Bridging the Semantic Void: A Dual-branch RAG Framework to Enhance GUI Component Description for BVI Users

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

Sheng Zhou*
zhousheng_zju@zju.edu.cn
School of Software Technology, Zhejiang University
Hangzhou, China

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

Liangcheng Li
liangcheng_li@zju.edu.cn
College of Computer Science and Technology, Zhejiang University
Hangzhou, China

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

ABSTRACT

Mobile applications are essential gateways to digital services, yet the lack of accurate alternative text for GUI components creates a significant semantic barrier for blind and visually impaired (BVI) users. While multimodal large language models offer a potential solution for automated labeling, they are often hindered by inconsistent labeling styles across various data sources and static knowledge boundary that exhibits limited generalization to novel UI design patterns beyond the model's training distribution. In this paper, we propose a robust generative framework for GUI component description that ensures the reliability and accuracy of screen reader feedback for BVI users. Based on a formative interview study, we identify [Identity] and [Action] as the core information dimensions required to establish functional certainty. Our framework addresses the domain gap through LoRA-based fine-tuning on a curated, standardized GUI-CD dataset, while simultaneously employing a dual-branch Retrieval-Augmented Generation (RAG) pipeline to recall visual and structural evidence during inference. Experimental results on benchmark datasets demonstrate that our approach achieves state-of-the-art performance. Qualitative analysis confirms that our framework effectively mitigates semantic mismatch, providing the reliable auditory feedback necessary for safe and independent mobile navigation.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; • **Human-centered computing** → **Accessibility systems and tools**; *Empirical studies in accessibility*.

*Corresponding author.

W4A'26, April 13-14, 2026, Dubai

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

KEYWORDS

Accessibility, Accessibility Technology, GUI Component Description, Interview Study, Generative AI

ACM Reference Format:
Yuxuan Wu, Sheng Zhou, Ziwei Wang, Liangcheng Li, and Jiajun Bu. 2026. Bridging the Semantic Void: A Dual-branch RAG Framework to Enhance GUI Component Description for BVI Users. In *Proceedings of In 23rd International Web for All Conference (W4A'26).* ACM, New York, NY, USA, 12 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Mobile applications have become the primary gateways to digital services, ranging from digital finance and healthcare to social infrastructure [11, 23, 45]. While these advancements offer significant convenience to the general public, they often introduce formidable accessibility barriers for blind and visually impaired (BVI) individuals [1, 25]. BVI users primarily navigate mobile interfaces through assistive technologies known as screen readers [43, 44], such as Android's TalkBack [15] and iOS's VoiceOver [4]. However, the efficacy of these tools relies on the availability of high-quality alternative text (Alt-text) to explain visual Graphical User Interface (GUI) components via synthesized speech [17, 41]. Empirical research indicates that a vast majority of functional icons and interactive elements remain unlabelled or possess only generic descriptions [40]. When a screen reader encounters a component without descriptive metadata, it typically announces "unlabelled button". This lack of information forces BVI users to rely on hazardous trial-and-error strategies [16, 24].

Recent developments in Multimodal Large Language Models (MLLMs), suggest a potential path for automating the generation of accessibility labels [10, 47, 51]. By jointly modeling visual pixels and textual semantics, these models can synthesize human-like descriptions for graphical elements [19, 28]. Nevertheless, our analysis identifies two challenges that prevent these foundation models from being reliable assistive technologies. First, existing GUI datasets (e.g., RICO [13]) suffer from *label heterogeneity*, where inconsistent and subjective naming conventions across different sources hinder the model's ability to learn a stable descriptive logic. Second, existing MLLMs operate under *static knowledge boundary* defined by their pre-training data. Because mobile application interfaces

iterate rapidly, models with static weights struggle to generalize to novel UI design patterns, leading to "semantic mismatch" in which components are assigned confident but incorrect functional descriptions. For BVI users, these descriptions are more harmful than missing descriptions, as they provide a false sense of functional certainty.

To establish the criteria for high-quality descriptions that ensure functional certainty, we conducted a semi-structured interview study with eight BVI participants. Comprised of university students and working professionals with an average of five years of screen reader experience, these individuals interact with mobile applications daily for several hours across diverse domains such as banking and navigation. The findings revealed that for BVI users, structural predictability is more important than absolute brevity, as they typically use screen readers at high reading speeds and can therefore comprehend detailed descriptions. Participants reported that novel components introduced in application updates often have unclear functionality, forcing time-consuming exploration to understand an element's purpose. Moreover, inconsistent labeling styles across applications create unnecessary cognitive friction.

Driven by these insights, we prioritize structural predictability and functional certainty in our system design through three principal contributions:

- **Automated Data Synthesis and Unification**: A multi-stage pipeline leveraging MLLMs to unify labels from disparate sources, producing the *GUI-CD* dataset of 33,205 samples following a standardized [Identity] + [Action] structure.
- **Domain-Specific Adaptation**: Fine-tuning the Qwen2-VL backbone with Low-Rank Adaptation (LoRA) on our standardized corpus to align the model's internal knowledge with visual appearance and functional description of GUI components.
- **Dual-branch Retrieval Augmentation**: A dual-branch RAG framework that retrieves evidence from visual and outline repositories, providing grounded, real-world examples to overcome static knowledge boundary during inference.

Experimental results demonstrate that our framework outperforms state-of-the-art baselines across multiple benchmarks. Analysis of the generated descriptions confirms that our approach successfully provides the [Identity] and [Action] information that BVI users requested during our interviews. Overall, by bridging the semantic void of raw GUI pixels with reliable semantic feedback, this work provides a scalable pathway toward a more inclusive and accessible digital future for BVI users.

2 RELATED WORK

Generating accurate GUI component descriptions is essential for enabling accessible interaction for blind and visually impaired (BVI) users. We review related advances in assistive technologies, accessibility auditing, and multimodal models for GUI understanding.

2.1 Assistive Technologies and Missing Semantic Metadata

Screen readers such as TalkBack [15] and VoiceOver [4] enable blind and visually impaired (BVI) users to interact with graphical user interfaces by converting interface elements into speech output [6, 36]. In practice, the usability of these systems depends heavily on the availability of semantic metadata, including component roles and descriptive labels. However, many mobile applications often remain unlabelled or possess generic descriptions, causing screen readers to announce interface elements with generic placeholders such as 'unlabeled button'.

Early attempts to infer missing semantics relied on rule-based layout analysis and UI reverse engineering [3, 14, 35, 37, 53]. These methods leverage spatial heuristics and visual patterns to recover widget structure, but they struggle with modern mobile interfaces that exhibit high visual diversity and dynamic behaviors. As a result, such approaches offer limited scalability and robustness in real-world applications.

2.2 GUI Component Description Generation

The availability of large-scale GUI datasets, notably RICO [13], enabled the adoption of learning-based methods for GUI understanding and description generation. Initial neural frameworks treated this task as a vision-to-language problem, where visual representations of UI components are mapped to natural language labels [5]. Specifically, Widget Caption [31] utilize encoder-decoder architectures to generate localized descriptions for individual elements, and Screen2Words [50] employs a multimodal approach to provide screen-level summaries by aggregating component features.

While these methods demonstrate the feasibility of automatic component description generation, they often depend on dataset-specific annotation schemes [42]. Inconsistent labeling conventions across various data sources frequently lead to generic or ambiguous outputs that fail to capture the actual interactive intent of a component. This lack of descriptive precision significantly reduces the effectiveness of automated tools in downstream applications such as screen-reader interaction [8, 46], as it fails to provide the functional certainty necessary for BVI users to navigate complex interfaces independently.

