AFRAgent : An Adaptive Feature Renormalization Based High Resolution Aware GUI agent

Neeraj Anand Rishabh Jain Sohan Patnaik Balaji Krishnamurthy Mausoom Sarkar

Media and Data Science Research, Adobe

1. Effect of AFR Block on accuracy

Here, we study the effect of AFR block for Low-res and High-res fusion after full training on AiTW dataset. The introduction of the Adaptive Feature Renormalization (AFR) block significantly enhances model performance, as demonstrated by the results in Table 1. When compared to the baseline InstructBLIP model, $\text{AFRAgent}_{\text{Low-res}}$, which applies low-resolution feature enrichment, shows notable improvements across multiple categories, with overall accuracy increasing from 76.11% to 77.06%. This demonstrates that even low-resolution enrichment contributes significantly to a more accurate model. Further improvements are observed with $\text{AFRAgent}_{\text{High-res}}$, which incorporates high-resolution feature enrichment. This variation yields the highest performance, with accuracy rising to 78.01% overall. These results underscore the effectiveness of the AFR block in improving model accuracy, with high-resolution enrichment providing the most significant gain, highlighting the value of our approach in enhancing feature representation for UI tasks in mobile applications. We also include the detailed comparison across all subsets in Tab 7.

2. Effect of AFR Block on different backbone and scale

In addition to the main experiments, we also evaluate the model-agnostic nature and scalability of AFR by applying it to LLaVA-1.5 (7B, direct projection). Specifically, we integrate the Q-Former module (185M) from InstructBLIP to enrich visual tokens.

Similar to AFRAgent, we incorporate high-resolution information into LLaVA using a combination of the Q-Former and AFR. For each input image, we generate 4 non-overlapping crops at higher resolution. These cropped features are independently passed through the Q-Former, which performs cross-attention between the high-resolution crop features and the query tokens. The enriched query tokens capture localized details from each crop and are then aggregated to form a compact high-resolution representation. Through AFR, these query tokens are used to modulate and enhance the original low-resolution image tokens, allowing the final representation to jointly capture both global context and fine-grained spatial details. This enriched set of image tokens is then fed into the LLM for response generation. Compared to directly scaling the backbone or applying cross-attention at the decoder side, this approach is significantly more efficient, as it leverages the lightweight Q-Former while still injecting high-resolution visual cues.

Table 2 reports the results on Meta-GUI and two independently trained AITW subsets. AFR yields consistent gains across scales and model types: overall completion improves from 88.26 $\rightarrow$ 90.56 for LLaVA and from 87.93 $\rightarrow$ 90.83 for InstructBLIP, with similar improvements observed across the AITW subsets.

3. Number of Crop Images Ablation

To investigate the impact of the number of crop images on the performance of AFRAgent , we conduct experiments on the AITW dataset, evaluating both the general and single data splits. The results are summarized in Table 3. In this analysis, we explore configurations with $C = 2, 3, 4, 5$ crops, where the mobile screenshot is horizontally divided into C equally spaced regions. These crops are processed individually by the vision encoder, and their embeddings are concatenated to form a comprehensive representation. This combined feature representation is then cross-attended with the Q-former, followed by the high-resolution Adaptive Feature Renormalization (AFR) block, which integrates high-resolution details into the final output.

The results demonstrate that the accuracy improves steadily as the number of crops increases from $C = 2$ to $C = 4$ for both the general and single splits. Notably, on the General split, performance peaks at $C = 4$ with an accuracy of 70.91%, slightly dropping to 70.85% for $C = 5$. A similar trend is observed on the single split, with the highest accuracy of 86.30% at $C = 4$, followed by a marginal decline to 86.18% for $C = 5$. The decrease in performance for

Method	General	Install	GoogleApps	Single	WebShop.	Overall
InstructBlip[4]	70.66	79.59	73.05	84.99	72.26	76.11
AFRAgent$_{\text{Low-res}}$	68.84	80.27	73.46	90.65	72.06	77.06
AFRAgent$_{\text{High-res}}$	**70.67**	**80.89**	**74.16**	**91.06**	**73.27**	**78.01**

Table 1. AFR Block serves as an effective enrichment operation for both low res and high res feature enrichment during unfied training. Full results can be found below in Sec 5

Method	Params	Act. Type	Item Acc.	Direction Acc.	Utter. (BLEU)	Input (F1)	Input (EM)	Action (CR)	General	Single
LlaVA-1.5	7.06B	91.2	92.63	96.09	**74.02**	93.33	88.53	88.26	65.7	87.17
LlaVA-1.5_AFR	7.25B	**92.05**	**93.44**	**96.29**	72.75	**97.54**	**94**	**90.56**	**66.34**	**88.2**
InstructBlip	4.02B	91.3	92.85	96.09	65.9	97.5	**94.57**	87.93	68.39	81.37
InstructBlip$_{\text{AFR}}$	4.03B	**93.28**	**95.06**	**97.02**	**67.6**	**97.94**	94.44	**90.83**	**70.91**	**86.3**

Table 2. Impact of AFR on performance across Meta-GUI and AITW subsets. AFR consistently improves both LLaVA and InstructBLIP backbones, highlighting model-agnostic gains for different types and scale for action accuracy and input understanding

Dataset	2 Crops	3 Crops	4 Crops	5 Crops
General	70.30	70.64	70.91	70.85
Single	85.68	86.12	86.30	86.18

Table 3. Effect of varying the number of crop images on the performance of AFRAgent for the AITW dataset (General and Single splits).

