

OS-Oracle: A Comprehensive Framework for Cross-Platform GUI Critic Models

Supplementary Material

1. OS-Oracle Dataset Details and Statistics

Unlike prior GUI critic datasets that focus on a single platform or interaction modality, as shown in Table 1, OS-Oracle provides a unified large-scale corpus that spans **desktop**, **web**, and **mobile** environments. Existing datasets typically emphasize only mobile or only desktop tasks, which limits a critic model’s ability to generalize across heterogeneous GUI ecosystems. In contrast, OS-Oracle offers balanced multi-platform coverage and serves as a comprehensive foundation for training robust cross-platform critics.

Table 1. CUA critic model dataset statistics.

Data source	Platform			Samples
	Mobile	Web	Desktop	
SEAgent [1]	✗	✗	✓	4,245
GUI-Critic-Train [2]	✓	✗	✗	11k
UI-Genie-RM-517k [3]	✓	✗	✗	517k
Ours	✓	✓	✓	310k

Table 2 presents an overview of OS-Oracle training corpus, detailing the positive and negative samples contributed by each dataset. OS-Oracle synthesizes data from seven open-source datasets, resulting in a total of **162,760** positive samples and **150,667** negative samples. Overall, approximately **58.32%** of the data comes from mobile environments, **23.17%** from desktop, and **18.51%** from web tasks.

Table 2. OS-Oracle datasets statistics overview.

Training dataset	Platform	#Positive	#Negative
AndroidControl	Mobile	52,466	52,162
AMEX	Mobile	9,370	7,154
AITZ	Mobile	11,485	9,107
GUI-Odyssey	Mobile	20,198	20,834
Mind2Web	Web	6,326	6,578
ScaleCUA-Web	Web	22,846	22,270
AgentNet	Desktop	40,069	32,562
Total		162,760	150,667

2. OS-Critic Bench

OS-Critic Bench is a curated cross-platform evaluation suite comprising **738** carefully designed test cases spanning three

major GUI environments: 438 mobile, 152 desktop, and 148 web tasks. The benchmark is constructed from representative sources across all platforms: for Mobile, the AndroidControl and GUIOdyssey test sets; for Web, the guiact test set and a dedicated, non-overlapping subset of ScaleCUA-Web; and for Desktop, AgentNet-Bench. For each test case, the model receives a screenshot, the task instruction, the step-by-step action history, and a candidate action. The model must determine whether the candidate action is appropriate for completing the task, and its final verification is compared against human-annotated ground truth. OS-Critic Bench assesses the model’s robustness under heterogeneous UI layouts, interaction modalities, and error patterns.

As illustrated in Figure 1, OS-Critic Bench spans not only common GUI operations such as `click`, `type`, `scroll`, and `terminate`, but also platform-specific interaction patterns. For mobile environments, the benchmark includes actions such as `swipe`, `long_press`, and `open`; for desktop scenarios, it incorporates operations like `right_click`, `key`, `move`, and `drag`. To better align with everyday human-GUI behavior, we calibrate the action distribution on each platform so that the relative frequency of actions mirrors realistic usage patterns. We further enforce a balanced 1:1 ratio between positive and negative instances to mitigate distributional biases and discourage trivial hacks, making OS-Critic Bench a realistic and high-quality benchmark for critic evaluation.

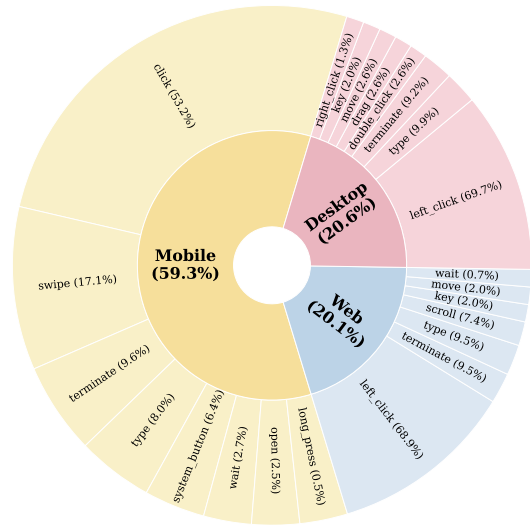


Figure 1. The action distribution of OS-Critic Bench.

3. Unified Prompt

We adopt a unified prompt format for both training and inference of OS-Oracle-7B. Specifically, the mobile version of the prompt is provided in Table 3, while the desktop and web versions are shown in Table 4.

