

This is the appendix for *PUO-Bench: A Panel Understanding and Operation Benchmark with A Privacy-Preserving Framework*. In this document, we present more visualization details to demonstrate the contribution of PUO benchmark.

- In Sec. A, we provide more details about the training process;
- In Sec. B, we present prompts and examples to demonstrate how LJS-F1 and LJS-Acc are calculated;
- Sec. C presents the human evaluation of the LJS score, showing its variation;
- Sec. D describes how the parser on the edge side is trained;
- Sec. E visualizes the result differences between the zero-shot and fine-tuned models.

A Training Details

During the training process, we fine-tuned all VLMs using LoRA on a combined dataset consisting of QA pairs from four distinct tasks to measure the model’s performance on the PUO task.

The fine-tuning was conducted using LLaMA Factory [64], leveraging the SFT stage with a learning rate of 1e-4 for 10 epochs on a machine equipped with 8 A100 GPUs. We allowed the model to generate up to 4096 new tokens per response. To ensure diverse and natural responses, we set the temperature to 0.95 and applied Top-p sampling (nucleus sampling) with a p-value of 0.7.

B Details about LLM-as-Judge

In Section 6, we present two metrics for panel description, function estimation, and multi-step operation planning: LJS-F1 and LJS-Acc, as defined in (5) and (6), respectively. It is important to note that our approach differs from the vanilla method [63], which uses an LLM to judge the quality of outputs using continuous scores. In contrast, we employ discrete, binary scoring to assess whether the prediction matches the GT.

B.1 LJS-F1 for panel description

For panel description, we use the prompt like this:

The prompt for LJS-F1 in panel description

The following markdown content provides two descriptions of the same control panels on an appliance. According to these descriptions, output a number 'A' indicating how many elements are in the first description, a number 'B' indicating how many elements are in the second description, and a number 'C' indicating how many elements are in both the first and the second description. The 'element' means a button or a knob.

*“‘md
{ ground truth }
““*

*“‘md
{ prediction }
““*

Analysis should be present. In the last line of the answer, outputting three numbers bounded with square brackets: "[A, B, C]", indicating element number in the first and the second description, and the number of common elements.

In Fig. 8, we present an example to show how LLM judges the element count in ground truth and prediction, and how the number of common elements is estimated.



Figure 8: LLM-F1 for evaluating panel description task. $2C/(A + B) = 0.67$ here.

569 B.2 LJS-Acc for function estimation

570 For function estimation, we only let the LLM to determine whether the prediction and ground truth
571 describe the same element in the function level:

The prompt for LJS-Acc in function estimation

The following markdown content provides two descriptions of two buttons or knobs on an appliance. Judge whether the two pieces of content describe the same button/knob and the same function.

