Towards an Inclusive Mobile Web: A Dataset and Framework for Focusability in UI Accessibility

Ming Gu
College of Computer Science and Technology, Zhejiang University
Hangzhou, Zhejiang, China
guming444@zju.edu.cn

Lei Pei
School of Software Technology, Zhejiang University
Ningbo, Zhejiang, China
pei_lei@zju.edu.cn

Sheng Zhou*
Zhejiang Key Laboratory of Accessible Perception and Intelligent Systems, Zhejiang University
Hangzhou, Zhejiang, China
zhousheng_zju@zju.edu.cn

Ming Shen
School of Software Technology, Zhejiang University
Ningbo, Zhejiang, China
shenming2023@zju.edu.cn

Yuxuan Wu
School of Software Technology, Zhejiang University
Ningbo, Zhejiang, China
wux521@zju.edu.cn

Zirui Gao
College of Computer Science and Technology, Zhejiang University
Hangzhou, Zhejiang, China
gaozirui@zju.edu.cn

Ziwei Wang
College of Computer Science and Technology, Zhejiang University
Hangzhou, Zhejiang, China
wangziwei98@zju.edu.cn

Shuo Shan
Ant Group
Hangzhou, Zhejiang, China
shanshuo.ss@antgroup.com

Wei Jiang
Ant Group
Hangzhou, Zhejiang, China
jonny.jw@antgroup.com

Yong Li
Ant Group
Hangzhou, Zhejiang, China
liyong.liy@antgroup.com

Jiajun Bu
College of Computer Science and Technology, Zhejiang University
Hangzhou, Zhejiang, China
bjj@zju.edu.cn

Abstract

The rapid growth of mobile web technologies has revolutionized how people manage daily activities, emphasizing the critical need for accessible mobile user interfaces (UIs) that accommodate users with disabilities and situational impairments. Current AI-driven UI understanding methods show promise but primarily target general UI modeling, neglecting nuanced, user-centric accessibility requirements. To bridge this gap, we first conducted a formative study with 12 visually impaired participants. Our study uncovers selective-accessible issues, a new class of accessibility challenges requiring finer granularity and selective focus on UI components, which existing methods largely overlook. Our findings also reveal that the severity of issues varies across interaction stages, with earlier stages posing a more significant impact. Building on these insights, we propose a comprehensive framework of three accessibility stages: focusability, information, and functionality (FIF), encompassing 12 sub-tasks under 3 overarching tasks. Identifying UI element focusability prediction (UFP) as a pivotal yet underexplored task within FIF, hindered by the absence of dedicated datasets, we introduce a new dataset (NOS) with 117,480 annotated components addressing accessibility issues comprehensively. To further enhance UFP, we introduce Graph-based UI Focusability Prediction (GIFT), a method leveraging graph neural networks to model UFP-targeted UI relationships. User studies validate the dataset's quality, while experiments show GIFT 's effectiveness in improving UFP outcomes. Our code and datasets are publicly available to support further web inclusivity advancements at https://github.com/eaglelab-zju/NOS.

CCS Concepts

• **Human-centered computing** → **Accessibility technologies**.

Keywords

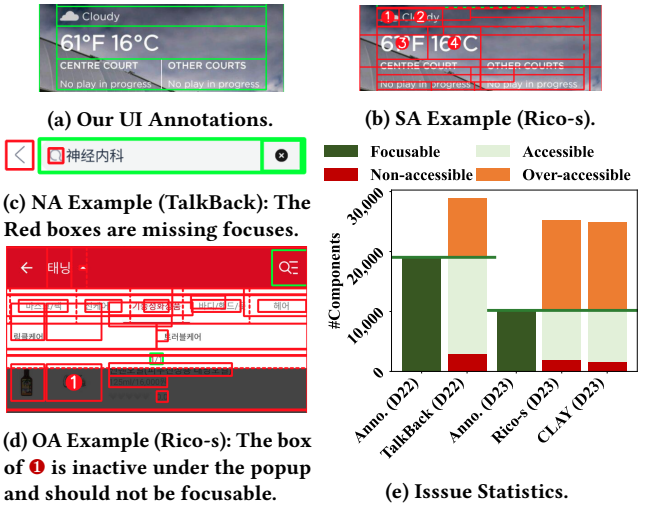
Mobile Web, Web Accessibility, UI Accessibility, Web Inclusivity

ACM Reference Format:
Ming Gu, Lei Pei, Sheng Zhou, Ming Shen, Yuxuan Wu, Zirui Gao, Ziwei Wang, Shuo Shan, Wei Jiang, Yong Li, and Jiajun Bu. 2025. Towards an Inclusive Mobile Web: A Dataset and Framework for Focusability in UI Accessibility. In *Proceedings of the ACM Web Conference 2025 (WWW '25), April 28-May 2, 2025, Sydney, NSW, Australia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3696410.3714523

*Corresponding author.

WWW '25, Sydney, NSW, Australia

ACM ISBN 979-8-4007-1274-6/25/04
https://doi.org/10.1145/3696410.3714523



(a) Our UI Annotations. (b) SA Example (Rico-s).

(c) NA Example (TalkBack): The Red boxes are missing focuses.

(d) OA Example (Rico-s): The box of ❶ is inactive under the popup and should not be focusable.

(e) Isssue Statistics.

Figure 1: (a)-(d) Examples of UI Accessibility Issues. (e) Comparison of Existing Datasets with Our Annotations (Anno. D22 & D23). The NA and OA ratios of <TalkBack, Rico-s [26], CLAY [24]> are <10.3%, 7.9%, 7.0%> and <33.8%, 59.5%, 59.0%>.

1 Introduction

The rapid evolution of mobile web technologies has significantly transformed how people manage daily activities, highlighting the critical need for universally accessible mobile user interfaces (UIs) [44]. However, many UIs still lack accessibility for individuals with disabilities and situational challenges [23, 28], and accessibility challenges of mobile web applications are reportedly more severe than those of native applications on mobile platforms as assistive services cannot analyze webView elements properly [32]. *UI accessibility*, which seeks to ensure that interfaces are perceivable, understandable, and operable for users with different requirements, is crucial for achieving web inclusivity [31]. Nevertheless, existing UI analysis methodologies often fall short by concentrating on general characteristics, such as the type identification of UI elements [35], instead of addressing comprehensive accessibility user needs [1]. These shortcomings highlight the pressing need for a user-centered approach to UI modeling that accommodates the full spectrum of accessibility requirements, thereby advancing web inclusivity.

To gain deep insights into the challenges and requirements of UI accessibility, we conducted a formative study involving 12 blind and low-vision (BLV) participants who regularly use mobile applications in their daily lives. The study uncovered critical accessibility issues and needs across multiple stages and granularity levels. **First**, participants highlighted that the UI interaction stages (namely reaching, comprehending, and operating) significantly impact the issue severity, even when the same issue is encountered. Accessibility issues arising in earlier stages, such as the initial reaching and focusing phase, pose greater barriers, underscoring the importance of resolving issues at this critical entry point. **Second**, in addition to the well-documented non-accessible (NA) [50] and over-accessible (OA) [28] issues, participants identified a new category: selective-accessible (SA) issues. While NA and OA issues refer to inaccessible or redundant UI elements, SA issues emphasize appropriate granularity in UI accessibility, requiring a selective focus on components. *Participants stressed that an ideal UI accessibility solution must involve a unified approach to addressing NA, OA, and SA issues.*

Building on the findings of our formative study, we proposed a novel framework structured around three core access stages: Focusability, Information, and Functionality (FIF). This framework encompasses 12 sub-tasks with eight dedicated to focusability and two each focused on information and functionality. These sub-tasks align with three overarching goals: UI Focusability Prediction (UFP), UI Accessibility Information Generation (UAG), and UI Functionality Repairment (UFR). Our analysis reveals that while UAG and UFR tasks can be effectively addressed using established AI-generated content (AIGC) and software engineering methods, *the UFP task has been critically overlooked despite its foundational role in addressing accessibility issues at the initial focusing stage.* Furthermore, an investigation into the focus results of Android default screen reader, TalkBack [17], alongside existing UI datasets such as Rico-semantic (Rico-s)[27] and CLAY[24], uncovers considerable issues with focusability. For instance, as shown in Figure 1e, the TalkBack, Rico-s, and CLAY datasets exhibit 10.3%, 7.9%, and 7.0% NA issues and 33.8%, 59.5%, and 59.0% OA issues, respectively. These results are consistent with *the dissatisfaction expressed by participants in our formative study regarding UI accessibility.* More specifically, Figure 1b illustrates the incorrect focus granularity of SA issues in the Rico-s dataset, contrasted with ground-truth annotations in Figure 1a. Likewise, Figure 1c shows NA issues in TalkBack's focus results, while Figure 1d demonstrates OA issues in Rico-s.

