Interspeech 2025
17-21 August 2025, Rotterdam, The Netherlands

TinyClick: Single-Turn Agent for Empowering GUI Automation

Pawel Pawlowski[*1], *Krystian Zawistowski*[*1], *Wojciech Lapacz*[1], *Adam Wiacek*[1], *Marcin Skorupa*[1], *Sebastien Postansque*[1], *Jakub Hoscilowicz*[1,2]

[1]Samsung R&D, Poland
[2]Warsaw University of Technology, Poland

Abstract

We present an UI agent for user interface (UI) interaction tasks, using Vision-Language Model Florence-2-Base. The agent's primary task is identifying the screen coordinates of the UI element corresponding to the user's command. It demonstrates very strong performance on Screenspot and OmniAct annotations, while maintaining a very small size of 0.27B parameters and minimal latency. Moreover, training needs small compute budget of 56 GPU-hours (worth about 40 USD). Relevant improvement comes from vision-specific multi-task training and MLLM-based data augmentation. We hope that decreased needs for expensive compute resources and manually annotated data will allow to facilitate more inclusive and sustainable research of UI agents.

Index Terms: UI agent, human-computer interaction, on-device models

1. Introduction

In voice assistants, we see a shift towards MLLM-based solutions. NLU and rule-based commands are replaced with agents, that can interact with the system in similar way as the user. Single-turn agents perform actions specified by the user in the UI environment, e.g. clicking on appropriate icons. Examples of such commands are "choose first item in product list", "close the application" or "open cart". The agent can perform actions based on such commands and the current GUI screenshot captured from the user's device. Currently, UI agents demonstrate moderate accuracy [1, 2], despite significant computational cost.

We present single-turn agent based on Florence-2 model[1] [4], which offers state-of-the-art performance for on-device system, while having only 0.27B of parameters. When finetuning the model we investigated multiple approaches, public datasets and data preparation techniques. High-level diagrams are provided in Figure 1 (agent inference) and Figure 2 (agent training). Most important improvements come from multitask training. It uses multiple purpose-oriented objectives to reinforce more efficient UI representation in the model. Our key contributions are as follows:

- TinyClick performs similarly to a family of much larger MLLM solutions, achieving 73.8% accuracy on Screenspot and 58.3% on OmniAct-Annotations.
- Training on vision-specific multitask data (such as generate captions and explanations) outperforms training on click-commands alone.

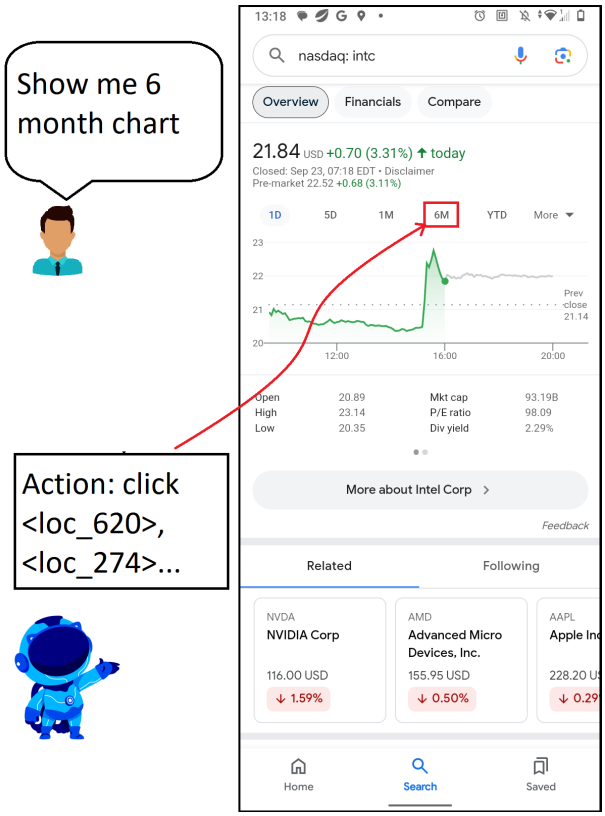


Figure 1: *Example command of the downstream task. TinyClick receives a screenshot and user command and predicts bounding box of the UI element. Stickers from Flaticon [3].*

- MLLM-based augmentation of corpora with commands and boxes significantly improves model performance.

Our results suggest that extensive visual pretraining (as performed by Florence-2 authors) provides strong foundation for single-turn agent model, even if performed on the data not related to GUI and agents.