2.3 Multimodal Models for GUI Understanding

The rapid development of Multimodal Large Language Models (MLLMs), such as BLIP-2 [29], LLaVA [33], Flamingo [2], and Qwen2-VL [51], have significantly advanced vision–language reasoning across a wide range of tasks. Applied to GUI understanding, these models exhibit strong zero-shot and few-shot capabilities. Domain-specific systems such as CogAgent [19] further improve performance by incorporating UI-oriented pretraining and task design.

Despite their representational capacity, most MLLMs operate under static model parameters and closed training distributions [27]. Rapid iteration in mobile UI design introduces new visual patterns and interaction paradigms that may not be covered during training, leading to mismatches between appearance and functionality. Retrieval-augmented generation (RAG) [18, 21, 54] offers a mechanism to incorporate external knowledge at inference time, but retrieval based solely on visual similarity can conflate visually similar yet functionally distinct components. Our work addresses this challenge by introducing a dual-branch retrieval strategy that jointly considers visual appearance and outline semantic, improving robustness for component description generation in evolving GUI environments.

3 FORMATIVE INTERVIEW STUDY

3.1 Participants

We recruited 8 BVI participants with varying levels of technical proficiency. As summarized in Table 1, the cohort encompasses a diverse range of ages, occupations, and technical backgrounds, providing a holistic view of the current barriers in mobile interaction.

- **Demographics:** Participants were aged between 18 and 40, including 3 full-time students and 5 working professionals (e.g., Teacher, Trainer, Accessibility Tester).
- **Technical Proficiency:** All participants are daily users of mobile applications, utilizing both Android (Xiaomi, Huawei, Redmi) and iOS platforms.
- **Assistive Tools:** They primarily rely on screen readers such as TalkBack (Android), VoiceOver (iOS), Tiantan [1] to interact with their devices.

Table 1: Demographic and Technical Profile of the Interview Participants (B: Blind, LV: Low Vision).

ID	Age	Visual	Occupation	Main Device/OS	Tools
P1	21	B	Student	Redmi K70 / And.	Tiantan
P2	22	B	Student	Huawei / And.	Tiantan
P3	25	B	Student	Xiaomi 12 / And.	TalkBack
P4	35-40	LV	Teacher	iPhone / iOS	VoiceOver
P5	31	B	Acc. Tester	iPhone 16P / iOS	VoiceOver
P6	28	B	Teacher	iPhone 16PM / iOS	VoiceOver
P7	29	B	Trainer	iPhone 15P / iOS	VoiceOver
P8	30-40	B	Teacher	iPhone 14 / iOS	VoiceOver

3.2 Semi-structured Interview Design

To gain a deep understanding of the practical challenges faced by the BVI community, we adopted a semi-structured interview format. This approach was chosen because it provided a consistent framework across all eight sessions while allowing the interviewer the flexibility to follow up on specific personal anecdotes, such as P8's difficulties with banking apps or P4's frustrations with educational platforms. Each session was conducted anonymously to ensure that participants felt comfortable sharing their honest, and often critical, experiences with current technology. The interviews lasted approximately 25 minutes and were recorded and transcribed for thematic analysis. The interview script was carefully organized into the following three progressive phases:

Current Assistive Tool Habits: This opening phase assessed the participants' history and daily routines with assistive technologies. Rather than a binary inquiry into tool usage, we investigated the "how" and "why" behind their choices. We explored the specific screen readers used—ranging from native TalkBack (Android) and VoiceOver (iOS) to localized third-party tools like Tiantan. For experienced users such as P1 (7-8 years of usage), we investigated their perceived evolution of these tools and how their device preferences (e.g., Xiaomi, Huawei, or Apple) influence their daily efficiency.

[1] https://www.tatans.cn/product.html

Detailed Identification of Interaction Barriers: In the second phase, we asked participants to recall specific instances where their digital journey was interrupted. We focused on three types of failures:

- *The "Unlabeled" Problem:* How often they encounter the "unlabeled button" announcement and the mental effort required to guess its function.
- *Recognition Inaccuracy:* We specifically followed up on the "guessing" behavior of current AI features, such as the reported 50% error rate in image recognition.
- *Contextual Logic Failures:* We discussed scenarios where a description might exist but is placed in the wrong order or fails to explain the logic of a complex page, such as a multi-step checkout process in a shopping app or a sliding verification puzzle.

By asking for specific app examples, we identified that barriers are most severe in "high-stakes" apps, where a single wrong tap could lead to unintended financial transactions or data loss.

Expectations for Ideal GUI Descriptions: The final phase was forward-looking, designed to extract the "Gold Standard" of what a description should contain. We asked users to ignore current technical limitations and describe their "ideal" auditory feedback when focusing on a non-text element. This is where the concepts of [Identity] and [Action] emerged. We asked whether they preferred to know the physical shape of an icon (e.g., "a star") or its functional result (e.g., "add to favorites"). We also explored their tolerance for sentence length, given that many users listen at extremely high speeds. These questions were crucial in determining that BVI users prioritize "functional certainty" over "artistic detail," wanting to know exactly what a component represents and what will happen the moment they perform a gesture.

3.3 Key Findings and Design Rationale

Through thematic analysis of the interview transcripts, we identified several critical requirements that directly shaped the design of the *GUI-CD* dataset.

3.3.1 The Necessity of [Identity] and [Action]. The most prominent finding was that BVI users suffer from a "semantic void" when encountering graphical elements.

- **[Identity] - What is it?** Participants (P2, P3, P5) emphasized the frustration of hearing "unlabeled button." P5 specifically noted that without a name, they cannot distinguish between similar-looking icons (e.g., different types of "arrows"). This confirms our design that every description must first establish the component's **Identity**.
- **[Action] - What does it do?** Participants (P4, P6, P8) highlighted that knowing the icon's shape is secondary to knowing its outcome. P6 stated, "I need to know what will happen after I click it, not just that it's a gear icon." This feedback led us to include a mandatory **Action** field in our dataset template to explain the interactive consequence (e.g., "used to open settings").

3.3.2 Information Density and Structural Sequence. A common assumption in accessibility design is that alternative text must be as brief as possible to save time. However, a significant finding from

our interviews (P1, P8) suggests that for proficient screen reader users, the absolute word count is a secondary concern. P1 specifically observed that because experienced users often set their screen readers to extremely high speeds—sometimes processing several hundred words in just a few seconds—the primary bottleneck is not the duration of the speech, but the cognitive effort required to parse an inconsistently structured label. When labels vary randomly in their phrasing (e.g., switching between "Search icon," "Click here to find products," and "Magnifying glass"), the user must actively listen to every word to extract the component's purpose.

Rationale for Standardized Templates: To minimize this cognitive friction, we deliberately moved away from fragmented labels in favor of a highly structured descriptive template: *"The functional description of this image is that it represents [Identity], which is used to [Action]."* This design serves a specific auditory function: the repetitive prefix acts as a "fixed carrier sentence." Much like how a sighted user skims a page for bold headers, a BVI user can learn to mentally bypass the familiar carrier sentence and focus their attention exclusively on the varying semantic tokens: [Identity] and [Action].

