# 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

## Abstract

*There is a growing demand for mobile user interface (UI) automation, driven by its broad applications across industries. With the advent of visual language models (VLMs), GUI automation has progressed from generating text-based instructions for humans to autonomously executing tasks, thus optimizing automation workflows. Recent approaches leverage VLMs for this problem due to their ability to 1) process on-screen content directly, 2) remain independent of device-specific APIs by utilizing human actions (e.g., clicks, typing), and 3) apply real-world contextual knowledge for task understanding. However, these models often have trouble accurately identifying widgets and determining actions due to limited spatial information in vision encoder features. Additionally, top-performing models are often large, requiring extensive training and resulting in inference delays. In this work, we introduce AFRAgent , an instruct-BLIP-based multimodal architecture that achieves superior performance in GUI automation while being less than **one-fourth** the size of its nearest competitor. To enhance image embeddings in the large language model (LLM) pipeline, we propose an adaptive feature renormalization-based (a token-level affine transformation) technique that effectively enriches low-resolution image embeddings and fuses high-resolution details. We evaluate AFRAgent on Meta-GUI and AITW benchmarks, establishing a new state-of-the-art baseline for smartphone automation.[1] The code is available at* https://github.com/neerajanand321/AFRAgent
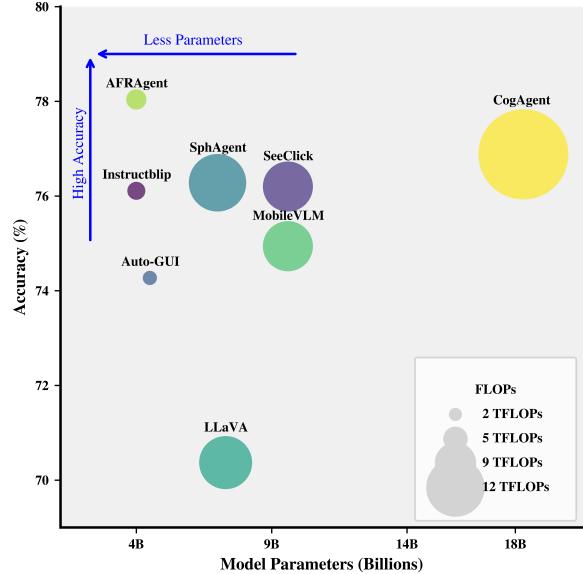
Figure 1. AFRAgent outperforms existing methods in UI automation on AITW dataset, achieving state-of-the-art results with significantly fewer parameters and reduced computational overhead.

## 1. Introduction

Advancements in deep learning have made previously challenging tasks achievable, particularly with the emergence of transformer-based architectures. Progress in natural language processing (NLP), computer vision, and multimodal models has significantly enhanced our capacity to address complex tasks. Among these tasks, GUI automation is crucial due to its wide-ranging real-world applications and substantial economic impact. A model must accomplish a user-defined objective, such as scheduling calendar events or preparing presentations on specific topics on a desktop or mobile operating system. While this problem has been explored even before the emergence of large language models (LLMs) [47], the development of LLMs has led to notable improvements in action prediction accuracy [57], strategic planning for sequential steps [22, 46], and generalization across various environments and applications [46]. Previous approaches, such as verbalization of screen content combined with prompt engineering and fine-tuning [12, 39], have leveraged LLMs to predict necessary subsequent actions. However, since user interfaces are fundamentally

---

* equal contribution

† Contact: neeraja,rishabhj,soha@adobe.com

[1] Project Webpage: Website

intended for visual engagement, text-only methods prove inadequate compared to multimodal models incorporating spatial information.

Large visual language models (VLMs) have demonstrated effectiveness across diverse tasks, including optical character recognition (OCR) [4], image captioning [1], visual grounding [34, 52], question answering [3], and reasoning [9]. Recently, VLMs have also demonstrated promising potential in automating user-defined workflows. For instance, Auto-GUI [56] introduced a chain-of-thought approach that combines past action steps with future planning to determine the following action. SeeClick [11] emphasized the importance of identifying icons and widgets on the screen, pretraining their network with GUI-grounded data to enhance alignment for automation tasks. CogAgent [16] incorporated both low and high-resolution vision encoders to capture fine-grained details from mobile screenshots, while CoCo-Agent [31] provided layout information as input to the LLM and simplified action prediction into a more semantic structure to improve performance.

Despite these advances, current methods have several limitations. First, many approaches rely on external tools like OCR and icon detectors [41, 54] to transform the interface into text-based inputs for language models. This dependence can introduce inefficiencies, as lengthy inputs increase processing time, risk exceeding model input limits, and may suffer from information loss or conversion errors. Additionally, extracting layout information from third-party applications can be challenging. In contrast, AFRA-gent efficiently processes mobile screenshots directly without needing external tools. Secondly, current approaches rely on computationally intensive models with parameter counts ranging from 7B [31] to 18B [16], making edge deployment difficult despite its critical importance for preserving privacy in UI interactions. Our approach significantly reduces this computational burden by utilizing a more efficient 4B parameter model while maintaining competitive performance, making edge deployment more feasible. Thirdly, existing UI automation methods typically use either Llava-style projections [26, 31] or learnable query embeddings [28] to extract screen information. However, recent research [17, 55] suggests that a combination of direct projection and learnable query embeddings provides enhanced results. Lastly, existing VLMs often incorporate high-resolution information through methods that substantially increase memory and computational overhead, such as increasing the size of the vision encoder [10], splitting images into multiple crops [27] resulting in additional LLM tokens, or using cross-attention to integrate information at each layer of the LLM [16]. To address these issues, we propose an Adaptive Feature Renormalization (AFR) technique that effectively merges image embeddings with learned query projections without adding a signifi-

cant computational burden. Additionally, we enhance the InstructBLIP architecture by integrating AFR to combine low-resolution and high-resolution features via Q-Former, demonstrating the adaptability of our approach. In summary, our work makes the following technical contributions

- We propose Adaptive Feature Renormalization, a novel technique to enrich the learned query projections with the image embeddings effectively.
- We adapt InstructBlip architecture to fuse high resolution information into the low-resolution embeddings via AFR.
- Our 4-billion parameter model achieves state-of-the-art results on key GUI benchmarks, significantly surpassing larger models and demonstrating superior performance(Fig 1).
- We provide comprehensive quantitative and qualitative evidence, along with ablation studies, to demonstrate the effectiveness of each component.