Therefore, addressing UFP is essential for advancing UI accessibility efforts, but the absence of applicable UI datasets and methodological frameworks hinders progress. To overcome these challenges, we propose a new dataset that addresses NA, OA, and SA issues (NOS) in UI focusability. This dataset contains annotations for 117,480 UI components across 2,000 pages, validated through a user evaluation where both BLV and sighted users confirmed its alignment with accessibility requirements. We also introduce a novel Graph-based UI FocusabiliTy Prediction (GIFT) method, leveraging graph structures to extract UFP-targeted contextual UI relationships. GIFT is the first approach to address the UFP task, emphasizing the critical role of heterophilic relationships and hierarchical individuality for effective UFP. Experimental and ablation studies validated GIFT's effectiveness in tackling UFP tasks.

The main contributions of this work are:

- **Formative Study and Framework for UI Accessibility**: We conducted a formative study on UI accessibility, identified a new class of selective-accessible issues, and established a unified framework FIF addressing UI accessibility for the mobile web.
- **New UI Accessibility Task**: We introduced UFP as a critical yet underexplored task in UI accessibility, highlighting its impact on subsequent accessibility stages of information and functionality.
- **New Dataset**: We developed a novel dataset with annotations addressing NA, OA, and SA issues, validated through user evaluations and supporting further web inclusivity advancements.
- **Method and Experiments**: We proposed the first method for UFP, Graph-based UI Focusability Prediction (GIFT), demonstrating the importance of graph-based approaches for the UFP task.

2 Formative Study

To better understand the real issues and needs of UI accessibility, we conducted a formative study involving 12 BLV participants. Using semi-structured interviews, we addressed the following questions:

- **Q1**: What is the mobile web UI experience of BLV users, and what are the most critical issues?
- **Q2**: Are there new accessibility issues that have not been noticed?
- **Q3**: What is the ideal UI experience for BLV users?

2.1 Method

We recruited 12 BLV participants through online outreach and compensated each with 25 USD for a 45-minute in-person study. All participants use assistive technologies (ATs) like TalkBack daily to interact with mobile applications and represent various professions, including teaching (P1-2), students (P3-7), software engineering (P8-10), and massage therapy (P11-12). Eight representative mobile applications (apps) were selected from diverse categories essential for daily life, such as online shopping and renting. Participants were provided access to these apps three days before the interview and were encouraged to explore them using TalkBack, noting any accessibility issues encountered. Additionally, we sampled example pages from each app that represented typical accessibility issues, focusing on two categories: ❶ NA: Non-accessible issues [2, 33] occur when UI components are inaccessible to users relying on ATs. ❷ OA: Over-accessible issues [28] arise when UI components that should not be accessible to either BLV or sighted users are improperly exposed to ATs.

During the interviews, participants interacted with and assessed these sampled pages, which helped evaluate the identified issues and prompted discussions about their broader app experiences. Participants shared additional challenges encountered during daily use and their expectations for an ideal UI experience. For more details of the formative study, please refer to Appendix E.

2.2 Findings

Several key findings were derived from the interviews.

Interaction Stages Affecting Issue Manifestation (Q1). Participants frequently reported encountering both NA and OA issues, but the severity of their impact varied depending on the interaction stage. Participant P3 explained, "*NA can result in missing information or functionality. Missing unimportant information is acceptable, but overlooking a focusable component prevents users from discovering certain functions, hindering normal use. For example, once when filling out registration information, I repeatedly navigated the page but couldn't find the submit button*". Similarly, P9 shared: "*Some sharing buttons on a page were inaccessible, denying me the same rights as others to use the feature*". Synthesizing participants' insights revealed two key points: (1) For NA issues, the impact varies across three interaction stages: accessing UI elements, retrieving information, and operating functionality. The inability to access elements has the most significant impact, followed by non-functional elements. Missing information has the least impact, as it can sometimes be inferred from context. (2) For OA issues, as UI elements are always accessible, their impact is generally less severe. However, excessive redundancy or irrelevant information can hinder comprehension and significantly increase interaction time.

New Issues Related to UI Granularity (Q2). The study revealed a previously underexplored category of issues tied to the granularity of UI elements, which were overlooked in prior research. These granularity-related issues affect how UI components are grouped or divided for accessibility. Participant P6 highlighted an example where fine-grained grouping in some pages was implemented unreasonably: "*The statement 'Buy one, get one free for $10' was split into two focuses: 'Buy one, get one free' and 'for $10'. This division could mislead users into believing the second item is free regardless of price or that the deal only applies if the total cost is $10*". P12 added: "*Overly large groups of content can also be problematic. Listening to extensive content in a single focus is exhausting. If I accidentally move to the next focus midway through listening and then attempt to return, I must start over from the beginning, which is frustrating.*" P1 highlighted another issue: "*Some pages group regions containing text, buttons, and other components into a single focusable element. If the group does not implement button functionality properly, it prevents users from operating the functionality.*" These insights identify a distinct category of accessibility issues beyond NA and OA. These issues, which we term ❸ SA: Selective-accessible issues, represent accessibility problems arising from the selective focus behavior of UI components. They stem from improper granularity in UI component access, affecting both the transmission of UI information and the exposure of UI functionality.

Ideal User Experience Expectation Beyond NA and OA (Q3). Participants unanimously agreed that ideal accessibility extends beyond resolving NA and OA issues. As one participant summarized: "*On the basis of eliminating NA and OA, an ideal design provides multiple levels of focus granularity and allows users to switch between levels to better understand combined information at different layers.*"

3 FIF: Unified Framework of UI Accessibility

Building upon the findings of our formative study, we propose a novel standardized framework for UI accessibility that categorizes issues into three interaction stages: Focusability reaching, Information comprehending, and Functionality actioning (FIF). Initially, UI components must ensure proper focusability to achieve reachable accessibility. Subsequently, they should deliver accurate information for comprehensible accessibility (e.g., UI content, logic, changes, etc.). Finally, ATs must be able to execute component functionalities to meet actionable accessibility requirements.

3.1 Sub- and Overarching tasks of FIF

3.1.1 Sub-tasks. The FIF framework comprises 12 sub-tasks, systematically labeled from **[T1]** to **[T12]**, ensuring a comprehensive and structured approach. Among these, reachable accessibility is the largest category, consisting of eight sub-tasks ([T1]–[T8]) and addressing NA ([T1]), OA ([T2]–[T4]), and SA ([T5]–[T8]) issues. In contrast, comprehensible and actionable accessibility each involves two sub-tasks, focusing on the completeness of information delivery ([T9]–[T10]) and functionality exposure ([T11]–[T12]).

Focusability: Reachable Accessibility. Reachable accessibility addresses focusability issues in UI components.

[T1] ***Non-accessible Component Recovery***. Components that support independent interaction or convey standalone semantic information should be identified as accessible.

[T2] ***Duplicate Component Pruning***. When multiple consecutive nodes represent the same UI region, the most information-rich node should be marked as focusable, while redundant nodes are designated as non-focusable.

[T3] ***Invisible Component Filtering***. Nodes without visual presence or semantic relevance should be marked as non-focusable.

[T4] ***Inactive Component Filtering***. Visible nodes not within the active view hierarchy, such as those obscured by overlays or hidden behind pop-ups, should be designated as non-focusable.

[T5] ***Semantic Component Aggregation***. For components that have granular focus requirements but for which the information is better conveyed as a whole, only the outermost node should be marked as focusable, while the inner nodes should be non-focusable.

[T6] ***Complex Component Segmentation***. In cases where numerous inner nodes convey complex information, these nodes may be segmented and marked as focusable to enhance user experience.

[T7] ***Minimal-functional Component Focusability***. Inner nodes capable of independent interaction should be marked as focusable rather than falsely grouped.

[T8] ***Hierarchical Consistency Enforcement***. Consistency should be maintained in labeling nodes across the same hierarchical level (e.g., list items within a single list).

Information: Comprehensible Accessibility. Comprehensible accessibility emphasizes ensuring that information conveyed by UI components is meaningful and interpretable for BLV users.

[T9] ***Component Captioning***. For components, especially those grouping multiple items, an appropriate caption should be assigned for better semantic clarity.

[T10] ***Component Ordering***. Each component should be correctly ordered based on its contextual importance or role.

Functionality: Actionable Accessibility. Actionable accessibility pertains to ensuring that the functionality of UI components is fully accessible and operable through assistive technologies.

[T11] ***Function Alignment***. The functionalities exposed to BLV users who access the mobile web through assistive technologies should align with those exposed to non-disabled users.

[T12] ***Inclusive Functional Alternatives***. Functional Accessibility has the unique task of alternative functional support, which is to provide alternative forms of certain functionality that is inaccessible to people with certain disabilities. For example, text CAPTCHAs need to have alternative forms like voice CAPTCHAs so that visually impaired users can pass them.

3.1.2 Overarching tasks. Three overarching tasks are further proposed to standardize the handling of the 12 sub-tasks: (1) UI Focusability Prediction (**UFP**), a newly introduced task, addresses the NA, OA, and SA issues of UI focusability as a unified prediction problem. (2) UI Accessibility Information Generation (**UAG**) focuses on generating accessible semantic information like component caption and order. (3) UI Functionality Repairment (**UFR**) repairs the functional issues through software engineering methods.