The decision to place [Identity] before [Action] is also evidence-based. Participants (P2, P3, P5) repeatedly expressed a sense of exclusion when hearing "unlabeled button," as it provides no mental image of the interface. By establishing the *Identity* first (e.g., "a gear-shaped icon"), we provide the user with a spatial and visual anchor of what is on the screen, fulfilling the "perceivability" requirement. Immediately following this with the *Action* (e.g., "used to open system settings") fulfills the "operability" requirement by explaining the consequence of interaction. This sequence ensures that the user receives the "what" and the "why" in a predictable order, allowing them to make rapid navigation decisions without waiting for a verbose, unstructured sentence to conclude.

3.3.3 Reliability and the Need for Knowledge Augmentation. The semi-structured interviews highlighted a significant problem regarding the reliability of current automated accessibility tools. Throughout the sessions, the most frequent complaint from the eight participants was the "unpredictability" of automatic descriptions. P1 pointed out a stark reality: in daily use, the error rate of image recognition features in mainstream screen readers is often as high as 50%. For a sighted person, a wrong description of a photo is merely a minor error, but for a BVI user, a wrong description of a functional button can lead to serious consequences. This frequent failure creates a situation where users feel they cannot fully trust their devices, especially when using unfamiliar apps.

Participants P5 and P8 shared specific anxieties about using high-stakes applications, such as mobile banking or government service platforms. They explained that if an icon is misidentified—for instance, if a "Delete" button is described as "Save" or "Next"—it can lead to irreversible mistakes, such as losing important data or mismanaging personal finances. Because current AI models often "guess" the meaning of an icon based on its appearance, they are prone to making mistakes when an app developer uses a non-standard design. This deficiency in reliability is what prevents many BVI users from exploring new digital services independently, as they often have to wait for a sighted person to help them confirm the button's actual purpose.

Rationale for Knowledge Augmentation via RAG: These insights directly led to our decision to use a Retrieval-Augmented Generation (RAG) approach rather than relying solely on a fixed AI model. The fundamental problem with a standard model is that it only knows what it was taught during its original training. When a new app comes out with a unique icon style, the model has no way to "look up" the correct answer and instead creates a description that might sound confident but is factually wrong. By introducing a knowledge-based system, we ensure that the model's output is anchored in a library of real-world GUI examples.

Our dual-branch design looks at both the visual style and the physical shape of an icon. This was our way of solving what P6 called 'visual confusion. P6 mentioned that many icons look exactly the same but perform completely different actions depending on which app is being used. For example, if the system encounters a brightly colored "Home" icon it has not seen before, the visual branch may have difficulty processing the unfamiliar color.But the text branch will recognize the familiar triangular roof and square base, allowing the system to find "Home" icons in its database and provide an accurate description. This method of "searching and comparing" rather than "guessing" is specifically designed to give BVI users the functional certainty they asked for during our interviews. It transforms the AI from an unreliable guesser into a tool that provides evidence-based information, directly addressing the safety and reliability concerns raised by the participants.

4 METHODOLOGY

The objective of our framework is to bridging the semantic void of raw GUI pixels with reliable semantic feedback. As illustrated in Figure 1, we propose a pipeline that transitions from offline data governance to online knowledge-augmented inference.

4.1 Problem Formulation

The core objective of GUI component description is to translate raw digital interface elements into accessibility-oriented natural language descriptions. Formally, we define a GUI component c through a multi-modal input tuple $\mathbf{X} = (I, S, M)$. Here, I represents the local visual crop of the target component, S denotes the global screenshot which provides essential layout and hierarchical context, and M signifies the structural metadata, including normalized bounding box coordinates and component class types (e.g., *button, checkbox*).

The task is to learn a mapping function $\mathcal{F} : (I, S, M) \rightarrow Y$, where $Y = \{y_1, y_2, \ldots, y_T\}$ is a sequence of linguistic tokens that describe the component's identity and function. In practical accessibility scenarios, this mapping is hindered by two primary challenges. First, *label heterogeneity* across existing datasets leads to inconsistent and messy annotations, which prevents the model from learning a standard way to describe components. Second, the *static knowledge boundary* of pre-trained weights makes it difficult for the model to describe new UI designs that were not included in the training data.

To overcome these challenges, we decompose the generative process into two optimization sub-tasks:

(1) **Parametric Alignment:** Use an automated pipeline to unify labeling styles and train the model with Parameter-Efficient

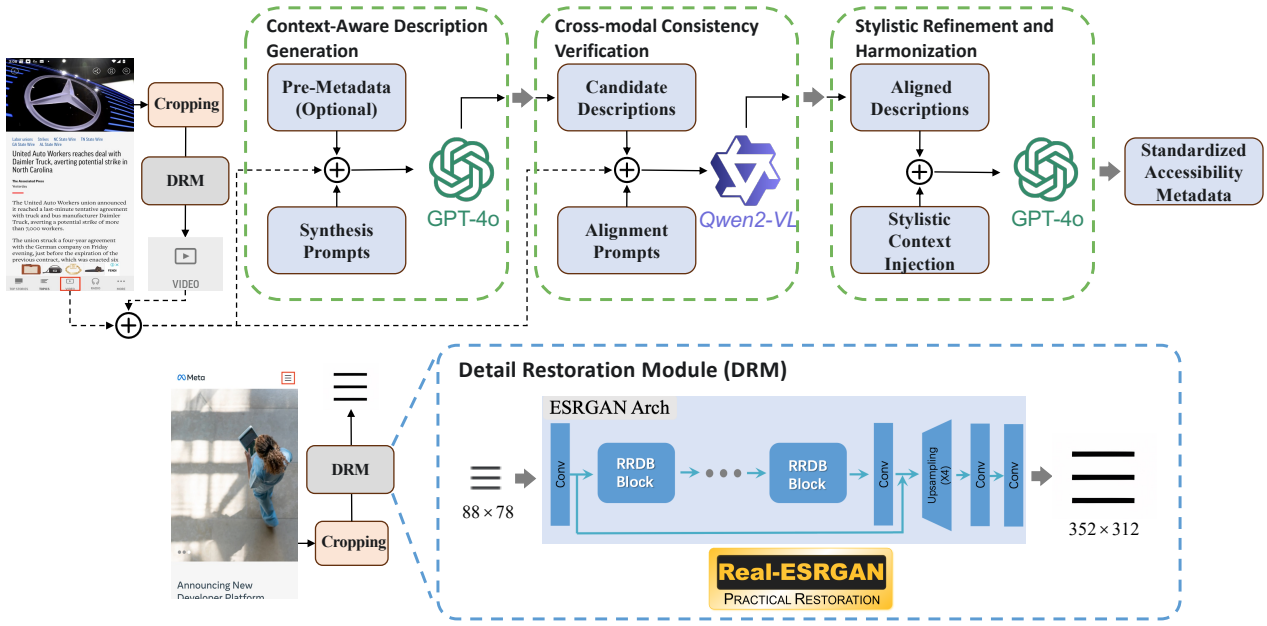

Figure 1: The automated pipeline for GUI component description unification. Raw screenshots are processed through a super-resolution DRM module and multi-stage refinement (generation, verification, and harmonization) involving Multimodal Large Language Models (MLMMs) to generate standardized, accessibility-oriented metadata.

Fine-Tuning (PEFT) by minimizing the negative log-likelihood to learn how to describe GUI components.

(2) **Non-parametric Augmentation:** Enhance the model's descriptive capacity through cross-modal retrieval over repositories D_{img} and D_{text}. By forming retrieved context set C during inference, the model utilizes external information to accurately describe new components.