$C = 5$ can be attributed to the distortion introduced when resizing the crops to square images for the vision encoder. The $C = 4$ configuration, on the other hand, achieves a better balance by maintaining an aspect ratio closer to the original screen dimensions before resizing. Based on these findings, we conclude that using $C = 4$ crops provides the best trade-off between preserving the original aspect ratio and enhancing the feature representation. Consequently, $C = 4$ is chosen as the default configuration for training AFRAgent to achieve optimal performance.

4. Effect of Fusion Strategy

In the main paper, we briefly analyzed the effectiveness of our Adaptive Feature Renormalization (AFR) technique as a fusion strategy for F_{enrich} and F_{target}, comparing it against a residual approach defined as $F_{enriched} = F_{target} + \text{MLP}(F_{enrich})$, Mixture of Experts (MoE) method [17], highResProj and AnyRes setting of Qwen2-VL across the single and general data splits. We presented results that indicated AFR consistently outperformed these fusion strategies, demonstrating its robustness and efficacy for feature integration in multimodal tasks.

Here, we provide a detailed breakdown of the results across multiple metrics: Action Type Accuracy, Text Input Accuracy, and Action Matching Score for both the General and Single splits of the AITW dataset. As shown in Table 4, AFR consistently achieves the highest performance across all metrics, demonstrating its effectiveness as a fusion strategy. These results underscore the superiority of AFR in integrating features across diverse dimensions, significantly surpassing the baseline methods in every evaluated aspect. In the main paper, we highlight only the final Action Matching Score for the General and Single subset to emphasize the improvement brought by AFR due to limited space

5. Dataset and Metric

For the empirical evaluation, we leverage two widely used benchmarks for GUI automation: **AITW** and **META-GUI**. These datasets provide comprehensive resources for developing and evaluating models in smartphone GUI automation tasks. Detailed dataset statistics are summarized in Table 5.

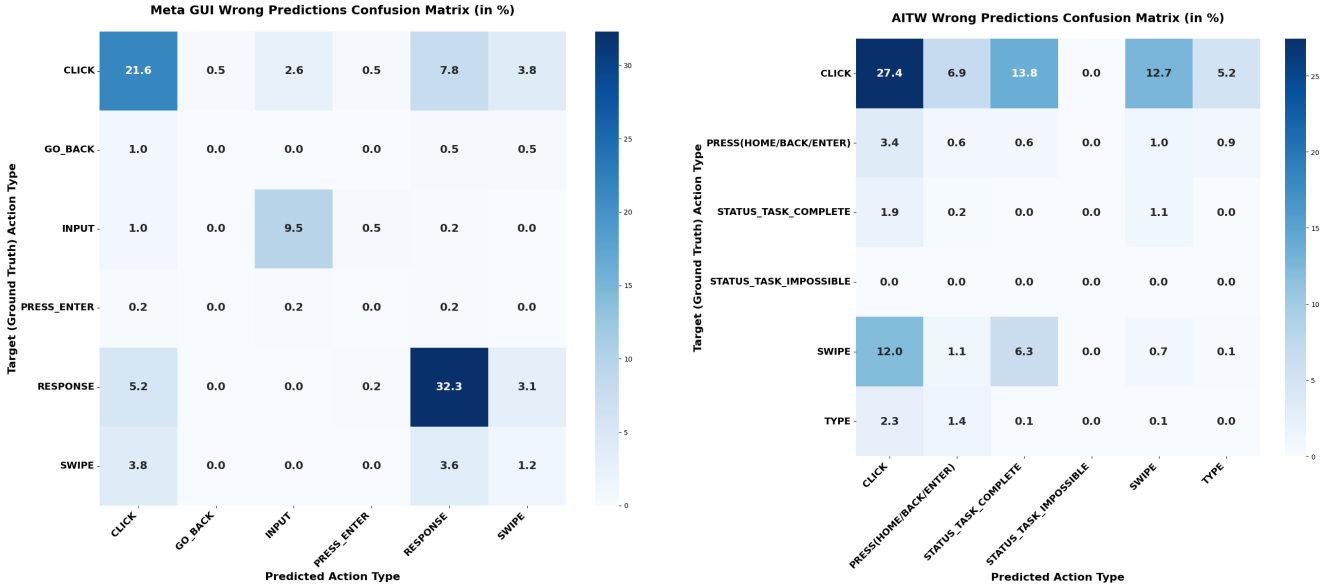
AITW (Android In The Wild) AITW [12] is the largest and most diverse dataset available for smartphone GUI automation. It comprises **715k episodes** of task executions, each containing natural language goal instructions and a sequence of screenshot-action pairs illustrating step-by-step task completion. These episodes are collected across a variety of Android apps and websites, ensuring data diversity in terms of screen resolutions, device types, and operating systems.

The dataset is structured into five subsets based on task categories:

- **General:** Multi-step tasks unrelated to specific app ecosystems.
- **Install:** Tasks related to installing, setting up, or configuring applications.
- **Google Apps:** A large subset focused on tasks performed in Google applications.