UI Focusability Prediction (UFP). The eight sub-tasks ([T1]-[T8]) associated with NA, OA, and SA can be formulated as a prediction problem. Fundamentally, these tasks seek to determine whether a component should be focusable to AT users. While the NA and OA issues ([T1]-[T4]) can be framed as classification problems, selective accessibility presents additional complexity. First, responses are not necessarily unique for components intended for selective AT exposure. For instance, tasks like [T5] (semantic component aggregation) and [T6] (complex component segmentation) may be interrelated, with choices between aggregation or segmentation resembling a recommendation or ranking problem due to varying user preferences. Therefore, we unify the first eight tasks under the UI Focusability Prediction (UFP) task.

UI Accessibility Information Generation (UAG). Following UFP's determination of component accessibility, the tasks ([T9]-[T10]) remain to accurately convey content and function-related information to AT users. We formalize these accessibility requirements, which pertain to content, functionality, logical structure, and more, as a generation task termed UI Accessibility Information Generation (UAG).

UI Functionality Repairment (UFR). Actionable problems are managed primarily via the ActionList attribute in code [28], making it largely a task of logistical assessment within software engineering (SE), rather than a challenge requiring complex knowledge inference of visual semantic or relational understanding.

3.2 Limitations of Existing UI Modeling for FIF

It is apparent that UFR represents a typical software engineering task, while UAG aligns closely with recent advancements in Artificial Intelligence Generated Content (AIGC). However, UI Focusability Prediction (UFP), a novel task proposed in this work, significantly diverges from existing UI tasks.

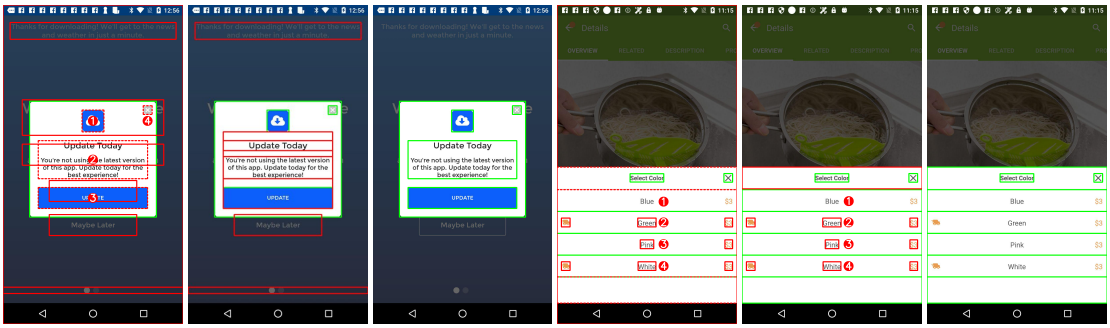
Limitations of Traditional Software Engineering for UFP. Traditional SE methods for UI accessibility primarily rely on hard-coded rules, static analysis, and structural heuristics [28]. While these methods are effective in detecting predefined accessibility issues, they fall short in providing the multi-modal context necessary for determining focusability at a granular level based on complex semantics and diverse user needs. These limitations underscore the demand for an intelligent, multi-modal approach to UFP.

Limitations of Existing UI Modeling for UFP. The UFP task introduces a nuanced layer of UI understanding, presenting two key difficulties: (1) *Context-Dependent Selective Focusability*. UFP determines components that should be selectively accessible, such as visible but inactive elements beneath overlays or widgets of multiple elements as an independent functional unit. In contrast, existing tasks typically assume all visible components are accessible, making them ill-equipped to meet the requirements. (2) *Unified Consideration of Multiple Accessibility Needs*. UFP integrates NA, OA, and SA issues holistically, unlike existing methods that address these issues in isolation. For instance, UI grouping [48] is limited to clustering components but fails to balance grouping and segmentation dynamically for accessibility needs, while UI detection [5] holds potential for resolving NA issues by visual information but fails OA issues marking components under overlays as non-focusable despite their visibility. A detailed comparison of existing UI tasks and their limitations is provided in Table 7 within Appendix D.

This study is the first to address UI accessibility at focusability, which can mitigate many issues related to information or function, as often caused by UI focusability issues. In the following sections, we provide a holistic approach to advancing UFP in the mobile web by introducing a tailored dataset and novel method.

Table 1: Comparison of NOS with Other Existing UI Datasets. *Inh.* represents the inheritance relationships of the datasets. *Comp.* is short for components. *VH.* is view hierarchy. *OS.* means epen source. *AL.* denotes accessibility labels.

	Dataset	Inh.	Year	#Comp.	#Pages	#Apps	VH.	Labels	Tasks	OS.	AL.
D1	**ERICA** [12]		2016	-	18,000	1,011	-	-	User Interaction Analysis	✗	✗
D2	**Rico** [11]		2017	-	66,261	9,700	✓	View Hierarchy	UI Layout Generation	✓	✗
D3	**Rico-semantic** [26]	D2	2018	1,369,685	66,261	9,700	✓	View Hierarchy	UI Layout Generation	✓	✗
D4	**ReDraw** [29]		2018	431,747	14,382	6,538	✓	Component Type	UI Component Recognition and Classification	✓	✗
D5	**Widget Caption** [25]	D2	2020	61,285	21,750	6,470	✓	Component Description	UI Component Caption	✓	✗
D6	**LabelDroid** [8]		2020	19,245	13,145	7,594	✗	Component Type	UI Component Recognition	✓	✗
D7	**Wireframe** [7]	D6	2020	-	54,987	7,748	✓	Query and Target UI	UI Search/UI Component-Matching	✓	✗
D8	**VINs** [4]	D2	2021	-	2,740	9,700	✓	Query and Target UI	UI Search	✗	✗
D9	**Screen2words** [42]	D2	2021	-	22,417	6,269	✓	Screen Description	Screen Summarization	✓	✗
D10	**CLAY** [24]	D2	2022	1,427,915	59,555	9,700	✓	Component Type	UI Component Recognition	✓	✗
D11	**META-GUI** [37]		2022	-	18,337	11	✓	Question-Answer	Task-oriented Dialogue	✓	✗
D12	**MUD** [15]		2023	-	18,132	3,300	✓	Component Type	UI Component Recognition & UI Retrieval	✓	✗
D13	**DroidTask** [45]	D11	2023	-	362	13	✓	Action Type	Mobile Task Automation	✓	✗
D14	**AITW** [30]		2023	-	5,689,993	159	✗	Instruction	Mobile Device Control	✓	✗
D15	**Auto-UI** [55]	D14	2023	-	1,276,752	159	✗	Instruction	Mobile Device Control	✓	✗
D16	**Ferret-UI** [52]	D5&D9	2024	-	123,702	-	✗	-	Referring, Grounding, and Reasoning	✗	✗
D17	**Mobile3M** [46]		2024	-	3,098,786	49	✓	Graph Structure	Page Navigation	✗	✗
D18	**AutoGUI** [38]		2024	-	702,000	-	✗	Question-Answer	UI Functionality Grounding	✓	✗
D19	**GUI-WORLD** [6]		2024	-	12,379	-	✗	Question-Answer	UI Understanding & Instruction Following	✓	✗
D20	**ScreenAI-SA** [3]	D2	2024	22,078	4,200	-	✓	Component Type	Screen Annotation & Navigation	✓	✗
D21	**NOS-raw**		2024	1,634,104	31,097	490	✓	UI Focusability	UI Focusability Prediction	✓	✓
D22	**NOS-raw-labeled**	D21	2024	66,031	1,000	208	✓			✓	✓
D23	**Rico-labeled**	D3	2024	51,449	1,000	-	✓			✓	✓
D24	**NOS**	D22&23	2024	117,480	2,000	-	✓			✓	✓



(a) Rico-s for [T7] (b) CLAY for [T7] (c) NOS for [T7] (d) Rico-s for [T8] (e) CLAY for [T8] (f) NOS for [T8]

Figure 2: SA Examples: (2a)-(2c) highlight erroneous red components in Rico-s and CLAY violating the requirements of [T7] Minimal-functional Component Focusability, while (2d)-(2f) illustrate violations of [T8] Hierarchical Consistency Enforcement.

4 NOS: A New Dataset for UI Accessibility

Existing UI datasets are not directly applicable to UFP due to two limitations: (1) *Absence of Focusability-Related Context Information.* UFP necessitates detailed component information and hierarchical relationships to indicate focusability at multiple semantic levels. However, as summarized in Table 1, 15 out of 20 existing datasets lack proper UI *component labels* or *view hierarchies*. (2) *Lack of Accessibility Labels.* UFP requires labels that specify focusability of granularity for NA, OA, and SA issues, which are generally missing in existing datasets with only component types provided and not adequate for focusability determination. Examples illustrating the inapplicability are presented in Figure 2 with additional examples in Figure 4 within Appendix C. To address the challenges, we introduce a novel dataset addressing the NA, OA, and SA issues (NOS).

