

VISUAL PROMPTING WITH ITERATIVE REFINEMENT FOR DESIGN CRITIQUE GENERATION

Anonymous authors

Paper under double-blind review

ABSTRACT

Feedback is crucial for every design process, such as user interface (UI) design, and automating design critiques can significantly improve the efficiency of the design workflow. Although existing multimodal large language models (LLMs) excel in many tasks, they often struggle with generating high-quality design critiques—a complex task that requires producing detailed design comments that are visually grounded in a given design’s image. Building on recent advancements in iterative refinement of text output and visual prompting methods, we propose an iterative visual prompting approach for UI critique that takes an input UI screenshot and design guidelines and generates a list of design comments, along with corresponding bounding boxes that map each comment to a specific region in the screenshot. The entire process is driven completely by LLMs, which iteratively refine both the text output and bounding boxes using few-shot samples tailored for each step. We evaluated our approach using Gemini-1.5-pro and GPT-4o, and found that human experts generally preferred the design critiques generated by our pipeline over those by the baseline, with the pipeline reducing the gap from human performance by 50% for one rating metric. To assess the generalizability of our approach to other multimodal tasks, we applied our pipeline to open-vocabulary object and attribute detection, and experiments showed that our method also outperformed the baseline.

1 INTRODUCTION

Critiques are essential for design, providing feedback to help designers improve their work (Duan et al., 2024a; Wang et al., 2021; Xu et al., 2014). However, obtaining design critiques is often costly and time-consuming, hindering the design process. Hence, automating design critiques has become an important goal in many design fields. In this paper, we focus on automating critiques for user interface (UI) design—a prevalent task in industry that directly impacts the user experience (Stone et al., 2005). Obtaining UI design feedback typically requires expert reviews or user testing with target end users, which may be expensive and not always readily available. This makes automated critique extremely valuable, as it can provide instant feedback for designers to quickly iterate on (Duan et al., 2024a). Furthermore, automated design feedback can serve as a reward function for automated UI generation, which has started to gain traction (Gajos et al., 2010; Gajjar et al., 2021).

UI design critique is often complex and open-ended, involving feedback that covers multiple dimensions of the design (e.g., aesthetics and usability) (Nielsen & Molich, 1990; Hartmann et al., 2008) and addresses both the overall design and specific problematic regions of the UI, based on design principles or guidelines. This makes automated UI critique a very challenging task. Given a UI screen and a set of design guidelines, the model needs to understand the screen, reason with UI design principles to detect violations in the UI design (both semantically and spatially), and then explain and contextualize the feedback in the way that human designers can understand and act upon (Duan et al., 2024b) (Figure 1). Essentially, automated UI design critique is a challenging task that presents an opportunity to develop various multimodal capabilities in models.

Multimodal LLMs have made tremendous progress in a variety of multimodal tasks, such as visual question answering (VQA) and visual understanding, due to their extensive knowledge and generalization capabilities. Although multimodal LLMs appear to be readily usable for design critique, a multimodal task, there remains a significant gap in quality between the feedback generated by

054 these LLMs compared to that of human design experts (Duan et al., 2024b). In addition, multimodal
055 LLMs often struggle with achieving accurate visual grounding (Duan et al., 2024b; Dorckenwald
056 et al., 2024), making it difficult for them to mark relevant regions in the UI screenshots, which is
057 crucial for contextualizing feedback for designers (Duan et al., 2024b).

058 Recent advances in prompting techniques have improved both visual grounding and text generation
059 performance. For example, Fang et al. (2024) introduced a visual prompting technique that adds
060 visual markers to an image, which helps multimodal LLMs better ground objects. Madaan et al.
061 (2023); Xu et al. (2024a) proposed a method called *iterative refinement*, where an LLM’s output is
062 repeatedly refined by itself or another model until the output is deemed correct. *iterative refinement*
063 has been shown to improve the LLM’s performance for text-only tasks like code optimization and
064 machine translation. Building on these prompting methods, we develop a novel technique for UI de-
065 sign critique generation (Figure 1). Our approach iteratively refines both design comment text and
066 their corresponding bounding boxes, utilizing visual prompting to assist in bounding box generation
067 and refinement. For iterative refinement of bounding boxes, we introduce a novel technique that dis-
068 plays a zoomed-in patch of the bounding box candidate to help the refinement process. Our approach
069 is implemented through an architecture that coordinates multiple multimodal LLMs (Figure 2).

070 We evaluated our pipeline for UI critique using UICrit, a public dataset (Duan et al., 2024b), with
071 two state-of-the-art multimodal LLMs: Gemini-1.5-pro (Team et al., 2024) and GPT-4o (OpenAI
072 et al., 2024). Our experiments demonstrated that the pipeline consistently improved the design
073 feedback output across both models, on both automatic metrics and human expert evaluation. To
074 assess the broader applicability of our method to other multimodal tasks, we tested it on open-
075 vocabulary object and attribute detection, where it consistently increased the mAP by up to 9.1.
076 These experiments demonstrate the potential of our method to be a useful technique in the broader
077 scope of tasks, beyond design critique generation, pushing the boundary of what prompting can
078 achieve for complex multimodal tasks. Our paper makes the following contributions:

- 079 • A modular multimodal prompting framework that orchestrates six LLMs (Figure 2) for
080 generation, refinement, and validation of design critiques, which takes in an image and a
081 task prompt, and generates a list of text items visually grounded in the image.
- 082 • A set of LLM prompting techniques for iterative refinement of both text and bounding
083 boxes that ground the text within the image. We introduce a technique for visual ground-
084 ing refinement, where we include a zoomed-in patch around the bounding box candidate
085 (Figure 3) in the prompt to assist in fine-grained visual grounding.
- 086 • Extensive experiments with the proposed prompting framework on UI design critique, a
087 challenging multimodal task, and a study of its performance on open vocabulary object and
088 attribute detection. These experiments showed that our pipeline consistently outperformed
089 the baseline methods across two distinct multimodal tasks and domains.

092 2 RELATED WORK

093 2.1 AUTOMATED UI DESIGN CRITIQUE WITH LLMs

094
095
096 Prior work have studied the capabilities of using LLMs for UI design critique. Duan et al. (2024a)
097 explored the performance of zero-shot (text-only) GPT-4 in critiquing UI mockups, using a JSON
098 representation of the UI. They identified gaps between the feedback capabilities of general-purpose
099 LLMs and human experts. To address this, Duan et al. (2024b) collected a dataset (UICrit) consisting
100 of human-annotated UI design critiques (grounded within UI screenshots via bounding boxes) for
101 UI screens that could be applied to train general purpose LLMs. Their UI design critique model
102 takes in a UI screenshot and outputs critiques grounded in screenshot regions. Their method showed
103 a significant improvement in LLM-generated feedback with just few-shot sampling from UICrit,
104 although the feedback quality still falls short of human experts. Similarly, Wu et al. (2024) generated
105 a synthetic dataset of UI design comments and trained a CLIP model (Radford et al., 2021) to assess
106 UI designs. We apply our approach to the design critique task, which augments the method of
107 Duan et al. (2024b) by incorporating iterative refinement of both the design comments and their
corresponding bounding box positions on the UI screen.

2.2 PROMPTING LLMs WITH ITERATIVE REFINEMENT

Iterative refinement on LLM output has been explored in prior studies, as a means to improve LLM performance. Madaan et al. (2023) developed an approach called “SELF-REFINE”, where a single LLM generates an initial output and then iteratively provides feedback on its own output and revises the output based on the feedback. They applied this technique across a diverse set of tasks, such as math reasoning and dialogue response, and found that SELF-REFINE resulted in an 20% average performance gain. Similarly, Zhou et al. (2023) utilized this iterative self-refinement technique on long-horizon sequential task planning in robotics, leading to higher success rates. However, Xu et al. (2024b) found that LLMs often exhibit *self-bias* (i.e. a tendency to favor its own generated output) during self-refinement across a variety of tasks and languages. To account for this, Xu et al. (2024a) developed “LLMRefine”, a method for text generation that uses a separate model to provide detailed feedback, along with a simulated annealing method to iteratively refine the LLM’s output. We also utilize iterative refinement in our pipeline, and we extend this method to multimodal tasks by refining both text and bounding boxes that associate the text with relevant regions in the image. Following the method in LLMRefine, we use separate LLMs for generation and refinement to prevent self-bias.