AITW	InstructBlip*	Residual	MoE	AFR$_{Low_res}$	highResProj	Qwen2-VL	AFR$_{High_res}$
Performance Metrics							
FLOPs (T)	3.19	3.2	3.54	3.2	6.46	17.08	**5.47**
Size (Billion)	4.02	4.03	4.03	4.03	**4.03**	8.29	**4.03**
General Subset							
Action Type Acc.	87.02	87.18	86.95	**87.32**	87.42	87.35	**87.65**
Text Input Acc.	68.98	68.92	69.09	**69.35**	68.80	69.28	**69.77**
Action Matching Score	69.56	69.61	69.75	**70.2**	69.74	70.15	**70.91**
Single Subset							
Action Type Acc.	92.81	92.92	92.98	**93.57**	93.91	92.94	**93.93**
Text Input Acc.	84.64	84.76	84.88	**84.97**	85.13	84.86	**85.24**
Action Matching Score	85.03	85.13	85.25	**85.37**	86.17	85.21	**86.3**

Table 4. Comparison of different fusion strategies for multimodal feature integration on the AITW dataset. We evaluate FLOPs, model size, and key performance metrics, including Action Type Accuracy, Text Input Accuracy, and Action Matching Score, across the General and Single splits. AFR consistently outperforms other fusion approaches, demonstrating its effectiveness in both low-resolution and high-resolution settings. HighResProj represents a high-resolution projection baseline, while Qwen2-VL employs an AnyRes strategy. AFR Agent achieves the best overall performance while maintaining efficiency. * denotes 257 tokens in the Qformer.



(a) Meta-GUI benchmark.

(b) AiTW benchmark.

Figure 1. Failure analysis of AFRAgent across both benchmarks. On Meta-GUI, errors primarily occur in RESPONSE predictions and CLICK actions, indicating challenges in precise dialog wording and spatial accuracy. On AiTW, the model struggles mostly with CLICK actions and premature task completions, reflecting limitations in coordinate precision and multi-step task reasoning. These plots highlight consistent failure patterns across benchmarks and suggest directions for improving UI understanding and high-resolution feature integration.

- **Single:** Simple single-step tasks, ideal for evaluating atomic action predictions.
- **Web Shopping:** Tasks involving navigating e-commerce websites and making purchases.

A key strength of AITW lies in its realistic task instructions that closely simulate real-world scenarios. For instance, a typical instruction may ask the agent to *"Add an event to the calendar for tomorrow at 3 PM."* These instructions are complemented by corresponding screenshots and actions, enabling detailed modeling of task dependencies.

AITW	Episode	Screen	Goal
General	9476	85,413	545
Install	25,760	250,058	688
Google Apps	625,542	4,903,601	306
Single	26,303	85,668	15,366
Web Shopping	28,061	365,253	13,473
META-GUI	**Episode**	**Dial. turn**	**Screen**
Train	897	3692	14,539
Dev	112	509	1875
Test	116	483	1923

Table 5. Statistics for AITW and Meta GUI showcasing the diversity and versatility of these datasets for the GUI automation task

META-GUI META-GUI [13] is a multimodal dataset designed to train conversational agents for mobile GUIs. It comprises **1k episodes** with over **18k steps** spanning 11 applications across six domains, including **weather**, **calendar**, and **search**. Unlike AITW, META-GUI emphasizes multi-turn interactions between the agent and the user.

Each episode is segmented into dialogue turns, allowing the agent to verify its current state or seek clarification about the next step. For example, after executing an action, the agent may ask, *"Is this what you are looking for?"* This interaction style makes META-GUI uniquely suited for building and evaluating systems that combine GUI automation with conversational capabilities. To handle such interactions effectively, we integrate **dialogue history** (D_{hist}) into the model inputs, alongside the action history. This approach follows the methodology outlined in CoCo-Agent [11]. Dialogue histories include utterances from both the user (u_{user}) and the agent (u_{agent}), enabling models to contextualize current and future actions effectively.

Dataset Statistics Table 5 provides a detailed breakdown of the statistics for AITW and META-GUI, including episode counts, dialogue turns, screen interactions, and goal diversity for AITW. In our study, we used only 10% of Google Apps category to reduce the training time and to prevent overfitting, consistent with other approaches[18]. We also performed coordinate normalization for action type click and axis transformation for action type swipe for AITW following previous work [18]. In summary, the combination of AITW and META-GUI provides a diverse and comprehensive resource for GUI automation research, supporting tasks ranging from single-step atomic actions to complex, multi-turn conversational interactions. These datasets enable the development of advanced models capable of effectively automating and interacting with smartphone GUIs in realistic settings.

ScreenSpot ScreenSpot is a recently introduced dataset designed to evaluate GUI grounding capabilities, which are essential for constructing visual GUI agents. Unlike prior datasets that focus primarily on Android environments [5], ScreenSpot encompasses a diverse set of GUI platforms, including mobile (iOS, Android), desktop (macOS, Windows), and web interfaces.

The dataset consists of over 600 interface screenshots and 1,200+ instructions annotated with actionable elements. Each instruction requires a vision-language model (VLM) to identify and localize specific UI components such as icons, widgets, and text elements. ScreenSpot presents a challenging grounding task, as it contains a substantial number of non-textual UI elements that are more difficult to locate compared to text-based interactions.