4.1 Introduction of the Proposed Dataset

The proposed dataset, NOS[1], is presented as a labeled version (NOS: NOS-raw-labeled & Rico-labeled) and an extensive unlabeled version (NOS-raw). See Table 1 for statistics.

[1]**URL**: https://doi.org/10.5281/zenodo.14802776 https://github.com/eaglelab-zju/NOS

Crawler and Cleaning. To obtain heuristic focus results from the default Android Assistive Technology TalkBack, which is absent in existing datasets, we crawled a new batch of UI data. This data captures each page's TalkBack focus results, serving as a baseline to illustrate the severity of NA, OA, and SA issues. The dataset, termed NOS-raw, consists of 1,634,104 components collected from 31,097 pages across 490 apps. The raw data included redundant and invalid nodes extracted from the view hierarchy of each page. To ensure quality, we implemented a cleaning process to filter out invalid components, such as those with negligible size or completely outside the screen. See Algorithm 1 in Appendix B for more details.

Labeling and Verification. Due to resource constraints, we sampled 1,000 pages each from NOS-raw and Rico for focusability labeling, resulting in a labeled subset with 117,480 components. The annotations were carried out by three annotators with varying levels of accessibility-related experience (one month, six months, and two years). The annotations were subsequently verified by three verifiers with more extensive experience (one month, one year, and five years). Before annotation, a senior expert with five years of accessibility experience provided training and addressed

Table 2: User Evaluation Scores. *S.* represents sighted users. The highest values are bolded.

Participants			1	2	3	4	5	6	7	8	9	10	11	12	AVG.		13	14	15	AVG.
BLV	NA	TalkBack	2	2	1	3	3	3	2	3	2	2	3	3	2.42		2	2	3	2.33
		Annotated	5	5	5	5	4	5	4	4	4	5	5	5	**4.67**		5	4	4	**4.33**
	OA	TalkBack	1	2	2	3	1	3	4	4	3	2	3	2	2.50		1	1	1	1
		Annotated	5	4	4	4	3	4	5	5	5	4	4	5	**4.33**	S.	5	5	5	**5**
	SA-aggregation	TalkBack	5	4	0	5	5	5	4	5	4	4	4	3	**4.00**		2	3	3	2.67
		Annotated	2	2	2	3	3	4	5	4	4	5	5	4	3.58		5	5	5	**5**
	SA-segmentation	TalkBack	0	1	0	1	2	2	1	2	2	2	1	1	1.25		1	1	1	1
		Annotated	5	5	5	5	5	5	5	5	4	5	5	5	**4.92**		5	5	5	**5**
	Frequency	NA	3	4	3	3	2	2	3	3	4	4	3	4	**3.17**					
		OA	4	3	2	3	2	2	3	4	2	3	3	4	2.92					
		SA	2	3	2	1	2	1	2	2	2	3	3	3	2.17					
	Severity	NA	5	5	5	5	4	4	4	4	4	5	4	4	**4.42**					
		OA	5	4	2	3	3	1	3	3	4	4	3	4	3.25					
		SA	3	4	3	3	3	3	5	4	4	4	4	5	3.75					

any questions raised by the annotators. To further validate the labeling quality, we conducted a user study to validate the dataset annotations, as described in the following section.

4.2 User Evaluation

We further conducted a user study to evaluate the criteria and results of the annotation. The interviews were designed to address the following questions:

- **q1**: Do the focusability results after annotation better meet the needs of users with vision impairments?
- **q2**: Do BLV users think of these UI accessibility issues as critical and frequently encountered?
- **q3**: Do sighted accessibility developers/evaluators share a consistent understanding when provided with the same experience?

Method. We recruited the same 12 BLV participants (P1-12) as in Section 2 with 3 new sighted participants (P13-15) and compensated them with 25 USD for their participation in a round of 45-minute in-person study. The three sighted participants had prior experience in mobile accessibility development (P13) or evaluation (P14-15). Three sets of examples for NA, OA, and SA issues are prepared, each consisting of a pair of UI pages representing pre- and post-annotation focusability conditions. The pre-annotation pages simulated TalkBack's focusability results, while the post-annotation pages reflected the application of our labeling criteria. To ensure user experience consistency, we programmatically implemented focusable component access and text-to-speech functionality for both pre-annotation and post-annotation focuses within the examples. Participants navigated the pages using swipe gestures on their screens, simulating the behavior of assistive technologies like TalkBack. After interacting with each pair of pages, participants rated their experiences based on three metrics: *accessibility*, *encounter frequency*, and *issue severity*, with scores ranging from 0 (lowest) to 5 (highest). This was followed by a Q&A-style interview to explore their perceptions. Please refer to Appendix E for details.

Findings. We analyze the 3 research questions based on the results in Table 2 and participant comments, drawing the following key findings. **First**, the results support the reasonability of the annotations, as most annotated examples received higher satisfaction scores from both BLV and sighted participants with their scores generally consistent with each other (**q1**). **Second**, regarding issue frequency and severity, NA issues emerged as the most frequent and severe, while SA issues were the least frequent, and OA issues were the least severe (**q2**). **Moreover**, comprehensive UI aggregation

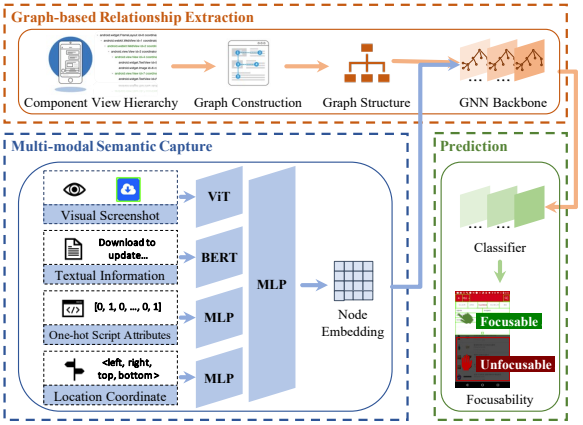


Figure 3: Overall Framework of GIFT.

based on a deep understanding of visual semantics can enhance user experience, as evidenced by the difference in scoring on SA aggregation annotations between sighted and BLV users and the change in opinions of BLV users in interviews (**q3**). Specifically, sighted participants found the aggregated results significantly more accessible, with scores of 5 compared to TalkBack's 2.67. Meanwhile, BLV users initially scored TalkBack focus results higher (4.00) than aggregated results (3.58). However, this changed after further clarification in interviews. Two BLV participants initially remarked that "*the lack of component aggregation is not a serious problem*" and expressed a preference for "*receiving as much focus as possible while also accessing the textual information*". However, after the interviewer clarified that the segmented components in the examples collectively served as a single functional unit (e.g., a product description) and did not provide additional value when segmented, the participants revised their perspectives. They then acknowledged that "*such aggregation is beneficial because over-segmented components can lead to misjudgment, making the process complicated, cumbersome and time-consuming.*" This underscores the critical role of addressing SA issues, as effectively aggregated component groups can significantly reduce the time and effort required for BLV users to navigate and comprehend UI content.

5 Methodology and Experiments

5.1 Graph-based UI Focusability Prediction

Building on the introduced dataset, we propose a novel framework, Graph-based UI FocusabiliTy Prediction (GIFT), designed to effectively address the UFP task. The overall architecture of GIFT is illustrated in Figure 3, comprising three key components: a multi-modal semantic capture module, a graph-based relationship extraction module, and a prediction module. The *primary contribution* lies in the development of the graph-based relationship extraction module and the recognition that heterophilic graph neural networks (GNNs) with hierarchical individuality preservation are essential for capturing nuanced relational patterns critical to this task.

5.1.1 Multi-modal Semantic Capture. We capture each UI component's semantic information using basic models across four feature sources: visually cropped screenshots, textual information, one-hot

Table 3: Relationship Requirements for UFP Tasks.