4.2 Automated Data Synthesis and Unification

To build a generative model that provides reliable description for BVI users, the quality of the training data is paramount. Existing datasets like RICO [13] often contain conflicting or overly brief labels that fail to explain the actual function of a component. For instance, a 'magnifying glass' icon might be labeled by its visual shape, its function as 'search,' or a meaningless name like 'icon1.' This makes feedback unpredictable, because BVI users cannot consistently know what an icon does when different apps describe it in different ways. To solve this, we designed an automated pipeline (as shown in Figure 1) that combines visual restoration with a three-stage refinement process involving GPT-4o [22] and Qwen2-VL [51]. This ensures that every component in our *GUI-CD* dataset is associated with a high-quality description that follows the [Identity] and [Action] requirements identified in our interview study, while providing a training signal that is both visually grounded and linguistically consistent with the Web Content Accessibility Guidelines (WCAG) [12].

4.2.1 Detail Restoration Module (DRM). One of the biggest technical challenges in GUI analysis is the low resolution of individual icons. Mobile UI elements are often stored as tiny bitmaps. When these are cropped directly from a screenshot, a 88×78 pixel icon becomes a low-fidelity visual feature, in which fine lines and distinctive shapes are lost.

To address this challenge, we implemented the Detail Restoration Module (DRM). Instead of using standard resizing methods which only make the blur bigger, we use the Real-ESRGAN architecture [52]. This choice is based on its practical ability to recover sharp edges and remove digital artifacts. As illustrated in the bottom part of Figure 1, the module utilizes Residual-in-Residual Dense Blocks (RRDB). These blocks allow the network to learn complex spatial features and reconstruct the original sharp vectors of the UI design.

For an input crop $I_{crop} \in \mathbb{R}^{H \times W \times 3}$, the restored image $I_{restored}$ is obtained by:

$$I_{restored} = \mathbf{G}_{sr}(I_{crop}; \theta_{sr}) \tag{1}$$

where $\mathbf{G}_{sr}$ is the generator network upsampling the input by a factor of 4×. By performing a 4× upsampling, we transform a low-quality 88×78 crop into a high-fidelity 352×312 image. This restoration is crucial because it provides the subsequent multimodal large language models (MLLMs) with a clear visual foundation to identify the component's [Identity].

4.2.2 Multi-stage Annotation Refinement Pipeline. Following the visual restoration of the component, we implement a multi-stage refinement pipeline to transform the high-fidelity image into standardized, accessibility-oriented metadata. Rather than relying on a single generative pass, this process is structured as a sequential workflow (see the green modules in Figure 1) to ensure that the final descriptions are both logically sound within the application's context and factually accurate based on the visual appearance.

Stage 1: Context-Aware Description Generation. The first stage is about understanding the "logic" behind an icon. In a mobile app, the meaning of a component often depends on what is around it. For example, a simple "plus" sign icon could mean "add a friend" in a social app, but "increase quantity" in a shopping app. To capture this correctly, we provide GPT-4o with the restored image $I_{restored}$ along with its metadata, such as its class name and the text labels near it. By looking at these multi-source features, the model can infer the specific *functional affordance* of the element. This ensures that the generated [Action] matches the real purpose of the button in that specific app context, rather than just being a generic guess.

Stage 2: Cross-modal Consistency Verification. The second stage is designed to prevent incorrect descriptions where the generated text describes a function that does not match the component's visual appearance. We use Qwen2-VL as a visual auditor to compare the text description from Stage 1 with the actual pixels of the upsampled component. It checks if the [Identity] mentioned in the text (like "a house icon") really matches what is visible in the image. If there is a mismatch, the sample is removed from the dataset. This verification step ensures that our dataset is grounded in visual reality and is safe for blind users to rely on.

Stage 3: Stylistic Refinement and Harmonization. The final stage is focused on making the descriptions consistent. Our interview study showed that BVI users prefer a predictable pattern so they can find information quickly. When every app uses a different way of speaking, it increases the user's cognitive load. We use GPT-4o in this stage to normalize the language of all the labels we have verified. We force every description to follow a strict and standardized template: *"The functional description of this image is that it represents [Identity], which is used to [Action]."* By using this fixed structure, as detailed in Figure 2, we allow users to quickly skip the "intro" part of the sentence and focus their attention only on the key information: what the icon is and what it does. This standardization turns a messy collection of labels into a high-quality corpus that supports efficient navigation.

4.3 Domain-Specific Adaptation via LoRA

We utilize Qwen2-VL as our backbone model. Unlike standard Vision Transformers (ViT), Qwen2-VL supports *Naive Dynamic Resolution*, allowing it to process input sequences of varying lengths corresponding to the aspect ratio of GUI components.

To specialize the model for the GUI domain without catastrophic forgetting, we apply Low-Rank Adaptation (LoRA) [20]. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, the weight update is constrained by a low-rank decomposition:

$$W = W_0 + \Delta W = W_0 + BA \tag{2}$$

where $B \in \mathbb{R}^{d\times r}$ and $A \in \mathbb{R}^{r\times k}$ are trainable parameters with rank $r \ll \min(d, k)$. We specifically target the query (W_q) and value (W_v) projection matrices in the self-attention layers of the transformer. This approach allows the model to learn the relationship between specific visual patterns of GUI icons and their functional actions.

```
You are an AI assistant creating standardized
accessibility label from GUI description. Your goal is
to ensure clarity and consistency for visually impaired
users.

## Your Task
1. Parse the input text to identify two key pieces of
information:
  * The name of the component (the `[Identity]`).
  * The primary function it performs (the `[Action]`).
2. Reconstruct these two pieces of information into the
mandatory output format.

## Mandatory Output Format
The output MUST strictly adhere to this template:
`The functional description of this image is that it
represents [Identity], which is used to [Action].`

## Critical Rules
* Factual Integrity: Do NOT add or invent information.
All information must be extracted from the input text.
* Error Handling: If you cannot confidently extract BOTH
a clear `[Identity]` AND its `[Action]` from the input
text, you MUST output the exact string: `ERROR:
INSUFFICENT INFORMATION`.
* No Commentary: Your response MUST contain ONLY the
final formatted string OR the error string. Do not
include any other text.

## Step-by-Step Examples
---
Example 1:
* Input TEXT: "A cafe icon labeled “Cafe” that opens the
cafe section when activated."
* Thinking Process:
    1. `[Identity]` = "a cafe icon"
    2. `[Action]` = "open the cafe section"
    3. Assemble into the template.
* Required Output: `The functional description of this
image is that it represents a cafe icon, which is used
to open the cafe section.`

---
Example 2:
* Input TEXT: "This is the settings button, shaped like
a gear. It navigates to the settings screen."
* Thinking Process:
    1. `[Identity]` = "a settings button"
    2. `[Action]` = "navigate to the settings screen"
    3. Assemble into the template.
* Required Output: `The functional description of this
image is that it represents a settings button, which is
used to navigate to the settings screen.`

---
Example 3 (Error Handling):
* Input TEXT: "A decorative background image with blue
gradients."
* Thinking Process:
    1. `[Identity]` = "A decorative background image"
    2. `[Action]` = Not found. The component has no
function.
    3. Condition for error is met. Output the error
string.
* Required Output: `ERROR: INSUFFICIENT INFORMATION`.
---
```

Figure 2: The Chain-of-Thought (CoT) prompt template used for Stylistic Refinement and Harmonization.