2.3 MULTIMODAL TASKS

Previous work has investigated a variety of grounded multimodal tasks using LLMs, where an LLM takes in a visual input (such as an image) and generates outputs that are connected to specific objects, regions, or attributes within the visual input. Liu et al. (2023) introduced Grounding DINO, a transformer-based model that supports open-vocabulary object detection and can identify arbitrary objects within an image. However, it struggles with complex queries involving multiple objects and intricate spatial relationships. To address this limitation, Zhao et al. (2024) developed LLM-Optic, which uses an LLM to break down complex queries into specific objects, employs Grounding DINO to detect candidate objects, and finally uses a multimodal LLM to select the most suitable objects for the query. Beyond object detection, Bravo et al. (2023) introduced open vocabulary object and attribute detection, which identifies and grounds both objects and their corresponding attributes in an image in an open vocabulary setting. In robotics, multimodal LLMs were used to help systems understand the physical world. Fang et al. (2024) introduced MOKA, which utilizes multimodal LLMs to solve complex robotic manipulation tasks by breaking them into multiple steps. Their approach incorporates visual prompting, where visual markers are added to the image, to aid in object grounding as part of the robot’s step-by-step instructions. Chen et al. (2024) examined the capabilities of multimodal LLMs for evaluation across three tasks: pair comparison, scoring, and ranking. They found that while LLMs performed well on pair comparison, they struggled with the other tasks, suggesting that further improvements are needed before LLMs can be reliable validators. Visual grounding is a vital component of our method, and we utilize visual prompting to enhance bounding box generation and refinement. Although multimodal validation has limitations, our ablation studies indicate that incorporating it to validate the generated text and bounding boxes generally improved performance.

3 TASK

UI design critique generation was first proposed as a grounded multimodal task by Duan et al. (2024b). The model takes in a UI screenshot and a set of design guidelines as input and outputs a list of design critiques. Each design critique comprises two components: a text comment that identifies a specific issue in the UI and a bounding box that highlights the relevant region of the screenshot (see Figure 1). For example the text comment might state “*The expected standard is to use clear contrast for readability. In the current design, the label ‘Best’ is difficult to see on the image due to its high transparency. To fix this, reduce the transparency of the box and apply a solid color so that the text ‘Best’ is readable.*”) and the bounding box will enclose the orange ‘Best’ tag in the UI screenshot in 1.

As discussed earlier, the UI design critique task is particularly challenging because the model must understand and apply UI design principles to identify design issues in the screenshot. Furthermore, determining the exact region of the screen (i.e., the bounding box) for a comment is not always straightforward. For example, a comment might note that the text in the UI has poor contrast with the background, but not specify which text element is problematic, requiring the model to identify



Figure 1: Illustration of the UI Design Critique Task, which takes in a UI screenshot and a set of design guidelines and outputs a list of design comments with corresponding bounding boxes (Bbox).

the problematic elements and also determine their bounding box. While our focus in this paper is on UI design critique, our task is representative of many multimodal tasks that require visually grounded text generation.

4 METHOD

We developed a prompting pipeline that uses multiple LLMs to generate UI design critiques. It consists of six distinct LLMs that are organized into three modules: *Text Generation & Refinement*, *Validation*, and *Bounding Box Generation & Refinement*. These modules communicate with each other to complete the task. Figure 2 illustrates the workflow of the pipeline, showing the main inputs and outputs of each LLM, which are numbered by the order of execution. We break down the entire task into separate generation and refinement steps for both text and bounding boxes, as decomposing complex tasks has been shown to improve performance (Khot et al., 2023).

As shown in the figure, the LLM output of each step is conditioned on that of the previous step. Since *Bounding Box Generation & Refinement* is conditioned on the text predictions, and text refinement, in turn, is conditioned on the bounding box predictions, we introduce the *Validation* module between the *Text* and *Bounding Box* modules to ensure that each refinement step is based on more accurate inputs. Additionally, each LLM is provided with targeted few-shot examples to improve its accuracy, as well as a text prompt containing specific instructions for that step, which is derived from the input task prompt. To provide as much guidance as possible, we included the UI design guidelines in the input task prompt, which are also included in the instructions prompts for relevant steps. The specific inputs, outputs, and few-shot examples for each LLM are detailed in the following sections, and the instructions prompt for each step can be found in Appendix A.3.

Text Generation LLM (TextGen) The pipeline begins with the TextGen LLM that takes an image and its instructions prompt (derived from the task prompt) as input, and generates a list of ungrounded text items (design comments) for the image. We decided to start with text generation and condition the bounding box generation on the generated text, instead of the other way around. This decision is based on our observation that for design critique, LLMs tend to perform poorly on visual grounding from scratch (i.e., without guidance from text), which makes the subsequent refinements much more error-prone.

Text Filtering LLM (TextFilter) To reduce the chance of bounding box generation being conditioned on incorrect text items (i.e., incorrect design comments), we add an additional filtering step to remove invalid or irrelevant text items. The TextFilter LLM takes as input a list of generated text items from TextGen, along with the image, and outputs a filtered list of valid text items. While previous studies (Chen et al., 2024; Shankar et al., 2024) have shown that LLMs may not always be reliable evaluators, Liu et al. (2024) demonstrated that few-shot examples can improve performance. We designed few-shot examples for TextFilter by injecting invalid items into a correct list of text items, using this augmented list as input and the original correct list as the expected output. This illustrates how to filter out invalid items.

Bounding Box Generation LLM (BoxGen) The BoxGen LLM generates bounding boxes based on the filtered text items from TextFilter. The LLM takes in one text item at a time, as well as

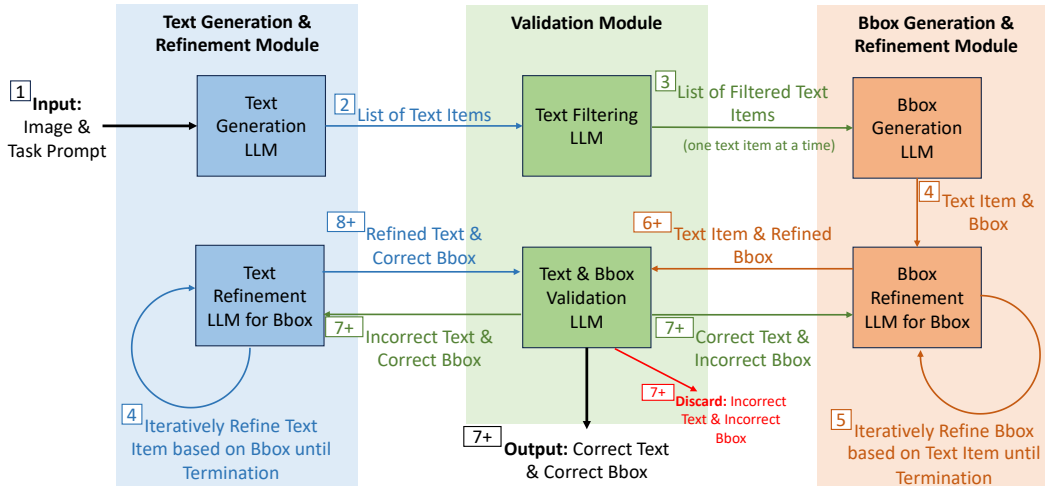


Figure 2: The figure illustrates our prompting pipeline, which takes an image and a task prompt as input and outputs text items with their corresponding bounding boxes on the image. The pipeline consists of six distinct LLMs, organized into three modules: Text Generation and Refinement, Validation, and Bounding Box (Bbox) Generation and Refinement. Targeted few-shot examples are provided for each LLM. The main inputs and outputs for each LLM are shown, and Section 4 details all the inputs, outputs, and few-shot examples for each LLM. Each input/output is numbered with their order of generation, and numbers with a '+' indicate multiple iterations of input/output.

the image, and predicts a relevant region on the image via bounding box coordinates. Following the visual prompting technique from Duan et al. (2024b), we augment the screenshot by adding coordinate markers along its edges (Figure 3) to help the LLM associate coordinates with specific regions in the screen.