“*md*
{ ground truth }
“

572

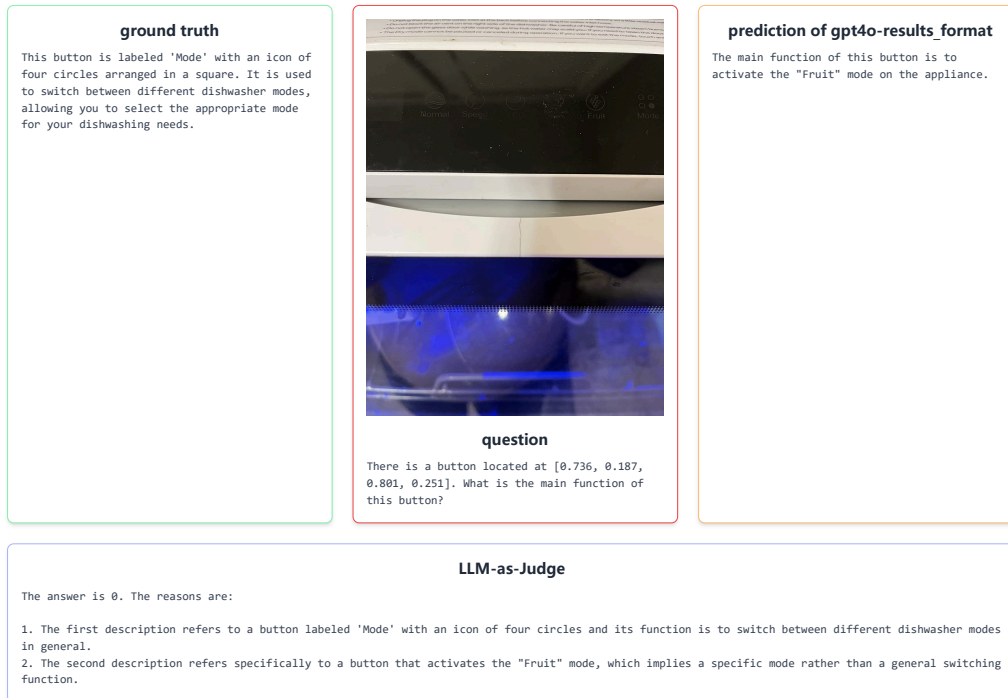


Figure 9: LLM-as-Judge in LLM-Acc for evaluating function estimation task. For the example, the judge result is false.

“*md*
{ prediction }

““
The result should be start with 'The answer is 1/0', with 1 indicating they are the same and 0 indicating they are different. No analysis should be output if the answer is 1, but the reason should be provided if and only if the answer is 0.

Fig. 9 demonstrates an example of how the LLM determines whether the prediction and the ground truth describe the same element. LJS-Acc is calculated on the entire test set.

B.3 LJS-Acc for goal-based planning

For goal-based planning, we let LLM judge whether the plan in ground truth and prediction complete the same target. The prompt is:

The prompt for LJS-Acc in goal-based planning

The content in the following two markdowns describes the operational processes required to achieve a certain goal. The first one is the correct sequence of operational steps. Determine whether the second process is correct according to the first one. Some steps might be interchangeable, which should be consider correct. Overall, if there are no significant deviations (operating the wrong elements, omitting some key elements, etc.), it should be considered correct.

“*md*
{ ground truth }

““

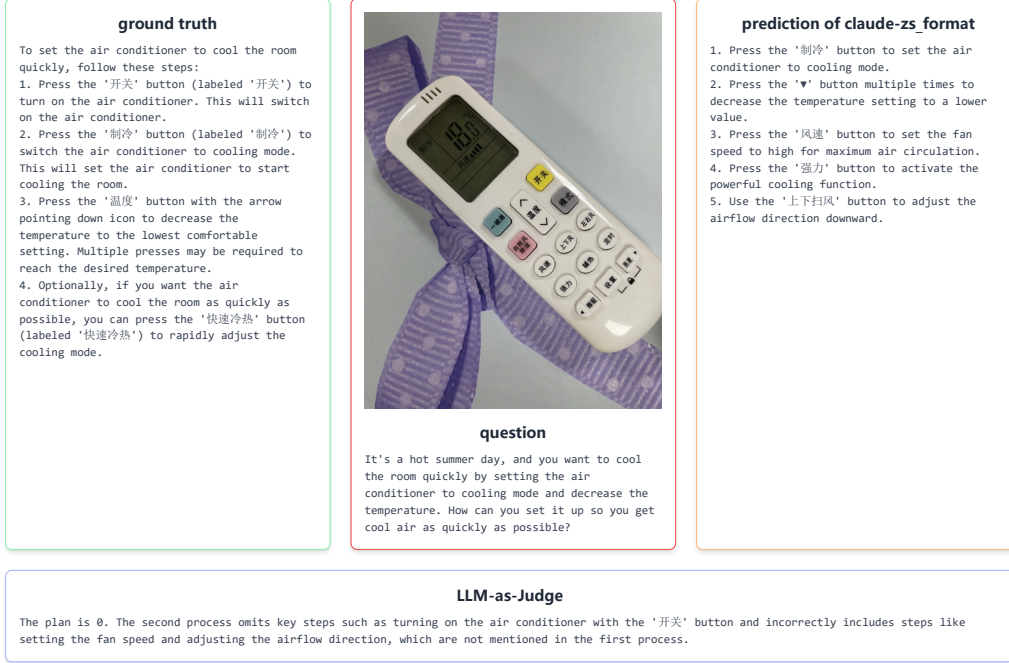


Figure 10: LLM-as-Judge in LLM-Acc for evaluating goal-based planning task. For the example, the judge result is false since the predicted plan omits the step to turn on the air conditioner.