4.4 Dual-branch Cross-modal Retrieval Enhancement

Static models fail to adapt to new UI design trends (e.g., the transition from skeuomorphism to flat design). We propose a dual-branch RAG framework to inject real-time knowledge into the generation process.

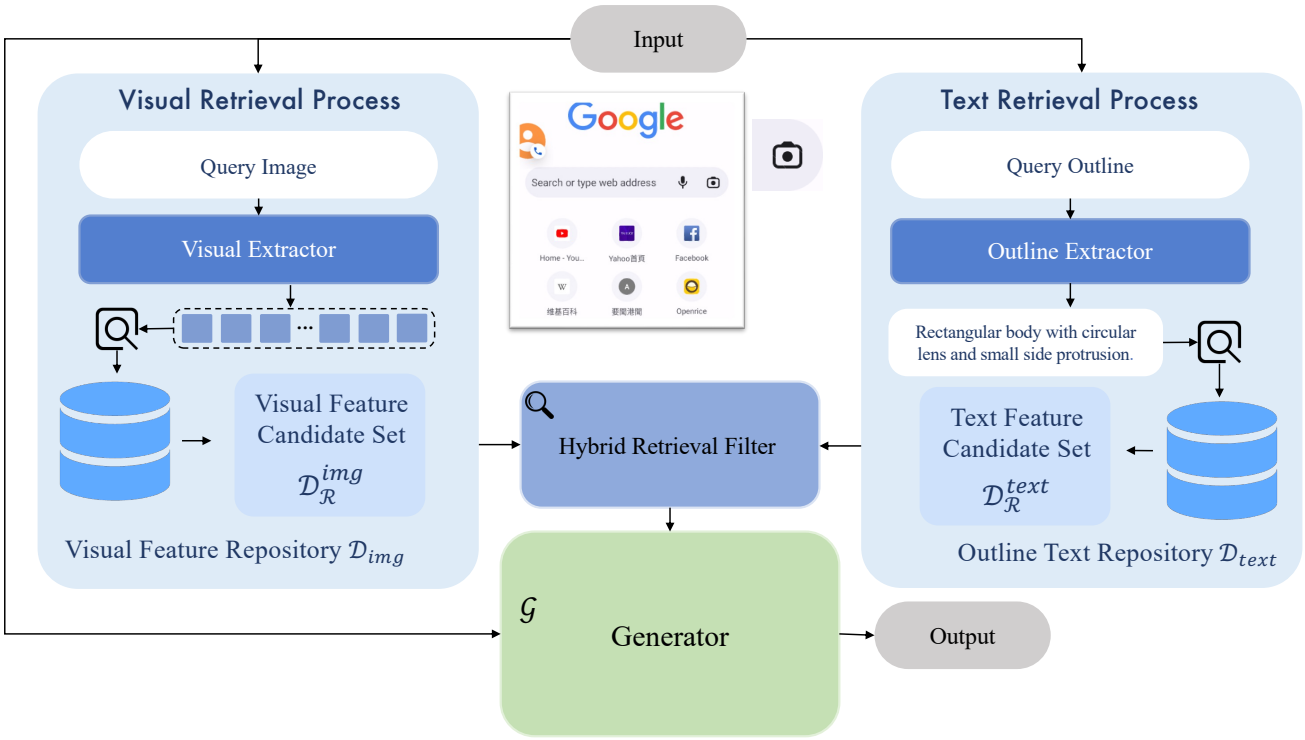

Figure 3: The proposed Dual-branch RAG framework for GUI description. The system processes a query image through two parallel paths: the Visual Retrieval Process (left) and the Outline-based Text Retrieval Process (right). A Hybrid Retrieval Filter is then used to choose the most accurate candidates to guide the final description.

4.4.1 Visual Retrieval Process. The purpose of the visual branch is to find components in our library that look similar to the one currently focused on by the user. For this process, we use a pre-trained CLIP [39] (Contrastive Language-Image Pre-training) model as our Visual Extractor. CLIP encodes GUI icons as vectors in a shared embedding space, such that images with similar visual features are closer together.

Specifically, for any query component image q, the CLIP vision encoder E_{clip} is used to turn the raw pixels into a high-dimensional feature vector $\mathbf{z}_q$. To make the retrieval fast, we processed a subset of components in our *GUI-CD* dataset and stored their vectors in the Visual Feature Repository D_{img}. We use Approximate Nearest Neighbor (ANN) search [30] to compare the query vector against this repository. The similarity between the query and each stored item is calculated using the cosine distance formula:

$$\mathcal{D}_R^{img} = \text{Top-K}\left(\frac{\mathbf{z}_q \cdot \mathbf{v}_i}{\|\mathbf{z}_q\|\|\mathbf{v}_i\|}\right), \forall \mathbf{v}_i \in D_{img} \tag{3}$$

The result of this calculation is the Visual Feature Candidate Set $\mathcal{D}_R^{img}$, which is a list of existing GUI elements that most closely resemble the query in terms of color, texture, and basic shape. This branch provides the system with a set of visual examples that the model can use to compare against the structural information from the text branch.

4.4.2 Text Retrieval Process. The second branch of our retrieval framework is the text branch. This branch is designed to provide a structural backup when an icon's visual style is too unusual for the visual extractor to recognize. As mentioned in our interviews, while the colors and artistic themes of apps change frequently, the basic physical shape of a button usually stays the same to remain recognizable. To use this stability, we designed Outline Extractor model. This model uses an encoder-decoder structure and is specifically trained to describe the physical geometry of a GUI component in plain text.

The process for this branch consists of two main steps:

(1) **Outline Text Generation:** The custom model takes the query image as input and outputs a string of text T_{out} that describes its physical outline. The model is trained to ignore visual details like color, shadows, or textures and focus only on the shapes.
(2) **Text-based Knowledge Retrieval:** Once the outline description T_{out} is generated, the system uses it as a query to search the Outline Text Repository D_{text}. This repository is a database that contains outline descriptions of GUI components and links them to their verified [Identity] and [Action] labels.

By searching based on the physical outline instead of the raw pixels, the system can correctly identify the purpose of a button even if its visual theme is completely new or out-of-distribution. This process

results in the Text Feature Candidate Set $\mathcal{D}_R^{text}$. This set provides the generator with structural evidence, which acts as a reliable reference to ensure the final description is functionally accurate for BVI users.

Algorithm 1 Dual-branch Cross-modal Retrieval Pipeline

Require: Query image I_q, Repositories D_{img}, D_{text}
Ensure: Final retrieved context set C
// Branch 1: Visual Retrieval Process
$\mathbf{z}_q \leftarrow \text{Visual\_Extractor}(I_q)$
$\mathcal{D}_R^{img} \leftarrow \text{ANN\_Search}(\mathbf{z}_q, D_{img}, \text{Top-}K)$
// Branch 2: Text Retrieval Process
$T_{out} \leftarrow \text{Outline\_Extractor}(I_q)$
$\mathcal{D}_R^{text} \leftarrow \text{Text\_Search}(T_{out}, D_{text}, \text{Top-}K)$
// Aggregation and Filtering
$\mathcal{R}_{all} \leftarrow \mathcal{D}_R^{img} \cup \mathcal{D}_R^{text}$
$C \leftarrow \text{Hybrid\_Retrieval\_Filter}(\mathcal{R}_{all}, I_q)$
return C

4.4.3 Hybrid Retrieval Filter. After collecting potential matches from both retrieval paths, the system needs a reliable way to select the most accurate information. To do this, we first aggregate the raw results produced by the two branches: Visual Feature Candidate Set $\mathcal{D}_R^{img}$ and Text Feature Candidate Set $\mathcal{D}_R^{text}$. These sets are combined into a single candidate pool $\mathcal{R}_{all}$:

$$\mathcal{R}_{all} = \mathcal{D}_R^{img} \cup \mathcal{D}_R^{text} \tag{4}$$

This pool is then sent into the Hybrid Retrieval Filter (located in the center of Figure 3). The filter acts as the final decision-maker that evaluates and ranks all the suggestions.

