TinyClick: Single-Turn Agent for Empowering GUI Automation

Pawel Pawlowski*, **Krystian Zawistowski, Wojciech Lapacz, Adam Wiacek**
Marcin Skorupa, Sebastien Postansque, *Jakub Hoscilowicz
Samsung R&D Poland
*Warsaw University of Technology

Abstract

We present a single-turn agent for graphical user interface (GUI) interaction tasks, using Vision-Language Model Florence-2-Base. Main goal of the agent is to click on desired UI element based on the screenshot and user command. It demonstrates strong performance on Screenspot and OmniAct, while maintaining a compact size of 0.27B parameters and minimal latency. Main improvement comes from multitask training and MLLM-based data augmentation. Manually annotated corpora are scarce, but we show that MLLM augmentation might produce much better result. On Screenspot and OmniAct, our model outperforms both GUI-specific models (e.g., SeeClick) and MLLMs (e.g., GPT-4V). Model is available at hugging-face.co/Samsung/TinyClick.

1 Introduction

Vision-language models (VLMs) have many applications in working with GUIs (Nakano et al., 2022; Rawles et al., 2023). One of them is to perform actions specified by the user in the UI environment, e.g. clicking on appropriate icons or entering text into appropriate fields. Examples of such commands are "Choose first option 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. Moreover, as shown in Table 2 of (Liu et al., 2024), standard MLLMs exhibit poor performance in single-turn agent tasks (only 0-11% accuracy), despite high computational cost.

We present single-turn agent based on Florence-2 (Xiao et al., 2023) model, which offers state-of-the-art performance, while being only 0.27B model. When finetuning the model we investigated multiple approaches, public datasets and data preparation techniques. High-level summaries are provided in Figure 1 (agent inference) and Figure 2 (agent training). Most important improvement comes from multitask training. It uses multiple purpose-oriented objectives to reinforce more efficient UI representation in the model. Furthermore, Florence-2 appears to be good choice for agent model. Our key contribution is as follows:

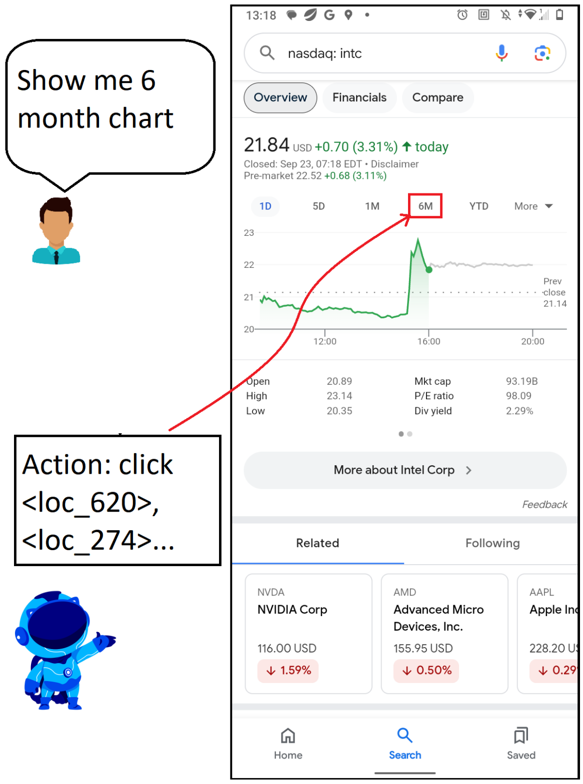

Figure 1: Example command of the downstream task. TinyClick receives screenshot and user command and predicts bounding box of the UI element.

- TinyClick significantly outperforms the baselines, achieving 73% accuracy on Screenspot and 57% on OmniAct.

*Correspondence to: p.pawlowski@partner.samsung.com, j.hoscilowic@samsung.com.

- Training on multitask data (such as generate captions and explanations) outperforms training on click-commands alone.

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

In addition, our contribution fills important niche, as tiny 0.27B model greatly outperforms state-of-the-art MLLM (GPT4, Gemini etc) on UI control.

