Hidden Forms: A Dataset to Fill Masked Interfaces from Language Commands

Anirudh Sundar [1], **Christopher Richardson** [2] *, **William Gay** [1], **Benjamin Reichman** [1], **Larry Heck** [1]
[1] Georgia Institute of Technology, USA
[2] Google Inc., USA
asundar34, larryheck@gatech.edu

Abstract

This paper introduces Hidden Forms (hFORMS), a dataset of natural language commands paired with user interfaces with masked visual context. By obscuring specific UI elements, the dataset challenges Computer-Using Agents to parse natural language instructions and infer the correct bounding box locations by leveraging UI context. Furthermore, hFORMS contains three distinct masking strategies representing progressive difficulty levels. Additionally, we explore parameter-efficient fine-tuning approaches using Vision-Language models from the `Llama` and `Qwen` series, demonstrating that fine-tuning on mobile domains results in more than 5x improvement in zero-shot domain adaptation performance when identifying bounding boxes on the desktop and web domains.

1 Introduction

Recent work in NLP has seen the extension of language modeling techniques to develop Computer-Using Agents (CUAs) (Gemini Team, 2024; Anthropic, 2025; OpenAI, 2025). CUAs execute natural-language user requests by interacting with elements of the graphical user interface (GUI), such as buttons, menus, and text fields. Current CUAs perform about half as well as humans on popular agent evaluation benchmarks such as OS-World (Xie et al., 2025), WebArena (Zhou et al., 2024), and VisualWebArena (Koh et al., 2024) with the average human performance hovering around 75%. Furthermore, leaderboards for these benchmarks [1] are populated by proprietary models, or models that use a significant number of parameters ($>$ 70B) (Qin et al., 2025), raising concerns about locally deployable solutions. This requires developing parameter-efficient techniques with smaller language models.

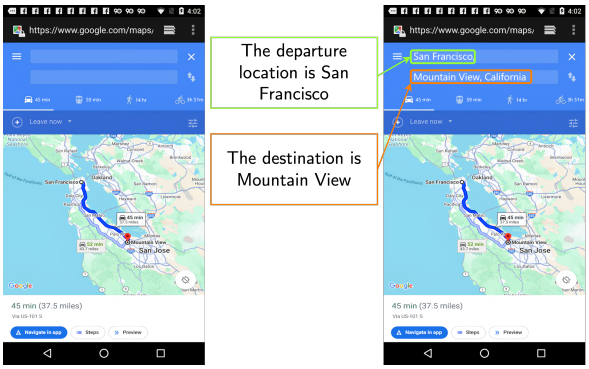

Figure 1: An example of the task in hFORMS. The task is to identify the bounding box location for the content on the screen. The interface contains multiple elements that are hidden from the model.

Motivated by the prevalence of mobile interactions, several datasets have been developed to build CUAs in the mobile domain (Zhang et al., 2023; Wang et al., 2025; Rawles et al., 2024). However, leaderboards for these datasets are also dominated by closed-source systems such as GPT-4, Claude, and Gemini. Locally deployable systems require developing smaller-scale models that run on low-resource hardware, e.g. single-GPU devices. However, building generalist UI-understanding capabilities in smaller vision-language models requires additional domain-specific training. To address these limitations, we build upon existing paired natural-language GUI datasets and introduce **Hidden Forms** [2] (hFORMS). hFORMS addresses a crucial auxiliary task in UI understanding: given a natural language description of GUI content or an action command, the system must infer the correct bounding box location to place this information. Importantly, the ground truth location of the element is concealed from the system, compelling it to leverage on-screen contextual cues to successfully complete the task.

*Work done while at Georgia Tech
[1] https://os-world.github.io/

[2] The dataset and code is available at https://github.com/avalab-gt/hFORMS.

Proceedings of the 1st Workshop for Research on Agent Language Models (REALM 2025), pages 89–96

2 Related Work

Prior work on UI modeling focused on the identification and classification of visual elements on mobile screens (Chen et al., 2020; Bunian et al., 2021; Zhang et al., 2021; Wu et al., 2023). Following the development of vision-language models, more recent work focused on jointly modeling referring expressions within the context of mobile interfaces (Bai et al., 2021; Li et al., 2020; Hsiao et al., 2022; Heck et al., 2024).