Task	Required Relationship for Task	Fig.
[T1]	**Syntactic and Positional Relationships:** Components with similar semantics as siblings typically share a similar spatial or hierarchical placement. If sibling components are focusable and one is not, this may indicate a missing focus assignment.	4a-4c
[T2]	**Redundancy and Nesting Relationships:** Components with redundant semantics or functions are often related through similarity or a nesting structure within a common view hierarchy or visual location, signaling shared roles or functionality.	4d-4f
[T3]	**Visibility Hierarchical Relationships:** Invisible components generally occupy distinct subtrees in the view hierarchy compared to visible ones, reinforcing their non-interactive status.	4g-4i
[T4]	**Active-Inactive Layer Relationships:** Inactive components are frequently situated outside the active subtree (e.g., the background of a popup layer) and may have grayscale overlays indicating inactivity. These hierarchical relationships assist in identifying components that remain unaffected by the overlay.	4j-4l
[T5]	**Compositional Relationships for Unified Semantics:** Components that collectively form a semantic unit, such as an icon paired with text in menus, should be grouped for focus, reflecting their shared functional semantic relationships.	4m-4o
[T6]	**Segmentation Relationships for Clarity:** Groups of components that convey complex semantics should be segmented based on their semantic similarity relationships to avoid overwhelming users with excessive information in a single focus.	4p-4r
[T7]	**Contextual and Functional Independence Relationships:** Small components with unique functions distinct from their parent or child elements should be independently focusable, underscoring a relational understanding of their design context.	2a-2c
[T8]	**Consistent Hierarchical Relationships:** Hierarchical components should provide a consistent focus experience, with a relational understanding of hierarchy essential.	2d-2f

script attributes, and location coordinates. For visual content, we use the ViT [14] architecture to embed each component's cropped screenshot. Textual data is embedded using BERT [13], including the component's inner text, content_description, view_id_resource_name, and class_name. The one-hot script attributes and location coordinates are encoded through MLPs. The one-hot vector captures the presence of specific node attributes (i.e., [`checkable`, `checked`, `clickable`, `long_clickable`, `context_clickable`, `text`, `focusable`, `focused`, `content_description`, `selected`, `enabled`]), while the location coordinates use `<left, top, right, bottom>` to define the component's position. Note that the selection of node attributes is hand-crafted, and additional accessibility-related attributes could be incorporated. However, as our goal is to demonstrate the effectiveness of GIFT rather than to optimize feature engineering, we only maintain a consistent attribute selection across all experiments to ensure fairness. Finally, the resulting embeddings are fused via an MLP layer for a unified representation.

5.1.2 Graph-based Relationship Extraction. To determine a component's focusability, relational analysis is crucial. Table 3 shows the relationship requirements for each task from [T1] to [T8]. Apparently, not only SA issues depend on relational understanding for proper focus selection, but NA and OA issues can also benefit from relationship extraction. Given the significance of these relationships in UFP tasks, we draw inspiration from the tree structure of the view hierarchy (i.e., a specialized graph form), and propose adding a graph-based relationship extraction module to GIFT. We construct the graph by leveraging the inherent view hierarchy and employing a GNN backbone to learn underlying relationships.

A critical question then arises: *What type of graph-based method best suits this scenario?* We argue that not all GNNs are appropriate for UFP. Specific GNNs with heterophily handling and hierarchical individuality capture capabilities offer distinct advantages due to two factors: (1) *Heterophilic Neighborhoods*: UI view hierarchy graphs typically exhibit heterophilic relationships, where nodes often connect to functionally distinct neighbors, while similar nodes (e.g., sibling components) are situated as unconnected siblings at the same depth. (2) *Hierarchical Individuality*: Tree structures create shared traits at the same depth but distinctive characteristics across different depths. Most GNNs aggregate neighbors uniformly, which cascades across neighborhoods of different hops. While this maintains depth-based commonalities, it may erase distinctions across depths. Thus, heterophilic GNNs [19, 36] that maintain individuality across hops are ideal for UFP rather than homophilic ones [18, 22] , a need largely unaddressed by existing graph-based UI understanding methods [24]. This setup enables GIFT to capture both relational structures and hierarchical individuality crucial for accurate focus prediction.

5.1.3 Inductive Binary Prediction. After relationship extraction, each component's embedding undergoes classification through an MLP, predicting focusability as a binary outcome. Since each page has its graph derived from the view hierarchy, GIFT operates inductively, processing each page independently rather than using a fixed graph as in transductive GNNs. When a new page is introduced, a new graph is generated with components as nodes, which then pass through the three modules. This design enables GIFT to generalize efficiently without prior exposure to specific page structures, adapting to each page's unique structure.

5.2 Experiments

In this section, we conduct extensive experiments to address the following research questions:

- **RQ1**: How does the proposed GIFT perform compared with graph-agnostic methods? What kind of GNNs are most suitable?
- **RQ2**: What are the contributions of each component in GIFT?
- **RQ3**: How do existing Multimodal Large Language Models perform relative to GIFT on the UFP task?

5.2.1 Baselines and Experimental Settings. We evaluate GIFT across three versions of the NOS dataset, i.e., Rico-labeled, NOS-labeled, and NOS (Mixed). **Baselines**: (1) Heuristic approaches: TalkBack; (2) Graph-agnostic methods: DeTR (CLAY-trans) [5, 24], MLP and Transformer [39] (GIFT variants without the graph-based module); (3) Graph-based methods (GIFT +GNNs): GCN [22], GAT [40], GraphSAGE [20], SIGN [16], OrderedGNN [36], SGFormer [47], IGNN [19]; (4) Multimodal Large Language Models (MLLMs): Intern2-VL-8B [10], CogAgent-18B [21], Qwen2-VL-7B [43], MiniCPM 2.6 V-8B [51]. **Experimental Setup**[2][3]: The NOS dataset is randomly split into train/valid/test sets in a 60%/20%/20% ratio, repeated three times. We report the mean and standard deviation of the performance metrics across these splits, i.e., macro F1-score (MaF1) and area under the curve (AUC). For all experiments, we use pre-trained ViT and BERT parameters, which remain frozen during training.

[2]**Code**: https://github.com/eaglelab-zju/NOS
[3]**DOI**: https://doi.org/10.5281/zenodo.14803014

Table 4: Performance Comparison across Rico-labeled, NOS-raw-labeled, and NOS (Mixed) Datasets. *R.* means rank, and *A.R.* is short for average rank. Bolded scores indicate the highest values, while underlined scores are the second-highest.

	Model	Rico-labeled				NOS-raw-labeled				NOS (Mixed)				
		MaF1±Std	R.	AUC±Std	R.	MaF1±Std	R.	AUC±Std	R.	MaF1±Std	R.	AUC±Std	R.	A.R.
Heuristic	Talkback	—		—		72.27±1.56	10	—		—		—		
Graph-agnostic	DeTR (CLAY-trans)	65.14±0.39	9	87.35±0.88	9	74.87±1.75	9	88.57±0.94	9	71.96±1.38	9	88.11±0.92	9	9.00
	GIFT +**MLP**	72.56±1.26	6	91.49±0.66	6	79.55±0.58	8	92.51±0.55	8	77.68±0.32	8	92.83±0.50	8	7.33
	GIFT +**Transformer**	74.96±1.16	5	93.08±0.35	4	80.75±0.26	6	93.37±0.52	7	78.81±0.39	6	93.47±0.06	7	5.83
Graph-based GIFT +	**GCN**	64.02±0.39	10	87.27±0.32	10	67.75±2.72	11	84.77±1.85	10	68.42±1.27	10	87.96±0.22	10	10.17
	GAT	75.28±2.00	3	92.62±0.85	5	82.38±0.58	4	94.59±0.58	4	82.51±1.00	2	95.38±0.16	2	3.33
	GraphSAGE	71.58±0.99	8	90.66±0.37	8	80.89±0.46	5	93.75±0.38	5	80.01±0.90	5	93.89±0.21	5	6.00
	SIGN	71.99±0.80	7	90.82±1.28	7	80.08±0.50	7	93.50±0.54	6	77.94±0.49	7	93.55±0.60	6	6.67
	OrderedGNN	75.15±2.22	4	93.13±1.09	3	82.27±1.24	3	95.01±0.55	2	82.24±1.24	3	95.21±0.34	4	3.16
	SGFormer	75.88±1.64	2	93.32±0.64	2	82.94±0.61	2	94.91±0.32	3	82.19±0.44	4	95.26±0.05	3	2.67
	IGNN	**77.59±2.65**	**1**	**94.33±0.97**	**1**	**84.37±0.62**	**1**	**95.81±0.21**	**1**	**83.87±0.49**	**1**	**95.91±0.09**	**1**	**1.00**

Table 5: Ablation Studies. *lbd* is short for labeled.

Model	Rico-lbd		NOS-raw-lbd		NOS (Mixed)	
	MaF1	AUC	MaF1	AUC	MaF1	AUC
w/o Visual	70.72	91.68	78.20	91.67	73.92	91.78
w/o Textual	75.97	93.87	83.88	95.52	82.52	95.44
w/o Attributes	67.50	89.85	77.11	91.88	74.75	92.18
w/o Location	76.27	93.70	84.09	95.79	82.78	95.70
w/o Graph	72.56	91.49	79.55	92.51	77.68	92.83
GIFT +**IGNN**	**77.59**	**94.33**	**84.37**	**95.81**	**83.87**	**95.91**

Table 6: Results of MLLMs on NOS (Mixed).