## 2. Related Works

*Agents* are systems that perceive their environment, make decisions, and pursue goals [47]. Recent advances in language models have improved agents' adaptability and generalization [32, 45], enabling memory, planning, and applications in domains like software development [23], scientific research [6], and web-based tasks, especially autonomous GUI navigation using (M)LLMs.

**Autonomous GUI Navigation** Autonomous navigation in graphical user interfaces (GUIs) leverages these advancements to enable agents to interact with and control complex digital environments. LLM-based methods such as WebAgents [15], Mind2Web [12], WebArena [58], m-BASH [39], and AutoWebGLM [21] utilize textual representations of screens (e.g., HTML, accessibility trees, OCR) along with task goals to guide navigation. These models effectively process and summarize verbose structures, but they struggle with challenges like HTML verbosity, high token counts, and loss of fine-grained visual details, limiting their ability to fully replicate human-like interactions with GUIs. In contrast, multimodal approaches combine textual and visual information to improve agent performance. Models like WebGum [14], AppAgent [53], Mobile-Agent [44], Auto-GUI [56], UI-TARS[35], and MobileVLM [48] rely solely on screenshots, leveraging visual data to navigate and interact with GUIs naturally. OmniParser [30] uses GPT-4V and introduces a robust screen parsing technique, achieving state-of-the-art zero-shot performance. CogAgent [16] incorporates a specialized high-resolution visual module with alignment pre-training, while CoCo-Agent [31] combines both textual information and screenshots. These multimodal agents overcome many limitations of text-based methods by providing richer context and a deeper understanding of UI elements, though they face challenges in efficiency due to large model sizes and limited high-resolution

visual capabilities, reducing effectiveness in tasks requiring precise visual understanding.

**Low-Resolution Feature Enrichment Strategies** Various strategies have been developed to pass information from vision encoders to LLMs, with the most widely used being LLaVA-style projections and the Q-Former-based Instruct-BLIP architecture. Zhang et al. [55] demonstrated that while InstructBLIP performs well on tasks like ScienceQA [29], which involves multiple-choice question answering across various science topics, it underperforms on tasks that require deeper reasoning, such as POPE [24], MME [51], and GQA [19], where LLaVA-style direct projection has proven more effective. To balance these strengths, Zhang et al. [55] proposed a Soft Mixture of Experts (MoE) model with a gating network that learns optimal weights to combine Q-Former and LLaVA-style features. Despite incorporating learnable noise to prevent over-reliance on either feature type, this approach faced challenges with training instability and yielded limited improvements in both performance and generalizability. We also compare our method with the MoE approach in Sec 6.Alternatively, BLIVA [17] uses a projection layer to input both Q-Former embeddings and direct projections into the LLM, but this significantly increases token count and computational costs. Inspired by previous advances in computer vision [2, 18, 20, 33], we introduce Adaptive Feature Renormalization (AFR), which enhances Q-Former features with additional learned embeddings. AFR retains the computational efficiency of the Q-Former approach while capturing the benefits of direct projection, balancing performance with resource efficiency

**High-Resolution Feature Enrichment Strategies** To merge high-resolution information, existing methods [10, 27, 38] often divide the image into smaller regions, process each independently through the vision encoder, and then pass all resulting embeddings to the LLM via a projection layer. Although effective, this approach can become computationally expensive with large LLMs due to the increase in token count. For example, InternVL [10] introduces up to 3k visual tokens during training and 10k during inference, adding substantial computational burden. To mitigate these costs, SliME [55] uses a local compression layer with a Q-Former and a text-guided patch-selection router, while CogAgent [16] employs a high-resolution cross module that injects features via cross-attention layers. AFR, in contrast, provides a simpler, compute-efficient solution by enriching existing features with high-resolution details without excessive overhead.

## 3. Methodology

**UI Automation Task:** GUI automation task requires the agent to execute a series of actions to achieve a specific objective. Formally, consider the GUI automation task $T$, defined as a free-form text instruction provided by the user. The current state of the user interface is captured as a screenshot, denoted by $S_t$, where $t$ represents the time step in the sequence of actions. The agent interacts with the GUI environment through a series of actions $A = \{a_1, a_2, \ldots, a_t\}$, where each action $a_t \in \mathcal{A}_{\text{UI}}$ (where $\mathcal{A}_{\text{UI}} = \{$click, type, scroll, swipe, press back, press home, press enter, task complete$\}$) [56] is taken at time step $t$, transitioning the system from state $S_{t-1}$ to state $S_t$. Given the task $T$, the current state $S_t$, and the history of actions taken up to time step $t$, denoted as $H_t = \{a_1, a_2, \ldots, a_t\}$, the goal of the model is to predict the following action $a_{t+1}$. This can be formalized as learning a policy $\pi(a_{t+1}|T, S_t, H_t)$ which outputs the probability distribution over possible action space $\mathcal{A}_{\text{UI}}$ given the task, current state, and action history. The process continues iteratively until the agent reaches a terminal state $S_g$, where $S_g$ satisfies the conditions defined by the task $T$.

**Architecture of AFRAgent:** We leverage InstructBLIP backbone for the task of GUI Automation. Consider the image encoder $E$, the QueryFormer $Q$, and the large language model $L$ as the core components of the architecture of AFRAgent $M_{\text{Agent}}$. The image encoder $E$ takes the current screenshot $S_t$ as input and outputs a set of visual embeddings $I_t$ by dividing the input image into patches. Formally,

$$I_t = [i_1, i_2, \ldots, i_N] = E(S_t), \quad (1)$$

where $i_n \in \mathbb{R}^{d_I}$, $n = 1, 2, \ldots, N$ and $N$ is the number of visual patches of $S_t$. The Q-Former $Q$ generates instruction-aware visual representations by attending to both visual embeddings and task instructions. Let $\mathcal{T}_Q = [e_{q_1}^{(0)}, e_{q_2}^{(0)}, \ldots, e_{q_M}^{(0)}]$ represent the learnable query token embeddings, where each query token $t_{q_m} \in \mathbb{R}^{d_Q}$ has dimension $d_Q$. Additionally, we embed the goal task instruction $T$ and the action history $H_t$ with a learnable embedding layer $g_Q(T, H_t) = [e_1^{(0)}, e_2^{(0)}, \ldots, e_{L_T}^{(0)}]$, where each token embedding $e_l^{(0)} \in \mathbb{R}^{d_Q}$, and $L_T$ is the number of tokens in the verbalized goal task instruction and the action history. We concatenate the query token embeddings $\mathcal{T}_Q$ with the goal task and action history embeddings $g_Q(T, H_t)$, forming a combined input to the Q-Former denoted by the following equation.