Table 4: Human evaluation results on LJS(MAE).

TASK	zero-shot		fine-tuned			PPF		avg.
	GPT-4o	Claude-3.7	Yi-VL	LLAVA	Qwen-VL	(w/ GPT-4o)	(w/ Claude)	
description	0.097	0.176	0.074	0.115	0.081	0.073	0.076	0.099
function	0.130	0.110	0.090	0.170	0.120	0.130	0.080	0.112
planning	0.090	0.140	0.110	0.120	0.120	0.150	0.110	0.120

“*md*
{ prediction }

“

The result should start with "The plan is 1/0", where 1 indicates that the steps are correct, and 0 indicates they are not. No analysis should be output. The reason should be provided if and only if the plan is 0.

Fig. 10 demonstrates an example on how to cool the room using an air conditioner.

C Human evaluation on the LJS score

To evaluate the reliability of LJS in assessing the consistency between predicted answers and the ground truth (GT), we conducted a human evaluation study.

A total of 10 participants took part in the study. Each participant was presented with 10 randomly selected questions from each of the three tasks, with 7 model predictions per question, resulting in $10 \times 3 \times 7 = 210$ Question-Answer-GT triplets. The human annotators independently judged whether each prediction was semantically consistent with the corresponding GT.

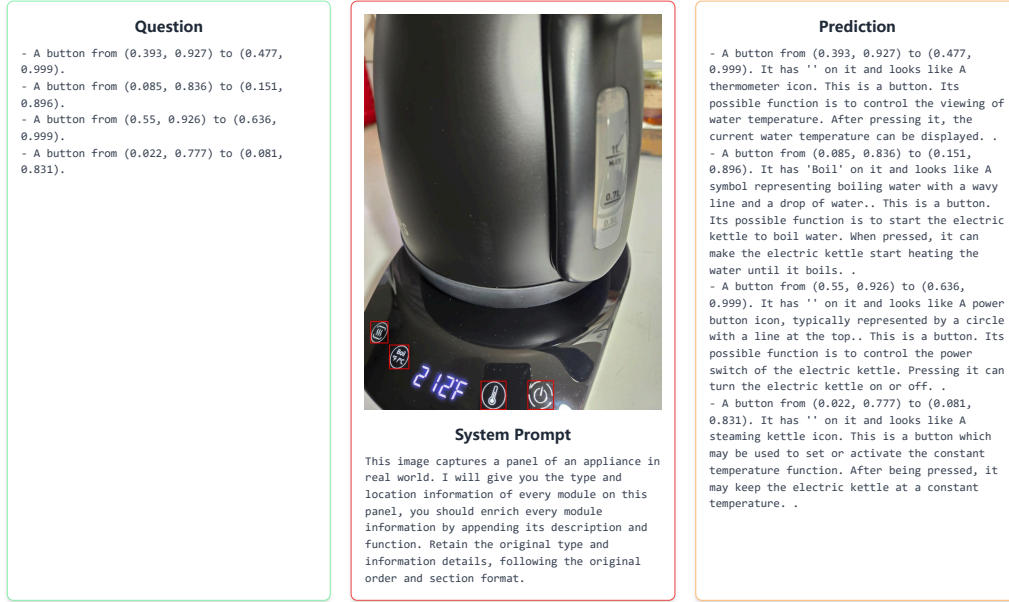


Figure 11: An example of parser with YOLO results as the question.

The results are presented in Tab. 4. The MAE measures the average deviation between the LLM’s consistency judgments and the majority vote of human annotators:

$$MAE = \frac{1}{N} \sum_{i=1}^N |S_{llm}^{(i)} - S_{human}^{(i)}|, \quad (8)$$

where $S_{llm}^{(i)}$ and $S_{human}^{(i)}$ indicate the scores for each triplets judged by LLM and human, respectively. This metric provides an interpretable measure of how closely the LLM’s evaluations align with human consensus. The average errors in the last column indicate that the LLM-as-judge makes mistakes at a rate of approximately 0.1, which is relatively stable across evaluations. This suggests that LLM-based metrics such as LLM-F1 and LLM-Acc can be considered acceptable, provided that the error margin is taken into account.