Across platforms, the prompt follows a consistent structure consisting of three main components:

1. **Task framing and critic instructions:** clear guidance defining the critic’s role and expected output format.
2. **Action space specification:** a detailed description of all permissible actions. We align our definitions with the action spaces used in the Qwen2.5-VL mobile agent and computer-use agent implementations to ensure compatibility and coverage.
3. **Task context:** including the current task instruction, the step-by-step action history, and the candidate action to be evaluated.

This unified design ensures consistent behavior across platforms while capturing the unique interaction patterns of mobile, desktop, and web environments.

4. Additional Discussion on Online Pre-Critic Performance

Although OS-Oracle-7B demonstrates clear gains in the online pre-critic setting, the improvements on AndroidWorld and OSWorld are relatively modest when UI-TARS is used as the base agent. We attribute this phenomenon primarily to the limited instruction-following capability of UI-TARS, which restricts its ability to effectively incorporate critic feedback and revise inappropriate actions. In other words, the bottleneck is not only the critic’s judgment quality, but also the agent’s capacity to utilize that feedback during interaction.

This observation is further supported by the comparative results in Table 5. When applied to UI-TARS, both GPT-4o and GUI-Critic-R1 may even degrade the agent’s performance, whereas OS-Oracle-7B consistently brings improvement. This suggests that stronger critic judgments alone do not necessarily translate into better online task success if the underlying agent cannot reliably interpret and act on the feedback.

To further verify this point, we additionally evaluate the pre-critic setting with a stronger general-purpose agent, Qwen2.5-VL-72B. In this case, OS-Oracle-7B yields substantial gains on both AndroidWorld and OSWorld, with improvements exceeding 5% on each benchmark, and also outperforms GUI-Critic-R1 by a clear margin. These results provide complementary evidence that the effectiveness of OS-Oracle-7B is not limited to offline critic evaluation, but can also transfer to realistic online environments when paired with an agent that has stronger feedback utilization ability.

5. Case Study

5.1. Pre-Critic in Online Benchmark

To evaluate whether our critic model can provide reliable, step-level guidance in realistic execution settings, we conduct pre-critic experiments on both OSWorld and AndroidWorld online environments. In this setup, the base agent first proposes an action; the critic then evaluates whether the action is appropriate. If the critic judges the action as incorrect, the agent is prompted to regenerate an alternative action for up to three attempts before execution proceeds. This procedure reflects how critic models can be deployed to stabilize long-horizon GUI tasks and prevent irreversible errors.

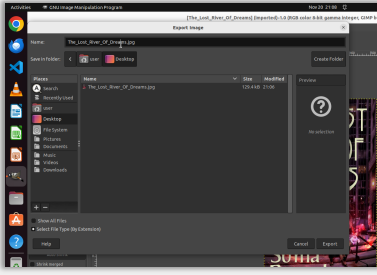
Fig. 2 and Fig. 3 illustrate representative cases in OSWorld and AndroidWorld where GPT-4o provides hallucinated or incorrect critic judgments, while our OS-Oracle-7B offers accurate, state-grounded assessments.

5.2. GRPO vs. CP-GRPO

We conduct experiments between OS-Oracle-7B and its GRPO-finetuned variant OS-Oracle-7B-GRPO in AndroidWorld Environment. As illustrated in Figure 4, OS-Oracle-7B-GRPO occasionally produces critiques that are inconsistent with its final verification outcome—indicating a misalignment between intermediate reasoning and the final judgment. In contrast, OS-Oracle-7B maintains stable and coherent behavior, delivering both correct critiques and consistent verification results. This example highlights the importance of consistency-preserving objectives such as CP-GRPO in stabilizing critic reasoning and ensuring reliable decision quality.

References


- [1] Zeyi Sun, Ziyu Liu, Yuhang Zang, Yuhang Cao, Xiaoyi Dong, Tong Wu, Dahua Lin, and Jiaqi Wang. Seagent: Self-evolving computer use agent with autonomous learning from experience. *arXiv preprint arXiv:2508.04700*, 2025. 1
- [2] Yuyang Wanyan, Xi Zhang, Haiyang Xu, Haowei Liu, Junyang Wang, Jiabo Ye, Yutong Kou, Ming Yan, Fei Huang, Xiaoshan Yang, et al. Look before you leap: A gui-critic-r1 model for pre-operative error diagnosis in gui automation. *arXiv preprint arXiv:2506.04614*, 2025. 1
- [3] Han Xiao, Guozhi Wang, Yuxiang Chai, Zimu Lu, Weifeng Lin, Hao He, Lue Fan, Liuyang Bian, Rui Hu, Liang Liu, et al. Ui-genie: A self-improving approach for iteratively boosting mllm-based mobile gui agents. *arXiv preprint arXiv:2505.21496*, 2025. 1



Task Instruction: Could you assist me in placing my photo on the desktop and renaming it to export.jpg?