2. Related Works

In recent years there is much interest in Vision-Language Models (VLMs) and Multimodal Large Language Models capable of UI understanding [1, 2, 5, 6].

Our focus are UI agents navigating UIs according to natural language commands of the user: such as AppAgent [7], MobileAgent [8], CogAgent [9], Auto-UI [10], V-Zen [1]. A

Correspondence to <k.zawistowsk@samsung.com> and <jakub.hoscilowicz.dokt@pw.edu.pl>
[*]First and second author contributed equally.
[1]huggingface.co/microsoft/Florence-2-base

10.21437/Interspeech.2025-176

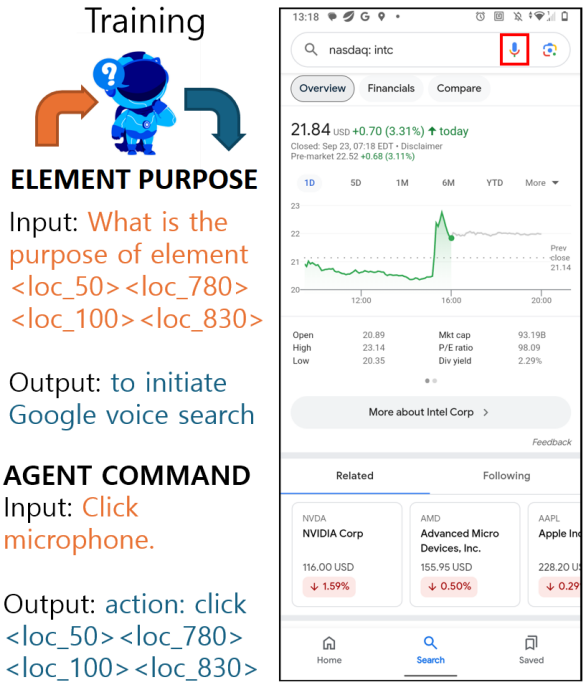


Figure 2: *During the training, the model receives a question and generates an answer. Both question and answer can contain location tokens of the specific UI element. Here, the first question is about element description and the second one is a command to click specific item. Multiple different tasks can be associated with a single UI element, allowing the model to gain a better understanding of the UI. Stickers from Flaticon [3].*

related problem is grounding, which involves finding UI elements corresponding to a given phrase (Ferret [11], Ferret-v2 [12]). Also, language-only LLM are used as agents for the Internet, using primarily HTML representation [13, 14]. Pix2Struct [15] was pretrained to reproduce simplified HTML representation of screenshot (including that for masked parts of the UI). SeeClick [16], one of best agent models so far, uses HTML data for grounding-style pretraining. Among multitask training approaches [17] utilizes 10 pre-training tasks that resemble real-world tasks. Tree-of-Lens was proposed to interpret screen content based on a user-indicated point [18].

Simultaneously with or after the initial version of this paper released in October, other models in the paradigm of SeeClick and CogAgent emerged, featuring typically large (7B or bigger) MLLM trained on UI grounding task, which produced similar or better accuracy compared to TinyClick, but with vastly larger costs. This includes [19, 20, 21]. Best one, AGUVIS, uses Qwen2-VL as backbone model [22], which was already extensively trained on agentic use cases and rivals SeeClick in accuracy.

3. Method

In this work, we use Florence-2 Base [4], a 0.27B vision transformer with language modelling head trained for different vision tasks. The model latency is approximately 250 ms, allowing for cheaper inference as well. Vision encoder uses large 768x768 image resolution, which might be important for large screenshots.

	Size	Screenspot	OmniAct-Annot.
AutoUI-Base	0.41B	1.97	1.25
Qwen2-VL[19]	9.6B	55.0†	-
GPT-4o[20]	-	18.2†	-
SeeClick	9.6B	55.1†	36.8
PaliGemma-WaveUI[26]	3B	53.9	52.3
Florence-2-base	0.27B	4.6	-
TinyClick (ours)	0.27B	73.8	58.3
w/o multi-task FT		69.4	50.4
WaveUI-only		64	58

Table 1: *TinyClick accuracy results. Goal here is choosing location to click based on user command (point prediction). The metric presented in this table is accuracy as presented in Section 4.3. All values are in percentage (%). Cited results are marked by †. Screenspot results were recalculated as mean of accuracies for mobile, desktop and web. 0.41B for AutoUI includes vision model from Blip2.*

While Florence-2 uses text transformer, it is designed to handle detection and grounding tasks, encoding coordinates as single tokens.