D Edge-side model

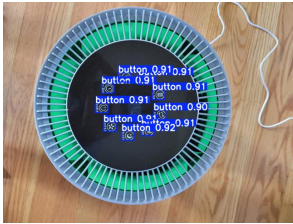
In PPF, a small VLM is deployed at the edge side to estimate the function of each detected elements by YOLO. For this model, the prompt is fixed and follows the formulation of

The prompt for estimating function in the edge side

This image captures a panel of an appliance in real world. I will give you the type and location information of every module on this panel, you should enrich every module information by appending its description and function. Retain the original type and information details, following the original order and section format.

A button from (left₁, top₁) to (right₁, bottom₁).
A button from (left₂, top₂) to (right₂, bottom₂).
A button from (left₃, top₃) to (right₃, bottom₃).
 ...

Some examples are presented in Fig. 11 and Fig. 12. In the top row, we demonstrate the question-answer pairs for training the edge-side parsing model, while the bottom row presents two examples of enhanced results.



📍 A knob from (0.348, 0.643) to (0.679, 0.858).

👁️ It has 'START + 30 SEC TIME WEIGHT PROG LEVEL' on it.

⚙️ This is a knob that may be used to select different functions of the microwave oven, such as starting heating, increasing by 30 seconds each time, defrosting, setting the time, defrosting by weight, or adjusting the power level. Rotating this knob can select different options.

👂 It has '**SPEED**' on it and looks like **fan** icon.

Figure 12: Examples of the parsing of button and knob that are enhanced on the edge-side model.

604 E Visualization comparison of model predictions

From Fig. 13 to Fig. 16, we compare the zero-shot and fine-tuned results of Qwen-2.5-7B VL model on four tasks, which demonstrate that zero-shot model is short of knowledge on panel understanding and operation.