$$X_Q^{(0)} = [E_Q^{(0)}; E_T^{(0)}] = [e_{q_1}^{(0)}, e_{q_2}^{(0)}, \ldots, e_{q_M}^{(0)};$$
$$e_1^{(0)}, e_2^{(0)}, \ldots, e_{L_T}^{(0)}] \quad (2)$$

where $X_Q \in \mathbb{R}^{(M+L_T) \times d_Q}$. The Q-Former then takes this concatenated input $X_Q$ along with the visual embeddings $I_t = [i_1, i_2, \ldots, i_N]$, and employs a cross attention layer and attends to the user task embeddings using self-attention layers followed by a feed-forward network (FFN), repeated
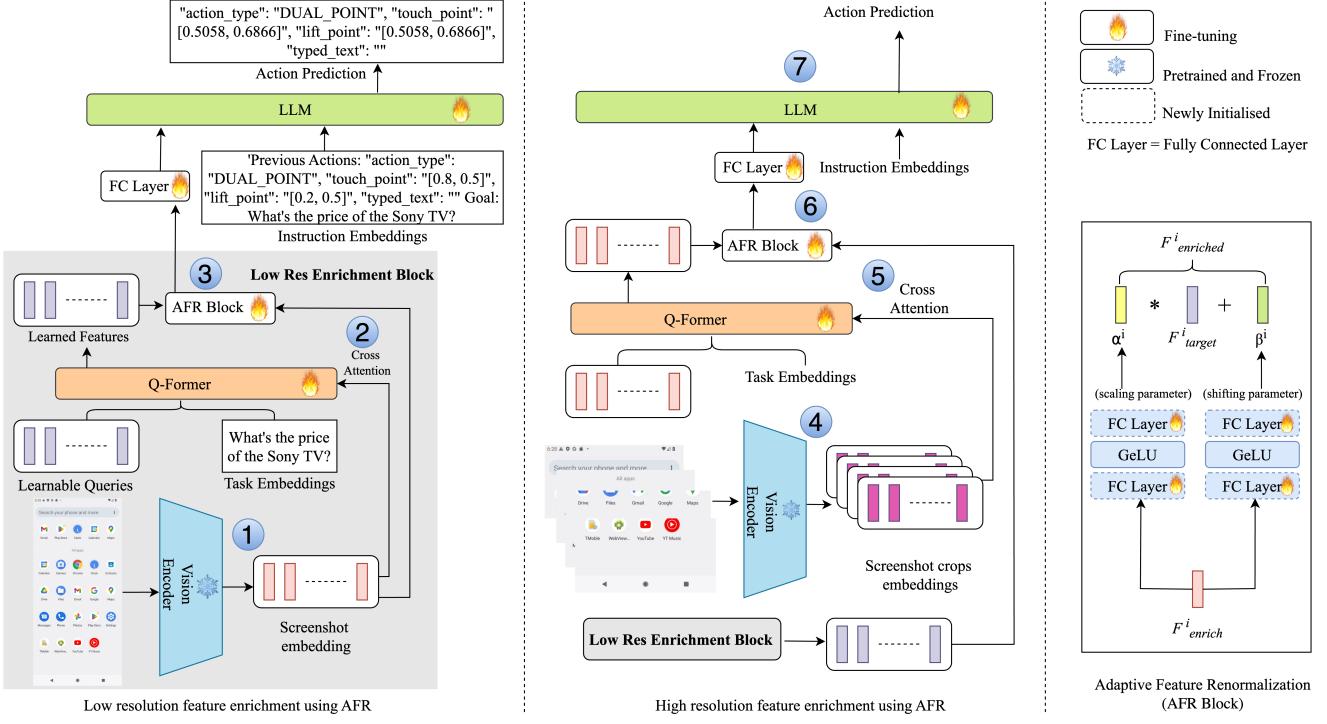
Figure 2. Core architectural improvements in AFRAgent on the AITW dataset. **Low-resolution enrichment with AFR:** Image embeddings are generated via BLIP (1) and cross-attended with the Q-Former (2). These control scaling and shifting parameters that renormalize Q-Former features (3). **High-resolution enrichment with AFR:** The image is divided into crops, processed through BLIP (4) to obtain tokens, which cross-attend with the Q-Former (5) and enrich the low-res features (6). Final enriched features are projected to the LLM for action prediction (7).

for $Z$ layers. These operations help extract the relevant image features useful for the particular task from the original image embeddings. The following equations describe the operation of Q-former.

$$E_Q^{(z-1)}, E_T^{(z-1)} = SelfAttention(E_Q^{(z-1)}, E_T^{(z-1)}),$$
$$E_{Q_{\text{cross}}}^{(z)} = CrossAttention(E_Q^{(z-1)}, I_t)),$$
$$E_Q^{(z)}, E_T^{(z)} = FFN(E_{Q_{\text{cross}}}^{(z)}), FFN(E_T^{(z-1)}), \quad (3)$$

where $z = 1, 2, \cdots, Z$ denote the layer index. The query token representations thus obtained at the last layer ($Z$) is aligned to the LLM embedding space through a learnable projection layer. Further, we concatenate the projected Q-Former output from $E_Q^{(Z)}$ i.e., the output after the projection layer, with the goal task and action history embeddings $g_L(T, H_t)$ of the underlying LLM $L$, forming the final input to the LLM:

$$X_{LLM} = [e_{q_1}^Z, e_{q_2}^Z, \ldots, e_{q_M}^Z; e_1^L, e_2^L, \ldots, e_{L_T}^L], \quad (4)$$

where $X_{LLM} \in \mathbb{R}^{(M+L_T) \times d_L}$. Given this as the input, the LLM $L$ outputs the next predicted action $\hat{a}_{t+1} =$ $L(X_{LLM})$. Given the predicted action and the ground-truth action, we enforce the standard cross-entropy loss to train the agent $M_{\text{Agent}}$(Fig 2).