To ensure real-world applicability, ScreenSpot samples were curated to avoid overlap with existing training data. Experienced annotators manually collected GUI interfaces and labeled bounding boxes for actionable elements. For mobile and desktop, common applications and operations were selected, while web-based samples were drawn from diverse website categories (e.g., development, shopping, forums, and tools) within the WebArena environment [20].

Evaluation Metrics For AITW, we use the action matching score to assess model performance. Two actions are considered a match if they share the same action type. Specifically, for the *click* action, a match is determined if the predicted and ground-truth locations fall within a 14% screen distance of each other. For *scroll* actions, they are considered equivalent if they have the same primary scroll axis (vertical or horizontal).

For the evaluation of META-GUI, we employ the completion rate as the primary metric for action prediction. An action is deemed completed only if both the action type and its parameters are correctly predicted. Additionally, we use accuracy metrics for action type prediction, item prediction, and direction prediction. For input prediction, we evaluate performance using token-level exact match and F1 score. Response generation quality is measured using the BLEU score.

To assess grounding performance on ScreenSpot, we use click accuracy, which quantifies the proportion of instances where the predicted click location falls within the ground-truth bounding box of the target UI element [8, 19]. This metric is particularly relevant for real-world GUI automation tasks, where precise localization of interactive elements is essential for accurate task execution.

6. Detailed results on AITW and Meta-GUI

Due to space limitations in the main paper, we provide a comprehensive comparison of various methods across all

Method	Params	Act. Type	Item Acc.	Direction Acc.	Input (F1)	Input (EM)	Utter. (BLEU)	Action (CR)
LayoutLM [15]	343M	82.22	71.98	94.87	90.56	83.04	50.43	67.76
LayoutLM_{v2}	426M	85.60	64.38	92.95	70.76	47.37	58.20	64.48
BERT [6]	340M	87.52	82.84	93.59	97.24	93.57	62.19	78.42
LLaVA [9]	7.3B	87.47	77.49	98.18	96.06	-	67.24	76.27
$\text{LLaVA}_{\text{w/history}}$	7.3B	91.68	81.23	97.62	96.93	-	66.57	81.08
m-BASH [13]	340M	90.80	85.90	96.42	94.23	91.23	63.11	82.74
CoCo-Agent [11]	7.3B	92.59	91.72	**98.39**	96.15	-	65.90	88.27
AFRAgent	4B	**93.28**	**95.06**	97.02	**97.94**	**94.44**	**67.6**	**90.83**

Table 6. Comparative performance on Meta-GUI benchmark, evaluating action completion rate (CR), action type, item and direction accuracy and input text metrics (F1, exact match), and response quality (BLEU). AFRAgent achieves state-of-the-art accuracy across most metrics, demonstrating efficiency with fewer parameters and enhanced input handling.

Method	Params	General	Install	GoogleApps	Single	WebShop.	Overall
Structured Layout Setting							
$\text{CoCo-Agent[11]}_{\text{seperate}}$	7.3B	69.92	80.60	75.76	88.81	74.02	77.82
$\text{CoCo-Agent[11]}_{\text{unified}}$	7.3B	70.96	81.46	76.45	91.41	75.00	79.05
$\text{AFRAgent}_{\text{unified}}$	4B	71.62	80.81	76.26	90.78	75.10	78.92
Pure Multimodal Setting							
PaLM-2*[1]	-	-	-	-	-	-	39.6
ChatGPT*†	-	5.93	4.38	10.47	9.39	8.42	7.72
MM-Navigator†[16]	-	41.66	42.46	49.82	72.83	45.73	50.54
$\text{MM-Navigator}_{\text{w/text}}$†[16]	-	43.55	42.44	49.18	48.26	76.34	51.92
$\text{MM-Navigator}_{\text{w/history}}$†[16]	-	43.01	46.14	49.18	78.29	48.18	52.96
OmniParser†[10]	-	48.3	57.8	51.6	77.4	52.9	57.7
BC[12]	-	-	-	-	-	-	68.7
$\text{BC}_{\text{w/history}}$[12]	-	63.7	77.5	75.7	80.3	68.5	73.1
$\text{LLaMA-2*}_{\text{unified}}$	7B	28.56	35.18	30.99	27.35	19.92	28.40
$\text{Auto-GUI}_{\text{seperate}}$[18]	4.5B	65.94	77.62	76.45	81.39	69.72	74.22
$\text{Auto-GUI}_{\text{unified}}$[18]	4.5B	68.24	76.89	71.37	84.58	70.26	74.27
$\text{LLaVA}_{\text{unified}}$[9]	7.3B	58.93	72.41	70.81	83.73	65.98	70.37
$\text{InstructBlip}_{\text{unified}}$[4]	4B	70.66	79.59	73.05	84.99	72.26	76.11
$\text{MobileVLM}_{\text{unified}}$[14]	9.6B	69.58	79.87	74.72	81.24	71.70	74.94
$\text{MobileVLM}_{\text{seperate}}$[14]	9.6B	70.27	78.86	**76.86**	87.06	71.42	77.05
$\text{SeeClick}_{\text{unified}}$[3]	9.6B	67.6	79.6	75.9	84.6	73.1	76.2
$\text{SphAgent}_{\text{unified}}$[2]	7B	68.2	80.5	73.3	85.4	**74**	76.28
$\text{CogAgent}_{\text{unified}}$[7]	18.3B	65.38	78.86	74.95	**93.49**	71.73	76.88
$\text{AFRAgent}_{\text{Unified/Low-res}}$	4B	68.84	80.27	73.46	90.65	72.06	77.06
$\text{AFRAgent}_{\text{seperate/High-res}}$	4B	**70.91**	80.56	73.88	86.3	72.6	76.85
$\text{AFRAgent}_{\text{unified/High-res}}$	4B	70.67	**80.89**	74.16	91.06	73.27	**78.01**