3.1. Multitask Training

Performance of transformer models can be improved by training on many related tasks. This is true of LLM [23], Florence-2 [4]), and models trained on UI [17, 24, 15]. [25, 24] also demonstrate training small model to predict natural language explanations of training data. Florence-2 is pretrained on object recognition, phrase grounding, captioning, segmentation and similar vision tasks. It was also trained for OCR, but not on UI data. To adapt Florence-2 for single-turn agent we investigated multitask training for UI, using tasks such as:

- Element captioning - generating descriptions or purposes or action expectations of UI elements based on their location on the screen.
- Element location - locating UI elements based on their visual description.
- Object detection - detection of all clickable UI elements.
- Agent action - locating an UI element to click or point to click based on user command.
- Question answering, based on screen content.

We used publicly available corpora for single-turn agent training, consisting mainly of commands and locations (bounding boxes). To prepare our training data, we used available MLLM annotations or software-based metadata and also re-annotated the data with own MLLM pipeline. Element captioning, expectation, location and purpose were mainly based on MLLM annotations, while for object detection we used Android XML UI metadata. This approach is not identical to [25, 24, 19, 9], or anything that one would use with MLLM for symbolic or common sense reasoning. We do not focus on Chain-of-Though data, as the Florence2 model has no pretraining of this sort. We instead focus on augmentation of visual tasks, similarly to original Florence-2 training approach [4]

4. Experiments

4.1. Training Datasets

We use following public datasets: WaveUI [27] (we do not use WebUI due to license issue and remove Screenspot and

		CogAgent[9]	SeeClick[16]	*TinyClick*	AGUVIS-G[19]	UGround
Size [B]		18	9.6	*0.27*	9.6	7
Mobile	Text	67	78	*86.8*	88.3	82.8
	Icon	24	52	*62*	78.2	60.3
Desktop	Text	74.2	72.2	*90.2*	88.1	82.5
	Icon	20	30	*54.2*	70.7	63.6
Web	Text	70.4	55.7	*80.9*	85.7	80.4
	Icon	28.6	32.5	*56.3*	74.8	70.4
Average		47.4	53.4	*71.7*	81.8	73.3

Table 2: *Accuracy results on ScreenSpot. other results than TinyClick reported by [16, 19].*

Dataset	Rows	Use	Type	huggingface.com link
AMEX OD	98.5k	30%	object detection (metadata)	Yuxiang007/AMEX
AMEX Functionality	296k	50%	commands	Yuxiang007/AMEX
AMEX Purpose	296k	50%	multitask (MLLM)	-
AMEX Expectation	28.9k	100%	multitask (MLLM)	-
Wave UI	44.2k	100%	commands	agentsea/wave-ui
Wave UI expectation/caption/purpose	3x44.2k	100%	multitask (MLLM)	agentsea/wave-ui
GUI Course web-single	102.8k	100%	commands	yiye2023/GUIAct
GUI Course caption/expectation	2x51.4k	100%	multitask (MLLM)	-
RICO ScreenQA	56.6k	100%	visual question answering	rootsautomation/RICO-ScreenQA
Android Control	51.9k	100%	commands	GCloud, see p. 3

Table 3: *Trainset composition for best training configuration (Table 1).*

OmniAct annotations, as these are benchmarks), AMEX [28], Mind2Web [13], GUI Odyssey [29] (it is not included in our final training), GUI Course [30], AndroidControl [2] [31], ScreenQA [32]. For WaveUI we use commands (for agent action task) as well as provided MLLM-generated expectation (expected outcome), purpose and captions of UI elements. We use similar approach to WaveUI to augment AMEX and GUI Course, where commands or functionalities of UI elements are provided with UI elements location. We use InternVL2-26B [33] to annotate these data with purposes, captions, or expectations. For AMEX, we use the 'functionality' field (manually annotated purpose of UI element) as well as Android XML annotations. Information about our training set that produced results in Table 1 can be found in Table 3, showing that training uses 845k rows in total.

4.2. Benchmarks

We use two benchmarks: Screenspot[3] [16] and OmniAct[4] annotations [34]. The first one contains 1200 test cases divided into 3 groups: mobile, web, and desktop. For Omniact, we do not use multi-turn commands it provides, but instead use about 3400 box-annotations with icon description provided with the dataset.