**Adaptive Feature Renormalization:** As discussed in Sec 2, combining LLAVA-style features with Q-Former features demonstrates superior performance on various multimodal reasoning benchmarks. Inspired by this observation, we introduce an Adaptive Feature Renormalization (AFR) technique to enhance the input representation for the LLM. AFR leverages both image embeddings and Q-Former outputs to enrich the final LLM input representation of the image.

The AFR block takes as input two sets of features: 1) the enriching features, denoted as $F_{\text{enrich}} \in \mathbb{R}^{M \times d_Q}$, and 2) the target features to be enriched, denoted as $F_{\text{target}} \in \mathbb{R}^{M \times d_Q}$. Formally, AFR computes two learnable parameters, $\alpha$ and $\beta$, where $\alpha, \beta \in \mathbb{R}^{M \times d_Q}$, that scale and shift $F_{\text{target}}$ based on $F_{\text{enrich}}$, producing the enriched features $F_{\text{enriched}} \in \mathbb{R}^{M \times d_Q}$. This operation is depicted by the fol-

lowing equation:

$$\alpha, \beta = FFN_\alpha(F_{\text{enrich}}), FFN_\beta(F_{\text{enrich}}),$$
$$F_{\text{enriched}} = (\alpha \odot F_{\text{target}}) \oplus \beta, \tag{5}$$

**Motivation behind AFR:** Affine transformations have been widely utilized in prior work for feature modulation. For instance, StyleGAN [18, 20] employs Adaptive Instance Normalization (AdaIN) to transfer style content in GAN-based image generation, while SPADE [33] applies Spatial Adaptive Normalization for semantic image synthesis. Despite their diverse applications, these methods highlight the versatility of affine transformations in injecting information through feature modulation. By scaling and shifting the target features based on the enriching features, we hypothesize that the information from the global enriching features is effectively injected into each target token, effectively renormalizing them and leading to a more informative representation for downstream tasks. We also verify this intuition using Grad-Cam [37] visualizations in Sec 6. Next, we describe the integration of Low-Res Image Embeddings and High-Res representations using AFR to obtain enriched features for UI automation.

**Low-resolution Enrichment using AFR:** We inject the image embeddings into the Q-Former representations using the AFR operation. As defined in Sec 3, the image embeddings $I_t$ for a given UI screenshot $S_t$ serve as the enriching features $F_{\text{enrich}}$, while the Q-Former representations of the query tokens $E_Q^{(Z)}$ act as the target features $F_{\text{target}}$. We compute the scaling and shifting parameters $\alpha^{\text{Image}}$ and $\beta^{\text{Image}} \in \mathbb{R}^{M \times d_Q}$, where $M$ is the number of output query token representations and $d_Q$ is their dimensionality. The enriched Q-Former features, denoted by $E_Q^{\text{Image}} \in \mathbb{R}^{M \times d_Q}$, are computed as follows:

$$\alpha^{\text{Image}}, \beta^{\text{Image}} = FFN_\alpha(I_t), FFN_\beta(I_t),$$
$$E_Q^{\text{Image}} = (\alpha^{\text{Image}} \odot E_Q^{(Z)}) \oplus \beta^{\text{Image}}, \tag{6}$$

In our case, patch embeddings or learned queries capture fine-grained semantic details of specific regions on the smartphone screen. By modulating these core features before feeding them into the LLM, we inject valuable information while preserving the original strengths of Instruct-BLIP's pre-trained features.

**High-resolution Enrichment using AFR:** Through empirical evidence, we found that most errors in action recognition for UI automation stem from the limited input resolution of the image encoder. This highlights the need for high-resolution features within the visual language model

(VLM) to enhance accuracy. Existing methods [10, 27, 38] often address this by dividing the image into smaller sections, processing each section independently through the vision encoder, and passing all resulting embeddings to the LLM via a projection layer. This setting can become prohibitively expensive computationally in the case of large LLMs due to the additional token count. Therefore, we leverage the AFR Block in order to fuse the high resolution information onto the Q-Former representation enriched with image embeddings. For a screenshot $S_t$, we obtain the high resolution image $\tilde{S}_t$ and split it into $C$ crops horizontally $S_t = [S_t^{(1)}, S_t^{(2)}, \cdots, S_t^{(C)}]$ to better preserve the aspect ratio during resizing. We pass each of the crops to the vision encoder to obtain the image embeddings denoted by $\tilde{I}_t^{(c)} = [\tilde{i}_1^{(c)}, \tilde{i}_2^{(c)} \cdots, \tilde{i}_N^{(c)}]$. Since the query tokens defined earlier were used to process the low resolution image, to capture fine-grained information via the high resolution image embeddings, we introduce a new set of learnable query token embeddings denoted by $\tilde{\mathcal{T}}_Q = [\tilde{e}_{q_1}^{(0)}, \tilde{e}_{q_2}^{(0)}, \cdots, \tilde{e}_{q_M}^{(0)}]$ whereas the goal task and action history embeddings remain the same $g_Q(T, H_t)$. Using the same Q-Former $Q$, we obtain the high-resolution output query token representation $\tilde{E}_Q^Z$ using the following equation.

$$\tilde{E}_Q^{(Z)}, \tilde{E}_T^{(Z)} = Q(\tilde{E}_Q^{(0)}; E_T^{(0)}; [\tilde{I}_t^{(1)}, \tilde{I}_t^{(2)} \cdots \tilde{I}_t^{(C)}]), \tag{7}$$

where we feed the new query token embeddings for high resolution, the task and action history embeddings, and concatenate the image embeddings of each of the crops as the input to the Q-Former. Using the AFR Block, we use $\tilde{E}_Q^{(Z)}$ as $F_{\text{enrich}}$ and $E_Q^{\text{Image}}$ as $F_{\text{target}}$ to obtain $E_Q^{\text{High}}$, denoted by the following equation.

$$\alpha^{\text{High}}, \beta^{\text{High}} = FFN_\alpha(\tilde{E}_Q^{(Z)}), FFN_\beta(\tilde{E}_Q^{(Z)}),$$
$$E_Q^{\text{High}} = (\alpha^{\text{High}} \odot E_Q^{\text{Image}}) \oplus \beta^{\text{Image}}, \tag{8}$$

After obtaining the final enriched features $E_Q^{\text{High}}$, we apply the projection layer on top and pass the visual features, task instruction and action history to the LLM for output generation. The model is trained using the language modelling loss on the predicted output as discussed in Sec 3.