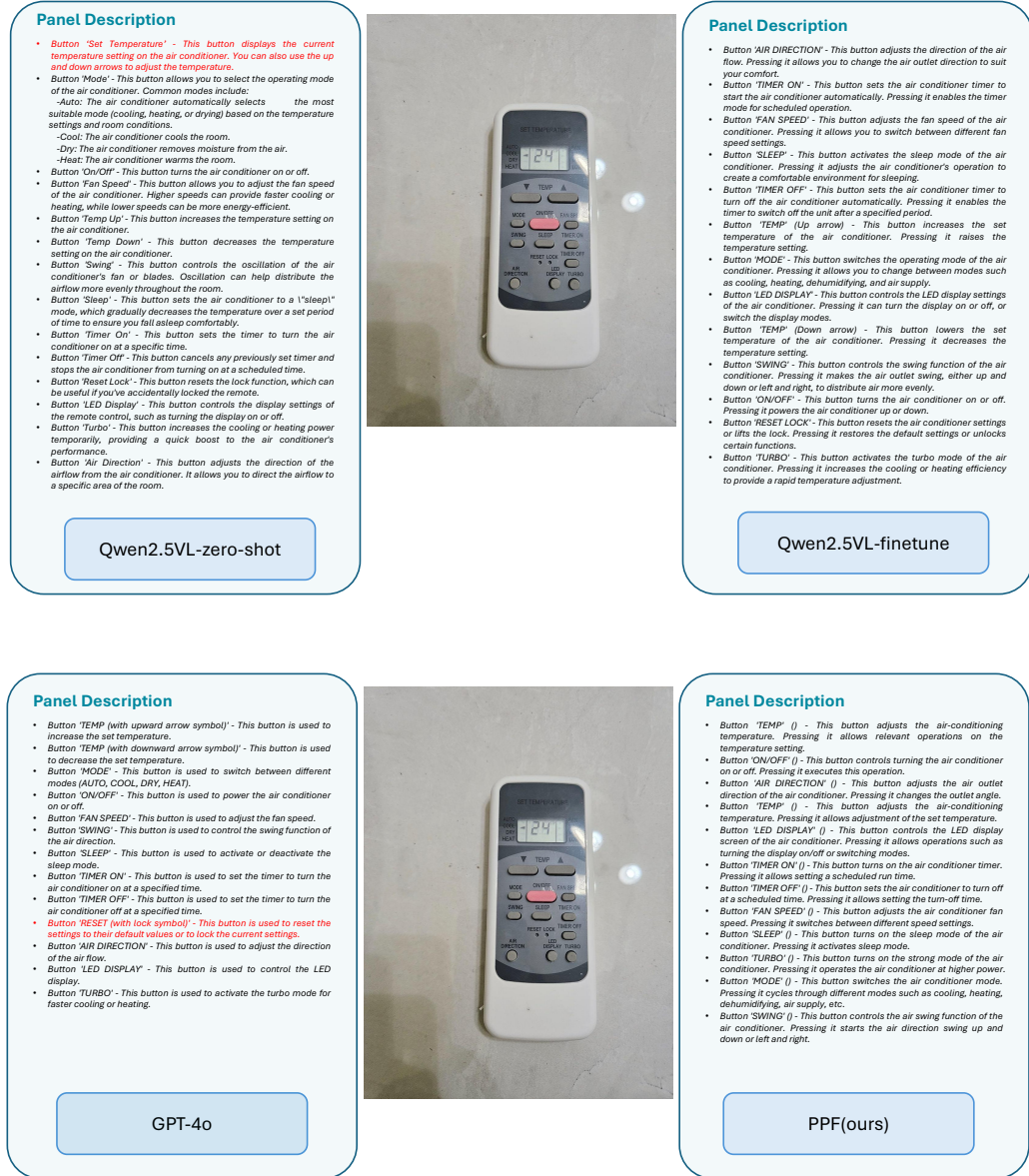


Figure 13: An example demonstrating models' ability in the *panel description* task. The zero-shot VLM mistakenly identifies the character "set temperature" at the top as a button. On the other hand, GPT-4o misidentifies the reset indicator as a button.