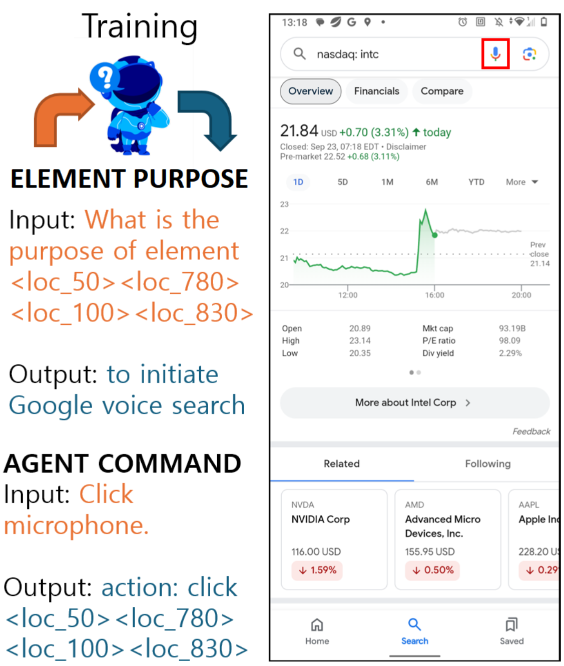

Figure 2: During the training model receives question and generates an answer, both answer and question can consist of location tokens of specific UI element. In this example, the first question is about element description and the second one is a command to click specific item. Multiple different tasks can be associated with a single UI element, allowing the model to gain a better understanding of the UI.

2 Related Works

In recent years there is much interest in Vision-Language Models (VLMs) and Multimodal Large Language Models capable of UI understanding (Rahman et al., 2024; Baechler et al., 2024; Gur et al., 2018; Li and Li, 2023).

Our focus are autonomous agents navigating UIs according to natural language commands of the user: such as AppAgent (Zhang et al., 2023), MobileAgent (Wang et al., 2024), CogAgent (Hong et al., 2023), Auto-UI (Zhang and Zhang, 2024), V-Zen(Rahman et al., 2024). A related problem is grounding, which involves finding UI elements corresponding to a given phrase (Ferret (You et al., 2023), Ferret v2 (Zhang et al., 2024a)). Also, language-only LLM are used as agents for web, using primarily HTML representation (Deng et al., 2023; Gur et al., 2024). Pix2Struct (Lee et al., 2023) was pretrained to reproduce simplified HTML representation of screenshot (including that for masked parts of the UI). SeeClick, one of best agent models so far, (Cheng et al., 2024) uses HTML data for grounding-style pretraining. Among multitask training approaches (Gao et al., 2024) 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 (Fan et al., 2024).

3 Method

In this work, we use Florence-2 Base(Xiao et al., 2023), a 0.27B vision transformer with language modelling head trained for different vision tasks. The model latency is approximately $250ms$, allowing also for cheaper inference. Vision encoder uses 768x768 image resolution. Larger than in other related works on agents (AutoUI (Zhang and Zhang, 2024) uses 224x224), it might be important for large screenshots. While Florence-2 uses text transformer, it is designed to handle detection and grounding tasks, encoding coordinates as single tokens.

3.1 Multitask Training

It was demonstrated that performance of transformer models can be improved by training on many related task (Chung et al., 2022), including Florence2 (Xiao et al., 2023)), and models trained on UI (Gao et al., 2024; Zhang et al., 2024b; Lee et al., 2023). (Hsieh et al., 2023; Zhang et al., 2024b) 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 Florence2 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.

	Size	Screenspot[1]	OmniAct
AutoUI-Base	0.41B[2]	1.97	1.25
Qwen-VL	9.6B	5.2	-
GPT-4V	-	16.2	-
SeeClick	9.6B	53.4	36.8
Florence-2 (vanilla)[3]	0.27B	0	0
TinyClick (ours)	0.27B	**73.8**	**58.3**
w/o multi-task FT		69.4	50.4

Table 1: TinyClick achieves the best results and significantly outperforms other known methods. Goal here is choosing location to click based on user command (point prediction). The metric presented in this table is accuracy as presented in Chapter 4.3. All values are in percentage (%).

- Element locating - 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 (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. Experiments confirm the viability of our approach, beating present baselines with much smaller and faster model.