	Accuracy	Recall	Precision	MaF1
Intern2-VL-8B	26.04±1.23	99.66±0.19	25.18±1.24	40.62±1.46
CogAgent-18B	26.29±0.91	97.21±0.49	25.12±1.16	39.91±1.42
Qwen2-VL-7B	35.31±0.31	96.77±0.57	27.46±1.29	42.97±1.44
MiniCPM 2.6V-8B	25.60±1.17	**99.97±0.02**	25.30±1.19	40.37±1.35
GIFT	**91.76±0.12**	85.04±0.84	**82.63±1.80**	**83.87±0.49**

*5.2.2 Performance Analysis (**RQ1**).* Table 4 presents the performance results. Several observations can be drawn: (1) The heuristic focus model, TalkBack, achieves an F1 score of 72.27, ranking 10th out of 11 models. This result underscores the significant challenges in achieving focus accessibility and highlights the need for advanced, intelligent models to address the UFP task effectively. (2) Graph-based models generally outperform graph-agnostic models, particularly when a suitable GNN backbone is selected. The GCN variant demonstrates the lowest performance likely due to its limitations in handling heterophily [36]. In contrast, heterophilic GNNs and graph transformers, such as OrderedGNN, SGFormer, and IGNN, significantly outperform other GNNs. (3) Variants of GIFT that incorporate transformers and GAT exhibit competitive performance, benefiting from their adaptive attention mechanisms. Conversely, the DeTR architecture, which is also utilized in CLAY as a cross-attention transformer variant for the UI type recognition task, shows the third-lowest performance. This suggests that DeTR's cross-attention mechanism, which aligns each node's view hierarchy embedding with the entire page screenshot, is less effective for UFP compared to methods that leverage visual information from cropped screenshots of individual components directly.

*5.2.3 Ablation Studies (**RQ2**).* The ablation studies are presented in Table 5. (1) Results show that removing the visual screenshots and one-hot script attribute results in the largest performance decline, underscoring the critical role of visual semantics and script attributes associated with component functionality in the UFP task. (2) Excluding the graph component leads to the third-largest drop in performance, highlighting the importance of graph-based relational modeling within this architecture. (3) The textual and location components contribute moderately, with slight performance decreases when omitted. While these components add some distinguishing information, they are less essential than the other information. (4) The complete GIFT achieves the highest scores, confirming the advantage of integrating all components into the model.

*5.2.4 Comparison with MLLMs (**RQ3**).* Table 6 presents the results of several MLLMs on UFP. While the recall values for all MLLMs approach 100%, the accuracy, precision, and macro F1-scores are consistently low. This indicates a tendency for the MLLMs to overpredict UI components as focusable, failing to effectively address components with OA and SA issues. However, findings from the formative study and user evaluations underscore the significance of addressing OA and SA issues. Over-segmentation or inappropriate focusability in such components often leads to semantic ambiguity, impeding effective navigation and usage—critical barriers identified by BLV participants. These results highlight *the limited customizability of MLLMs for accessibility-specific scenarios and their inability to adapt to diverse accessibility requirements effectively.*

6 Conclusion

This paper advances the field of mobile web accessibility by introducing the unified UI accessibility framework of three stages: focusability, information, and functionality (FIF), guided by insights from a formative study with 12 BLV users. We identified key challenges across the stages, introducing the novel concept of selective-accessible (SA) issues. FIF structures accessibility tasks into a unified approach with 12 sub-tasks of 3 overarching tasks. Recognizing UI focusability prediction as a pivotal yet underexplored task, we contribute a tailored dataset (NOS) and propose the Graph-based UI Focusability Prediction (GIFT) method, which leverages GNNs to model UFP-targeted contextual relationships. Experimental and user evaluations validate the effectiveness of our dataset and method, enhancing mobile web accessibility.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (Grant No.62372408).

References

[1] Maram Fahaad Almufareh, Sumaira Kausar, Mamoona Humayun, and Samabia Tehsin. 2024. A conceptual model for inclusive technology: advancing disability inclusion through artificial intelligence. *Journal of Disability Research* 3, 1 (2024), 20230060.

[2] Abdulaziz Alshayban, Iftekhar Ahmed, and Sam Malek. 2020. Accessibility issues in android apps: state of affairs, sentiments, and ways forward. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 1323–1334.

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

[4] 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*. 1–14.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European conference on computer vision*. Springer, 213–229.

[6] Dongping Chen, Yue Huang, Siyuan Wu, Jingyu Tang, Liuyi Chen, Yilin Bai, Zhigang He, Chenlong Wang, Huichi Zhou, Yiqiang Li, et al. 2024. GUI-WORLD: A Dataset for GUI-oriented Multimodal LLM-based Agents. *arXiv preprint arXiv:2406.10819* (2024).

[7] Jieshan Chen, Chunyang Chen, Zhenchang Xing, Xin Xia, Liming Zhu, John Grundy, and Jinshui Wang. 2020. Wireframe-based UI design search through image autoencoder. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 29, 3 (2020), 1–31.

[8] Jieshan Chen, Chunyang Chen, Zhenchang Xing, Xiwei Xu, Liming Zhu, Guoqiang Li, and Jinshui Wang. 2020. Unblind your apps: Predicting natural-language labels for mobile gui components by deep learning. In *Proceedings of the ACM/IEEE 42nd international conference on software engineering*. 322–334.

[9] Liuqing Chen, Yunnong Chen, Shuhong Xiao, Yaxuan Song, Lingyun Sun, Yankun Zhen, Tingting Zhou, and Yanfang Chang. 2024. EGFE: End-to-end Grouping of Fragmented Elements in UI Designs with Multimodal Learning. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*. 1–12.

[10] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. 2023. InternVL: Scaling up Vision Foundation Models and Aligning for Generic Visual-Linguistic Tasks. *arXiv preprint arXiv:2312.14238* (2023).

[11] 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 ACM symposium on user interface software and technology*. 845–854.

[12] Biplab Deka, Zifeng Huang, and Ranjitha Kumar. 2016. ERICA: Interaction mining mobile apps. In *Proceedings of the 29th annual symposium on user interface software and technology*. 767–776.

[13] Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[14] Alexey Dosovitskiy. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[15] Sidong Feng, Suyu Ma, Han Wang, David Kong, and Chunyang Chen. 2024. MUD: Towards a Large-Scale and Noise-Filtered UI Dataset for Modern Style UI Modeling. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–14.

[16] Fabrizio Frasca, Emanuele Rossi, Davide Eynard, Ben Chamberlain, Michael Bronstein, and Federico Monti. 2020. Sign: Scalable inception graph neural networks. *arXiv preprint arXiv:2004.11198* (2020).

[17] Google. 2020. Get started on Android with TalkBack - Android Accessibility Help. https://support.google.com/accessibility/android/answer/6283677?hl=en Accessed: 2024-12-02.

[18] Ming Gu, Gaoming Yang, Sheng Zhou, Ning Ma, Jiawei Chen, Qiaoyu Tan, Meihan Liu, and Jiajun Bu. 2023. Homophily-enhanced structure learning for graph clustering. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 577–586.

[19] Ming Gu, Zhuonan Zheng, Sheng Zhou, Meihan Liu, Jiawei Chen, Tanyu Qiao, Liangcheng Li, and Jiajun Bu. 2024. Universal Inceptive GNNs by Eliminating the Smoothness-generalization Dilemma. *arXiv preprint arXiv:2412.09805* (2024).

[20] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

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

[22] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[23] Jonathan Lazar, Daniel F Goldstein, and Anne Taylor. 2015. *Ensuring digital accessibility through process and policy*. Morgan kaufmann.

[24] Gang Li, Gilles Baechler, Manuel Tragut, and Yang Li. 2022. Learning to denoise raw mobile UI layouts for improving datasets at scale. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–13.

[25] 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. *arXiv preprint arXiv:2010.04295* (2020).

[26] Thomas F Liu, Mark Craft, Jason Situ, Ersin Yumer, Radomir Mech, and Ranjitha Kumar. 2018. Learning design semantics for mobile apps. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 569–579.

[27] Thomas F Liu, Mark Craft, Jason Situ, Ersin Yumer, Radomir Mech, and Ranjitha Kumar. 2018. Learning design semantics for mobile apps. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 569–579.

[28] Forough Mehralian, Navid Salehnamadi, Syed Fatiul Huq, and Sam Malek. 2022. Too much accessibility is harmful! automated detection and analysis of overly accessible elements in mobile apps. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*. 1–13.

[29] Kevin Moran, CB Cardenas, M Curcio, R Bonett, and D Poshyvanyk. 2018. The ReDraw dataset: A set of Android screenshots, GUI metadata, and labeled images of GUI components. (2018).

[30] Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. 2024. Androidinthewild: A large-scale dataset for android device control. *Advances in Neural Information Processing Systems* 36 (2024).

[31] Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O Wobbrock. 2017. Epidemiology as a framework for large-scale mobile application accessibility assessment. In *Proceedings of the 19th international ACM SIGACCESS conference on computers and accessibility*. 2–11.

[32] Navid Salehnamadi, Abdulaziz Alshayban, Jun-Wei Lin, Iftekhar Ahmed, Stacy Branham, and Sam Malek. 2021. Latte: Use-case and assistive-service driven automated accessibility testing framework for android. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–11.

[33] Navid Salehnamadi, Abdulaziz Alshayban, Jun-Wei Lin, Iftekhar Ahmed, Stacy Branham, and Sam Malek. 2021. Latte: Use-case and assistive-service driven automated accessibility testing framework for android. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–11.