The filter is designed as a 3-layer Transformer model specifically for ranking tasks. It takes each candidate from the pool $\mathcal{R}_{all}$ and calculates an independent score based on how well its features match the icon currently focused on by the user. By processing this data through its three layers, the filter identifies which [Identity] and [Action] pairs are factually correct. Based on these scores, the filter selects the Top-K most reliable samples to form the final retrieved context set C. The pipeline of Dual-branch Cross-modal Retrieval is summarized in Algorithm 1.

4.4.4 Knowledge Fusion and Joint Inference. The final stage of our framework is the knowledge fusion process, where the retrieved evidence is combined with the original input. Instead of relying solely on what the model "remembers" from its training phase, our system performs "open-book" reasoning by looking at verified examples from the retrieved context set C. These candidates are placed into Retrieval-Augmented Prompt, providing the model with clear references for the component's [Identity] and [Action].

The fine-tuned generator $\mathcal{G}$ then performs joint attention over the input image I_q and the evidence in C to produce the final output:

$$y_t = \arg\max_{y \in \mathcal{V}_{vocab}} P(y \mid y_{<t}, I_q, C; \Theta_{LoRA}) \tag{5}$$

By grounding the generation process in this external, filtered knowledge, the system can accurately describe new or unfamiliar app components. This fusion ensures that the final auditory feedback is not a guess but is based on verified real-world examples, following the standardized structure requested by BVI users during our interviews.

5 EXPERIMENTS

In this section, we conduct extensive evaluations to assess the performance of our proposed framework. We begin by describing the experimental setup, followed by a comparative analysis against state-of-the-art baselines. Finally, we provide ablation studies and qualitative examples to demonstrate the efficacy of our dual-branch RAG mechanism and standardized data unification.

5.1 Experimental Setup

5.1.1 Datasets. We evaluate our model on three benchmark datasets representing different GUI complexities:

- **Widget Caption [31]:** A benchmark dataset derived from RICO [13], consisting of 61,285 UI components and 162,859 human-annotated descriptions. Each component typically possesses 2 to 5 distinct natural language labels.
- **AMEX [7]:** A large-scale, high-resolution dataset containing over 104,000 screenshots with fine-grained functional descriptions. It serves as our primary benchmark for complex semantic understanding.
- **GUI-CD (Ours):** Our unified dataset synthesized from Widget Caption and AMEX via the pipeline described in Section 4.2, ensuring stylistic consistency.

5.1.2 Evaluation Metrics. To quantify the linguistic fluency and semantic accuracy of the generated accessibility labels, we utilize a suite of standard automated metrics: **BLEU-2** [38], **ROUGE-L** [32], and **CIDEr** [48]. While BLEU and ROUGE-L effectively evaluate surface-level n-gram overlap and structural sequence consistency, we prioritize CIDEr as our primary performance indicator. This decision is based on CIDEr's ability to utilize Term Frequency-Inverse Document Frequency (TF-IDF) weighting, which emphasizes rare but informative functional keywords. In the context of assistive technology, this metric more accurately captures the consensus between the model's output and human annotations, particularly regarding the component's [Identity] and [Action]. By rewarding the correct identification of these critical information dimensions, CIDEr provides a reliable proxy for the "functional certainty" required by BVI users during mobile navigation.

5.1.3 Baselines. We compare our approach against two groups of state-of-the-art models to evaluate performance across general vision-language knowledge and specialized GUI understanding:

- **General MLLMs:** We select BLIP2-2.7B [29] and InternVL2-8B [9] as representatives of high-capacity foundation models to assess how general vision knowledge performs on digital interfaces. We also include the Qwen2-VL-7B [51], which serves as the original backbone of our framework, to measure the specific improvements gained from our domain adaptation and RAG modules.
- **GUI-Specialized Models:** We select Pix2Struct-large [26], a model pre-trained on structural parsing tasks to internalize UI layouts. Additionally, we include CogAgent [19],

which features a high-resolution visual encoder for perceiving micro-scale icons, and OmniParser [49], a framework designed to identify interactive components and their functional descriptions.

5.2 Implementation Details

The model is implemented using the PyTorch framework. For LoRA fine-tuning, we set the rank $r = 8$ and the scaling factor $\alpha = 16$. The training is conducted with a learning rate of 1×10^{-5} using the AdamW [34] optimizer and a cosine learning rate scheduler. All experiments are performed on a server equipped with 8 × NVIDIA RTX 3090 Ti GPUs. For the RAG module, the initial candidate pool size is set to 20, and the final top-$K = 5$ context samples are injected into the generator.

5.3 Main Results and Comparative Analysis

The performance of our proposed framework is evaluated against a wide range of baselines on the Widget Caption and AMEX benchmarks, as well as our unified *GUI-CD* dataset. The quantitative results presented in Table 2 and Table 3.

Table 2: Performance comparison on Widget Caption and AMEX datasets. B-2, R-L, and CIDEr denote BLEU-2, ROUGE-L, and CIDEr, respectively. Bold and underlined entries indicate the best and second-best performance.

Model	Widget Caption			AMEX		
	B-2	R-L	CIDEr	B-2	R-L	CIDEr
BLIP2-2.7B [29]	1.2	3.5	4.1	4.2	9.8	14.2
InternVL2-8B [9]	11.7	18.8	62.9	2.1	18.8	45.3
Qwen2-VL-7B [51]	12.9	27.2	92.6	12.9	27.2	94.6
Pix2Struct-large [26]	17.1	24.5	96.3	1.7	21.6	50.1
CogAgent [19]	12.4	13.1	46.2	19.1	<u>43.8</u>	93.3
OmniParser$_{blip2}$ [49]	<u>18.4</u>	<u>33.5</u>	<u>126.2</u>	<u>20.2</u>	39.4	<u>132.5</u>
Ours (LoRA + RAG)	**21.5**	**38.2**	**143.5**	**39.5**	**52.8**	**185.3**

5.3.1 Performance on Public Benchmarks. As shown in Table 2, our framework (LoRA + RAG) achieves the highest scores across all metrics on both the Widget Caption and AMEX datasets. On the AMEX dataset, which is characterized by high-resolution interface designs and complex functional logic, our model achieves a CIDEr score of **185.3**. Compared to the Qwen2-VL-7B score of 94.6, these results indicate that our domain-specific adaptations are effective in improving descriptive quality.

For BVI users, this means that screen reader feedback is more likely to match the actual purpose of an icon, which is essential for providing functional certainty. While specialized baselines such as OmniParser and CogAgent demonstrate strong results in UI parsing and coordinate detection, their generated descriptions often lack sufficient semantic specificity. This gap exists because those models are primarily optimized for action-execution tasks (e.g., clicking) rather than providing the semantic detail needed for assistive technology.

Table 3: Performance comparison on the proposed GUI-CD dataset. The models are categorized into General MLLMs and GUI-Specialized Models to highlight the domain gap. Bold and underlined entries indicate the best and second-best performance.