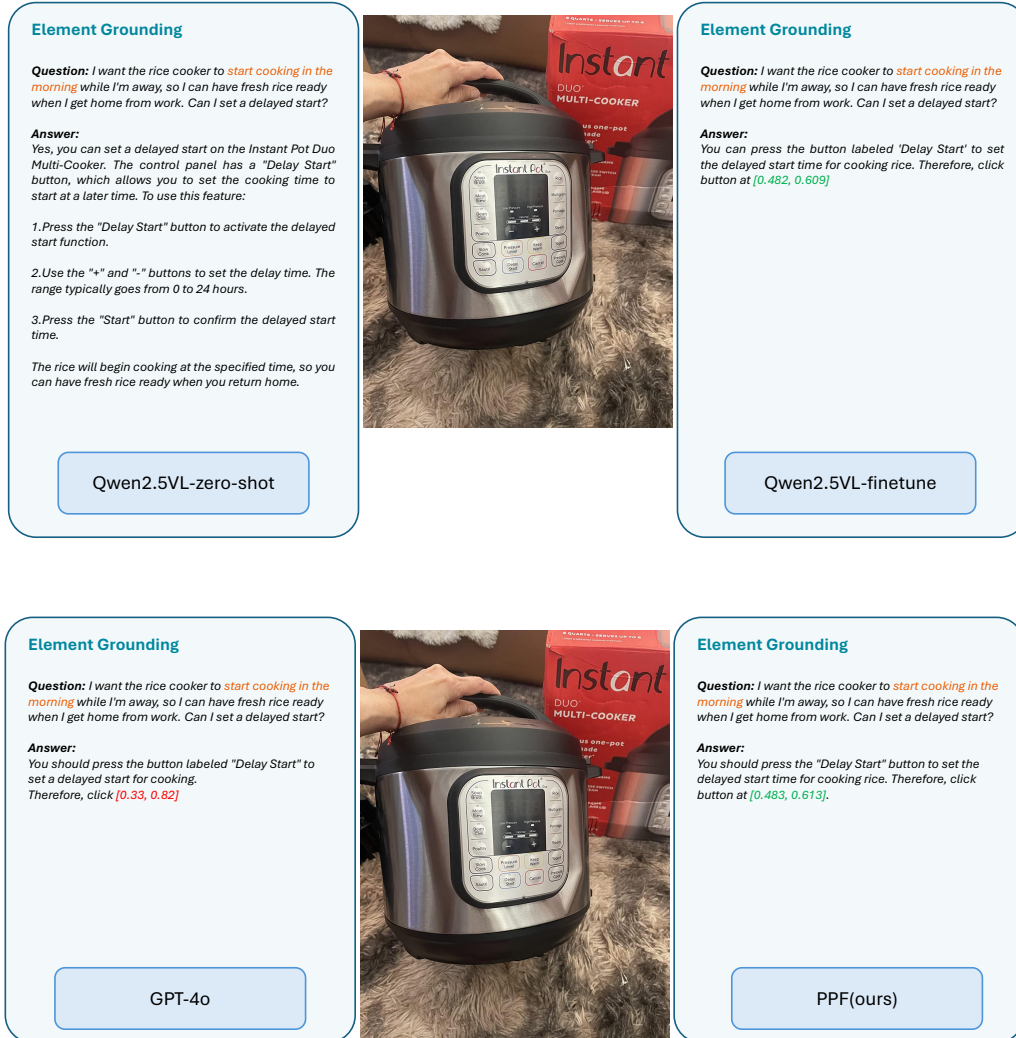


Figure 14: An example demonstrating models' ability in the *element grounding* task. The zero-shot VLM cannot generate a coordinate for visualization, while GPT-4o has produced an incorrect 2d coordinate.

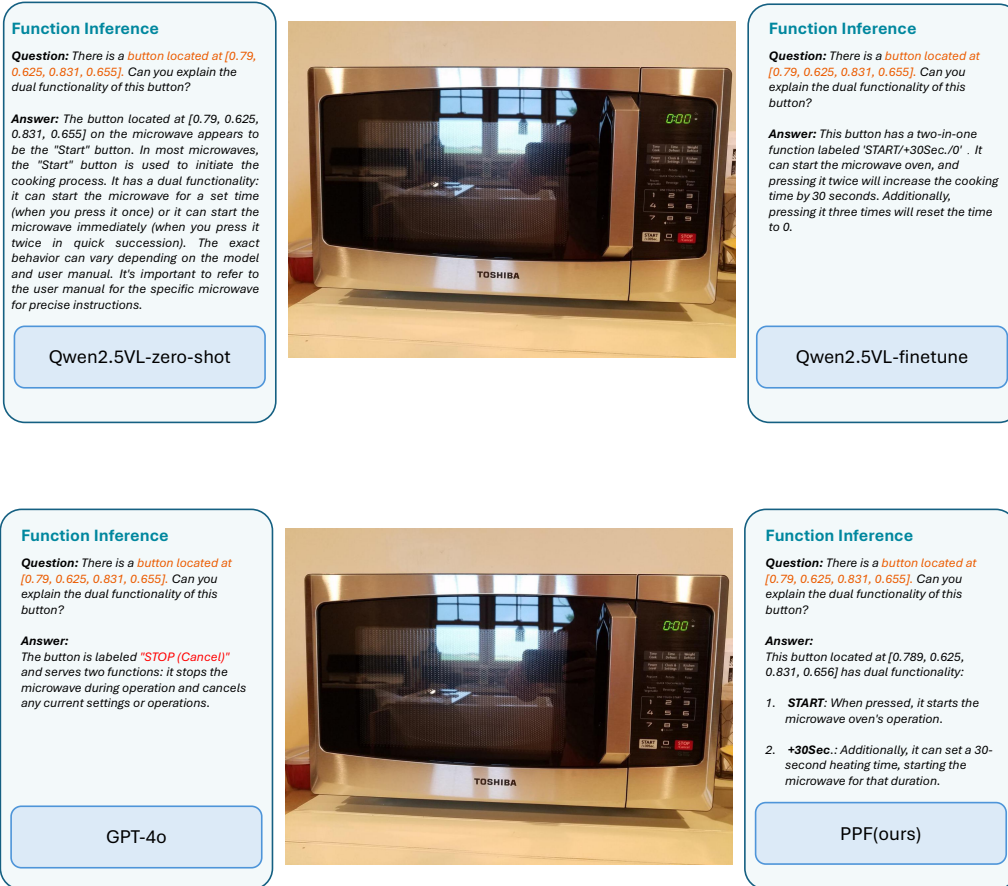


Figure 15: An example demonstrating models' ability in the *function inference* task. The zero-shot model failed to recognize the function of "+30 sec", while the result of GPT-4o is completely wrong, which is entirely different from the specified button.