Bounding Box Refinement LLM (BoxRefine) To avoid self-bias during iterative refinement (Xu et al., 2024b), we use a separate LLM to iteratively refine the generated bounding box from the previous step. The BoxRefine LLM takes in several inputs, as shown in Figure 3. Similar to BoxGen, BoxRefine takes in the coordinate-marker enhanced screenshot image and a filtered text item. Additionally, BoxRefine takes in the bounding box coordinates that was predicted by BoxGen, and a close-up view of the image region specified by the predicted bounding box coordinates. In this zoomed-in image patch, the bounding box is displayed as a blue box, with some surrounding region of the box included for additional context. The zoomed-in image patch also has coordinate markers along the edges to help the LLM refine the bounding box coordinates based on this close-up view.

The LLM assesses the quality of the current bounding box based on all these inputs. If the bounding box is deemed accurate by the BoxRefine LLM, the iterative refinement process terminates. Otherwise, the LLM returns the refined coordinates, which are then re-evaluated by the LLM. This process is repeated until the LLM either confirms the bounding box as correct or the maximum number of iterations is reached. Previous work (Madaan et al., 2023) has shown that the history of refinements provides helpful information. Thus, we include the history of the LLM’s refinements for the input bounding box as an input at each iteration, which enables the model to learn from past adjustments. Few-shot examples are generated by creating a synthetic refinement sequence with gradually reduced noise in the perturbation of a sampled bounding box’s coordinates. Algorithm 1 in the Appendix details our methods for bounding box perturbation and the generation of few-shot examples for bounding box refinement.

Text & Bounding Box Validation LLM (Validation) After determining the bounding box for the text item, the Validation LLM determines if the bounding box and text are correct and can be used in the final output, or if they require further refinement. The Validation LLM takes as input the entire image, a zoomed-in image patch for the proposed region specified by the bounding box, and the text item, and assesses the accuracy of critique generation as one of the following:

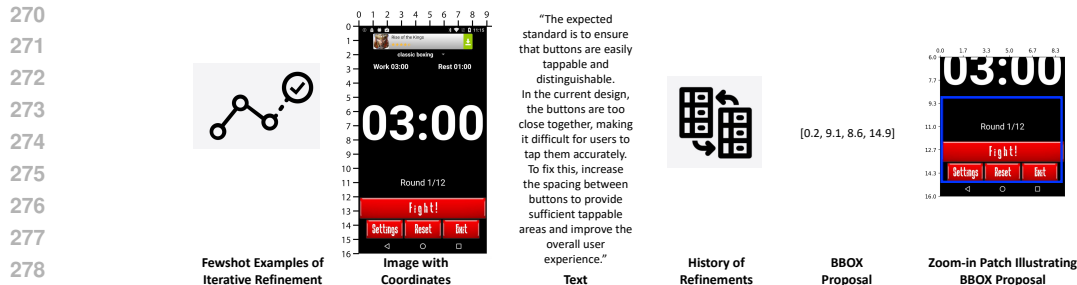


Figure 3: An example of the inputs to the Bounding Box Refinement LLM.

1. *Both Text & Box Correct*: Both the bounding box and the text item are accurate, and the pair is returned. The pipeline moves onto the next text item in the filtered list.
2. *Incorrect Text*: The bounding box correctly identifies a region in the UI screenshot with design issues, but the text item is incorrect (e.g., does not adequately describe the design issues in the region). The pair is sent to the TextRefine LLM for text refinement.
3. *Incorrect Bounding Box*: The text item is correct (e.g., describes a valid design issue in the UI screenshot), but the bounding box is incorrect (e.g., does not accurately enclose the region described in the critique). The bounding box and text item are sent back to the BoxRefine LLM for further refinement of the bounding box.
4. *Both Incorrect*: Both the text and the bounding box are incorrect. The pair is discarded and the pipeline moves onto the next text item in the filtered list.

Few-shot examples are generated differently for each case; the bounding box is perturbed for the Incorrect Bounding Box case (Algorithm 1 in the Appendix), the text item is perturbed for the Incorrect Text case, and both the bounding box and text are perturbed for the Both Incorrect case. In addition, text and bounding box pairs that are sent for further refinement are sent back to this LLM for validation, after they have been refined.

Text Refinement LLM (TextRefine) The TextRefine LLM is used to refine incorrect text items conditioned on bounding boxes that correctly identify relevant regions in the image, as determined by the Validation LLM. This iterative refinement process mirrors the bounding box refinement procedure. The LLM takes as input the entire image, a zoomed-in image patch focused on the bounding box, and the text item, and refines the text iteratively until it determines that the text is accurate for the region shown in the bounding box. Few-shot examples are generated either by perturbing the text (if possible) or by selecting irrelevant text items from the few-shot dataset and then ranking them by increasing semantic similarity to simulate the refinement process. The refined text item and bounding box are then returned to the Validation LLM.

5 EXPERIMENTS

5.1 DATASET

We used the UICrit dataset¹, collected by Duan et al. (2024b), to evaluate our pipeline for the design critique task. Each UI screenshot in this dataset was annotated by three experienced human designers, providing feedback that includes a list of text-based design critiques with their corresponding bounding boxes, numerical ratings for usability, aesthetics, and overall design quality, as well as a description of what the screen is designed for. The dataset contains a total of 11,344 design critiques for 1,000 screenshots. For evaluation, we used the UI screenshots from UICrit as input images, included the three sets of design guidelines used by Duan et al. (2024b) in the task prompt, and evaluated the model’s output against the comments and bounding boxes of the screen from the dataset (depending on the experiment). For few-shot examples, we sampled from a split of UICrit that is separate from the examples used for the evaluation. The few-shot sampling methods used at each step is detailed in Appendix A.2.1.

¹<https://github.com/google-research-datasets/uicrit>

Table 1: IoU values from the Ablation study on the different components of bounding box generation. IR stands for Iterative Refinement, and VP stands for Visual Prompting.

Methods	UI Critique IoU \uparrow	
	Gemini _{1.5tn}	GPT _{4tn}
Zero-shot	0.120	0.233
Zero-shot, VP	0.180	0.249
Few-shot, VP	0.267	0.319
Few-shot, VP, Zero-shot IR	0.279	0.319
Few-shot, VP, Few-shot IR	0.357	0.345

5.2 BASELINE

We used the few-shot pipeline developed by Duan et al. (2024b) for their UI critique task as the baseline. Their pipeline consists of the Text Generation LLM (Figure 2) with few-shot sampling, followed by an LLM for bounding box generation that uses visual prompting (i.e., coordinates marked on the screenshot edges) without few-shot examples.

