Formatting Instructions For NeurIPS 2024

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Computer-Using Agents (CUAs) are designed to interact with graphical user interfaces (GUIs) in a human-like manner, capable of opening applications, executing command-line instructions, and performing diverse tasks. Despite the advanced task-parsing capabilities of the underlying Large Language Models (LLMs), existing CUAs exhibit significant limitations in GUI grounding. This gap often stems from difficulties in translating the LLM's latent understanding of a task into precise, actionable outputs. Currently, most CUA models rely on pre-trained LLMs that directly output numerical coordinates for clicks or actions. We hypothesize this is suboptimal, as LLMs may lack the fine-grained numerical grounding required for precise coordinate generation.

To address this limitation, we propose an alternative approach. Instead of regressing coordinates, we adapt an action expert based on diffusion, similar to those used in Vision-Language-Action (VLA) models. We posit that a diffusion-based action head can more effectively translate the LLM's innate task comprehension into robust GUI interactions, bypassing the challenges of direct coordinate output. In this paper, we test this hypothesis by training and evaluating an LLM equipped with this action head on foundational computer interaction tasks: clicking GUI elements and inputting text.

1 Introduction

Computer-Use Agents (CUAs) typically implement tasks in two stages:

1. **High-level planning:** The CUA must understand the overall objective from a given prompt and decompose it into a sequence of individual steps.
2. **Action generation:** After outlining these general steps, the CUA must translate them into concrete, executable actions.

For example, during action generation, a high-level step like "Open application xyz" must be converted into a low-level action string, such as `click(x, y)` or `type("message")`.

A key failure point for GUI agents is this final action generation step, often referred to as "GUI grounding". While Vision-Language Models (VLMs) can demonstrate strong latent grounding by internally attending to the correct GUI element, they frequently fail to translate this internal understanding into precise, executable actions, such as `click(x, y)` with correct `x` and `y` coordinates How.

Current state-of-the-art (SOTA) approaches for improving GUI grounding abilities primarily fall into two categories:

1. **Visual Input Augmentation:** Modifying the input screenshot by drawing auxiliary markers, such as axes or grids, to enhance spatial reasoning and grounding How, Ziyang et al. [2024].

2. **Data Scaling:** Training agents on larger and higher-quality datasets of trajectories to improve generalization and robustness Gonzalez-Pumariega et al. [2025], Wang et al. [2025].

However, despite these advances, existing methods, to our knowledge, still rely on autoregressive text token generation to produce actions.

This approach presents several critical issues:

1. **Invalid Action Formulation:** Token-level generation can produce syntactically invalid or nonsensical actions that the execution environment cannot parse. Furthermore, models may hallucinate coordinates outside the screen's bounds (e.g., outputting `click(401, 200)` when the screen width is only 400).
2. **Poor Numerical and Spatial Understanding:** Action generation via text relies on the model's numeracy, which is often a weakness. This is critical for GUI tasks that require precise spatial and numerical reasoning (e.g., adapting to different screen resolutions or understanding that an element "above" another must have a smaller y-coordinate).

Recent advances in **Vision-Language-Action (VLA)** models have demonstrated the effectiveness of introducing dedicated *action heads* for downstream control. In such systems, the core multimodal encoder processes visual and textual context to form a latent representation, while the action head specializes in translating this latent intent into structured, low-level actions. This separation allows VLAs to maintain semantic reasoning in the backbone while achieving precise motor or spatial control through the action-specific module while improving both training stability and generalization to unseen environments Li et al. [2024].

Inspired by these developments, we propose extending the same principle to **Computer-Use Agents (CUAs)**. Specifically, we introduce an explicit *Action Head* to decouple high-level reasoning from low-level GUI execution. Instead of relying on autoregressive token generation, the Action Head directly maps multimodal latent features to executable actions.