4.3. Results Analysis

The accuracy shown in Table 1 is calculated as the average of binary outcomes, whether predicted click point (or bounding box center) falls within the original ground-truth bounding box (1 if it does, 0 otherwise). For Screenspot, we report the arithmetic mean accuracy achieved on 3 data subgroups (see Chapter 4.2), according to SeeClick publication [16].

The results show strong performance improvement over other approaches, such as SeeClick [16], AutoUI [10], and other MLLMs (see Table 1). The results are also stronger than publicly available PaliGemma-WaveUI-3B-896[26], despite the fact, that this larger model has been trained on very similar multitask dataset, with some possible contamination with our testsets. This is also true with TinyClick trained on WaveUI alone, which beats Paligemma by 10% on Screenspot. Our own, further experiments with finetuning Paligemma did not produce results, as only LoRa with small batch size was operational on 80GB A100 GPU and compute budgets needed are much larger, that those needed for TinyClick. This suggests that Florence2 pretraining on grounding and detection is important, facilitating good results in a much smaller model and compute resources.

While Florence2 architecture is similar to that of MLLM, it differs in size and in pre-training. It has very small language modeling part, that lacks textual reasoning training (except for simple grounding phrases), but it is trained on mixture of visual task. As we follow Florence-2 approach, similar finetuning strategy adapted to GUI allows to produce our result.

4.4. Methodology

Large size and real world significance of the observed effect facilitates our belief in significance and credibility of the findings, especially when using much different pretrained model, than previous solutions. This follows Gosset-Neyman-Pearson methodology discussed by [35] and [36], which advise prioritizing strong, dominant effects. In complex systems it is easy to find many statistically significant effects, but that rarely correlates with stability and real-world significance. We, on the other hand, saw a very high gain in accuracy, with much-decreased model size and compute budget, compared to other multimodal transformers used before. Secondly, we corroborate our claims by performing balanced experiments. Thirdly, we verified that the claims presented are supported by statistically significant

[2]Google Cloud link: `gresearch/android_control`

[3]`huggingface.co/datasets/rootsautomation/ScreenSpot`

[4]`huggingface.co/datasets/Writer/omniact`

results. Smallest significant (2σ) difference in accuracy is 1.8% for OmniAct and 2.6% for Screenspot (calculated from Beta conjugate prior). This also accounts for the observed variance of repeated experiments (about 1% for Screenspot). Results below the significance threshold are not considered conclusive. We used 3 epochs as initial setting and tested training up to 5 epochs, which did not produce statistically significant improvement. We trained models on 4xA100 GPU (except Table 5, which used 1xA100 and TinyClick-WaveUI that used 1xA100 with 4 epochs). Training the best checkpoint required about 56 GPU-hours (18h on 4xA100). The best checkpoint result in Table 1 is the average accuracy for 3 training runs. Other results were calculated for single training run.

4.5. Dataset insights

We performed an ablation study, removing the numerically largest AMEX, multitask data and MLLM multitask data (see Table 6). Multitask data seems more important than larger amount of commands, and MLLM data more important than metadata. AMEX produces only small gain, despite large number of examples. Some of multitask data (as WaveUI's) can be used either as grounding objectives (generate location given phrase) and annotation objectives (generate phrase given location). The latter appears to provide stronger performance gain. To corroborate this, we used GUI Course Web single-action commands and annotated each example with expectation and caption using InternVL2-26B MLLM (method analogous to WaveUI). For training with 51k commands, 6k MLLM generated annotations of each type improved result (see Table 5), but only for annotation objective.

Similar ablation was performed on AMEX with metadata multitask (based on Android XML annotations) compared to MLLM multitask (generated by us) and 50% of commands (see Table 4). All multitask approaches yield 59% on Screenspot, but MLLM multitask is better on OmniAct, outperforming software-based metadata. We also tested MLLM-based labeling of screenshots without manual supervision; this however was much weaker than smaller high quality corpora on present benchmarks.

4.6. Fail analysis

30% of failed examples suggest spurious signals that were taught to produce right answer for wrong reasons. One such example is positional bias: model clicks on the left and the top of the screen, where various menus are often found. Other is misinterpretations of similar icons. 20% of failed examples are "missed" clicks: model clicks very closely to correct button, but slightly off-mark.