Table 7. Comparative performance of AFRAgent and state-of-the-art models on the AITW benchmark across various task categories with separate and unified training. AFRAgent achieves superior performance in the purely multimodal setting and competitive results in the structured layout setting, demonstrating robust action prediction accuracy while reducing computational and memory overhead.* represent that the model is only language based and † represent zero shot or few shot setting evaluation

evaluation metrics here. For AITW, we include AFRAgent evaluated under both the unified and separate training settings. In the unified training setting, the model was trained using all dataset splits combined, allowing it to learn from a broader distribution of GUI interactions. In the separate training setting, each dataset split was trained independently, and evaluation metrics were computed separately for each dataset. The complete results for these settings are reported in Table 7. Additionally, we provide a detailed evaluation of the AFRAgent Low-res configuration, where the model was trained using unified training while leveraging feature enrichment solely through Low resolution feature enrichment.

For Meta-GUI, we present the comparison of all meth-

ods across the full range of evaluation metrics, including Action Completion Rate (CR), Action Type Accuracy, Item Accuracy, Direction Accuracy, Input Text F1, Exact Match (EM) Score, and BLEU for response generation. The complete evaluation results are provided in Table 6, offering a detailed breakdown of performance across all metrics.

7. Failure Analysis of AFRAgent

Figures 1a and 1b provide a detailed breakdown of error patterns for AFRAgent on the Meta-GUI and AITW benchmarks. On Meta-GUI, the most frequent errors occur in RESPONSE predictions (32.3%), where the model produces conversational feedback that is semantically appropriate but fails to reproduce the exact wording expected in the ground truth. This suggests that errors arise more from challenges in precise dialog fidelity than from misunderstanding the underlying action. The second major error source involves CLICK actions (21.6%), primarily due to coordinate inaccuracies: the model often identifies the correct region of the screen but predicts click points that are slightly offset from the ground-truth locations, reflecting limitations in fine-grained spatial precision.

On AITW, the most prominent errors are in CLICK actions (66% overall), with a large fraction attributable to coordinate inaccuracies (27%). Another notable category of errors is premature completions (13.8%), in which the model prematurely declares task completion while additional interactions are still required. These cases reveal difficulties in maintaining robust multi-step reasoning and accurately assessing intermediate task states.

These findings point to the need for improved UI understanding and even better higher resolution incorporation.

8. Detailed Qualitative Analysis

Figures 2, 3, and 4 provide a comprehensive qualitative comparison of different methods for predicting the next action in various task scenarios. These figures illustrate the performance of AFRAgent alongside CogAgent [7] and Auto-GUI [18], highlighting the nuances of their action prediction capabilities across diverse challenges.

In the second task, where the web page displays a news section and the target action is to click on the "All" text, AFRAgent correctly identified and executed the action. This demonstrates its strong OCR capabilities, accurately recognizing and interacting with text elements. In comparison, CogAgent incorrectly clicked on the search bar, while Auto-GUI prematurely marked the task as complete without performing the action, revealing deficiencies in text recognition and task understanding. For the third task, the target was to click on the Google icon. AFRAgent successfully executed this action, demonstrating precise visual grounding and spatial awareness. Conversely, both CogAgent and Auto-GUI erroneously clicked on the Files icon, underscoring their limitations in distinguishing between similar but contextually different elements. In the fifth task, AFRAgent accurately clicked on the search icon, showcasing its consistent ability to localize and interact with specific UI elements. Competing methods failed in this instance, misinterpreting the action or failing to perform it altogether. Similarly, in the sixth task, AFRAgent excelled by successfully clicking on the Google icon within the layout, whereas both CogAgent and Auto-GUI failed, selecting incorrect actions. This further emphasizes AFRAgent's robust understanding of UI layouts and action prediction under spatial constraints. In the eighth task, AFRAgent predicted the action type as "Input text," effectively leveraging trajectory-level information to infer the correct action. This highlights its ability to incorporate contextual cues and prior steps in the task sequence, outperforming methods that focus more on immediate visual or textual inputs.

Overall, these examples underscore AFRAgent's superior performance across various qualitative benchmarks. Its ability to combine OCR, visual grounding, and trajectory-level reasoning enables more accurate and contextually appropriate predictions, setting it apart from competing methods in challenging multimodal tasks.

Grad-CAM Analysis for Individual Crops In the main paper, to assess the impact of feature enrichment, we visualize attention from the 18th layer of the vision encoder using Grad-CAM, averaged across the entire LLM response and encoder attention heads. Additionally, in Fig. 5, we present Grad-CAM visualizations for individual crops processed by the Q-Former, demonstrating that the model effectively attends to the relevant regions within each crop, accurately focusing on the areas essential for the given task.