Action History: ...
 click(start_box='(137,408)')
 click(start_box='(355,122)')

Current Action: hotkey(key='ctrl a')





<reasoning>
 The goal is to place the photo on the desktop and rename it to "export.jpg". The screenshot shows the "Export Image" dialog in GIMP, with the current file name "The_Lost_River_Of_Dreams.jpg" already selected. The proposed action is to use the hotkey "Ctrl + A", which typically selects all text in a text field.

1. **Analyze the Goal**: The user wants to rename the file to "export.jpg" and save it on the desktop.
2. **Examine the Screenshot**: The file name is already highlighted, indicating it is ready for editing. The file is set to be saved on the desktop.
3. **Review the History**: Previous actions involved navigating to this dialog, likely to prepare for saving the file.
4. **Evaluate the Proposed Action**: Using "Ctrl + A" is unnecessary because the text is already selected. The next logical step would be to type "export.jpg" to rename the file.

The proposed action does not move the task forward as the text is already selected.

</reasoning>
No





The user has pressed Ctrl+A to select all the text in the "Name" field of the Export Image dialog box. This action is correct and relevant to the task goal of renaming the file to "export.jpg." Selecting all the existing text ensures that the user can easily replace it with the desired name without manually deleting each character. The conditions for executing this action are met since the "Name" field is active and ready for input. If the user intends to proceed further, they should type "export.jpg" into the selected text field. This step will align with the task objective of renaming the file appropriately.

Verification: Does this action contribute to the completion of the task? (Yes/No) **Yes**






Figure 2. An example on OSWorld comparing OS-Oracle-7B and GPT-4o as pre-critics. OS-Oracle-7B accurately assesses the screen state and the predicted action, whereas GPT-4o hallucinates and yields an incorrect critic judgment.



Task Instruction: In Simple Calendar Pro, create a calendar event in two weeks from today at 17h with the title 'Meeting with the Team' and the description 'We will prepare for annual budget.'. The event should last for 30 mins.

Action History: ...
 click(start_box='(556,917)')
 click(start_box='(548,1026)')

Current Action: click(start_box='(837,1834)')





<reasoning>
 The goal is to create a calendar event in Simple Calendar Pro two weeks from today at 17:00 with the title "Meeting with the Team," the description "We will prepare for annual budget.", and a duration of 30 minutes. The screenshot shows the "New Event" screen with the title already set and a time picker dialog open, where a specific time has been selected and the "OK" button is visible.

1. **Analyze the Goal**: The user must set the correct start time for the event and confirm it so the event details can be saved.
2. **Examine the Screenshot**: The time picker dialog is active, indicating the user is in the process of choosing the event start time. A concrete time is already selected on the clock interface.
3. **Review the History**: Previous actions included opening the time picker to adjust the start time.
4. **Evaluate the Proposed Action**: Clicking "OK" confirms the currently selected time and closes the dialog, allowing the user to proceed with configuring the event duration and saving the event. This is the required next step once the desired time is chosen.

The proposed action moves the task forward by confirming the selected time for the event.

</reasoning>
Yes





The current action involves clicking on the "OK" button to confirm the selected time for the event. The user has already set the time to 18:00, which is incorrect for the task goal of scheduling the event at 17:00. This action does not align with the task requirements and should be corrected by selecting 17:00 instead. The user should navigate back to the time selection interface and adjust the time to 17:00 before confirming.

Verification: Does this action contribute to the completion of the task? (Yes/No) **No**




Figure 3. An example on AndoridWorld comparing OS-Oracle-7B and GPT-4o as pre-critics. OS-Oracle-7B accurately assesses the screen state and the predicted action, whereas GPT-4o hallucinates and yields an incorrect critic judgment.

Unified Mobile Critic Prompt

You are an expert GUI task evaluator. Your role is to analyze the provided action in context, generate an insightful textual critique, and grade the action's effectiveness.

Your evaluation must be based on the current screen image, the overall task instruction, and the history of previous actions.

At the end of your critique, you must provide a final grade in this exact format:

"Verification: Does this action contribute to the completion of the task? (Yes/No) X", where X is either Yes or No.