Model	Type	BLEU-2	ROUGE-L	CIDEr
BLIP2-2.7B [29]	General	4.8	11.9	7.7
InternVL2-8B [9]	General	0.2	6.4	1.1
Qwen2-VL-7B [51]	General	10.8	24.6	23.5
Pix2Struct-large [26]	GUI-Spec.	0.2	8.1	1.2
CogAgent [19]	GUI-Spec.	<u>31.2</u>	<u>53.2</u>	<u>104.8</u>
OmniParser$_{blip2}$ [49]	GUI-Spec.	8.6	16.8	73.2
Ours (LoRA+RAG)	GUI-Spec.	**35.4**	**53.9**	**114.8**

5.3.2 Impact of Data Quality on the GUI-CD Dataset. To analyze how annotation quality influences model behavior, we evaluated all models on our standardized *GUI-CD* dataset. As categorized in Table 3, the results highlight a clear performance gap between general-purpose foundation models and specialized GUI architectures.

The results show that General MLLMs, such as InternVL2-8B and BLIP2-2.7B, fail to generate accurate GUI descriptions, with CIDEr scores near 1.0. This confirms that general vision training is insufficient for interpreting the symbolic logic of mobile software. Without the parametric alignment provided by our standardized training set, these models cannot resolve the ambiguity of UI elements, often leaving the user in a "semantic void."

Among GUI-Specialized Models, CogAgent is the second-best performer with a CIDEr score of 104.8. Its high-resolution vision encoder provides a clear advantage in perceiving micro-scale icons. However, our proposed framework outperforms CogAgent by 10.0 CIDEr points. This difference indicates that visual resolution is only one part of the problem, the more significant factor is the consistency of the descriptive logic. By training on unified labels that follow a strict [Identity] + [Action] structure, our model learns to produce predictable and reliable descriptions that match the interaction patterns identified in our interviews.

The consistent performance across both tables validates our two-stage strategy. The first stage addresses labeling inconsistencies through the data unification pipeline, while the second stage overcomes the knowledge boundary of static weights through the dual-branch RAG module. This combination ensures the generator provides accurate feedback for both standard UI patterns and new designs that emerge after the initial training phase.

5.4 Ablation Study

To evaluate the specific contribution of each retrieval branch in our system, we conducted a series of ablation experiments on the AMEX dataset. We focused on how the visual branch and the text branch interact to improve the final auditory descriptions. Since the accuracy of the generated labels depends on the quality of the retrieved information, we introduce two auxiliary metrics to measure retrieval performance independently before the generation stage.

- **Recall@5 (R@5):** This metric measures the percentage of cases where the correct functional component is successfully found within the top-5 retrieved candidates. It represents the coverage of our knowledge repositories.
- **Precision@5 (P@5):** This calculates the proportion of relevant candidates among the top-5 results. It indicates the purity of the candidate set that is passed to the Hybrid Retrieval Filter.

Table 4: Ablation study of the dual-branch retrieval architecture on AMEX dataset.Bold entries indicate the best performance.

Retrieval Strategy	R@5 (%)	P@5 (%)	CIDEr
(a) Visual Branch Only	75.1	78.3	182.8
(b) Text Branch Only	72.4	74.9	180.5
(c) Full Dual-branch RAG	**82.4**	**85.1**	**185.3**

The results of these experiments are summarized in Table 4. As shown in row (a), relying only on the visual branch provides a baseline CIDEr score of 182.8. This branch is effective at identifying standard icons through their colors and textures. However, it is sensitive to changes in artistic style. When an application uses a new visual theme or a custom design, the visual similarity often drops. This sensitivity explains the Recall@5 of 75.1% for this branch. Row (b) shows the text branch, which focuses on geometric shapes. While its CIDEr score is lower at 180.5, this branch is much more robust against artistic changes. It identifies the component's [Identity] by analyzing its physical structure. This provides a reliable backup when the visual appearance is unfamiliar to the model.

The full dual-branch framework in row (c) achieves the highest performance across all metrics. By combining both branches, the system reaches a Recall@5 of 82.4% and a CIDEr score of 185.3. This performance gain proves that visual appearance and physical structure provide complementary evidence. In this full setup, the Hybrid Retrieval Filter acts as the final decision-maker. It ranks candidates that are supported by both visual and structural dimensions.

For BVI users, the dual-branch is crucial for providing functional certainty. When the visual look of an icon is confusing, the outline information ensures that the model still understands what the component is. This mechanism effectively overcomes the internal knowledge boundary of the pre-trained model. By using these dual anchors, the system ensures that the final [Action] generated for the screen reader is grounded in verified real-world examples. This reduces the risk of functional misidentification and provides a more reliable navigation experience.

5.5 Sensitivity Analysis: Impact of Top-K

The number of retrieved knowledge pieces, denoted as K, is a key factor that balances information richness and contextual noise. If K is too small, the generator may not receive enough functional evidence to accurately describe a new component. Conversely, if K is too large, the input prompt may become overly cluttered with irrelevant information, leading to confusion for the model. To find the optimal balance, we conducted a sensitivity experiment on the AMEX dataset, testing values of $K \in \{1, 3, 5, 7, 10\}$.

As illustrated in Figure 4, we observe two distinct trends in the results. First, the Recall@5 (represented by the orange line) shows a steady upward trend as K increases. This is expected, as searching for more candidates in the library increases the statistical probability that the correct [Identity] and [Action] pair will be included in the candidate set.

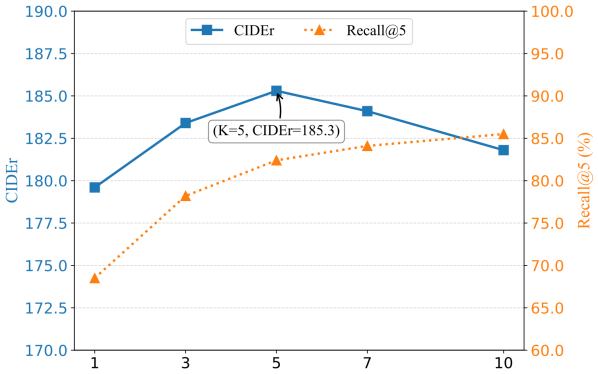

Figure 4: Impact of the number of retrieved knowledge pieces (Top-K) on performance. The graph shows Recall@5 (retrieval coverage) and CIDEr (generation quality) on the AMEX dataset.

However, the generation quality, measured by the CIDEr score (the blue line), follows a different pattern. As K moves from 1 to 5, the CIDEr score increases rapidly, reaching its peak at 185.3 when $K = 5$. This indicates that providing a moderate amount of external evidence helps the model significantly in grounding its descriptions in real-world GUI patterns.

Interestingly, when K increases further to 7 and 10, the CIDEr score begins to decline. We believe this happens because as we add more candidates, the system inevitably introduces "semantic noise"—examples that look similar visually but perform different functions. These irrelevant examples distract the generator's attention mechanism, making it harder for the model to pick the single correct answer.

This experiment proves that more information is not always better for BVI users. Providing a "noisy" or confusing description can be harmful to navigation safety. Based on these findings, we set $K = 5$ as the default value for our framework. This setting provides enough functional evidence to ensure functional certainty while keeping the description clean and accurate.

6 CONCLUSION

This paper established a generative framework for GUI component description grounded in the core dimensions of [Identity] and [Action]. By coupling domain-specific adaptation with a Dual-branch RAG mechanism, the proposed approach successfully overcame the systemic issues of label heterogeneity and the static knowledge boundary. Experimental results confirmed that this synergy provided BVI users with the functional certainty required for independent and safe mobile navigation, effectively bridging the semantic void of raw GUI pixels with reliable semantic feedback.