4 Experiments

4.1 Training Datasets

We use following public datasets: WaveUI (Daniel Jeffries, 2024) (we do not use WebUI due to license issue and remove Screenspot and OmniAct as these are benchmarks), AMEX (Chai et al., 2024), Mind2Web (Deng et al., 2023), GUI Odyssey (Lu et al., 2024) (it is not included in our final training), GUI Course (Chen et al., 2024), AndroidControl (Li et al., 2024), ScreenQA (Hsiao et al., 2024). For WaveUI we use commands (for agent action task) as well as provided MLLM-generated expectation, purpose and captions of UI elements. We use similar approach to augment AMEX and GUI Course, where commands or functionalities of UI elements are provided with UI elements location. We use InternVL2-26B to annotate these data with purposes, captions or expectations. For AMEX, we use 'functionality' field (manually annotated purpose of UI element) as well as Android XML annotations (if available for specific element).

Information about our trainset that produced results in Table 1 can be found in Appendix A, Table **??**, showing that training uses 845k rows in total.

4.2 Benchmarks

We use two standard benchmarks: Screenspot (Cheng et al., 2024) and OmniAct (Kapoor et al., 2024). The first one contains 1200 test cases divided into 3 groups: mobile, web and desktop. The second one contains 3000 examples mainly related to the use of desktop operating systems.

4.3 Results Analysis

The accuracy shown in Tables 1 is calculated as 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 (Cheng et al., 2024).

The results show strong performance improvement over other approaches, such as SeeClick (Cheng et al., 2024), AutoUI (Zhang and Zhang, 2024), and other MLLMs (see Table 1).

4.4 Methodology

We verified that presented claims are supported by statistically significant 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 observed variance of repeated experiments (about 1% for Screenspot). Results below significance threshold are deemed not conclusive. We used 3 epochs as initial setting and tested training up to 5 epochs, which did not produce statistically significant improvement (albeit appears to work minimally better). We trained models on 4xA100 GPU (except Table 3, which used 1xA100). Training best checkpoint requires about 56 GPU-hours (18h on 4xA100). Best checkpoint result in Table 1 is average accuracy for 3 training runs. Other results are calculated for single training runs.

[1]OmniAct included as validation set
[2]0.22B for base model and 0.19B for Blip2 for features
[3]Florence-2 was not pretrained on GUI data, so it is almost impossible for it to predict correct locations.

4.5 Dataset insights

We performed ablation study, starting with a mixture of multitask and agent command datasets and removing some (see Table 4). Multitask data is more important large amount of commands, and diversity is more important than quantity. 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 expectations, captions and reasoning with use of InternVL2-26B MLLM (analogous to WaveUI). With 51k commands, 6k MLLM generated annotations of each type improve result (see Table 3), 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 (functionality). All multitask approaches yield 59% on Screenspot, but MLLM multitask is better on OmniAct, outperforming software-based metadata. Training on 100% command data produces worse result than baseline (see Table 2). Other MLLM annotation approaches were investigated (using InternVL2). We tested labelling screenshots without manual supervision hoping for large amounts of cheap data. These data however so far did not outperform smaller high quality corpora for present benchmarks.

4.6 Fail analysis.

30% failed examples suggest spurious signals that were taught to produce right answer for wrong reasons. One 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% examples are "missed" clicks: model clicks very closely to correct button, but slightly off-mark. Both of these problems might be mitigated by multitask training, with model trained to reason about the UI and attend to specific parts of screenshot.

[4]Trainset like Table ?? but only 10% of GUI course and 10% of AMEX OD used.

	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 2: Annotation and multitask training on AMEX

	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 3: Annotation and multitask training on GUI course web single turn data.

	Screenspot
TinyClick[4]	72.5
w/o AMEX	71.9
w/o multitask	**67.3**
w/o multitask except metadata	<u>68.1</u>

Table 4: Training data ablation study of TinyClick model.

5 Conclusions

TinyClick model strongly improves present baselines, achieving 73.8% on Screenspot and 58.3% on OmniAct, 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 can strongly improve the performance.

6 Future Research

Presented contribution suggests few new research problems to investigate.