UIBert (Bai et al., 2021) consists of a dataset specifically for the task of UI understanding. Bai et al. (2021) introduce five tasks to learn representations by jointly modeling on-screen content and their captions obtained through OCR. They evaluate trained models for referring expression retrieval, a multiple-choice task where the goal is to retrieve the correct on-screen content given a natural language description. **ScreenQA** (Hsiao et al., 2022) is a dataset of questions and answers targeting content across multiple Android apps. Given an app screenshot, crowdworkers write questions and answers that address specific screen components. **ScreenSpot** (Cheng et al., 2024) is a benchmark dataset of screenshots and instructions from iOS, Android, Windows, MacOS, and webpages that evaluates the ability of models to identify the locations of screen content corresponding to natural language commands.

3 Building the hFORMS Dataset

hFORMS consists of three splits – $\text{hFORMS}_{\text{ScreenQA}}$, $\text{hFORMS}_{\text{UIBert}}$, and $\text{hFORMS}_{\text{ScreenSpot}}$. As the names indicate, the splits are built by modifying ScreenQA, UIBert, and ScreenSpot. While ScreenQA and UIBert contain Android app screenshots from RICO (Deka et al., 2017), ScreenSpot is a smaller dataset consisting of screenshots from WebArena (Zhou et al., 2024).

3.1 Bounding Box Identification

For ScreenQA, we first collect all question-answer pairs addressing a given screenshot and the associated bounding boxes on the GUI elements. Then, the bounding boxes are masked based on the strategies described in Section 3.2 to ensure that the GUI does not contain any of the elements whose positions are to be identified. The long-form versions of the answers are used to generate the dataset.

(a) Original UI (b) Static

(c) High-contrast (d) Low-contrast

Figure 2: Examples of the three different masking strategies in hFORMS.

Since UIBert contains bounding boxes for every element, masking all of which would make the task impossible, we select 20% of the bounding boxes for GUI elements at random. The number was empirically selected to produce a similar number of masked elements as the ScreenQA dataset. The textual component in $\text{hFORMS}_{\text{UIBert}}$ is obtained from the UIBert referring expressions.

Finally, since ScreenSpot contains only one bounding box corresponding to the action to be taken, the selection is trivial. We use

Dataset	Split	# Samples
$\text{hFORMS}_{\text{ScreenQA}}$	Train	62,373
	Validation	7,832
	Test	7,691
$\text{hFORMS}_{\text{UIBert}}$	Train	15,624
	Validation	471
	Test	565
$\text{hFORMS}_{\text{ScreenSpot}}$	Mobile	502
	Desktop	334
	Web	436

Table 1: Dataset statistics for the different splits in hFORMS

Masking	Model	IoU	BCP
ScreenQA			
Static	Llama-3.2-11B	37.19	58.86
	Qwen2.5-VL-7B	64.78	74.09
High-contrast	Llama-3.2-11B	54.12	77.81
	Qwen2.5-VL-7B	80.34	85.58
Low-contrast	Llama-3.2-11B	17.61	30.70
	Qwen2.5-VL-7B	39.56	57.38
UIBert			
Static	Llama-3.2-11B	36.58	54.34
	Qwen2.5-VL-7B	58.54	67.08
High-contrast	Llama-3.2-11B	44.05	67.08
	Qwen2.5-VL-7B	74.65	78.94
Low-contrast	Llama-3.2-11B	25.33	39.82
	Qwen2.5-VL-7B	39.33	51.68

Table 2: IoU and Box Center Prediction results on the ScreenQA split of hFORMS

hFORMS$_{\text{ScreenSpot}}$ to evaluate zero-shot domain adaptation capabilities and posit that masking a single element represents real-world situations.

hFORMS is formatted in the JSON Lines format, an example of the json schema is provided in Appendix B.

3.2 Masking GUI Information

The next step in building hFORMS is to mask the corresponding contextual information on the GUI. In this work, we experiment with three different masking strategies of varying levels of difficulty. The first masking strategy simply draws a white box over the identified bounding boxes. The second masking strategy masks the bounding boxes with a contrasting color. This results in an easier task that represents a multiple choice scenario where the system has to choose from a limited number of options to fill in content. The final masking strategy uses a color that is selected dynamically based on the pixel values around the bounding box. By choosing a color that is as similar as possible to the background, the corresponding GUI element is effectively hidden from the system, making the identification of the element a harder challenge than either of the previous strategies. Examples of the three masking strategies are provided in Figure 2.