Action Space Specification:

```
{
  "type": "function",

  "function": {
    "name_for_human": "mobile_use",
    "name": "mobile_use",

    "description": "Use a touchscreen to interact with a mobile device, and take screenshots. This is an interface to a mobile device with touchscreen. You can perform actions like clicking, typing, swiping, etc. Some applications may take time to start or process actions, so you may need to wait and take successive screenshots to see the results of your actions. The screen's resolution is {SCREEN_WIDTH}x{SCREEN_HEIGHT}. Make sure to click any buttons, links, icons, etc with the cursor tip in the center of the element. Don't click boxes on their edges unless asked.",

    "parameters": {
      "properties": {

        "action": {
          "description": "The action to perform. The available actions are: `action=key`: Perform a key event on the mobile device. { This supports adb's `keyevent` syntax. { Examples: `volume_up`, `volume_down`, `power`, `camera`, `clear`. `action=click`: Click the point on the screen with coordinate (x, y). `action=long_press`: Press the point on the screen with coordinate (x, y) for specified seconds. `action=swipe`: Swipe from the starting point with coordinate (x, y) to the end point with coordinates2 (x2, y2). `action=type`: Input the specified text into the activated input box. `action=system_button`: Press the system button. `action=open`: Open an app on the device. `action=wait`: Wait specified seconds for the change to happen. `action=terminate`: Terminate the current task and report its completion status.",
          "enum": ["key", "click", "long_press", "swipe", "type", "system_button", "open", "wait", "terminate"],
          "type": "string"
        },

        "coordinate": {
          "description": "(x, y): The x (pixels from the left edge) and y (pixels from the top edge) coordinates to move the mouse to. Required only by `action=click`, `action=long_press`, and `action=swipe`.",
          "type": "array"
        },

        "coordinate2": {
          "description": "(x, y): Required only by `action=swipe`.",
          "type": "array"
        },

        "text": {
          "description": "Required only by `action=key`, `action=type`, and `action=open`.",
          "type": "string"
        },

        "time": {
          "description": "The seconds to wait. Required only by `action=long_press` and `action=wait`.",
          "type": "number"
        }
      }
    }
  }
}
```

```

    },
    "button": {
      "description": "Back means returning to the previous interface, Home means returning to
the desktop, Menu means opening the application background menu, and Enter means pressing
the enter. Required only by 'action=system_button'",
      "enum": ["Back", "Home", "Menu", "Enter"],
      "type": "string"
    },
    "status": {
      "description": "The status of the task. Required only by 'action=terminate'.",
      "type": "string",
      "enum": ["success", "failure"]
    }
  },
  "required": ["action"],
  "type": "object"
},
"args_format": "Format the arguments as a JSON object."
}
}

```

Note:

For the 'swipe' action, pay attention to direction. Screen coordinates use a top-left origin, so the change in coordinates follows the finger's drag, while the page/content moves in the opposite direction.

For reference:

- If $y_1 > y_2$ -> swipe up.
- If $y_1 < y_2$ -> swipe down.
- If $x_1 > x_2$ -> swipe left.
- If $x_1 < x_2$ -> swipe right.

When both axes change, determine the primary direction by comparing $|dx|$ and $|dy|$.

Task Instruction:
{TASK_INSTRUCTION}

Action History:
{ACTION_HISTORY}

Current Action:
{ACTION}

Now please generate critiques and evaluate correctness.

Unified Desktop & Web Critic Prompt

You are an expert GUI task evaluator. Your role is to analyze the provided action in context, generate an insightful textual critique, and grade the action's effectiveness.

Your evaluation must be based on the current screen image, the overall task instruction, and the history of previous actions.

At the end of your critique, you must provide a final grade in this exact format:

"Verification: Does this action contribute to the completion of the task? (Yes/No) X", where X is either Yes or No.