REFERENCES

[1] Muna Al-Razgan, Sarah Almoaiqel, Nuha Alrajhi, Alyah Alhumegani, Abeer Alshehri, Bashayr Alnefaie, Raghad AlKhamiss, and Shahad Rushdi. 2021. A systematic literature review on the usability of mobile applications for visually impaired users. *PeerJ Computer Science* 7 (2021), e771.

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems* 35 (2022), 23716–23736.

[3] Domenico Amalfitano, Anna Rita Fasolino, Porfirio Tramontana, Salvatore De Carmine, and Atif M Memon. 2012. Using GUI ripping for automated testing of Android applications. In *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*. 258–261.

[4] Apple Inc. 2023. *VoiceOver User Guide for iPhone.* https://support.apple.com/guide/iphone/iph3e2e415f/ios Accessed: 2025-01-XX.

[5] Mars Ballantyne, Archit Jha, Anna Jacobsen, J Scott Hawker, and Yasmine N El-Glaly. 2018. Study of accessibility guidelines of mobile applications. In *Proceedings of the 17th international conference on mobile and ubiquitous multimedia*. 305–315.

[6] Yevgen Borodin, Jeffrey P Bigham, Glenn Dausch, and IV Ramakrishnan. 2010. More than meets the eye: a survey of screen-reader browsing strategies. In *Proceedings of the 2010 international cross disciplinary conference on web accessibility (W4A)*. 1–10.

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

[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] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. 2024. 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*. 24185–24198.

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

[11] Jyoti Choudrie, Sutee Pheeraphuttranghkoon, and Soheil Davari. 2020. The digital divide and older adult population adoption, use and diffusion of mobile phones: A quantitative study. *Information Systems Frontiers* 22, 3 (2020), 673–695.

[12] World Wide Web Consortium. 2018. *Web Content Accessibility Guidelines (WCAG) 2.1.* https://www.w3.org/TR/WCAG21/ Retrieved September 23, 2023.

[13] 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.

[14] Morgan Dixon and James Fogarty. 2010. Prefab: implementing advanced behaviors using pixel-based reverse engineering of interface structure. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1525–1534.

[15] Google. 2023. *TalkBack Screen Reader for Android.* https://support.google.com/accessibility/android/answer/6007100 Accessed: 2025-01-XX.

[16] Nora Griffin-Shirley, Devender R Banda, Paul M Ajuwon, Jongpil Cheon, Jaehoon Lee, Hye Ran Park, and Sanpalei N Lyngdoh. 2017. A survey on the use of mobile applications for people who are visually impaired. *Journal of Visual Impairment & Blindness* 111, 4 (2017), 307–323.

[17] William Grussenmeyer and Eelke Folmer. 2017. Accessible touchscreen technology for people with visual impairments: a survey. *ACM Transactions on Accessible Computing (TACCESS)* 9, 2 (2017), 1–31.

[18] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*. PMLR, 3929–3938.

[19] 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.

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

[21] Ziniu Hu, Ahmet Iscen, Chen Sun, Zirui Wang, Kai-Wei Chang, Yizhou Sun, Cordelia Schmid, David A Ross, and Alireza Fathi. 2023. Reveal: Retrieval-augmented visual-language pre-training with multi-source multimodal knowledge memory. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 23369–23379.

[22] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).

[23] Rashedul Islam, Rofiqul Islam, and Tohidul Mazumder. 2010. Mobile application and its global impact. *International Journal of Engineering & Technology* 10, 6 (2010), 72–78.

[24] Shaun K Kane, Chandrika Jayant, Jacob O Wobbrock, and Richard E Ladner. 2009. Freedom to roam: a study of mobile device adoption and accessibility for people with visual and motor disabilities. In *Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility*. 115–122.

[25] Akif Khan and Shah Khusro. 2021. An insight into smartphone-based assistive solutions for visually impaired and blind people: issues, challenges and opportunities. *Universal Access in the Information Society* 20, 2 (2021), 265–298.

[26] Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. 2023. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *International Conference on Machine Learning*. PMLR, 18893–18912.

[27] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems* 33 (2020), 9459–9474.

[28] Gang Li and Yang Li. 2022. Spotlight: Mobile ui understanding using vision-language models with a focus. *arXiv preprint arXiv:2209.14927* (2022).

[29] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*. PMLR, 19730–19742.

[30] Wen Li, Ying Zhang, Yifang Sun, Wei Wang, Mingjie Li, Wenjie Zhang, and Xuemin Lin. 2019. Approximate nearest neighbor search on high dimensional data—experiments, analyses, and improvement. *IEEE Transactions on Knowledge and Data Engineering* 32, 8 (2019), 1475–1488.

[31] 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).

[32] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.

[33] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Advances in neural information processing systems* 36 (2023), 34892–34916.

[34] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

[35] Atif M Memon, Ishan Banerjee, and Adithya Nagarajan. 2003. GUI ripping: reverse engineering of graphical user interfaces for testing.. In *Wcre*, Vol. 3. 260.

[36] Alan F Newell. 2008. Accessible computing–past trends and future suggestions: Commentary on "Computers and people with disabilities". *ACM Transactions on Accessible Computing (TACCESS)* 1, 2 (2008), 1–7.

[37] Tuan Anh Nguyen and Christoph Csallner. 2015. Reverse engineering mobile application user interfaces with remaui (t). In *2015 30th IEEE/ACM international conference on automated software engineering (ASE)*. IEEE, 248–259.

[38] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PmLR, 8748–8763.

[40] 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.

[41] Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O Wobbrock. 2018. Examining image-based button labeling for accessibility in Android apps through large-scale analysis. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*. 119–130.

[42] Deb Roy. 2000. Learning visually grounded words and syntax of natural spoken language. *Evolution of communication* 4, 1 (2000), 33–56.

[43] Richard S Schwerdtfeger. 1991. Making the GUI talk. *Byte* (1991), 118–128.

[44] Suraj Singh Senjam, Souvik Manna, and Covadonga Bascaran. 2021. Smartphones-based assistive technology: accessibility features and apps for people with visual impairment, and its usage, challenges, and usability testing. *Clinical optometry* (2021), 311–322.

[45] Ben Shneiderman. 2000. Universal usability. *Commun. ACM* 43, 5 (2000), 84–91.

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

[47] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805* (2023).

[48] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4566–4575.

[49] Jianqiang Wan, Sibo Song, Wenwen Yu, Yuliang Liu, Wenqing Cheng, Fei Huang, Xiang Bai, Cong Yao, and Zhibo Yang. 2024. Omniparser: A unified framework for text spotting key information extraction and table recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15641–15653.

[50] 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.

[51] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191* (2024).

[52] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. 2021. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF international conference on computer vision*. 1905–1914.

[53] Tom Yeh, Tsung-Hsiang Chang, and Robert C Miller. 2009. Sikuli: using GUI screenshots for search and automation. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*. 183–192.

[54] Shi Yu, Chaoyue Tang, Bokai Xu, Junbo Cui, Junhao Ran, Yukun Yan, Zhenghao Liu, Shuo Wang, Xu Han, Zhiyuan Liu, et al. 2024. Visrag: Vision-based retrieval-augmented generation on multi-modality documents. *arXiv preprint arXiv:2410.10594* (2024).