., Xu, M., Xu, E.Z., Zhao, C., Bansal, S., Batra, D., Cartillier, V., Crane, S., Do, T., Doulaty, M., Erapalli, A., Feichtenhofer, C., Fragomeni, A., Fu, Q., Fuegen, C., Gebreselasie, A., Gonzalez, C., Hillis, J., Huang, X., Huang, Y., Jia, W., Khoo, W., Kolar, J., Kottur, S., Kumar, A., Landini, F., Li, C., Li, Y., Li, Z., Mangalam, K., Modhugu, R., Munro, J., Murrell, T., Nishiyasu, T., Price, W., Puentes, P.R., Ramazanova, M., Sari, L., Somasundaram, K., Southerland, A., Sugano, Y., Tao, R., Vo, M., Wang, Y., Wu, X., Yagi, T., Zhu, Y., Arbelaez, P., Crandall, D., Damen, D., Farinella, G.M., Ghanem, B., Ithapu, V.K., Jawahar, C.V., Joo, H., Kitani, K., Li, H., Newcombe, R., Oliva, A., Park, H.S., Rehg, J.M., Sato, Y., Shi, J., Shou, M.Z., Torralba, A., Torresani, L., Yan, M., Malik, J.: Ego4d: Around the world in 3, 000 hours of egocentric video. arXiv:2110.07058 (2021)

21. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997)
22. Jain, U., Weihs, L., Kolve, E., Rastegari, M., Lazebnik, S., Farhadi, A., Schwing, A.G., Kembhavi, A.: Two body problem: Collaborative visual task completion. In: CVPR. pp. 6689–6699 (2019)
23. Jang, Y., Song, Y., Yu, Y., Kim, Y., Kim, G.: TGIF-QA: toward spatio-temporal reasoning in visual question answering. In: CVPR. pp. 1359–1367 (2017)
24. Kottur, S., Moura, J.M.F., Parikh, D., Batra, D., Rohrbach, M.: Visual coreference resolution in visual dialog using neural module networks. In: ECCV. pp. 160–178 (2018)
25. Lei, J., Yu, L., Bansal, M., Berg, T.L.: TVQA: localized, compositional video question answering. In: EMNLP. pp. 1369–1379 (2018)
26. Lei, J., Yu, L., Berg, T.L., Bansal, M.: TVQA+: spatio-temporal grounding for video question answering. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (eds.) ACL. pp. 8211–8225 (2020)
27. Li, Z., Li, Z., Zhang, J., Feng, Y., Zhou, J.: Bridging text and video: A universal multimodal transformer for audio-visual scene-aware dialog. IEEE ACM Trans. Audio Speech Lang. Process. **29**, 2476–2483 (2021)
28. Lin, K.Q., Wang, A.J., Soldan, M., Wray, M., Yan, R., Xu, E.Z., Gao, D., Tu, R., Zhao, W., Kong, W., Cai, C., Wang, H., Damen, D., Ghanem, B., Liu, W., Shou, M.Z.: Egocentric video-language pretraining. arXiv:2206.01670 (2022)
29. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with warm restarts. In: ICLR (2017)
30. Paszke, A., Gross, S., Massa, F.e.a.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS. pp. 8026–8037 (2019)
31. Sax, A., Zhang, J.O., Emi, B., Zamir, A.R., Savarese, S., Guibas, L.J., Malik, J.: Learning to navigate using mid-level visual priors. In: CoRL. pp. 791–812 (2019)
32. Schwartz, I., Schwing, A.G., Hazan, T.: A simple baseline for audio-visual scene-aware dialog. In: CVPR. pp. 12548–12558 (2019)
33. Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R., Fidler, S.: Movieqa: Understanding stories in movies through question-answering. In: CVPR. pp. 4631–4640 (2016)
34. Tong, Z., Song, Y., Wang, J., Wang, L.: Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. arXiv:2203.12602 (2022)
35. Wang, A.J., Ge, Y., Yan, R., Yuying, G., Lin, X., Cai, G., Wu, J., Shan, Y., Qie, X., Shou, M.Z.: All in one: Exploring unified video-language pre-training. arXiv:2203.07303 (2022)
36. Wang, J., Ge, Y., Cai, G., Yan, R., Lin, X., Shan, Y., Qie, X., Shou, M.Z.: Object-aware video-language pre-training for retrieval. In: CVPR. pp. 3313–3322 (2022)

37. Wortsman, M., Ehsani, K., Rastegari, M., Farhadi, A., Mottaghi, R.: Learning to learn how to learn: Self-adaptive visual navigation using meta-learning. In: CVPR. pp. 6750–6759 (2019)
38. Yan, R., Shou, M.Z., Ge, Y., Wang, A.J., Lin, X., Cai, G., Tang, J.: Video-text pre-training with learned regions. arXiv:2112.01194 (2021)
39. Yan, R., Xie, L., Tang, J., Shu, X., Tian, Q.: Higcin: Hierarchical graph-based cross inference network for group activity recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–1 (2020)
40. Yang, W., Wang, X., Farhadi, A., Gupta, A., Mottaghi, R.: Visual semantic navigation using scene priors. In: ICLR (2019)
41. Yu, L., Chen, X., Gkioxari, G., Bansal, M., Berg, T.L., Batra, D.: Multi-target embodied question answering. In: CVPR. pp. 6309–6318 (2019)