## 4. Experiments

This section describes our experimental setup, covering datasets, baselines, and evaluation metrics. Additional details are in the supplementary material. We evaluate our approach on two key datasets: Meta-GUI and AITW, which span diverse mobile GUI interaction tasks.

**Datasets** **Meta-GUI** [40] consists of 1k episodes with 18k steps across 11 applications in 6 domains, including

weather, calendar, and search. Each episode is structured as a multi-turn dialogue, with agents interacting with users and confirming or discussing next steps. We follow CoCo-Agent [31] by incorporating dialogue and action history as input. Additional details can be found in supplementary. **AITW** [36] is the largest dataset for smartphone GUIs, containing 715k episodes and 30k unique instructions across Android apps and websites. It includes multi-step categories (GoogleApps, Install, WebShopping, General) and a single-step category (Single), with each episode comprising a goal instruction and a sequence of screenshot-action pairs.

**Experiment Settings**  For Meta-GUI, we use the same settings as Meta-GUI [40] and CoCo-Agent [31]. We follow the standard (80/10/10) training/validation/test split of AITW used by Auto-GUI [56], SphAgent [7], and CogAgent[16] for fair comparison. Importantly, we fine-tune our model **without any UI pretraining**, unlike CogAgent [16], SeeClick [11], and SphAgent [7], demonstrating the robustness of our approach to outperform these methods without prior GUI grounding.

**Baselines and Implementation**  We compare AFRAgent with diverse baselines spanning different architectures and training paradigms, including multimodal methods using textual (HTML, accessibility trees) and visual inputs (screenshots). Key baselines include **MM-Navigator** [50], **LLaMA-2** [43], **Auto-GUI** [56], **CogAgent** [16], **CoCo-Agent** [31], **MobileVLM** [48], and **SphAgent** [7]. On Meta-GUI, we also compare with **LayoutLM** [49], **BERT**, **m-BASH** [40], and **CoCo-Agent**. LLaMA-2 and BERT are pure language models, while MM-Navigator uses few-shot GPT-4V. The rest are fine-tuned multimodal models. Notably, CogAgent [16] utilizes high-resolution ViT tokens that directly cross-attend with its LLM, whereas SphAgent [7] feeds high-resolution image tokens directly into the LLM. AFRAgent is trained for 12 epochs using Adam (lr=$5 \times 10^{-5}$), with 257 query tokens ($M$), Q-Former hidden size 768 ($d_Q$), 256 image patches ($N$), and LLM hidden size 2048. Screenshots are divided into 4 sections ($C$), and 8 past actions ($H_t$) are used as history. The AFR block has two fully connected layers with GeLU activation. All experiments were conducted on 8xA100 80GB GPUs. We found that AFR introduces no noticeable instability during training, and the model converges steadily without requiring special initialization. We observed consistent improvements in performance starting from the early epochs, indicating effective feature fusion from the AFR mechanism.

**Evaluation Metrics**  We use action matching [36] as our main metric to assess alignment with ground-truth actions. Click actions are considered matches if they are within a
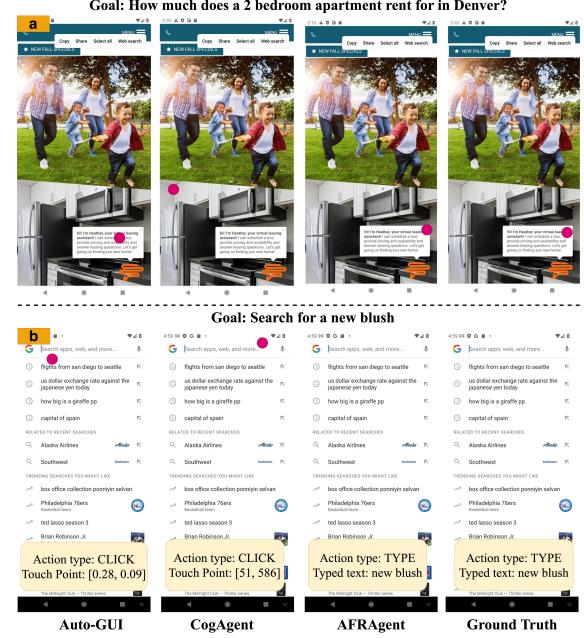


Figure 3. Qualitative comparison of various methods for predicting next action. The figure shows the scenarios where the agent is a) required to click the close icon of a pop-up window and b) type "new blush" in the search bar. While other methods failed, our method successfully predicted the correct coordinate for clicking and inputted the correct text, highlighting the superior performance of our approach.

14% screen distance or the same bounding box as the target UI element, while scrolls match if they share the same primary scroll axis (vertical or horizontal). Exact match accuracy is used for other action types, except *typed_text* and *dialogue* response. On AITW, predicted text is correct if it contains the label; on Meta-GUI, we use Action Completion Rate(CR) for accuracy and F1 score for input text matching. An action is correct only if all components (e.g., scroll direction, typed text) are accurately predicted.

## 5. Results

We assess AFRAgent on two benchmark datasets, Meta-GUI and AITW, spanning a broad range of UI automation tasks. On the Meta-GUI dataset (Tab. 1), AFRAgent establishes a new state-of-the-art, significantly improving upon existing methods across multiple key metrics. Specifically, we achieve a **2.56%** increase in action completion rate and a substantial **3.34%** improvement in target item prediction accuracy. Additionally, our model excels in input text matching metrics, underscoring its enhanced task comprehension and efficiency despite having fewer parameters.