5.3 IMPACT OF VISUAL PROMPTING & ITERATIVE REFINEMENT ON VISUAL GROUNDING

Table 1 presents an ablation study on the different components of the Bounding Box Generation and Refinement module (Figure 2), which illustrates the impact of visual prompting and iterative refinement on the visual grounding accuracy of two state-of-the-art multimodal LLMs: Gemini-1.5-pro and GPT-4o. For this evaluation, the module is given a UI screenshot and one of its comments from UICrit. Its output bounding box is evaluated against the ground-truth bounding box of that comment in UICrit by computing their IoU. The module consists of two LLMs (BoxGen and BoxRefine), and the BoxRefine LLM was only used for the conditions with iterative refinement (i.e., the last two rows of the table).

For Gemini-1.5-pro, each enhancement led to an improvement in the average IoU, with the final setup (used in our pipeline) achieving an average IoU nearly three times higher than zero-shot and almost double that of zero-shot with visual prompting, which was used in the baseline (Duan et al. (2024b)). For GPT-4o, improvements were seen at each step, except for zero-shot iterative refinement; when no few-shot examples were provided in the refinement prompt, GPT-4o did not refine any of the input bounding boxes. Additionally, while GPT-4o had better zero-shot performance, its IoU for the final setup was slightly worse than that of Gemini-1.5-pro. Nevertheless, iterative visual prompting led to substantial performance gains over zero-shot prompting for both LLMs, indicating that iterative visual prompting significantly enhances bounding box estimation.

5.4 PIPELINE ABLATION STUDY AND QUALITATIVE ANALYSIS

Table 2 presents the results of the ablation study for UI design critique for both LLMs, as well as the results for the baseline setup and multimodal Llama-3.2 11b (Dubey et al., 2024), which has been finetuned on the training split of UICrit for three epochs. Since UI design critique is open-ended, UICrit does not contain all the ground-truth design comments for each UI screenshot. Hence, we evaluated comment generation by computing the cosine similarity of sentenceBERT embeddings with each comment in the dataset for the UI screenshot and selecting the highest one (“Comment Similarity” in Table 2). The IoU was estimated by comparing the predicted bounding box with that of the most semantically similar comment (“Estimated IoU” in Table 2). The estimated IoU values are lower than those in Table 1, where the IoU was calculated directly from the input comments’ corresponding bounding boxes in UICrit. The estimated IoU is lower because it uses the bounding box of the most semantically similar comment in the dataset instead, which may not precisely match the comment for which the bounding box was generated.

Each step of the pipeline incrementally improved the comment similarity and estimated IoU for both LLMs. While GPT-4o and Gemini-1.5-pro showed similar values in terms of comment similarity, GPT-4o achieved a higher estimated average IoU. GPT-4o’s advantage could be due to its significantly larger size—nearly three times as many parameters as Gemini-1.5-pro. The complete pipeline also outperforms the baseline in both comment similarity and estimated IoU. Note that the comment

Table 2: Ablation study of the different steps of our pipeline on UI design critique. IR stands for Iterative Refinement. Note that we combine the results of the Validation step with results from the additional iterative refinement steps for bounding box and text. This is because these additional refinements are applied to a much smaller subset; specifically, only the pairs identified as having incorrect text or incorrect bounding boxes during the Validation step. We also include results from the baseline setup and finetuned Llama-3.2 11b.

Pipeline Steps	Comment Similarity \uparrow		Estimated IoU* \uparrow	
	Gemini _{1.5tn}	GPT _{4tn}	Gemini _{1.5tn}	GPT _{4tn}
Text Generation	0.651	0.680	N/A	N/A
+ Text Filtering	0.694	0.692	N/A	N/A
+ Bbox Generation	0.694	0.692	0.153	0.244
+ IR of Bbox	0.694	0.692	0.173	0.259
+ Validation, IR of Text & Bbox	0.702	0.701	0.199	0.275
Baseline (Duan et al., 2024b)	0.651	0.680	0.176	0.257
Finetuned Llama-3.2 11b	0.842		0.230	

similarity for the baseline is identical to that of the ‘Text Generation’ row. This is because both the pipeline and baseline start with TextGen, so we used the same initial comments from TextGen for both conditions for easier comparison. Fine-tuned Llama-3.2 achieves higher comment similarity than the pipeline, but its estimated IoU falls between those of Gemini-1.5-pro and GPT-4o for the complete pipeline.

We conducted a qualitative analysis of the outputs from the pipeline, baseline, and finetuned Llama-3.2, finding the pipeline outputs helpful comments with reasonable bounding boxes (more often than not) and generally outperforms the baseline and finetuned Llama. Compared to baseline, it reduces the generation of invalid and generic comments, while producing bounding boxes that are tighter, more specific, and closer to the target region. However, the pipeline sometimes eliminates valid comments. Also, we found that the baseline often generates very large bounding boxes that cover the majority of the screen. This would decrease the chance of the IoU being zero, which may have inflated its estimated IoU. We found that finetuned Llama only generated a very limited set of critiques, while our pipeline generates a considerably more diverse set of comments. Although finetuned Llama generally had better visual grounding, the bounding boxes tend to be larger and less specific. Section A.4.1 (Appendix) provides detailed results and example outputs. Section A.4.2 presents qualitative results and outputs for out-of-domain UIs (e.g. websites), demonstrating that our pipeline can still generate helpful feedback. Finally, Section A.5 includes a cost analysis of our pipeline and also contains example visualizations of bounding box and comment refinements.

5.5 HUMAN EVALUATION

Due to the open-ended nature of UI design critique, UICrit does not have the complete set of ground-truth design comments for each UI screen. Hence, we recruited human design experts to assess the validity of the feedback generated by our pipeline. For comparison, the experts also rated the comments generated by the *baseline* setup and human annotated comments from UICrit. We used the same procedure devised by Duan et al. (2024b), where each design comment was rated as invalid, partially valid and valid, and the set of design comments from each condition was ranked as a whole, based on overall quality and comprehensiveness. Unlike the method used by Duan et al. (2024b), where participants rated both comment quality and bounding box accuracy together, our evaluation presented participants with a screenshot marked with a ground-truth bounding box (determined and agreed upon by the authors) and asked them to rate the validity of the comment only for that region. This is to ensure a more rigorous and standardized approach to evaluate bounding box accuracy and a more focused evaluation on comment quality. See Section 5.5 (Appendix) for more details on the study method. Table 3 shows the average comment rating, the average comment set rank, and the average IoU for each of the three conditions for Gemini-1.5-pro’s output. We used the established ground-truth bounding boxes from comments rated as valid or partially valid to compute the IoU with predicted bounding boxes. For the “human” condition, the IoU was not computed as we displayed the bounding boxes from UICrit. The average Fleiss Kappa inter-rater reliability score (Fleiss et al., 1971) amongst the participants was 0.22 for comment quality and 0.29 for comment set ranking, indicating fair agreement.

Table 3: Human expert ratings on UI design comments generated by Gemini-1.5-pro, and IoU of the generated bounding boxes for human validated comments.

Methods	Comment Quality \uparrow	Comment Set Rank \downarrow	BBox IoU \uparrow
Baseline (Duan et al., 2024b)	0.45	2.3	0.423
Our Pipeline	0.47	2.0	0.451
Human	0.56	1.7	N/A

Across all the metrics, the pipeline outperformed the baseline, while human annotations remain the best. Interestingly, the average comment quality rating for human feedback was lower than expected, which may be attributed to the subjective nature of design critique (Nielsen & Molich, 1990) and the variability in dataset quality, potentially due to UICrit’s annotators’ limited design experience (Duan et al., 2024b). While the gap between our pipeline and the baseline is modest, it still closes 22% of the gap between the baseline and human condition. Notably, the average comment set rank of our pipeline is positioned midway between the human and baseline setups. The comment set from our pipeline was preferred over the baseline’s 58% of the time and was even favored over the human condition 38% of the time.