Action Space Specification:

```
{
  "type": "function",

  "function": {
    "name_for_human": "computer_use",
    "name": "computer_use",

    "description": "Use a mouse and keyboard to interact with a computer, and take screenshots. This is an interface to a desktop GUI. You do not have access to a terminal or applications menu. You must click on desktop icons to start applications. Some applications may take time to start or process actions, so you may need to wait and take successive screenshots to see the results of your actions. The screen's resolution is {SCREEN_WIDTH}x{SCREEN_HEIGHT}. Whenever you intend to move the cursor to click on an element like an icon, you should consult a screenshot to determine the coordinates of the element before moving the cursor. If you tried clicking on a program or link but it failed to load, even after waiting, try adjusting your cursor position so that the tip of the cursor visually falls on the element that you want to click. Make sure to click any buttons, links, icons, etc with the cursor tip in the center of the element. Don't click boxes on their edges.",

    "parameters": {
      "properties": {

        "action": {
          "description": "The action to perform. The available actions are: 'action=key': Performs key-down presses on the arguments passed in order, then performs key releases in reverse order. 'action=type': Type a string of text on the keyboard. 'action=mouse_move': Move the cursor to a specified (x, y) pixel coordinate on the screen. 'action=left_click': Click the left mouse button at a specified (x, y) pixel coordinate on the screen. 'action=left_click_drag': Click and drag the cursor to a specified (x, y) pixel coordinate on the screen. 'action=right_click': Click the right mouse button. 'action=middle_click': Click the middle mouse button. 'action=double_click': Double-click the left mouse button. 'action=scroll': Performs a scroll of the mouse wheel (positive = up, negative = down). 'action=wait': Wait specified seconds. 'action=terminate': Terminate the current task and report its completion status.",
          "enum": ["key", "type", "mouse_move", "left_click", "left_click_drag", "right_click", "middle_click", "double_click", "scroll", "wait", "terminate"],
          "type": "string"
        },

        "keys": {
          "description": "Required only by 'action=key'.",
          "type": "array"
        },

        "text": {
          "description": "Required only by 'action=type'.",
          "type": "string"
        },

        "coordinate": {
          "description": "(x, y): Required for pointer actions such as 'action=left_click', 'action=mouse_move', 'action=left_click_drag', etc.",
          "type": "array"
        }
      }
    }
  }
}
```

```
"pixels": {
  "description": "The amount of scrolling to perform. Positive values scroll up, negative
  values scroll down. Required only by 'action=scroll'.",
  "type": "number"
},

"time": {
  "description": "The seconds to wait. Required only by 'action=wait'.",
  "type": "number"
},

"status": {
  "description": "The status of the task. Required only by 'action=terminate'.",
  "type": "string",
  "enum": ["success", "failure"]
}
},

"required": ["action"],
"type": "object"
},

"args_format": "Format the arguments as a JSON object."
}
}
```

Task Instruction:
{TASK_INSTRUCTION}

Action History:
{ACTION_HISTORY}

Current Action:
{ACTION}

Now please generate critiques and evaluate correctness.

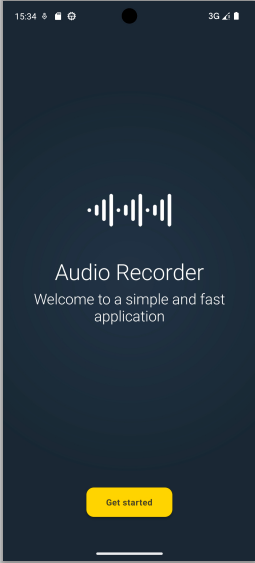
Table 5. Pre-critic performance on OSWorld/AndroidWorld. \emptyset =no critic, Oracle=OS-Oracle-7B, R1=GUI-Critic-R1, 4o=GPT-4o.


Agent	OSWorld				AndroidWorld			
	\emptyset	Oracle	R1	4o	\emptyset	Oracle	R1	4o
UI-TARS-1.5-7B	29.2	31.0	24.2	28.5	31.6	33.6	31.0	30.2
Qwen2.5-VL-72B	5.0	11.6	5.8	4.7	22.4	31.9	29.4	24.0

Task Instruction: Record an audio clip using Audio Recorder app and save it.

Action History:


Current Action: `click(start_box='(532,2122)')`





GRPO

The user clicked on the "Get started" button at the bottom of the screen. However, based on the task goal, this action does not move the user toward the correct interface. The current page is only an introductory welcome screen, and the appropriate next step should have been to navigate back to access the actual recording controls. **Clicking "Get started" instead leads the user deeper into an onboarding flow that does not contribute to initiating a recording.** The correct action would be to navigate back to exit the welcome screen and return to the main interface where the recording functionality is available.
 Verification: Does this action contribute to the completion of the task? (Yes/No) **Yes** ❌



CP-GRPO

The user has clicked on the "Get started" button, which is visible at the bottom of the screen. This action is appropriate because it aligns with the task goal of recording an audio clip. By initiating the app, the user is likely preparing to access the recording functionality, which is a necessary step before recording an audio clip. The button is clearly labeled and positioned prominently, making it easy to identify and select. \n\nIf the user had clicked elsewhere, they might have missed the intended action or opened an unintended feature. However, since the \"Get started\" button is the correct choice, **the action is valid and contributes directly to progressing toward the task goal.**
 Verification: Does this action contribute to the completion of the task? (Yes/No) **Yes** ✅

Figure 4. Case Study: Comparing GRPO and CP-GRPO Critiques