[34] Eldon Schoop, Xin Zhou, Gang Li, Zhourong Chen, Bjoern Hartmann, and Yang Li. 2022. Predicting and explaining mobile ui tappability with vision modeling and saliency analysis. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–21.

[35] Jeff Shuford. 2023. Contribution of Artificial Intelligence in Improving Accessibility for Individuals with Disabilities. *Journal of Knowledge Learning and Science Technology ISSN: 2959-6386 (online)* 2, 2 (2023), 421–433.

[36] Yunchong Song, Chenghu Zhou, Xinbing Wang, and Zhouhan Lin. 2023. Ordered GNN: Ordering Message Passing to Deal with Heterophily and Over-smoothing. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=wKPmPBHSnT6

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

[38] AutoGUI Team. 2024. AutoGUI-v1-702k Dataset. https://huggingface.co/datasets/AutoGUI/AutoGUI-v1-702k. Accessed: 2024-11-16.

[39] A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* (2017).

[40] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. 2017. Graph attention networks. *stat* 1050, 20 (2017), 10–48550.

[41] W3C Web Accessibility Initiative (WAI). 2008. Mobile Accessibility at W3C. https://www.w3.org/WAI/standards-guidelines/mobile/ Accessed: 2024-12-02.

[42] Bryan Wang, Gang Li, Xin Zhou, Zhourong Chen, Tovi Grossman, and Yang Li. 2021. Screen2words: Automatic mobile UI summarization with multimodal learning. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. 498–510.

[43] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024. Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution. *arXiv preprint arXiv:2409.12191* (2024).

[44] Web Accessibility Initiative (WAI). 2024. Introduction to Web Accessibility. https://www.w3.org/WAI/fundamentals/accessibility-intro/ Accessed: 2024-11-26. First published: February 2005. Last updated: 7 March 2024.

[45] Hao Wen, Yuanchun Li, Guohong Liu, Shanhui Zhao, Tao Yu, Toby Jia-Jun Li, Shiqi Jiang, Yunhao Liu, Yaqin Zhang, and Yunxin Liu. 2024. Autodroid: Llm-powered task automation in android. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. 543–557.

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

[47] Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. 2024. Simplifying and empowering transformers for large-graph representations. *Advances in Neural Information Processing Systems* 36 (2024).
[48] Shuhong Xiao, Yunnong Chen, Yaxuan Song, Liuqing Chen, Lingyun Sun, Yankun Zhen, Yanfang Chang, and Tingting Zhou. 2024. UI semantic component group detection: Grouping UI elements with similar semantics in mobile graphical user interface. *Displays* 83 (2024), 102679.
[49] Tao Xin, Jiying Zhu, Luling Wang, and Xiaowei Qin. 2023. Screen Recognition: Creating Accessibility Metadata for Mobile Applications using View Type Detection. In *2023 9th International Conference on Computer and Communications (ICCC)*. IEEE, 1787–1793.
[50] Shunguo Yan. 2016. IBM strengthens mobile app accessibility and usability. *IBM, Armonk, NY, USA* (2016).
[51] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. 2024. MiniCPM-V: A GPT-4V Level MLLM on Your Phone. *arXiv preprint arXiv:2408.01800* (2024).
[52] Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, and Zhe Gan. 2025. Ferret-ui: Grounded mobile ui understanding with multimodal llms. In *European Conference on Computer Vision*. Springer, 240–255.
[53] Mengxi Zhang, Huaxiao Liu, Shenning Song, Chunyang Chen, Pei Huang, and Jian Zhao. 2024. Are your apps accessible? A GCN-based accessibility checker for low vision users. *Information and Software Technology* 174 (2024), 107518.
[54] 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*. 1–15.
[55] Zhuosheng Zhang and Aston Zhang. 2023. You only look at screens: Multimodal chain-of-action agents. *arXiv preprint arXiv:2309.11436* (2023).

A Related Work

Accessibility in Mobile Web and UI Test. The mobile web constitutes a significant segment of the Web [41], and its accessibility challenges are reportedly more severe than those of native applications on mobile platforms as assistive services cannot analyze webView elements properly [32]. This highlights the urgent need to prioritize inclusivity in the mobile web. While UI accessibility testing tools tackle a range of accessibility issues, traditional methods rooted in software engineering [28, 50] primarily rely on hard-coded, rule-based component-level analysis. These methods often fail to consider visual, textual, or relational contexts, which are crucial for accurately predicting focus granularity. Recent research has sought to address these limitations through deep learning techniques. For instance, Zhang et al. [54] proposed an on-device object detection model that extracts UI elements from screenshots, leveraging heuristics to refine detections, group elements and determine navigation order. However, this method is constrained by its reliance on heuristics, and the absence of open-source datasets and code limits its adaptability and broader applicability. ALVIN [53], a GCN-based checker, focuses on addressing accessibility needs for users with low vision but does not address focusability issues pertinent to screen reader users. By relying solely on view hierarchy information, it overlooks the visual semantics required for effective focus prediction. In summary, *existing accessibility testing tools and techniques inadequately address the critical aspect of focusability, lacking the semantic and relational insights necessary for optimizing focus prediction. This underscores the need for an intelligent approach beyond rule-based and heuristic checks, as well as a deeper understanding of user-specific accessibility requirements.*

UI Understanding and Modeling. Certain research efforts in UI understanding and modeling, such as UI detection, type recognition, and tappability prediction [24, 34], contribute to the creation of accessibility metadata by providing foundational information [49, 54] (e.g., size, position, type, etc.). While these tasks can partially support UI focusability improvements, they struggle with selective focusability in complex scenarios, such as the grouping or segmentation of multiple UI elements. Similarly, studies on UI grouping [48] primarily aim to facilitate applications like UI code generation [9], rather than addressing accessibility concerns. Consequently, these approaches fail to address key focusability issues, such as non-accessible or over-accessible elements. Moreover, although AI-driven agents for UI manipulation [46] are gaining popularity, their pre-training tasks generally target broader objectives like general UI understanding, often overlooking user-centric accessibility requirements. *Overall, there is a noticeable gap in unified UI understanding tasks tailored specifically to accessibility, particularly in resolving focusability challenges. Developing such a task is crucial for ensuring the equitable application of intelligent techniques in UI understanding and modeling for users with disabilities.* A comprehensive analysis of existing UI tasks and their limitations in addressing accessibility issues is presented in Table 7 within Appendix D.

B Algorithms of GIFT

Algorithm 1 presents the detailed process of component cleaning.

C A Comparison with Existing UI Datasets and Additional Examples of Their Issues

We provide the details of 20 existing UI datasets in our open-source repository: https://github.com/eaglelab-zju/NOS/tree/master/appendix_details/A Comparison with Existing Datasets.md.

Figure 4 shows additional examples of existing dataset issues.

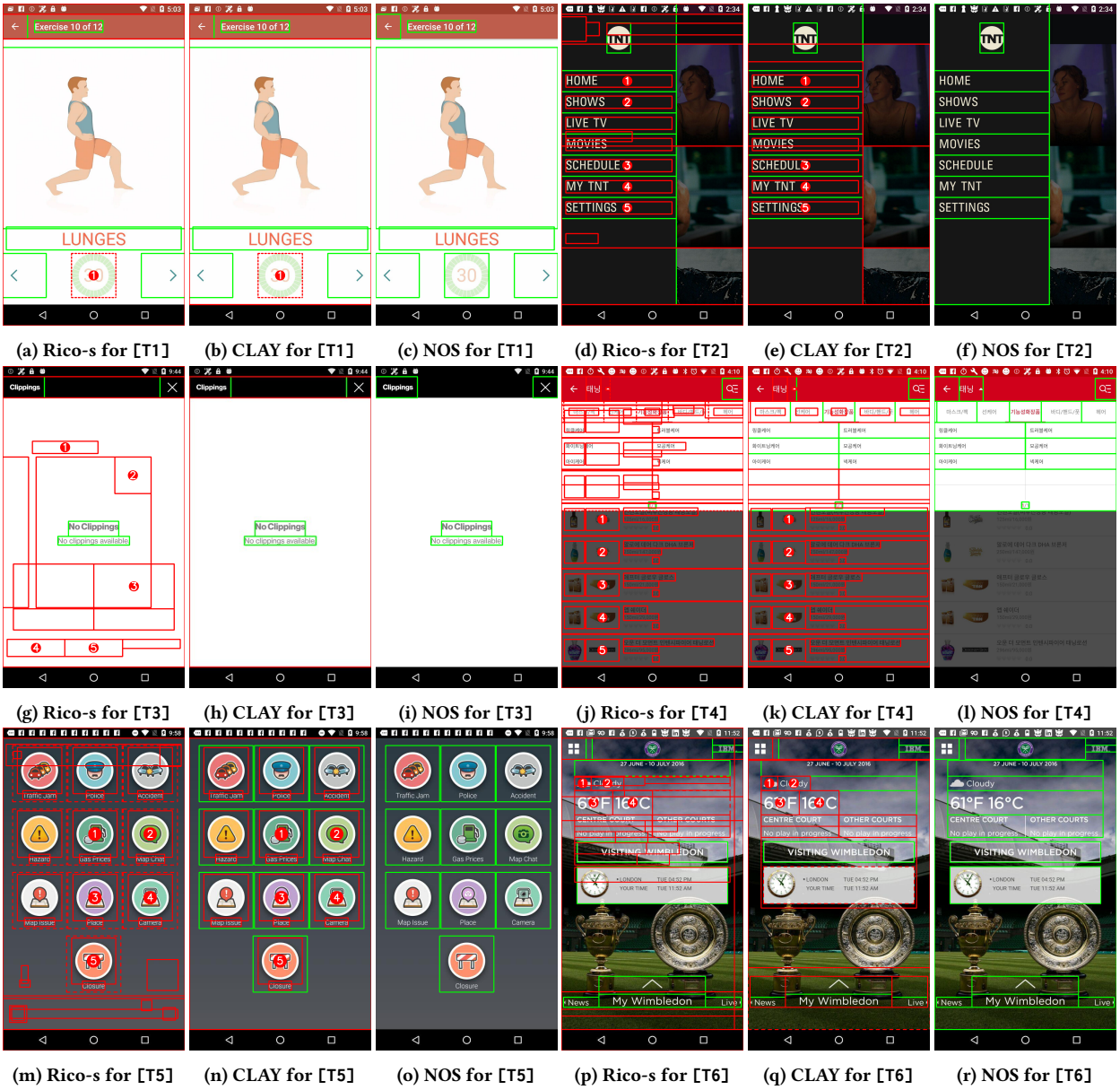
[T1] Non-accessible Component Recovery (Figure 4a- 4c): In both CLAY and Rico-s datasets, a critical text component labeled "30" is missing, which results in the loss of essential information for users. In Figure 4c, manual annotations addressed this issue by recovering the missing focus, thereby restoring the accessibility of independent semantic components.