6 GENERALIZATION TO OTHER TASKS

Our pipeline can be applied to other multimodal LLM tasks that involve generating visually grounded text. To assess if its performance enhancement generalizes to other tasks, we evaluate our pipeline on an existing vision-language modeling benchmark: Open Vocabulary Object and Attribute Detection (Bravo et al., 2023).

6.1 OPEN VOCABULARY OBJECT AND ATTRIBUTE DETECTION

Open vocabulary object and attribute detection, developed by Bravo et al. (2023), involves detecting objects and their associated attributes, along with bounding boxes marking their locations in the image (see Appendix A.1). During inference, the model is given a set of object classes and attributes to identify, including classes and attributes that were not seen during training, which tests the model’s ability to generalize to novel object classes and attributes (i.e., “open vocabulary”). Bravo et al. (2023) evaluated both attribute detection (OVAD) and object detection (OVD) in this open vocabulary setting. They collected a dataset² of human annotated object classes and attributes for 2,000 images from the MS COCO dataset (Lin et al. (2014)), including 80 object classes and 117 attribute categories. The object classes are divided into base and novel categories, with only the base classes seen during training. We used this dataset to evaluate our pipeline on this task. The task involves taking an image as input, along with a task prompt specifying the object and attribute classes. The output is evaluated against the ground truth object and attribute annotations. To meet the open-vocabulary criterion of this task, we sampled few-shot examples from the base classes only, from a split of their dataset, but used all the classes for evaluation. Appendix A.2.2 describes the fewshot sampling strategy in more detail.

6.2 COMPARISON WITH BASELINE

Table 4 presents the results of the ablation study for open-vocabulary object and attribute detection, using both Gemini-1.5-pro and GPT-4o. We used the same baseline described in Section 5.2, as it can also be applied to this task. We followed the evaluation method of Bravo et al. (2023), calculating the mean average precision (mAP) across all attribute (OVAD) and object categories (OVD). The predicted text and corresponding bounding box were matched with the ground truth by selecting the bounding box with the highest IoU, with a minimum threshold of 0.5, and comparing the object categories and attribute classes.

Our approach outperformed the baseline mAP for OVAD by 2.5 and OVD by 4.6 with Gemini-1.5-pro, and by 2.2 for OVAD and 9.1 for OVD with GPT-4o. The larger performance gain for OVD may be due to the fact that it is a simpler task, with only 80 object categories compared to 117 attribute

²<https://ovad-benchmark.github.io/>

Table 4: Ablation study on the open vocabulary attribute detection (OVAD) and object detection (OVD) for Gemini-1.5-pro and GPT-4o. IR stands for Iterative Refinement. Note that bounding boxes are required for computing the mAP, so we combined the results for the text generation, text filtering, and bounding box generation steps. Similar to Table 2, we combined the results of the Validation step with the additional iterative refinements of the bounding box and text.

Pipeline Steps	OVAD mAP \uparrow		OVD mAP \uparrow	
	Gemini _{1.5tn}	GPT _{4tn}	Gemini _{1.5tn}	GPT _{4tn}
Text Generation + Filtering + BBox	11.3	13.1	13.1	15.8
+ IR of BBox	12.6	14.0	15.8	17.8
+ Validation, IR of Comment & BBox	13.6	15.1	15.8	20.2
Baseline	11.1	12.9	11.2	11.1

categories, and attributes are often more nuanced and harder to detect. Additionally, GPT-4o slightly outperformed Gemini-1.5-pro, likely due to its much larger size. However, our pipeline still falls short of the fine-tuned model from Bravo et al. (2023) (mAP 18.8 for OVAD and 39.3 for OVD).

7 DISCUSSION

Our pipeline outperforms the baseline for UI critique in both comment quality and grounding accuracy, based on automatic metrics (e.g., IoU) and human expert ratings; its feedback was also more often preferred by human experts. This implies that the design feedback generated by our pipeline is more useful for human designers. Its performance improvement also generalizes to open-vocabulary object and attribute detection, suggesting the technique could be potentially applied to enhance other grounded multimodal LLM tasks.

While our technique outperforms the baselines for open vocabulary object and attribute detection, it falls short of the fine-tuned LLMs from Bravo et al. (2023). This is expected, since our pipeline does not involve parameter-tuning, whereas their fine-tuned LLMs were trained on significantly more data than the few-shot examples provided to our model. For design critique, our pipeline generates a significantly more diverse set of critiques compared to finetuned Llama 3.2, potentially making our pipeline more useful in practice. However, our pipeline still has room for improvement when compared to human expert design feedback. Despite its performance gap with human critique (which are expensive to acquire), the generalizability of our pipeline and its consistent performance improvement over the baseline demonstrate its potential to be a versatile and resource-efficient solution for improving multimodal LLM performance across different tasks and domains.

A reason for the performance gap could be that the LLM-based validation steps are not fully accurate (Shankar et al., 2024; Chen et al., 2024), which could lead to incorrect judgement of the bounding box and/or text accuracy. Future work can improve the validation step with better prompting strategies, or look into a human-in-the-loop approach, where human experts validate or refine the text and bounding boxes. The human-in-the-loop validation could both improve the immediate quality of the output and help the system learn from human inputs over time via targeted few-shot examples. This step can be integrated into a design tool where designers validate or refine the feedback, so the model learns to provide more accurate and personalized design critiques over time.

8 CONCLUSION

We introduce a novel prompting pipeline that improves both the quality and visual grounding of automated UI design critique by using visual prompting and iterative refinement of both text and bounding boxes. Our approach outperformed the baseline in human evaluations, generating higher quality comments with more accurate visual grounding. Additionally, we demonstrated the generalizability of our technique through performance gains in open-vocabulary object and attribute detection, suggesting its potential to enhance other grounded multimodal tasks. While our method has limitations, it offers a versatile and resource-efficient solution for improving multimodal LLM performance across various tasks and domains.