Model	Training Data	IoU	BCP
Mobile			
Llama-3.2-11B	None	1.21	0.40
	ScreenQA	2.43	9.56
	UIBert	6.56	13.55
Qwen2.5-VL-7B	None	5.95	26.10
	ScreenQA	12.93	33.67
	UIBert	31.35	52.99
Desktop			
Llama-3.2-11B	None	0.83	1.22
	ScreenQA	1.69	5.69
	UIBert	2.55	5.39
Qwen2.5-VL-7B	None	2.37	11.93
	ScreenQA	7.22	15.27
	UIBert	12.24	21.56
Web			
Llama-3.2-11B	None	0.74	0.92
	ScreenQA	1.00	5.50
	UIBert	2.08	4.82
Qwen2.5-VL-7B	None	3.93	20.41
	ScreenQA	17.26	35.09
	UIBert	28.83	42.43

Table 3: Zero-shot cross-domain performance of Llama-3.2-11B and Qwen2.5-VL-7B on hFORMS$_{\text{ScreenSpot}}$. The source domain dataset is provided under Training Data.

4 Results

4.1 Fine-tuning

We experiment with two open-source Vision-Language Models – `Llama-3.2-11B-Vision-Instruct-bnb-4bit` (Dubey et al., 2024) and `Qwen2.5-VL-7B-Instruct-bnb-4bit` (Bai et al., 2025). We use the 4-bit versions of the models as provided by the unsloth library [3] as the 4-bit models fit on a single GPU and represent compute situations when these models are typically utilized. We fine-tune models for the task of predicting the bounding box as a text sequence `x1<SEP>y1<SEP>x2<SEP>y2` where `x1,y1` and `x2,y2` represent the top-left and bottom-right corners of the bounding box. Additional details about the fine-tuning setup are provided in Appendix A.

We report performance on two metrics – the Intersection over Union (IoU) and Box Center Pre-

[3] https://unsloth.ai/

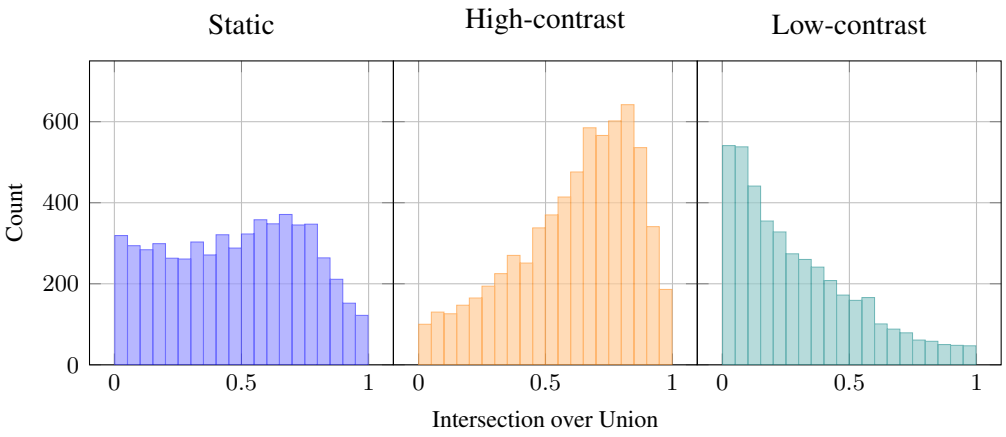

Figure 3: Histogram of IoU scores for the three different masking strategies - `Llama` on $\text{hFORMS}_{\text{ScreenQA}}$

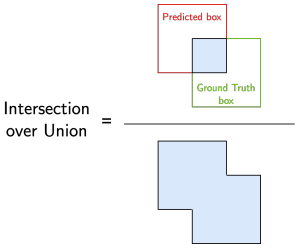

Figure 4: Intersection over Union of two bounding boxes.