- MLLMs, thus far showing weak performance on UI control, might benefit from training strategy similar to ours. These models are similar architecture to Florence2, but they are larger and designed for general purpose tasks.
- The model often fails when facing complicated structure of UI. Presented offline single-turn approach seems insufficient to deal with this problem, suggesting that online reinforcement learning or some novel approach might be better.
- Florence2, as a grounding and detection model, allows to adapt out-of-domain by si-

multaneous use of annotated images and natural language explanations.

Limitations

From user perspective, TinyClick is research artifact, not suited for real world use. It is limited to supporting only single-turn commands and does not achieve the level of accuracy or functionality needed to serve as a full-fledged assistant. Many device-specific actions, such as hardware buttons or touch gestures are not supported. Some of model predictions show biases, such as clicking unrelated elements on top of screen, where menus are found.

Furthermore, results show large difference between training on full suite of data and training on smaller datasets, suggesting importance of representative training dataset. Thus, user might experience much worse real-world accuracy when testing model on new apps and systems.

From scientific perspective, TinyClick beats prior baseline, but it is not yet fully answered, why such result was produced. Multitask training accounts for part of the improvement and also quality and diversity of training data is important. Other factors might be involved, such as Florence2 multitask pretraining on vision corpora, or Florence2 architecture (high resolution vision backbone, or use of location tokens), or favorable proportion of compute resources to model size. Answer to these questions requires further experiments, and possibly large investment for training new models from scratch. Other important yet poorly understood area is composition of datasets: with results strongly conditional on data distribution it is not clear what this data distribution should be in general, in regard to modalities, apps and systems and how robust are the default proportions we used. Similarly, robustness of training parameters was not tested.

Ethics

Model uses language-modelling architecture similar to MLLM, which could in principle expose it to output personal information or offensive, or misleading content. However, model is trained to predict agent actions and clicks and uses only 0.27B parameters, which makes such case unlikely. For similar reasons, it is unlikely that any personally identifiable information (such as photographs) are learned from the images.

Relevant risk comes from unsupported real-world usage, which could lead to loss of data or even more serious damage, if the model wrongly executes commands. We recommend to test the model only on emulator, virtual machine or other controlled environment. Any risk-sensitive application (such as banking, decision-making, controlling machines...) should be avoided.

Florence2 model is available under MIT license, while the datasets use open licenses that explicitly allow research use (CC BY, Apache 2.0 etc). We share our model checkpoint and code under MIT license, too.

References

Gilles Baechler, Srinivas Sunkara, Maria Wang, Fedir Zubach, Hassan Mansoor, Vincent Etter, Victor Cărbune, Jason Lin, Jindong Chen, and Abhanshu Sharma. 2024. Screenai: A vision-language model for ui and infographics understanding. *Preprint*, arXiv:2402.04615.

Yuxiang Chai, Siyuan Huang, Yazhe Niu, Han Xiao, Liang Liu, Dingyu Zhang, Peng Gao, Shuai Ren, and Hongsheng Li. 2024. Amex: Android multi-annotation expo dataset for mobile gui agents. *Preprint*, arXiv:2407.17490.

Wentong Chen, Junbo Cui, Jinyi Hu, Yujia Qin, Junjie Fang, Yue Zhao, Chongyi Wang, Jun Liu, Guirong Chen, Yupeng Huo, Yuan Yao, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. Guicourse: From general vision language models to versatile gui agents. *Preprint*, arXiv:2406.11317.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. 2024. Seeclick: Harnessing gui grounding for advanced visual gui agents. *Preprint*, arXiv:2401.10935.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

KentaurosAI Daniel Jeffries. 2024. Wave ui dataset.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. *Preprint*, arXiv:2306.06070.

Yue Fan, Lei Ding, Ching-Chen Kuo, Shan Jiang, Yang Zhao, Xinze Guan, Jie Yang, Yi Zhang, and Xin Eric

Wang. 2024. Read anywhere pointed: Layout-aware gui screen reading with tree-of-lens grounding. *Preprint*, arXiv:2406.19263.