REFERENCES

- 540
541
542 María A. Bravo, Sudhanshu Mittal, Simon Ging, and Thomas Brox. Open-vocabulary attribute de-
543 tecton. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*
544 (*CVPR*), pp. 7041–7050, June 2023.
- 545 Dongping Chen, Ruoxi Chen, Shilin Zhang, Yinuo Liu, Yaochen Wang, Huichi Zhou, Qihui Zhang,
546 Pan Zhou, Yao Wan, and Lichao Sun. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge
547 with vision-language benchmark, 2024.
- 548 Biplob Deka, Zifeng Huang, Chad Franzen, Joshua Hibsichman, Daniel Afegan, Yang Li, Jef-
549 frey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven de-
550 sign applications. In *Proceedings of the 30th Annual ACM Symposium on User Interface*
551 *Software and Technology*, UIST '17, pp. 845–854, New York, NY, USA, 2017. Association
552 for Computing Machinery. ISBN 9781450349819. doi: 10.1145/3126594.3126651. URL
553 <https://doi.org/10.1145/3126594.3126651>.
- 554 Michael Dorckenwald, Nimrod Barazani, Cees G. M. Snoek, and Yuki M. Asano. Pin: Positional
555 insert unlocks object localisation abilities in vlms, 2024. URL [https://arxiv.org/abs/](https://arxiv.org/abs/2402.08657)
556 [2402.08657](https://arxiv.org/abs/2402.08657).
- 557 Peitong Duan, Jeremy Warner, Yang Li, and Bjoern Hartmann. Generating automatic feedback
558 on ui mockups with large language models. In *Proceedings of the CHI Conference on Human*
559 *Factors in Computing Systems*, CHI '24, New York, NY, USA, 2024a. Association for Computing
560 Machinery. ISBN 9798400703300. doi: 10.1145/3613904.3642782. URL [https://doi.](https://doi.org/10.1145/3613904.3642782)
561 [org/10.1145/3613904.3642782](https://doi.org/10.1145/3613904.3642782).
- 562 Peitong Duan, Chin yi Chen, Gang Li, Bjoern Hartmann, and Yang Li. Uicrit: Enhancing automated
563 design evaluation with a uicritique dataset, 2024b. URL [https://arxiv.org/abs/2407.](https://arxiv.org/abs/2407.08850)
564 [08850](https://arxiv.org/abs/2407.08850).
- 565 Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, and et. al. The llama 3 herd
566 of models, 2024. URL <https://arxiv.org/abs/2407.21783>.
- 567 Kuan Fang, Fangchen Liu, Pieter Abbeel, and Sergey Levine. MOKA: Open-World Robotic Manip-
568 ulation through Mark-Based Visual Prompting. In *Proceedings of Robotics: Science and Systems*,
569 Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.062.
- 570 J.L. Fleiss et al. Measuring nominal scale agreement among many raters. *Psychological Bulletin*,
571 76(5):378–382, 1971.
- 572 Nishit Gajjar, Vinoth Pandian Sermuga Pandian, Sarah Suleri, and Matthias Jarke. Akin: Generating
573 ui wireframes from ui design patterns using deep learning. In *Companion Proceedings of the*
574 *26th International Conference on Intelligent User Interfaces*, IUI '21 Companion, pp. 40–42,
575 New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380188. doi:
576 [10.1145/3397482.3450727](https://doi.org/10.1145/3397482.3450727). URL <https://doi.org/10.1145/3397482.3450727>.
- 577 Krzysztof Z Gajos, Daniel S Weld, and Jacob O Wobbrock. Automatically generating personalized
578 user interfaces with supple. *Artificial intelligence*, 174(12-13):910–950, 2010.
- 579 Foziah Gazzawe. Comparison of websites and mobile applications for e-learning. 07 2017.
- 580 Jan Hartmann, Alistair Sutcliffe, and Antonella De Angeli. Towards a theory of user judgment of
581 aesthetics and user interface quality. *ACM Trans. Comput.-Hum. Interact.*, 15, 11 2008. doi:
582 [10.1145/1460355.1460357](https://doi.org/10.1145/1460355.1460357).
- 583 Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish
584 Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *The*
585 *Eleventh International Conference on Learning Representations*, 2023.
- 586 Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr
587 Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet,
588 Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision – ECCV 2014*, pp.
589 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1.
- 590
591
592
593

- 594 Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei
595 Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for
596 open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.
- 597
598 Yuxuan Liu, Tianchi Yang, Shaohan Huang, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng,
599 Feng Sun, and Qi Zhang. Calibrating LLM-based evaluator. In Nicoletta Calzolari, Min-Yen
600 Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings*
601 *of the 2024 Joint International Conference on Computational Linguistics, Language Resources*
602 *and Evaluation (LREC-COLING 2024)*, pp. 2638–2656, Torino, Italia, May 2024. ELRA and
603 ICCL.
- 604 Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri
605 Alon, Nouha Dziri, Shrimai Prabhunoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad
606 Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-
607 refine: Iterative refinement with self-feedback. In A. Oh, T. Naumann, A. Globerson, K. Saenko,
608 M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36,
609 pp. 46534–46594. Curran Associates, Inc., 2023.
- 610 Jakob Nielsen and Rolf Molich. Heuristic evaluation of user interfaces. In *Proceedings of the*
611 *SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, pp. 249–256, New York,
612 NY, USA, 1990. Association for Computing Machinery. ISBN 0201509326. doi: 10.1145/97243.
613 97281.
- 614 OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, and Lama Ahmad et. al. Gpt-4 technical
615 report, 2024. URL <https://arxiv.org/abs/2303.08774>.
- 616
617 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agar-
618 wal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya
619 Sutskever. Learning transferable visual models from natural language supervision. In Marina
620 Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine*
621 *Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR,
622 18–24 Jul 2021.
- 623 Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-
624 networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*
625 *Processing*. Association for Computational Linguistics, 11 2019. URL [https://arxiv.](https://arxiv.org/abs/1908.10084)
626 [org/abs/1908.10084](https://arxiv.org/abs/1908.10084).
- 627 Shreya Shankar, J. D. Zamfirescu-Pereira, Björn Hartmann, Aditya G. Parameswaran, and Ian
628 Arawjo. Who validates the validators? aligning llm-assisted evaluation of llm outputs with human
629 preferences, 2024.
- 630
631 Debbie Stone, Caroline Jarrett, Mark Woodroffe, and Shailey Minocha. *User interface design and*
632 *evaluation*. Elsevier, 2005.
- 633 Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, and et. al. Gemini 1.5:
634 Unlocking multimodal understanding across millions of tokens of context, 2024.
- 635
636 Chunxiao Wang, Shuai Lu, Hongzhong Chen, Ziwei Li, and Borong Lin. Effectiveness of one-click
637 feedback of building energy efficiency in supporting early-stage architecture design: An exper-
638 imental study. *Building and Environment*, 196:107780, 2021. ISSN 0360-1323. doi: <https://doi.org/10.1016/j.buildenv.2021.107780>. URL [https://www.sciencedirect.com/](https://www.sciencedirect.com/science/article/pii/S0360132321001876)
639 [science/article/pii/S0360132321001876](https://www.sciencedirect.com/science/article/pii/S0360132321001876).
- 640
641 Jason Wu, Xiaoyi Zhang, Jeff Nichols, and Jeffrey P Bigham. Screen parsing: Towards reverse
642 engineering of ui models from screenshots. In *The 34th Annual ACM Symposium on User In-*
643 *terface Software and Technology*, UIST '21, pp. 470–483, New York, NY, USA, 2021. Associa-
644 tion for Computing Machinery. ISBN 9781450386357. doi: 10.1145/3472749.3474763. URL
645 <https://doi.org/10.1145/3472749.3474763>.
- 646
647 Jason Wu, Yi-Hao Peng, Amanda Li, Amanda Swearngin, Jeffrey P. Bigham, and Jeffrey Nichols.
Uiclip: A data-driven model for assessing user interface design, 2024. URL [https://arxiv.](https://arxiv.org/abs/2404.12500)
[org/abs/2404.12500](https://arxiv.org/abs/2404.12500).

702 generate joint task and screenshot embeddings, and cosine similarity to determine relatedness. For
703 filtering, we augmented the dataset’s comments with LLM-generated comments deemed incorrect
704 by annotators (Duan et al. (2024b)). For bounding box generation, refinement, and subsequent steps
705 that operate on individual comments, we sampled few-shot examples by selecting the most seman-
706 tically similar comments and their corresponding bounding boxes from a split of UICrit. We used
707 sentenceBERT (Reimers & Gurevych, 2019) to embed the comment text for similarity ranking. For
708 validation, few-shot examples of invalid comments were selected from incorrect comments that were
709 marked by dataset annotators, or from irrelevant comments from other UIs. Finally, for text refine-
710 ment, multiple invalid comments were selected, following the process described earlier, and then
711 sorted by increasing cosine similarity to simulate the comment refinement process.

712 For bounding box refinement, we considered another technique to generate fewshot examples. This
713 technique involves selecting the first bounding box location based on visual similarity of the region it
714 contains in the fewshot UI to that of the region contained by the input bounding box proposal of the
715 input screenshot. This bounding box is then gradually moved closer to the ground truth bounding
716 box for the fewshot UI to simulate the refinement process. However, we found that the simpler
717 approach of randomly perturbing the bounding box actually gave better results (IoU 0.357 (random
718 perturbation, from Table 1) vs 0.333 (visual similarity match)).