References

[1] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023. 5

[2] Yuxiang Chai, Siyuan Huang, Yazhe Niu, Han Xiao, Liang Liu, Dingyu Zhang, Peng Gao, Shuai Ren, and Hongsheng Li. Amex: Android multi-annotation expo dataset for mobile gui agents. *arXiv preprint arXiv:2407.17490*, 2024. 5

[3] Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. 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, Bangkok, Thailand, 2024. Association for Computational Linguistics. 5

[4] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale

Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023. 2, 5

[5] Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th annual ACM symposium on user interface software and technology*, pages 845–854, 2017. 4

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. 5

[7] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14281–14290, 2024. 5, 6

[8] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10965–10975, 2022. 4

[9] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 5

[10] Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based GUI agent. *CoRR*, abs/2408.00203, 2024. 5

[11] Xinbei Ma, Zhuosheng Zhang, and Hai Zhao. CoCo-agent: A comprehensive cognitive MLLM agent for smartphone GUI automation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9097–9110, Bangkok, Thailand, 2024. Association for Computational Linguistics. 4, 5

[12] Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy P Lillicrap. Androidinthewild: A large-scale dataset for android device control. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. 2, 5

[13] Liangtai Sun, Xingyu Chen, Lu Chen, Tianle Dai, Zichen Zhu, and Kai Yu. Meta-gui: Towards multi-modal conversational agents on mobile gui. *arXiv preprint arXiv:2205.11029*, 2022. 4, 5

[14] Qinzhuo Wu, Weikai Xu, Wei Liu, Tao Tan, Jianfeng Liu, Ang Li, Jian Luan, Bin Wang, and Shuo Shang. Mobilevlm: A vision-language model for better intra-and inter-ui understanding. *arXiv preprint arXiv:2409.14818*, 2024. 5

[15] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1192–1200, 2020. 5

[16] An Yan, Zhengyuan Yang, Wanrong Zhu, Kevin Lin, Linjie Li, Jianfeng Wang, Jianwei Yang, Yiwu Zhong, Julian McAuley, Jianfeng Gao, et al. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation. *arXiv preprint arXiv:2311.07562*, 2023. 5

[17] Yi-Fan Zhang, Qingsong Wen, Chaoyou Fu, Xue Wang, Zhang Zhang, Liang Wang, and Rong Jin. Beyond llava-hd: Diving into high-resolution large multimodal models. *arXiv preprint arXiv:2406.08487*, 2024. 2

[18] Zhuosheng Zhang and Aston Zhang. You only look at screens: Multimodal chain-of-action agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3132–3149, Bangkok, Thailand, 2024. Association for Computational Linguistics. 4, 5, 6

[19] Zhizheng Zhang, Wenxuan Xie, Xiaoyi Zhang, and Yan Lu. Reinforced ui instruction grounding: Towards a generic ui task automation api. *arXiv preprint arXiv:2310.04716*, 2023. 4