	Screenspot	Omniact
AMEX alone, 50% commands	51.4	19.6
+ *metadata and MLLM multitask*	<u>59.6</u>	<u>25.0</u>
+ *metadata multitask*	59.2	25.0
+ *MLLM multitask*	**59.7**	**31.2**
+ *50% commands*	47.6	14.0

Table 4: *Annotation and multitask training on AMEX.*

5. Conclusions

TinyClick model strongly improves present baselines, demonstrating new approach to agentic models, based on small visual model and training procedure suited to training it. It is achieving 73.8% on Screenspot and 58.3% on OmniAct Annotations, while being much smaller than competing solutions and small enough to work with sub-second latency. Furthermore, we confirmed that augmenting manually annotated agent data with MLLM for multitask training improves the performance.

	Screenspot
TinyClick on 50% GUI Course web single	52.0
+ *6k annotations - text output*	**55.9**
+ *6k annotations - box output*	53.1
100% GUI Course web single	53.2

Table 5: *Annotation and multitask training on GUI Course web single turn data.*

	Screenspot
TinyClick	72.5
w/o AMEX	71.9
w/o multitask	**67.3**
w/o multitask except metadata	<u>68.1</u>

Table 6: *Training data ablation study of TinyClick model. Trainset like Table 3 but only 10% of GUI course and 10% of AMEX OD used.*

6. Future Research

The presented contribution suggests a few new research problems to investigate. TinyClick can be used as a component for multi-turn agent (such as [37]). MLLMs, thus far showing weak performance on UI control, might benefit from training strategy similar to ours. Florence-2, as a grounding and detection model, allows to adapt out-of-domain by simultaneous use of annotated images and natural language phrases. This might produce improvements in computer vision, especially with scarce visual data. The results suggest that cheap, automated MLLM augmentation is sufficient for better performance.

With this work, we seek to answer one significant concern: provide a baseline solution that is reasonably close to commercial viability and reasonably cheap, to facilitate more sustainable, inclusive, and democratized research of GUI agents.

Lots of ongoing research in the area deals with training 7B or larger multimodal transformer on GUI automation objective. Accuracies grow with compute resources and data. However, this often has very large costs and so far produces effects that have dubious real-world utility, where not only reliability is lacking but also privacy, latency, and inference cost remain unsolved. In result, a proof-of-concept field of technology, such as GUI agents currently are, generates excessive financial and environmental costs, that could be cut, perhaps, by an order of magnitude.

7. References

[1] A. Rahman *et al.*, "V-zen: Efficient gui understanding and precise grounding with a novel multimodal llm," 2024. [Online]. Available: https://arxiv.org/abs/2405.15341

[2] G. Baechler *et al.*, "Screenai: A vision-language model for ui and infographics understanding," 08 2024, pp. 3058–3068.

[3] Bukeicon and Flaticon.com, "Vector icons and stickers." [Online]. Available: www.flaticon.com

[4] B. Xiao *et al.*, "Florence-2: Advancing a unified representation for a variety of vision tasks," *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.

4818–4829, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:265128818

[5] I. Gur *et al.*, "Learning to navigate the web," *CoRR*, vol. abs/1812.09195, 2018. [Online]. Available: http://arxiv.org/abs/1812.09195

[6] G. Li and Y. Li, "Spotlight: Mobile UI understanding using vision-language models with a focus," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=9yE2xEj0BH7

[7] C. Zhang *et al.*, "Appagent: Multimodal agents as smartphone users," 2023. [Online]. Available: https://arxiv.org/abs/2312.13771

[8] J. Wang *et al.*, "Mobile-agent: Autonomous multi-modal mobile device agent with visual perception," 2024. [Online]. Available: https://arxiv.org/abs/2401.16158

[9] W. Hong *et al.*, "Cogagent: A visual language model for gui agents," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2024, pp. 14 281–14 290. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.01354

[10] Z. Zhang and A. Zhang, "You only look at screens: Multimodal chain-of-action agents," Bangkok, Thailand and virtual meeting, pp. 3132–3149, Aug. 2024. [Online]. Available: https://aclanthology.org/2024.findings-acl.186

[11] H. You *et al.*, "Ferret: Refer and ground anything anywhere at any granularity," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=2msbbX3ydD

[12] H. Zhang *et al.*, "Ferret-v2: An improved baseline for referring and grounding with large language models," in *First Conference on Language Modeling*, 2024. [Online]. Available: https://openreview.net/forum?id=EEPBOB2Xww

[13] X. Deng *et al.*, "Mind2web: towards a generalist agent for the web," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, ser. NIPS '23. Red Hook, NY, USA: Curran Associates Inc., 2024.