719 A.2.2 OPEN VOCABULARY OBJECT AND ATTRIBUTE DETECTION

721 For text generation (i.e., category and attributes) and filtering, we sampled images based on the
722 semantic similarity of their CLIP embeddings. Negative text samples for the filtering step were
723 generated by sampling irrelevant text from other images. For bounding box generation, refinement,
724 and subsequent steps applied to individual text items, we sampled few-shot examples by selecting
725 the most semantically similar text items and their corresponding bounding boxes from a split of their
726 annotated dataset. We used sentenceBERT Reimers & Gurevych (2019) to embed the text items for
727 similarity ranking. For validation, invalid text examples were perturbed by randomly swapping the
728 category or attributes, or by deleting or adding attributes. Similarly, for text refinement, few-shot
729 examples were generated by perturbing the text in decreasing amounts.

731 A.3 INSTRUCTIONS PROMPTS FOR PIPELINE

732 We provide the instructions prompt for each step of the pipeline for the UI Critique Task.

733 Text Generation: For these sets of guidelines: [Guidelines]. Please find
734 all the guideline violations in the UI provided. For violation found,
735 please provide an explanation that includes these three things: 1.
736 the expected standard (i.e. what good design should look like), 2.
737 the gap between the current design and the expected standard (i.e.
738 the critique for the design), and 3. how to fix the issue in the
739 current design. For formatting each violation, please include these
740 three things in separate sentences. For the expected standard (#1),
741 start the sentence with 'The expected standard is that...'. For the
742 gap (#2), start the sentence with 'In the current design, ...', and
743 for how to fix the design (#3), start the sentence with 'To fix this
744 ...'. Please end each violation explanation with two newline
745 characters (\n\n). Please be specific in your violation explanations,
746 making sure to refer to specific UI elements and groups in the UI.
747 After determining all guideline violations, please also share any
748 other design feedback you have for the UI and follow the same format
749 of providing the expected standard, the critique for the design, and
750 how to fix the issue. We will provide N examples of a UI screenshot
751 and a set of valid design comments. Please learn how to give valid
752 design comments from these examples and apply this knowledge to
753 determine valid design comments for the last UI. Please be specific
754 in your comments, referring to specific UI elements by their text
755 label or icon, like in the examples provided. Also, please do not
return any comments regarding user testing nor adherence to platform
standards.

756 Text Filtering: For the provided UI and a list corresponding design
757 comments, please filter out the incorrect design comments and return
758 a list tuples. Each tuple contains its index i in the list, followed
759 by True or False. The tuple would contain True if the design comment
760 at index i in the input list is a valid design comment, and False if
761 the design comment at index i is an invalid comment. Please analyze
762 the UI screenshot to determine whether or not each design comment is
763 valid. We will give N examples, where each UI screenshot is followed
764 by a list of its corresponding design comments and an output list of
765 tuples, where each tuple contains the list index and True/False
766 indicating the validity of the design comment at that index. Please
767 learn from these examples, analyzing the UI screenshot to see why
768 each comment was considered valid or invalid. Finally, we will give a
769 UI screenshot, followed by its corresponding design comments. Please
770 output a list of tuples consisting of the comment's list index and
771 an indication of each comment's validity, like in the provided
772 examples. Please output False for the design comment if it is about
consistency with the brand, user testing, or adherence to platform
standards. Please only output this list of tuples and nothing else.

773 Bounding Box Generation: You will be providing bounding boxes coordinates
774 for the provided UI screenshot and design comment. The bounding box
775 will enclose a relevant region in the screenshot that is discussed in
776 the design comment. You will use the coordinate axes along the edge
777 of the screenshot to determine the coordinates of the bounding box.
778 Please make sure you follow the provide coordinate axes, so that
779 vertical bounding box coordinates are between 0 and 16 and horizontal
800 bounding box coordinates are between 0 and 9, and format the
801 bounding box coordinates as (left, top, right, bottom). Please do not
802 output bounding boxes with area 0. Also, please only output the
803 bounding box and nothing else. We will provide N examples of design
804 comments, followed by the corresponding UI screenshot (with a
805 coordinate axis along its edge) and a correct bounding box for the
806 design comment in the UI screenshot based on the coordinate axis.
807 Please learn how to determine accurate bounding boxes for the design
808 comment in the UI screenshot based on these examples. We will provide
809 a final design comment and UI screenshot; please apply what you have
learned from the examples to determine an accurate bounding box for
this final design comment and UI screenshot only.

810 Bounding Box Refinement: You will be refining bounding boxes for a given
811 UI screenshot and design comment. The bounding box will enclose a
812 relevant region in the screenshot that is discussed in the design
813 comment. You will be given a proposed bounding box candidate and will
814 evaluate whether or not this bounding box accurately encloses the
815 region in the screenshot that is discussed in the comment. The
816 proposed bounding box coordinates, in the format of (left_coordinate,
817 top_coordinate, right_coordinate, bottom_coordinate) and is
818 displayed as a blue box in the screenshot patch that is also provided
819 , with some additional margin around the blue bounding box. Please
820 reflect on whether or not this bounding box is accurate and look
821 closely at the UI elements contained in the blue bounding box to
822 judge its accuracy and relevance to the design comment. If the
823 bounding box is not accurate, please output a new bounding box that
824 you think is accurate in the format of (left_coordinate,
825 top_coordinate, right_coordinate, bottom_coordinate), where each
826 coordinate is determined from the coordinate axes along the edge of
827 the UI screenshot provided earlier. Please make sure the new bounding
828 box you output is accurate, and refer to the coordinate axes along
829 the edge of the zoomed-in screenshot patch and the entire screenshot
830 (provided earlier) to determine the bounding box coordinates. If the
831 bounding box is accurate, please output 'BOUNDING BOX IS ACCURATE,
832 PLEASE TERMINATE'. Please only output either the updated bounding or
833 'BOUNDING BOX IS ACCURATE, PLEASE TERMINATE' and nothing else. We
834 will provide N examples of bounding box refinements for a given

810 design comment, UI screenshot, and bounding box candidate. Please
811 learn how to accurately refine bounding boxes for the design comment
812 in the UI screenshot based on these examples. We will provide a final
813 design comment, UI screenshot, and bounding box candidate; please
814 apply what you have learned from the examples to accurately refine
815 the bounding box candidate for this final design comment, UI
816 screenshot, and the zoomed in patch showing the bounding box
817 candidate.

818 Text and Bounding Box Validation: You are given a UI screenshot, design
819 comment for the UI screen, and a zoomed-in patch of the UI screenshot
820 showing the corresponding bounding box for the design comment.
821 Please evaluate the accuracy of the design comment and bounding box
822 with respect to the UI screenshot. The bounding box is displayed as a
823 blue box in the zoomed-in screenshot patch, and is supposed to
824 contain the region in the UI screen that is targeted by the design
825 comment. Please first evaluate if the design comment is valid for the
826 provided UI screenshot, i.e. if it correctly points out a design
827 issue and suggests an accurate way to fix it. Please analyze the
828 provided UI screenshot to assess the comment's validity. If the
829 design comment is valid, please next evaluate whether the blue box in
830 zoomed-in UI screenshot contains the region that is relevant to the
831 design comment. If the design comment is invalid and the blue box
832 still contains a region in the UI screenshot with design issues,
833 please return the label 'Incorrect Comment'. If the comment is valid,
834 but the blue box does not contain the region relevant to the comment
835 , please return the label 'Incorrect Bbox'. If the comment is invalid
836 and the blue box does not contain a region with design issues,
837 please return the label 'Both Incorrect'. Finally, if the design
838 comment is valid and the blue box correctly contains a region in the
839 UI that is relevant to the comment, please return the label 'Both
840 Correct'. Please only return the appropriate label and nothing else.
841 We will give N examples, the UI screenshot (labeled 'UI Screenshot'),
842 followed by the design comment (labeled 'Design Comment'), a zoomed-
843 in screenshot patch showing the blue bounding box (labeled 'Zoomed-in
844 Patch'), and finally the correct label (labeled 'Label') for the
845 accuracy of the UI screenshot, design comment, and corresponding
846 bounding box. Please learn from these examples, to see how to
847 correctly categorize the design comment and its corresponding
848 bounding box by accuracy. Finally, we will give a UI screenshot,
849 design comment, and a zoomed-in patch showing the corresponding blue
850 bounding box. Please apply what you have learned from the examples to
851 correctly classify the accuracy of the design comment and its
852 corresponding bounding box.