Yuan Gao, Kunyu Shi, Pengkai Zhu, Edouard Belval, Oren Nuriel, Srikar Appalaraju, Shabnam Ghadar, Vijay Mahadevan, Zhuowen Tu, and Stefano Soatto. 2024. Enhancing vision-language pre-training with rich supervisions. *Preprint*, arXiv:2403.03346.

Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. 2024. A real-world webagent with planning, long context understanding, and program synthesis. *Preprint*, arXiv:2307.12856.

Izzeddin Gur, Ulrich Rueckert, Aleksandra Faust, and Dilek Hakkani-Tur. 2018. Learning to navigate the web. *Preprint*, arXiv:1812.09195.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. 2023. Cogagent: A visual language model for gui agents. *Preprint*, arXiv:2312.08914.

Yu-Chung Hsiao, Fedir Zubach, Gilles Baechler, Victor Carbune, Jason Lin, Maria Wang, Srinivas Sunkara, Yun Zhu, and Jindong Chen. 2024. Screenqa: Large-scale question-answer pairs over mobile app screenshots. *Preprint*, arXiv:2209.08199.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *Preprint*, arXiv:2305.02301.

Raghav Kapoor, Yash Parag Butala, Melisa Russak, Jing Yu Koh, Kiran Kamble, Waseem Alshikh, and Ruslan Salakhutdinov. 2024. Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web. *Preprint*, arXiv:2402.17553.

Kenton Lee, Mandar Joshi, Iulia Turc, Hexiang Hu, Fangyu Liu, Julian Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. 2023. Pix2struct: Screenshot parsing as pretraining for visual language understanding. *Preprint*, arXiv:2210.03347.

Gang Li and Yang Li. 2023. Spotlight: Mobile ui understanding using vision-language models with a focus. *Preprint*, arXiv:2209.14927.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. 2024. On the effects of data scale on computer control agents. *Preprint*, arXiv:2406.03679.

Junpeng Liu, Yifan Song, Bill Yuchen Lin, Wai Lam, Graham Neubig, Yuanzhi Li, and Xiang Yue. 2024. Visualwebbench: How far have multimodal llms evolved in web page understanding and grounding? *Preprint*, arXiv:2404.05955.

Quanfeng Lu, Wenqi Shao, Zitao Liu, Fanqing Meng, Boxuan Li, Botong Chen, Siyuan Huang, Kaipeng Zhang, Yu Qiao, and Ping Luo. 2024. Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices. *Preprint*, arXiv:2406.08451.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2022. Webgpt: Browser-assisted question-answering with human feedback. *Preprint*, arXiv:2112.09332.

Abdur Rahman, Rajat Chawla, Muskaan Kumar, Arkajit Datta, Adarsh Jha, Mukunda NS, and Ishaan Bhola. 2024. V-zen: Efficient gui understanding and precise grounding with a novel multimodal llm. *Preprint*, arXiv:2405.15341.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. 2023. Android in the wild: A large-scale dataset for android device control. *Preprint*, arXiv:2307.10088.

Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *Preprint*, arXiv:2401.16158.

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. 2023. Florence-2: Advancing a unified representation for a variety of vision tasks. *Preprint*, arXiv:2311.06242.

Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. 2023. Ferret: Refer and ground anything anywhere at any granularity. *Preprint*, arXiv:2310.07704.

Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. 2023. Appagent: Multimodal agents as smartphone users. *Preprint*, arXiv:2312.13771.

Haotian Zhang, Haoxuan You, Philipp Dufter, Bowen Zhang, Chen Chen, Hong-You Chen, Tsu-Jui Fu, William Yang Wang, Shih-Fu Chang, Zhe Gan, and Yinfei Yang. 2024a. Ferret-v2: An improved baseline for referring and grounding with large language models. *Preprint*, arXiv:2404.07973.

Jiwen Zhang, Jihao Wu, Yihua Teng, Minghui Liao, Nuo Xu, Xiao Xiao, Zhongyu Wei, and Duyu Tang. 2024b. Android in the zoo: Chain-of-action-thought for gui agents. *Preprint*, arXiv:2403.02713.

Zhuosheng Zhang and Aston Zhang. 2024. You only look at screens: Multimodal chain-of-action agents. *Preprint*, arXiv:2309.11436.