Formally, given a latent representation $\mathbf{h} \in \mathbb{R}^d$ from the CUA backbone (e.g., a Vision–Language Transformer), the Action Head learns a parameterized mapping

$$\pi_\theta : \mathbb{R}^d \to \mathcal{A}, \tag{1}$$

where $\mathcal{A}$ denotes a continuous or structured action space (e.g., 2D coordinates, keypress distributions, or function signatures). Unlike autoregressive text decoders, π_θ operates in a continuous domain, enabling direct gradient-based optimization for spatial precision and constraint enforcement (e.g., bounding-box or screen-size clipping).

The goal of this paper is to improve GUI grounding by bridging the gap between the VLM's internal latent representation and its action output. In other words, we aim to ensure the VLM's output actions directly correspond to its internal spatial understanding of the screenshot.

Key Contributions

1. We adapt and train a dedicated action head specifically for core GUI interaction tasks. Our work focuses on the most fundamental GUI actions: left and right mouse clicks, and typing.

2 Design

Training We fine-tune an existing pre-trained model in two stages:

1. GUI grounding pre-training on the OSAtlas dataset, which contains over 2.3 million screenshots Wu et al. [2024].
2. An online reinforcement learning phase, using the Agent Lightning framework for agent management and rollouts Luo et al. [2025].

Baseline Model We select Holo1.5 3b as the backbone for our training. Holo1.5 3b is the current open source model that performs the best on GUI grounding tasks while maintaining a reasonably small parameter size compared to SOTA models. For our training purposes we freeze Holo1.5 3b during GUI grounding and only adjust the parameters of our actions head.

,

Table 1: Model Performance Comparison

Model Name	Param. Size	Type	Performance	Additional Info
DeepMiner-Mano-7B	7B	Specialized	- Osworld: 40.1%	
Seed1.5-VL		General		
Mobile-Agent-v3		Specialized		
GUI-owl	7b	Specialized	- Osworld: 23.1%	
uitars-1.5-7b	7b	Specialized	- Osworld: 27.5 $\pm$ 2.2%	
GUI-ARP 7b	7b	Specialized	- Screenspot-Pro: 91.8%	
			- Screenspot-pro: 60.8%	
UI-Venus 7b	7b	Specialized	- Screenspot-v2: 94.1%	Built on QWen 2.5 VL 7b
			- Screenspot-pro: 50.8%	
Holo1.5 7b	7b	Specialized	- Screenspot-v2: 93.3%	Built on Qwen
			- Screenspot-pro: 57.9%	Holo1.5 are natively built on high-res
Holo 1.5 3b	3b	Specialized	- Screenspot-v2: 91.7%	Built on Qwen
			- Screenspot-pro: 51.5%	

Action Head

Benchmarking We evaluate our model on three benchmarks. The first two focus on GUI grounding, while the last evaluates real-world performance.

1. ScreenSpot-V2: a benchmark for single-step grounding abilities across environments (mobile, desktop, etc.) Wu et al. [2024], where top models achieve 95% accuracy.
2. ScreenSpot-Pro: a high-resolution benchmark with 23 images across 3 operating systems Li et al. [2025], where top models achieve 65% accuracy.
3. OSWorld: an online environment for real-world evaluation across various operating systems Xie et al. [2024], where top models achieve 63% accuracy.

We adopt the evaluation methodology of Gou et al. [2025], using two settings:

1. Grounding setting: A planner model decomposes high-level instructions into simpler subtasks, which are fed to our model.
2. Standalone setting: Our model executes instructions directly, without a planner.

TODOs

1. Formularize inputs and outputs of architecture -> Get training data afterwards
2. Understand what action tokens VLMs output
3. Find out how to interpret continuous tokens outputted by -> absolute or relative
4. find out how we get the latent embeddings of VLMs

Future Improvements

1. Trajectory Selection: Generate higher quality data Gonzalez-Pumariega et al. [2025]

3 Evaluation

4 Related Works

5 Conclusion

References

How Auxiliary Reasoning Unleashes GUI Grounding in VLMs. https://arxiv.org/html/2509.11548v1/.