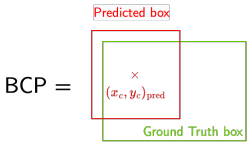

Figure 5: Example computation of Box Center Prediction. In this example, since the center of the predicted box (in red) lies inside the ground truth box (in green), the BCP score is 1, even though the IoU is less than 1.

diction (BCP). IoU is a metric that calculates the ratio between the overlapping area (intersection) and the combined area (union) of predicted and ground truth bounding boxes (Figure 4). The BCP metric measures whether the center of the predicted bounding box lies anywhere inside the ground-truth box. Since the UIBert and ScreenSpot datasets contain natural language statements that ask the system to click on UI elements, BCP is an appropriate metric since a click on any UI element is a successful hit. In this work, the center of the predicted box serves as a proxy for a click on a screen. BCP accuracy is predicted using the formula in Equation 1 and is exemplified in Figure 5. While IoU is optimized only when the two bounding boxes overlap completely, BCP awards partial credit for predicting a reasonably correct response.

$$\text{BCP} = \mathbb{1}_{(x_c, y_c)_{\text{pred}}} \in \text{Bounding Box}_{\text{GT}} \quad (1)$$

Table 2 presents parameter-efficient fine-tuning results from `Llama-3.2-11B` and `Qwen2.5-VL-7B` on the ScreenQA and UIBert splits of hFORMS. Consistent with our hypothesis, the low contrast masking is the hardest task, while the high contrast masking is the easiest on both splits of the dataset. Across all masking strategies, `Qwen2.5-VL-7B` performs better than `Llama-3.2-11B` since the pretraining data for `Qwen` includes screenshots from GUIs for agentic capabilities.

4.2 Zero-Shot Cross-Domain Adaptation

An important challenge when building CUAs is ensuring that they adapt to unseen domains since user interfaces often change with software updates, and users may request actions on newly developed apps not seen during training. Another concern is when dealing with screenshots with different aspect ratios and resolutions since mobile desktop, and web app windows are scaled differently. Alleviating this concern requires good zero-shot cross-domain adaptation capabilities. In this work, we evaluate the models in a zero-shot configuration on a modified version of the ScreenSpot dataset, which serves as the holdout domain. Since there is only one bounding box in the ScreenSpot dataset per screenshot, we experiment with only the Low-contrast masking strategy.

We evaluate the versions of the models fine-tuned on each of the two datasets separately and present the results in Table 3. As before, `Qwen2.5-VL-7B` performs better than

Llama-3.2-11B. Additionally, training on the UIBert split performs better than ScreenQA. We hypothesize that this is because the commands in UIBert are similar to the commands in ScreenSpot that address clicking related tasks while the ScreenQA dataset contains descriptions of content. Furthermore, the benefits of training on the UI-based datasets carries over, evidenced by better performance on the Mobile split of ScreenSpot when compared to Desktop and Web. Interestingly, though the models are fine-tuned on the UI-based datasets, the performance on Desktop and Web results in up to 5x improvement in IoU scores over the versions that are not trained (prompt available in Appendix C). We observe comparable performance between iOS and Android, Appendix D.

4.3 Performance Analysis

The results in Tables 2 and 3 raise questions regarding the nature of bounding box hits and misses. To understand this distribution, the histogram of IoU scores is presented in Figure 3. As observed in Figure 3, the distributions have significant differences between the different types of masking. Note that peaks at either extreme have been removed for clarity and the unmodified distributions are available in Appendix E. The high-contrast masking has a significant peak around IoU 0.8 while the low-contrast masking, a harder task, has a distribution that decreases as the IoU values increase. The static masking appears to be relatively uniform, which further supports the observations from Table 2 that the difficulty is in between the other two strategies.

5 Conclusion

This work introduces Hidden Forms (hFORMS), a dataset comprising natural language commands paired with user interfaces where relevant information is masked to help build UI understanding capabilities in Computer Using Agents. By obscuring UI elements, we challenge agents to parse natural language instructions and infer the correct bounding box locations by leveraging contextual cues. hFORMS presents three distinct masking strategies representing progressive difficulty levels. Additionally, we explore parameter-efficient fine-tuning approaches using Vision-Language models from the Llama and Qwen series, demonstrating that fine-tuning on mobile domains significantly improves zero-shot domain adaptation performance on the desktop and mobile domains.

Limitations

This work is limited by the fact that it is primarily situated in the mobile domain, and the desktop and webapps are restricted to zero-shot domain-adaptation experiments. Future work could address expanding the dataset to larger datasets for the other domains. Additionally, the hFORMS dataset is obtained by extending datasets from prior work. To make the challenge harder, future work could address collected samples specifically for this challenge by recruiting annotators. Finally, this work only evaluates models that fit on a single GPU, we leave the evaluation of much larger models to future work.

Acknowledgments

This work was supported by CoCoSys, one of seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

References

Anthropic. 2025. Claude's extended thinking.

Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, et al. 2021. Uibert: Learning generic multimodal representations for ui understanding. *arXiv preprint arXiv:2107.13731*.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Sara Bunian, Kai Li, Chaima Jemmali, Casper Harteveld, Yun Fu, and Magy Seif Seif El-Nasr. 2021. Vins: Visual search for mobile user interface design. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Jieshan Chen, Mulong Xie, Zhenchang Xing, Chunyang Chen, Xiwei Xu, Liming Zhu, and Guoqiang Li. 2020. Object detection for graphical user interface: Old fashioned or deep learning or a combination? In *proceedings of the 28th ACM joint meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1202–1214.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. 2024. Seeclick: Harnessing gui grounding for advanced visual gui agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9313–9332.

Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols,

and Ranjitha Kumar. 2017. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th Annual Symposium on User Interface Software and Technology*, UIST '17.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gemini Team. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Larry Heck, Simon Heck, and Anirudh S. Sundar. 2024. mForms : Multimodal form filling with question answering. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 11262–11271, Torino, Italia. ELRA and ICCL.

Yu-Chung Hsiao, Fedir Zubach, Maria Wang, and Jindong Chen. 2022. Screenqa: Large-scale question-answer pairs over mobile app screenshots. *CoRR*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. 2024. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 881–905, Bangkok, Thailand. Association for Computational Linguistics.

Yang Li, Gang Li, Luheng He, Jingjie Zheng, Hong Li, and Zhiwei Guan. 2020. Widget captioning: Generating natural language description for mobile user interface elements. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5495–5510, Online. Association for Computational Linguistics.

OpenAI. 2025. Operator System Card.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjun Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, Xiaojun Xiao, Kai Cai, Chuang Li, Yaowei Zheng, Chaolin Jin, Chen Li, Xiao Zhou, Minchao Wang, Haoli Chen, Zhaojian Li, Haihua Yang, Haifeng Liu, Feng Lin, Tao Peng, Xin Liu, and Guang Shi. 2025. Ui-tars: Pioneering automated gui interaction with native agents. *Preprint*, arXiv:2501.12326.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, et al. 2024. Androidworld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*.

Zhenhailong Wang, Haiyang Xu, Junyang Wang, Xi Zhang, Ming Yan, Ji Zhang, Fei Huang, and Heng Ji. 2025. Mobile-agent-e: Self-evolving mobile assistant for complex tasks. *Preprint*, arXiv:2501.11733.

Jason Wu, Siyan Wang, Siman Shen, Yi-Hao Peng, Jeffrey Nichols, and Jeffrey P Bigham. 2023. Webui: A dataset for enhancing visual ui understanding with web semantics. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Jing Hua Toh, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. 2025. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094.

Danyang Zhang, Zhennan Shen, Rui Xie, Situo Zhang, Tianbao Xie, Zihan Zhao, Siyuan Chen, Lu Chen, Hongshen Xu, Ruisheng Cao, et al. 2023. Mobile-env: Building qualified evaluation benchmarks for llm-gui interaction. *arXiv preprint arXiv:2305.08144*.

Xiaoyi Zhang, Lilian De Greef, Amanda Swearngin, Samuel White, Kyle Murray, Lisa Yu, Qi Shan, Jeffrey Nichols, Jason Wu, Chris Fleizach, et al. 2021. Screen recognition: Creating accessibility metadata for mobile applications from pixels. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–15.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. 2024. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*.

A Fine-tuning details

All fine-tuning and inference was run on Nvidia A40 GPUs with 48GB GDDR6 memory.

For our experiments, we fine-tune both our models using the 8-bit Adam optimizer with a learning rate of 2e-4 and 5 warmup steps. We use LoRA (Hu et al., 2022) to train adapters while keeping base weights frozen. We use a LoRA r = 16 and α = 16 with a dropout of 0, and adapter weights added to all linear layers, attention modules, across the vision and language layers. All models are trained for 10000 steps which was the numbers of steps at which the relative decrease in loss was less than 1%. All experiments use a random seed of 3407.

B JSON Schema


```
{
  "image": "image_files/5.jpg",
  "image_width": 1080,
  "image_height": 1920,
  "statement_l": "There are 12 exercises
      ↪ in total to do.",
  "statement_s": "12",
  "bbox": "509<SEP>116<SEP>569<SEP>169",
  "box_center": "539.0<SEP>142.5"
}
```


Figure 6: JSON Lines schema describing the dataset structure

C Prompt for Zero-Shot experiments

```
Look at the image and find the UI element
that matches this instruction. Return
ONLY the bounding box coordinates in this
EXACT format with NO text before or after:
x1<SEP>y1<SEP>x2<SEP>y2
```

D Breakdown of results for the mobile domain

Table 4 contains a breakdown of the results by different operating systems on the split for ScreenSpot mobile. The low-contrast masking strategy is used in this experiment.