[20] Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. 4

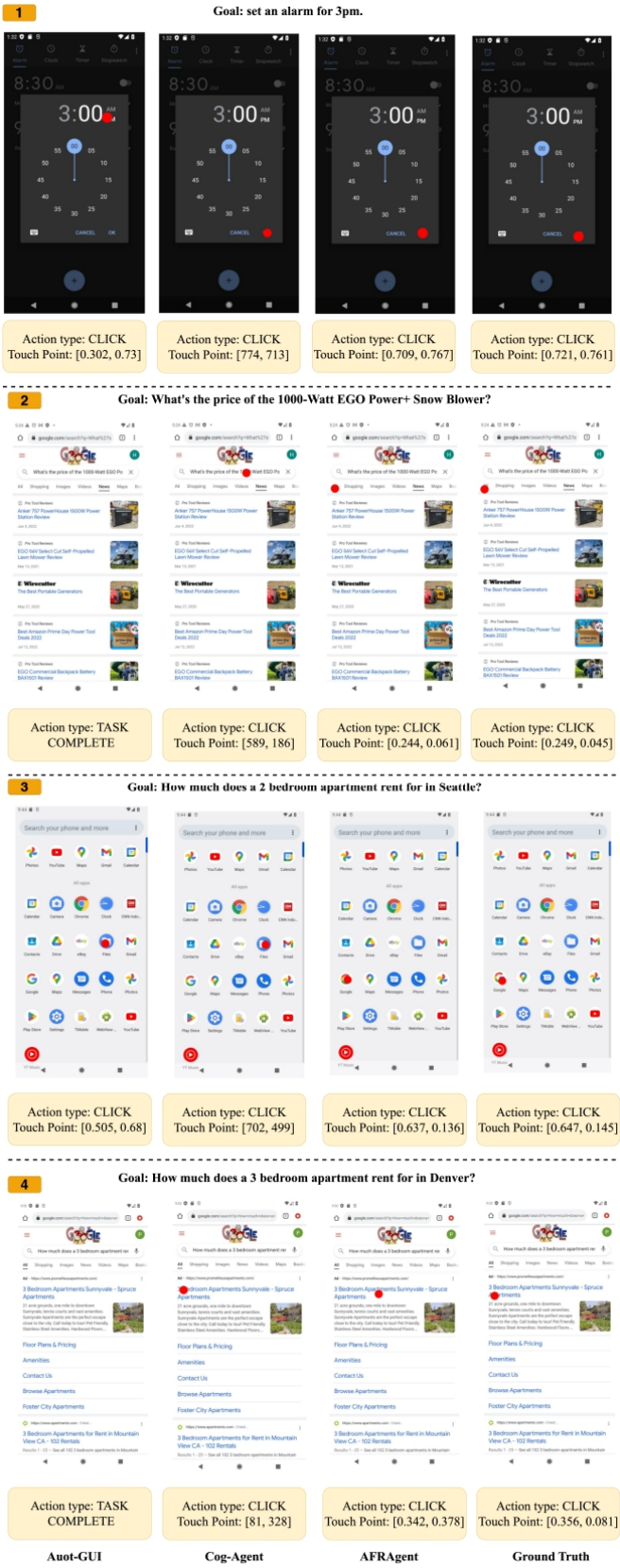


Figure 2. Qualitative comparison of different agents on UI automation

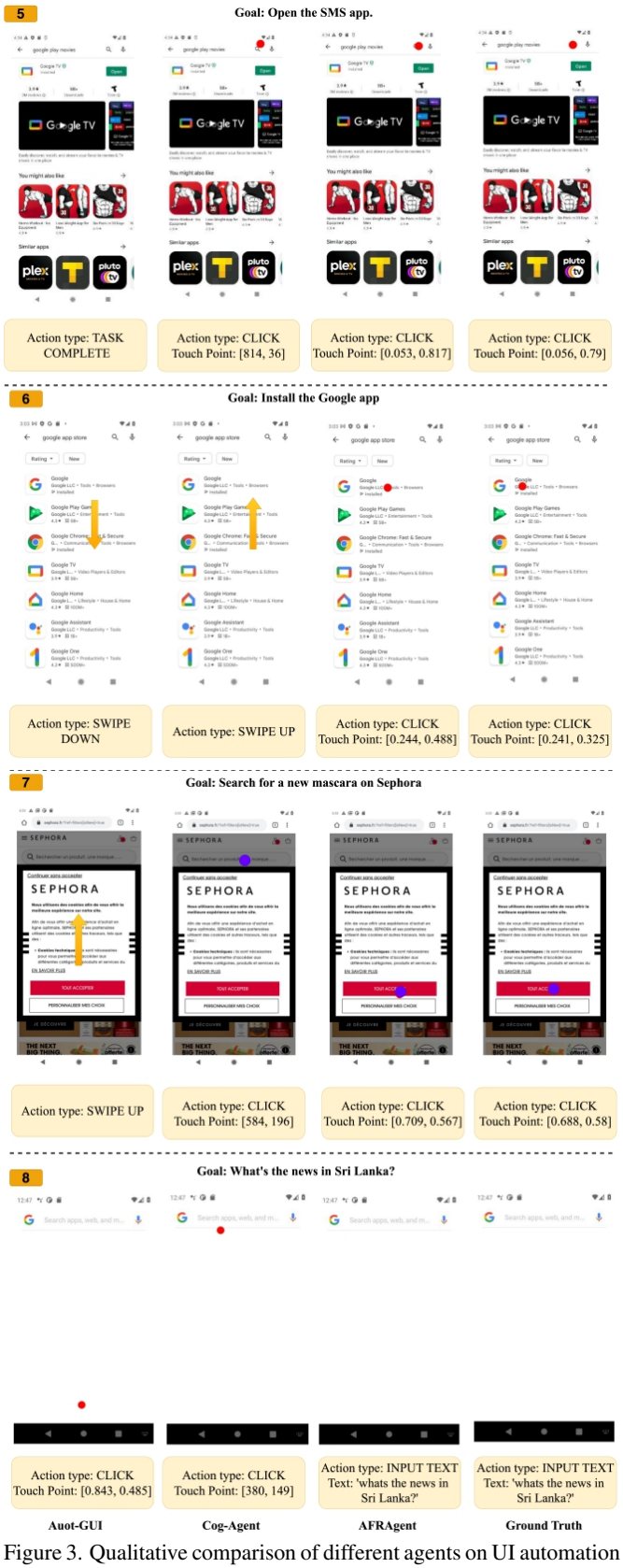


Figure 3. Qualitative comparison of different agents on UI automation