Gonzalo Gonzalez-Pumariega, Vincent Tu, Chih-Lun Lee, Jiachen Yang, Ang Li, and Xin Eric Wang. The Unreasonable Effectiveness of Scaling Agents for Computer Use, October 2025.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the Digital World as Humans Do: Universal Visual Grounding for GUI Agents, June 2025.

Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. ScreenSpot-Pro: GUI Grounding for Professional High-Resolution Computer Use, April 2025.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation, 2024. URL `https://arxiv.org/abs/2411.19650`.

Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K. Qiu, and Yuqing Yang. Agent lightning: Train any ai agents with reinforcement learning, 2025. URL `https://arxiv.org/abs/2508.03680`.

Haoming Wang, Haoyang Zou, Huatong Song, Jiazhan Feng, Junjie Fang, Junting Lu, Longxiang Liu, Qinyu Luo, Shihao Liang, Shijue Huang, Wanjun Zhong, Yining Ye, Yujia Qin, Yuwen Xiong, Yuxin Song, Zhiyong Wu, Aoyan Li, Bo Li, Chen Dun, Chong Liu, Daoguang Zan, Fuxing Leng, Hanbin Wang, Hao Yu, Haobin Chen, Hongyi Guo, Jing Su, Jingjia Huang, Kai Shen, Kaiyu Shi, Lin Yan, Peiyao Zhao, Pengfei Liu, Qinghao Ye, Renjie Zheng, Shulin Xin, Wayne Xin Zhao, Wen Heng, Wenhao Huang, Wenqian Wang, Xiaobo Qin, Yi Lin, Youbin Wu, Zehui Chen, Zihao Wang, Baoquan Zhong, Xinchun Zhang, Xujing Li, Yuanfan Li, Zhongkai Zhao, Chengquan Jiang, Faming Wu, Haotian Zhou, Jinlin Pang, Li Han, Qi Liu, Qianli Ma, Siyao Liu, Songhua Cai, Wenqi Fu, Xin Liu, Yaohui Wang, Zhi Zhang, Bo Zhou, Guoliang Li, Jiajun Shi, Jiale Yang, Jie Tang, Li Li, Qihua Han, Taoran Lu, Woyu Lin, Xiaokang Tong, Xinyao Li, Yichi Zhang, Yu Miao, Zhengxuan Jiang, Zili Li, Ziyuan Zhao, Chenxin Li, Dehua Ma, Feng Lin, Ge Zhang, Haihua Yang, Hangyu Guo, Hongda Zhu, Jiaheng Liu, Junda Du, Kai Cai, Kuanye Li, Lichen Yuan, Meilan Han, Minchao Wang, Shuyue Guo, Tianhao Cheng, Xiaobo Ma, Xiaojun Xiao, Xiaolong Huang, Xinjie Chen, Yidi Du, Yilin Chen, Yiwen Wang, Zhaojian Li, Zhenzhu Yang, Zhiyuan Zeng, Chaolin Jin, Chen Li, Hao Chen, Haoli Chen, Jian Chen, Qinghao Zhao, and Guang Shi. UI-TARS-2 Technical Report: Advancing GUI Agent with Multi-Turn Reinforcement Learning, September 2025.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. OS-ATLAS: A Foundation Action Model for Generalist GUI Agents, October 2024.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments, 2024.

Meng Ziyang, Yu Dai, Zezheng Gong, Shaoxiong Guo, Minglong Tang, and Tongquan Wei. VGA: Vision GUI Assistant - Minimizing Hallucinations through Image-Centric Fine-Tuning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1261–1279, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.68.

A Appendix / supplemental material

Optionally include supplemental material (complete proofs, additional experiments and plots) in appendix. All such materials **SHOULD be included in the main submission.**