Model	OS	IoU	BCP
ScreenQA			
Llama-3.2-11B	Android	3.68	13.77
	iOS	1.22	5.49
Qwen2.5-VL-7B	Android	10.16	31.58
	iOS	15.63	35.69
UIBert			
Llama-3.2-11B	Android	9.96	20.65
	iOS	3.27	6.67
Qwen2.5-VL-7B	Android	30.88	55.06
	iOS	31.81	50.98

Table 4: Breakdown of results by operating system on ScreenSpot-mobile. The data used to train the models is provided as a header.

E Histogram of IoU Scores for all combinations

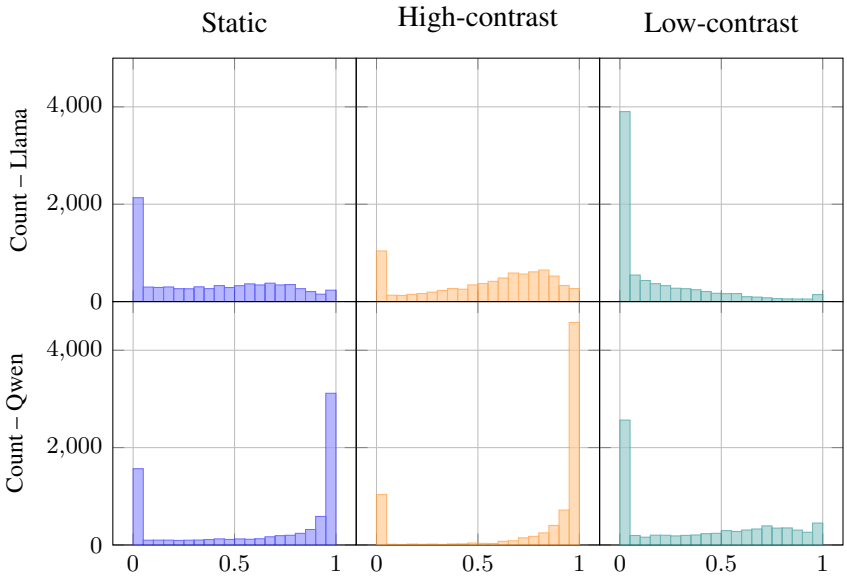

Figure 7: Histogram of Intersection over Union scores on ScreenQA.

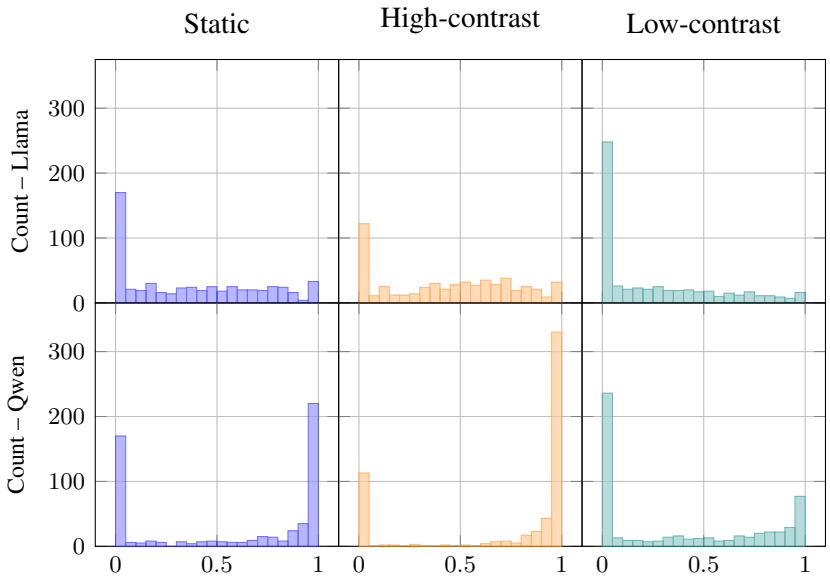

Figure 8: Histogram of Intersection over Union scores on UIBert.