AITW, a widely used large-scale Android benchmark, serves as a comprehensive testbed for evaluating our model.

| Method | Params | Act. Type | Item Acc. | Direction Acc. | Utter. (BLEU) | Input (F1) | Input (EM) | Action (CR) |
|---|---|---|---|---|---|---|---|---|
| LayoutLM [49] | 343M | 82.22 | 71.98 | 94.87 | 50.43 | 90.56 | 83.04 | 67.76 |
| LayoutLM$_{v2}$ | 426M | 85.60 | 64.38 | 92.95 | 58.20 | 70.76 | 47.37 | 64.48 |
| BERT [13] | 340M | 87.52 | 82.84 | 93.59 | 62.19 | 97.24 | 93.57 | 78.42 |
| LLaVA [26] | 7.3B | 87.47 | 77.49 | 98.18 | 67.24 | 96.06 | - | 76.27 |
| LLaVA$_{w/history}$ | 7.3B | 91.68 | 81.23 | 97.62 | 66.57 | 96.93 | - | 81.08 |
| m-BASH [40] | 340M | 90.80 | 85.90 | 96.42 | 63.11 | 94.23 | 91.23 | 82.74 |
| CoCo-Agent [31] | 7.3B | 92.59 | 91.72 | **98.39** | 65.90 | 96.15 | - | 88.27 |
| AFRAgent | 4B | **93.28** | **95.06** | 97.02 | **67.6** | **97.94** | **94.44** | **90.83** |

Table 1. Comparative performance on Meta-GUI benchmark evaluating Completion rate (CR), action type, item/direction accuracy, text metrics (F1, exact match), and response quality (BLEU). AFRAgent achieves state-of-the-art accuracy across most metrics, showing efficiency with fewer parameters and improved input handling.

| Method | Params | General | Install | GoogleApps | Single | WebShop. | Overall |
|---|---|---|---|---|---|---|---|
| **Structured Layout Setting** | | | | | | | |
| CoCo-Agent[31] | 7.3B | 70.96 | 81.46 | 76.45 | 91.41 | 75.00 | 79.05 |
| AFRAgent | 4B | 71.62 | 80.81 | 76.26 | 90.78 | 75.10 | 78.92 |
| **Pure Multimodal Setting** | | | | | | | |
| MM-Navigator*[50] | - | 43.01 | 46.14 | 49.18 | 78.29 | 48.18 | 52.96 |
| LLaMA-2 [43] | 7B | 28.56 | 35.18 | 30.99 | 27.35 | 19.92 | 28.40 |
| Auto-GUI [56] | 4.5B | 68.24 | 76.89 | 71.37 | 84.58 | 70.26 | 74.27 |
| LLaVA [26] | 7.3B | 58.93 | 72.41 | 70.81 | 83.73 | 65.98 | 70.37 |
| MobileVLM [48] | 9.6B | 69.58 | 79.87 | 74.72 | 81.24 | 71.70 | 74.94 |
| InstructBlip [28] | 4B | 70.66 | 79.59 | 73.05 | 84.99 | 72.26 | 76.11 |
| SeeClick [11] | 9.6B | 67.6 | 79.6 | **75.9** | 84.6 | 73.1 | 76.2 |
| SphAgent [7] | 7B | 68.2 | 80.5 | 73.3 | 85.4 | **74** | 76.28 |
| ShowUI [25] | 2B | 63.9 | 72.5 | 69.7 | 77.5 | 66.6 | 70 |
| CogAgent [16] | 18.3B | 65.38 | 78.86 | 74.95 | **93.49** | 71.73 | 76.88 |
| AFRAgent | 4B | **70.67** | **80.89** | 74.16 | 91.06 | 73.27 | **78.01** |

Table 2. Comparison of AFRAgent and prior models on AITW with unified training. AFRAgent excels in the multimodal setting and performs competitively in the structured layout setting, achieving strong action accuracy with lower compute and memory. * indicates few-shot setting.

We report results under two settings: **Structured Layout Setting** augments text inputs with layout data from Icon-Net [42], incurring high computational cost (∼1600 extra tokens/screen), an additional inference pass, and susceptibility to error propagation, limiting generalization across UIs. **Purely Multimodal Setting** evaluates VLMs using only screenshots and natural language goals, avoiding external tools or layout data. It has emerged as the preferred evaluation method, reflecting a model's intrinsic visual-semantic understanding. Despite being **78% smaller**, AFRAgent achieves superior overall accuracy in the multimodal setting, outperforming prior methods across most tasks. It also outperforms CoCo-Agent [31] on general and webshop splits in the structured setting, with significantly lower memory overhead.

Figure 3 further illustrates qualitative gains, demonstrating enhanced spatial awareness and task comprehension achieved by AFRAgent 's novel AFR block integration. In row (a), AFRAgent accurately identifies the relevant icon due to enhanced spatial perception from our high-resolution embedding integration. In row (b), AFRAgent precisely identifies the correct action and input text, highlighting improved task comprehension from rich spatial embeddings.

# 6. Analysis

| LVLMs | Params | Mobile | | Desktop | | Web | | Average |
|---|---|---|---|---|---|---|---|---|
| | | Text | Icon/Widget | Text | Icon/Widget | Text | Icon/Widget | |
| MiniGPT-v2*[8] | 7B | 8.4% | 6.6% | 6.2% | 2.9% | 6.5% | 3.4% | 5.7% |
| Qwen-VL*[4] | 9.6B | 9.5% | 4.8% | 5.7% | 5.0% | 3.5% | 2.4% | 5.2% |
| GPT-4V* | - | 22.6% | 24.5% | 20.2% | 11.8% | 9.2% | 8.8% | 16.2% |
| Fuyu[5] | 8B | 41.0% | 1.3% | 33.0% | 3.6% | 33.9% | 4.4% | 19.5% |
| CogAgent[16] | 18.3B | 67.0% | 24.0% | **74.2%** | 20.0% | **70.4%** | 28.6% | 47.4% |
| SeeClick[11] | 9.6B | 78.0% | 52.0% | 72.2% | 30.0% | 55.7% | 32.5% | 53.4% |
| AFRAgent | **4B** | **78.52%** | **53.14%** | 72.62% | **31.44%** | 64.51% | **33.44%** | **55.61%** |