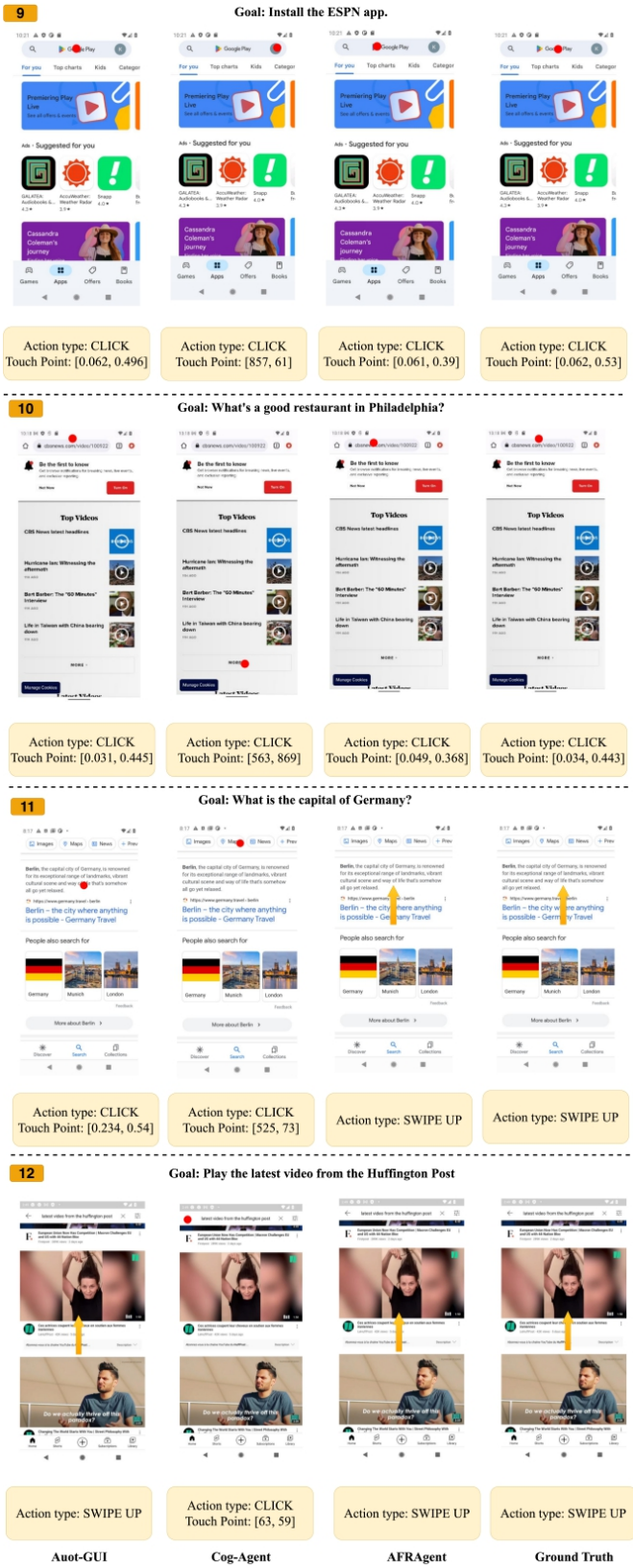


Figure 4. Qualitative comparison of different agents on UI automation

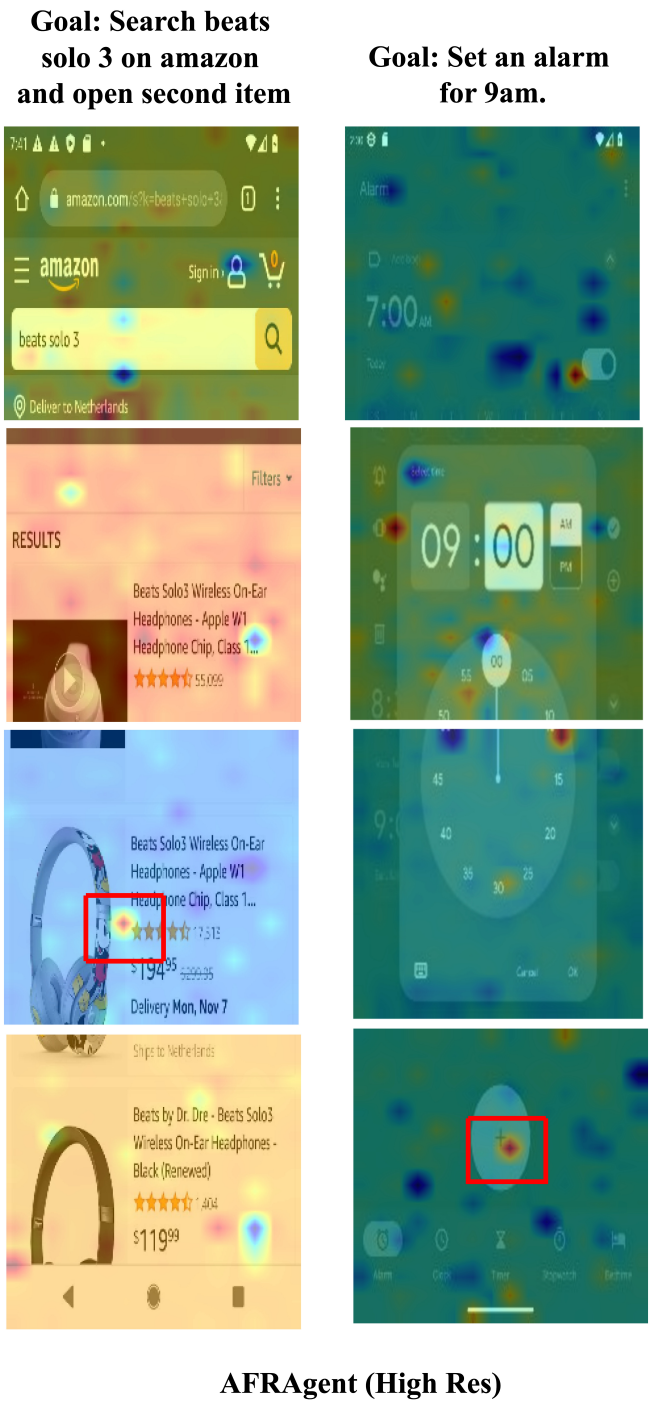


Figure 5. Grad-CAM visualization of individual crops showcasing the high resolution attention maps