[14] I. Gur *et al.*, "A real-world webagent with planning, long context understanding, and program synthesis," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=9JQtrumvg8

[15] K. Lee *et al.*, "Pix2struct: screenshot parsing as pretraining for visual language understanding," in *Proceedings of the 40th International Conference on Machine Learning*, ser. ICML'23. JMLR.org, 2023.

[16] K. Cheng *et al.*, "Seeclick: Harnessing gui grounding for advanced visual gui agents," in *Annual Meeting of the Association for Computational Linguistics*, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:267069082

[17] Y. Gao *et al.*, "Enhancing vision-language pre-training with rich supervisions," *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13 480–13 491, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268253650

[18] Y. Fan *et al.*, "Read anywhere pointed: Layout-aware gui screen reading with tree-of-lens grounding," 2024. [Online]. Available: https://arxiv.org/abs/2406.19263

[19] Y. Xu, Z. Wang, J. Wang, D. Lu, T. Xie, A. Saha, D. Sahoo, T. Yu, and C. Xiong, "Aguvis: Unified pure vision agents for autonomous gui interaction," 2024. [Online]. Available: https://arxiv.org/abs/2412.04454

[20] B. Gou *et al.*, "Navigating the digital world as humans do: Universal visual grounding for gui agents," 2024. [Online]. Available: https://arxiv.org/abs/2410.05243

[21] M. Deitke *et al.*, "Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models," 2024. [Online]. Available: https://arxiv.org/abs/2409.17146

[22] P. Wang *et al.*, "Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution," 2024. [Online]. Available: https://arxiv.org/abs/2409.12191

[23] H. W. Chung *et al.*, "Scaling instruction-finetuned language models," 2022. [Online]. Available: https://arxiv.org/abs/2210.11416

[24] J. Zhang *et al.*, "Android in the zoo: Chain-of-action-thought for gui agents," 2024. [Online]. Available: https://arxiv.org/abs/2403.02713

[25] C.-Y. Hsieh *et al.*, "Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes," in *Findings of the Association for Computational Linguistics: ACL 2023*. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 8003–8017. [Online]. Available: https://aclanthology.org/2023.findings-acl.507

[26] AgentSea, "Paligemma waveui model." [Online]. Available: https://huggingface.co/agentsea/paligemma-3b-ft-waveui-896

[27] D. Jeffries and K. Team, "Wave ui dataset," 2024. [Online]. Available: https://huggingface.co/datasets/agentsea/wave-ui

[28] Y. Chai *et al.*, "Amex: Android multi-annotation expo dataset for mobile gui agents," 2024. [Online]. Available: https://arxiv.org/abs/2407.17490

[29] Q. Lu *et al.*, "Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices," 2024.

[30] W. Chen *et al.*, "Guicourse: From general vision language models to versatile gui agents," 2024. [Online]. Available: https://arxiv.org/abs/2406.11317

[31] W. Li *et al.*, "On the effects of data scale on computer control agents," 2024. [Online]. Available: https://arxiv.org/abs/2406.03679

[32] Y.-C. H. others, "Screenqa: Large-scale question-answer pairs over mobile app screenshots," 2024. [Online]. Available: https://arxiv.org/abs/2209.08199

[33] Z. Chen *et al.*, "Intern vl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks," *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24 185–24 198, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:266521410

[34] R. Kapoor *et al.*, "Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web," 2024. [Online]. Available: https://arxiv.org/abs/2402.17553

[35] J. P. A. Ioannidis, "Why most published research findings are false," *PLOS Medicine*, vol. 2, no. 8, p. null, 08 2005. [Online]. Available: https://doi.org/10.1371/journal.pmed.0020124

[36] S. T. Ziliak and D. N. McCloskey, *The Cult of Statistical Significance: How the Standard Error Costs Us Jobs, Justice, and Lives*. University of Michigan Press, 2008. [Online]. Available: http://www.jstor.org/stable/10.3998/mpub.186351

[37] J. Hoscilowicz *et al.*, "Clickagent: Enhancing ui location capabilities of autonomous agents," 2024. [Online]. Available: https://arxiv.org/abs/2410.11872