Table 3. Performance comparison across Mobile, Desktop, and Web scenarios for Text and Icon/Widget elements on *ScreenSpot*[11].* represent non GUI-specific methods

| Method | TFLOPs | Latency |
|---|---|---|
| MobileVLM[48] | 8.82 | 2.16 s |
| CogAgent[16] | 11.86 | 3.42 s |
| InstructBlip* | 3.19 | 0.63 s |
| AFRAgent$_{\text{Low-res}}$ | 3.2 | 0.78 s |
| AFRAgent$_{\text{High-res}}$ | 5.47 | 1.24 s |

Table 4. AFRAgent achieves notably lower FLOPs and operates with substantially faster inference times compared to previous methods. * represent 257 tokens in Qformer

**Generalization to Web/Desktop:** We pre-trained our model on SeeClick [11] data before testing the cross-platform generalization on *ScreenSpot* [11] dataset. This dataset evaluates VLM's ability to locate text and icons across platforms by measuring click accuracy. We crop screenshots in the original $4 \times 1$ grid for mobile training and a $2 \times 2$ grid for web/desktop data. AFRAgent performs better than Fuyu [5], CogAgent, SeeClick, etc. across all platforms, as observed in Tab 3. However, CogAgent performs better in text inputs due to its more extensive pretraining data and greater model size.

**Grad-CAM Analysis**  To evaluate the impact of feature enrichment, we visualize attention from the 18th layer of the vision encoder using Grad-CAM [37] averaged across the entire LLM response and encoder attention heads. As seen in Fig. 4, AFRAgent effectively renormalizes features, allowing the model to focus on key regions after enrichment. For example, in row (b), the network correctly identifies the target product on Amazon following high-res fusion.

**Computational Complexity Analysis**  The compute overhead introduced by enriching low- and high-resolution features using AFR is analyzed in Table 4. AFRAgent $_{\text{Lowres}}$ runs nearly $3 \times$ faster with 54% fewer FLOPs than CogAgent, thanks to its lightweight AFR-based low-resolution enrichment. While $AFR_{HighRes}$ has $\sim 70\%$ cost over $AFR_{LowRes}$, both remain far cheaper than CogAgent, and $AFR_{LowRes}$ already improves over baselines(Additional details in supplementary). High-resolution enrichment increases computation due to additional ViT passes and cross-attention between high-resolution features and the Q-Former. The complexity of this cross-attention operation is:

$$\mathcal{O}\left((M + L_{\text{T}}) \times C \times N_{\text{N}} \times H_{\text{cross}} \times d_{\text{Q}}\right)$$

where $M$, $L_{\text{T}}$, $C$, $N_{\text{N}}$, $H_{\text{cross}}$, and $d_{\text{Q}}$ are query or text tokens, image crops, tokens per image, cross-attention heads, and Q-Former embedding size, respectively. Notably, our approach is much more efficient than CogAgent, which applies cross-attention to a large LLM decoder, while we leverage the compact Q-Former.
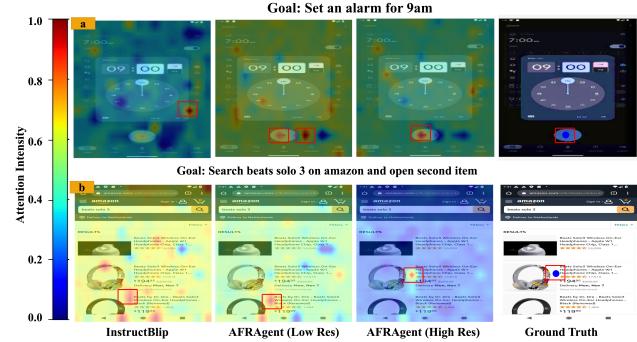


Figure 4. Grad-CAM visualizations showing AFRAgent's improved attention to task-relevant UI elements.

| Dataset | Residual | MoE | AFR$_{\text{Low-res}}$ | highResProj | Qwen2-VL | AFR$_{\text{High-res}}$ |
|---|---|---|---|---|---|---|
| FLOPs (T) | **3.2** | 3.54 | **3.2** | 6.46 | 17.08 | **5.47** |
| Size (Billion) | 4.03 | 4.03 | 4.03 | **4.03** | 8.29 | **4.03** |
| General | 69.61 | 69.75 | **70.2** | 69.74 | 70.15 | **70.91** |
| Single | 85.13 | 85.25 | **85.37** | 86.17 | 85.21 | **86.3** |

Table 5. Comparison of AFR with prior fusion approaches in both low-res and high-res settings.

**Comparison of Fusion Strategies**  We evaluate the effectiveness of AFR as a fusion strategy in low-resolution and high-resolution settings on the General and Single subsets of AITW. For low-resolution fusion, we compare AFR against a residual approach, defined as $F_{enriched} = F_{target} + \text{MLP}(F_{enrich})$, and the Mixture of Experts (MoE) method [55]. For the high-resolution setting, we assess Qwen2-VL using the *AnyRes* configuration and by directly providing high-resolution query tokens after projection to InstructBLIP (highResProj) for training (Table 5). Our results show that AFR consistently outperforms these fusion strategies, highlighting its robustness and effectiveness in multimodal feature integration. Furthermore, $AFR_{High\_res}$ offers superior computational efficiency compared to highResProj due to the lower token count (514 vs. 257 respectively) in LLM owing to feature fusion. Due to restricted space, we offer fine-grained analysis of Tab 5 and results for comprehensive training on AITW for the AFR method in the appendix.

## 7. Conclusion

We introduce AFRAgent , a lightweight, high-performance multimodal large language model that outperforms existing methods across multiple benchmarks. Our approach employs a novel Adaptive Feature Renormalization (AFR) technique to enrich global and high-resolution image features efficiently. We provide a comprehensive evaluation of AFRAgent through quantitative and qualitative analyses, demonstrating its effectiveness in UI automation.