[T2] Duplicate Component Pruning (Figure 4d- 4f): The pop-up list in Figure 4d- 4f exhibits redundancy in CLAY and Rico-s datasets, where each list item is represented by two overlapping bounding boxes. These redundant components reduce operational efficiency by requiring users to navigate duplicate focuses. In our NOS dataset, each list item is consolidated into a single focusable component, streamlining interaction.

Algorithm 1 Component Cleaning

Input: Left (l), right (r), top (t), and bottom (b) coordinates of the component node, screen height (H) and width (W), height of the bottom system panel (N), minimal width or height (M)
Output: $True$ if the node is valid, otherwise $False$

$t = \max(t, 0), l = \max(l, 0), b = \min(b, H - N), r = \min(r, W)$ ▷ Adjust boundaries to fit within screen edges, allowing partial out-of-screen nodes
if $l + M \geq r$ **or** $t + M \geq b$ **then**
  return $False$ ▷ Filter out nodes that are too small
end if
if $t \geq (H - N)$ **or** $l \geq W$ **or** $b \leq 0$ **or** $r \leq 0$ **then**
  return $False$ ▷ Filter out nodes completely outside the screen
end if
return $True$



(a) Rico-s for [T1] (b) CLAY for [T1] (c) NOS for [T1] (d) Rico-s for [T2] (e) CLAY for [T2] (f) NOS for [T2]

(g) Rico-s for [T3] (h) CLAY for [T3] (i) NOS for [T3] (j) Rico-s for [T4] (k) CLAY for [T4] (l) NOS for [T4]

(m) Rico-s for [T5] (n) CLAY for [T5] (o) NOS for [T5] (p) Rico-s for [T6] (q) CLAY for [T6] (r) NOS for [T6]

Figure 4: Examples of Various NA, OA, and SA Issues. (4a)-(4c): [T1] Non-accessible Component Recovery; (4d)-(4f): [T2] Duplicate Component Pruning; (4g)-(4i): [T3] Invisible Component Filtering; (4j)-(4l): [T4] Inactive Component Filtering; (4m)-(4o): [T5] Semantic Component Aggregation; (4p)-(4r): [T6] Complex Component Segmentation.

[T3] Invisible Component Filtering (Figure 4g- 4i): Figure 4g- 4i involve a page with two stacked views, where the lower layer is invisible. The Rico-s dataset fails to filter out invisible components, potentially misleading users about the content of the page. Both CLAY and manually annotated datasets retain only visible components as focusable elements. However, the CLAY dataset includes an unnecessary outermost bounding box.

[T4] Inactive Component Filtering (Figure 4j- 4l): In Figure 4j- 4l, a pop-up overlays inactive views. Neither the CLAY nor Rico-s datasets filter out components from the inactive views, complicating user interaction and comprehension of the pop-up functionality. The manual annotations (Figure 4 l) focus exclusively on the active pop-up, improving user understanding and efficiency.

[T5] Semantic Component Aggregation (Figure 4m- 4o): Figure 4m- 4o contain numerous icons. In CLAY and Rico-s datasets, icons and their corresponding labels are treated as separate components. The annotated dataset NOS (Figure 4 o) groups them into unified components, enhancing usability by aligning semantic and visual associations, thus improving operational efficiency.

Table 7: Comparison of UFP with Existing UI Tasks. *N.*, *O.* and *S.* are short for NA,OA and SA issues,respectively.

	Existing UI Tasks and Their Limitation for UFP	N.	O.	S.
1	**UI Detection**[11, 27]: Identifying visible components without assessing accessibility requirements of focusability.		✗	✗
2	**Type Recognition**[15, 24, 37]: Classifying component types without determining accessibility needs of focusability.	✗	✗	✗
3	**Function Inference**[3, 6, 38]: Inferring potential functionality but lacking granularity to decide on focus exposure for ATs.	✗	✗	✗
4	**Tappability Prediction**[24]: Identifying clickable elements without addressing focusability for non-clickable but accessible ones.	✗	✗	✗
5	**Widget Captioning**[25]: Generating captions for individual components without handling accessibility needs of focusability.	✗	✗	✗
6	**Screen Summarization**[42]: Providing an overview of the screen without considering element distinction needed for accessibility.	✗	✗	✗
7	**UI Component Suggestion**[15, 24, 37]: Suggesting components but lacking predictions on whether they should be focusable.	✗	✗	✗
8	**Touch Gesture Recognition**: Focusing on user gestures without addressing accessibility requirements.	✗	✗	✗
9	**User Intent Detection**[30, 55]: Detecting general intent without providing context-sensitive accessibility requirements.	✗	✗	✗
10	**User Flow Prediction**[37]: Predicting user flows without considering accessibility needs.	✗	✗	✗
11	**Screen Transition Prediction**[46]: Forecasting screen changes without attention to individual element accessibility.	✗	✗	✗
12	**UI Aesthetics Evaluation**: Evaluating visual appeal without considering accessibility needs for users with disabilities.	✗	✗	✗
13	**Command Grounding**[8]: Mapping commands to actions without predicting context-based focusability accessibility of components.	✗	✗	✗
14	**Interaction Modeling**[7]: Modeling multi-modal interactions yet lacking predictive accessibility within complex interfaces.	✗	✗	✗
15	**Conversation Perception**[37]: Interpreting conversational cues without addressing accessibility needs of focusability.	✗	✗	✗
16	**Conversation Interaction**: Facilitating conversation-based interactions without predicting component focusability needs.	✗	✗	✗
	UI Focusability Prediction (UFP): Achieving unified accessibility predictions on component focusability and granularity.	✓	✓	✓

[T6] Complex Component Segmentation (Figure 4p- 4r): The game information in Figure 4p- 4r is complex, with numerous text components in the middle treated as independent entities in both CLAY and Rico-s datasets. The manual annotations (Figure 4r) reorganize these components into logically grouped segments. This segmentation improves usability by presenting information in a more structured format without compromising comprehension.

[T7] Minimal-functional Component Focusability (Figure 2a- 2c): In Figure 2a- 2c, the active pop-up in the Rico-s dataset is treated as a single large component, with no separation focus for its internal text or buttons. The manually annotated dataset separates them into individual focusable components, enabling access to the elements of independent semantics and interactions. While the CLAY dataset also treats the pop-up's internal components as distinct, it includes additional components from inactive views beneath the pop-up, which may confuse users.

[T8] Hierarchical Consistency Enforcement (Figure 2d- 2f): Figure 2d- 2f depict a pop-up list with inconsistent component focuses in CLAY and Rico-s datasets. For example, the focus of the list item ❶ is missing, while other items feature nested focus boxes. The manual annotations unify the structure by designating the outermost focus box of each list item as a single focusable element.

D A Comparison with Existing UI Tasks

The limitations of 16 existing UI tasks for addressing NA, OA, and SA issues are demonstrated in Table 7, with a summary of them presented in our open-source repository https://github.com/eaglelab-zju/NOS/tree/master/appendix_details/A Comparison with Existing UI Tasks.md.

E Details of Formative Study and User Evaluation

The details of the formative study and user evaluation can be found in our open-source repository https://github.com/eaglelab-zju/NOS/tree/master/appendix_details/Formative Study.md.