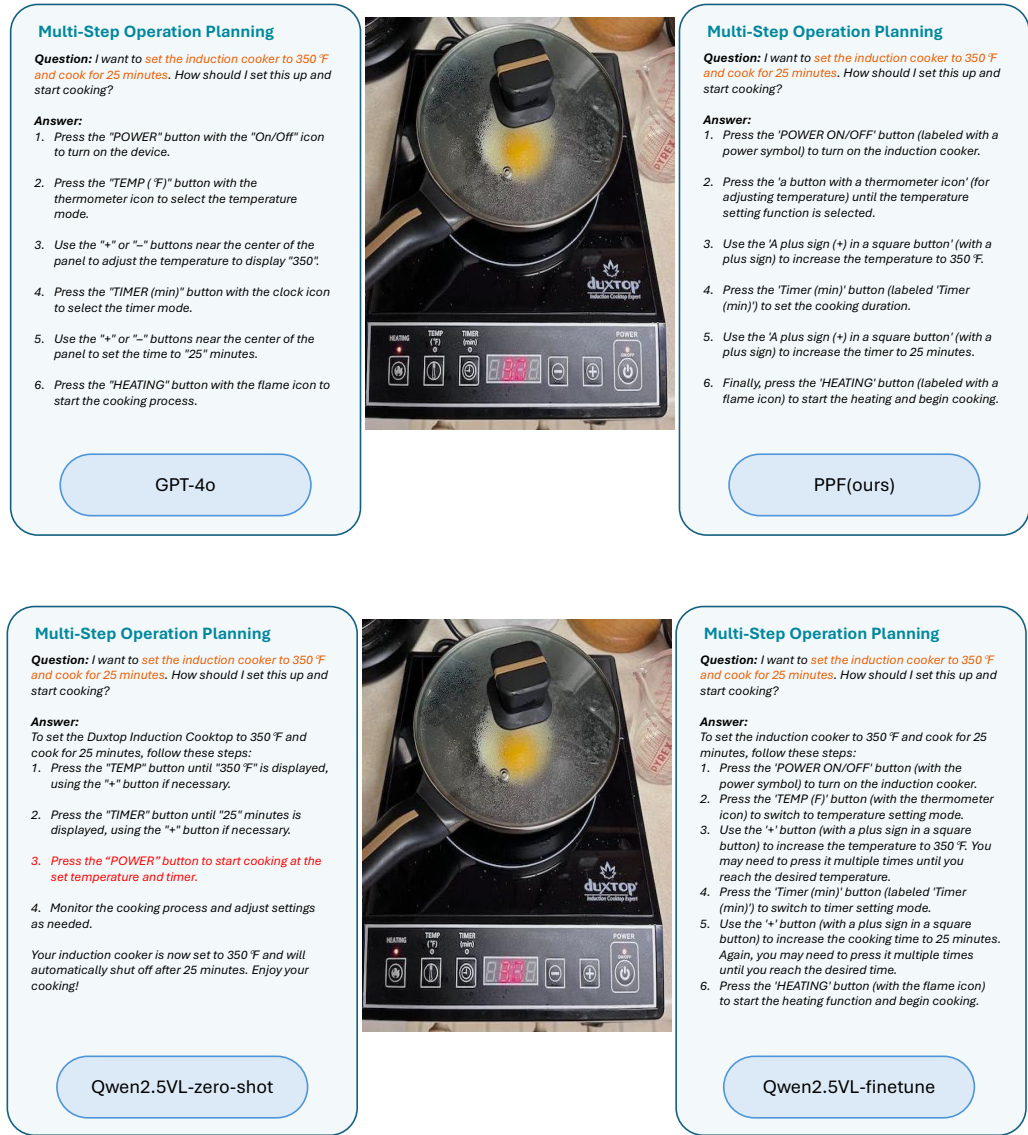


Figure 16: An example demonstrating models’ ability in the multi-step operation planning task. The zero-shot one advises to press the “power” button at the third step, which will turn off the induction cooker. This action deviates from the planning goal.