# References

[1] Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. Nocaps: Novel object captioning at scale. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8948–8957, 2019. 2

[2] Badour Albahar, Jingwan Lu, Jimei Yang, Zhixin Shu, Eli Shechtman, and Jia-Bin Huang. Pose with style: Detail-preserving pose-guided image synthesis with conditional stylegan. *ACM Transactions on Graphics (TOG)*, 40(6):1–11, 2021. 3

[3] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015. 2

[4] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023. 2, 7

[5] Rohan Bavishi, Erich Elsen, Curtis Hawthorne, Maxwell Nye, Augustus Odena, Arushi Somani, and Sagnak Tasırlar. Introducing our multimodal models, 2023. *URL https://www. adept. ai/blog/fuyu-8b*, 2, 2023. 7, 8

[6] Daniil A Boiko, Robert MacKnight, and Gabe Gomes. Emergent autonomous scientific research capabilities of large language models. *arXiv preprint arXiv:2304.05332*, 2023. 2

[7] 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. 6, 7

[8] Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, and Mohamed Elhoseiny. Minigpt-v2: large language model as a unified interface for vision-language multi-task learning. *arXiv preprint arXiv:2310.09478*, 2023. 7

[9] Yangyi Chen, Karan Sikka, Michael Cogswell, Heng Ji, and Ajay Divakaran. Measuring and improving chain-of-thought reasoning in vision-language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 192–210, Mexico City, Mexico, 2024. Association for Computational Linguistics. 2

[10] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024. 2, 3, 5

[11] 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. 2, 6, 7, 8

[12] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 2

[13] 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. 7

[14] Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. 2

[15] Izzeddin Gur, Hiroki Furuta, Austin V. Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. 2

[16] 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. 2, 3, 6, 7, 8

[17] Wenbo Hu, Yifan Xu, Yi Li, Weiyue Li, Zeyuan Chen, and Zhuowen Tu. Bliva: A simple multimodal llm for better handling of text-rich visual questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2256–2264, 2024. 2, 3

[18] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017. 3, 5

[19] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019. 3

[20] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 3, 5

[21] Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, and Jie Tang. Autowebglm: A large language model-based web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 5295–5306, New York, NY, USA, 2024. Association for Computing Machinery. 2

[22] Sunjae Lee, Junyoung Choi, Jungjae Lee, Munim Hasan Wasi, Hojun Choi, Steven Y. Ko, Sangeun Oh, and Insik Shin. Explore, select, derive, and recall: Augmenting llm with human-like memory for mobile task automation, 2024. 1

[23] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008, 2023. 2

[24] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 292–305, Singapore, 2023. Association for Computational Linguistics. 3

[25] Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Stan Weixian Lei, Lijuan Wang, and Mike Zheng Shou. Showui: One vision-language-action model for gui visual agent. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19498–19508, 2025. 7

[26] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 2, 7

[27] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024. 2, 3, 5

[28] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 2, 7

[29] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022. 3

[30] Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based GUI agent. *CoRR*, abs/2408.00203, 2024. 2

[31] 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. 2, 6, 7

[32] Osonde A Osoba, Raffaele Vardavas, Justin Grana, Rushil Zutshi, and Amber Jaycocks. Policy-focused agent-based modeling using rl behavioral models. *arXiv preprint arXiv:2006.05048*, 2020. 2

[33] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019. 3, 5

[34] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *CoRR*, abs/2306.14824, 2023. 2

[35] Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025. 2

[36] 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. 6

[37] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 5, 8

[38] Baifeng Shi, Ziyang Wu, Maolin Mao, Xin Wang, and Trevor Darrell. When do we not need larger vision models? In *European Conference on Computer Vision*, pages 444–462. Springer, 2025. 3, 5

[39] Liangtai Sun, Xingyu Chen, Lu Chen, Tianle Dai, Zichen Zhu, and Kai Yu. META-GUI: Towards multimodal conversational agents on mobile GUI. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6699–6712, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. 1, 2

[40] Liangtai Sun, Xingyu Chen, Lu Chen, Tianle Dai, Zichen Zhu, and Kai Yu. Meta-gui: Towards multimodal conversational agents on mobile gui. *arXiv preprint arXiv:2205.11029*, 2022. 5, 6, 7

[41] Srinivas Sunkara, Maria Wang, Lijuan Liu, Gilles Baechler, Yu-Chung Hsiao, Jindong Chen, Abhanshu Sharma, and James W. W. Stout. Towards better semantic understanding of mobile interfaces. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5636–5650, Gyeongju, Republic of Korea, 2022. International Committee on Computational Linguistics. 2

[42] Srinivas Sunkara, Maria Wang, Lijuan Liu, Gilles Baechler, Yu-Chung Hsiao, Jindong Chen, Abhanshu Sharma, and James W. W. Stout. Towards better semantic understanding of mobile interfaces. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5636–5650, Gyeongju, Republic of Korea, 2022. International Committee on Computational Linguistics. 7

[43] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 6, 7

[44] Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *CoRR*, abs/2401.16158, 2024. 2

[45] Xiaoling Wang and Housheng Su. Completely model-free rl-based consensus of continuous-time multi-agent systems. *Applied Mathematics and Computation*, 382:125312, 2020. 2

[46] Hao Wen, Yuanchun Li, Guohong Liu, Shanhui Zhao, Tao Yu, Toby Jia-Jun Li, Shiqi Jiang, Yunhao Liu, Yaqin Zhang, and Yunxin Liu. Empowering LLM to use smartphone for intelligent task automation. *CoRR*, abs/2308.15272, 2023. 1

[47] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents: theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995. 1, 2

[48] 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. 2, 6, 7, 8

[49] 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. 6, 7

[50] 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. 6, 7

[51] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *CoRR*, abs/2306.13549, 2023. 3

[52] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 69–85. Springer, 2016. 2

[53] Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. Appagent: Multimodal agents as smartphone users. *CoRR*, abs/2312.13771, 2023. 2

[54] Xiaoyi Zhang, Lilian De Greef, Amanda Swearngin, Samuel White, Kyle Murray, Lisa Yu, Qi Shan, Jeffrey Nichols, Jason Wu, Chris Fleizach, et al. 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, 2021. 2

[55] 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, 3, 8

[56] 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. 2, 3, 6, 7

[57] Longtao Zheng, Rundong Wang, and Bo An. Synapse:

Leveraging few-shot exemplars for human-level computer control. *CoRR*, abs/2306.07863, 2023. 1

[58] 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. 2