848 Text Refinement: You will be refining the design comment for a specific
849 region in a UI screenshot. You will be given a UI screenshot, a
850 zoomed-in patch of the screenshot with a blue box containing the
851 region of interest, and a design comment for the UI region inside the
852 blue box. Please evaluate whether or not the design comment
853 accurately describes the design issue for the UI region inside the
854 blue box. If the design comment is accurate, please output 'COMMENT
855 IS ACCURATE, PLEASE TERMINATE'. If the design comment is not accurate
856 , please refine the design comment to the accurate and output this
857 accurate design comment for the region of interest, following the
858 same format as the input design comment. We will provide N examples
859 of bounding box refinements for each UI screenshot, region of
860 interest, and design comment candidate for the region of interest.
861 Please learn how to accurately refine the design comment for the
862 region of interest in the UI screenshot based on these examples. We
863 will provide a final UI screenshot, region of interest, and design
864 comment candidate for the region of interest; please apply what you
865 have learned from the examples to accurately refine design comment
866 candidate for this final UI screenshot and region of interest. Please

864 only output the refined comment or 'COMMENT IS ACCURATE, PLEASE
865 TERMINATE' and nothing else.
866
867

868 A.4 QUALITATIVE ANALYSIS

869 A.4.1 QUALITATIVE ANALYSIS OF OUTPUTS FROM PIPELINE, BASELINE, AND FINETUNED 870 LLM 871

872 We qualitatively analyzed the outputs from our pipeline, baseline, and finetuned Llama-3.2 11b.
873 Figures 5, 6, and 7 illustrate the design feedback and corresponding bounding boxes generated by our
874 pipeline (using Gemini-1.5-pro) for a diverse set of 12 UIs. Figure 8 presents two examples where
875 our pipeline outperformed the baseline, and Figure 9 contains two examples where the baseline
876 performed better. To enable easier comparison between the two conditions, we used the same set
877 of initial comments from the TextGen module, as both the pipeline and baseline begin with this
878 module.

879 As shown in figures 5, 6, and 7, we found that, more often than not, the pipeline generates helpful
880 comments with reasonably accurate bounding boxes (highlighted in green). For the baseline, we ob-
881 served that it frequently generates very generic comments that would apply to any UI screen and are
882 usually not helpful, such as suggesting that at design should be tested with users or needs to be made
883 responsive as shown in Figure 8 (Baseline, top screenshot). These comments are usually eliminated
884 by the pipeline (Pipeline, top screenshot). Additionally, the pipeline successfully refined incorrect
885 comments, as shown by the red and green comments in the top screenshot, and filters out incorrect
886 comments during the validation stages as shown in both screenshots. For bounding boxes, those
887 generated by the pipeline are usually tighter and closer to the correct region compared to the base-
888 line, which often generates large, unspecific bounding boxes that encompass a significant portion of
889 the screen, as shown by the bounding boxes in Figures 8 and 9. This demonstrates the effectiveness
890 of iterative refinement and validation in improving bounding box accuracy. Furthermore, the large
891 bounding boxes generated by the baseline would decrease the chance of the IoU being zero, which
892 may have inflated the average IoU shown in Tables 2 and 3.

893 The pipeline sometimes eliminated valid comments, as shown in both examples in Figure 9, where
894 the green comments were accurate comments that were eliminated. In the top screenshot, the
895 pipeline retained only one inaccurate comment, although its bounding box was significantly im-
896 proved. In the bottom screenshot, the pipeline produced a less accurate bounding box around the
897 red buttons compared to the baseline, though these instances are rare.

898 We found that fine-tuned Llama-3.2 generated a very limited range of comments, primarily focusing
899 on text readability, visual clutter, and generic critiques about the need for improved visual appeal.
900 This limited range could be due to the over-representation of such critiques in UICrit. Figure 10
901 presents example outputs for two screenshots, comparing them with outputs from our pipeline. The
902 figure shows that, in addition to its limited range of critiques, the finetuned model also produces in-
903 accurate comments. In contrast, our pipeline generates a significantly more diverse set of comments
904 with tighter bounding boxes, though the bounding boxes are generally less accurate than those from
905 the fine-tuned model.

906 Overall, the pipeline generally outperforms the baseline qualitatively, reducing the generation of
907 invalid and generic comments and outputting bounding boxes that are tighter, more specific, and
908 closer to the target region. Furthermore, it generates a considerably more diverse set of comments
909 compared to finetuned Llama, though its visual grounding is less accurate.

910 A.4.2 QUALITATIVE ANALYSIS OF PIPELINE OUTPUTS FOR OUT OF DOMAIN UIs 911

912 Since UICrit consists of older UIs (from 2014) Duan et al. (2024b), we evaluated the pipeline’s
913 performance to determine whether it generalizes to modern UIs and other out-of-domain UIs, such
914 as websites, using only few-shot examples selected from UICrit. Figure 11 displays the generated
915 feedback for four modern Android UIs (the few-shot examples from UICrit are also Android UIs)
916 from 2024, taken from Mobbin³. Figure 12 presents feedback for four modern iOS UIs from 2024,
917

³<https://mobbin.com/>

918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971

The expected standard is that the design should use a clear and easy-to-read font. In the current design, the font is too small and difficult to read, especially on the smaller screens of mobile devices. To fix this, the font size should be increased and a more legible font should be used.

The expected standard is to ensure that text buttons are sufficiently sized and have an appropriate font width for clear visibility and usability. In the current design, the text button size is too small, and the font width is too low, resulting in poor visibility of the text. To fix this, the size of the text buttons should be increased to make them more prominent and easier to interact with. Additionally, adjusting the font width to a more appropriate level will improve the legibility of the text.

The expected standard is that the design should be clear and easy to understand. In the current design, it is not clear what the user is supposed to do on the screen. The title "Create or Edit Period" is not very descriptive. To fix this, make the purpose of the screen more clear. For example, the title could be changed to "Create a New Period" or "Edit an Existing Period".

The expected standard is that the design should be consistent. In the current design, the buttons are not consistent in style. The "Confirm" and "Cancel" buttons are different sizes and shapes than the buttons at the top of the screen. To fix this, make all of the buttons in the design consistent in style.

The expected standard is that the design should be consistent. In the current design, the buttons are not consistent in style. The "Confirm" and "Cancel" buttons are different sizes and shapes. To fix this, make the buttons consistent in style.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, there's an excess of text, creating a cluttered appearance. To fix this, we can consider condensing the text or simplifying the content to reduce clutter and improve readability.

The expected standard is that every element should have some connection to another element on the page. In the current design, the last paragraph texts are partially hidden by the OK button. To fix this, adjust the position of the OK button to avoid obscuring the text in the last paragraph.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small and the background makes the foreground text difficult to read. To fix this, we can increase the text font size and choose a different font color and choose a different contrasting background.

The expected standard is to have a clear visual separation between different sections of content to improve readability and visual hierarchy. In the current design, the advertisement lacks clear visual separation from the rest of the content. To fix this, add a distinct border or spacing around the advertisement to visually separate it from the app's content.

The expected standard is the design should appropriately communicate the content to its intended audience. In the current design, the icons at the top are difficult to understand. To fix this, try choosing more suitable icons to carry the intended message.

The expected standard is that the text should be easy to read and understand. In the current design, the font is too small and the text is too dense, making it difficult to read. To fix this, the designer should use a larger font size and more line spacing. The expected standard is to have higher contrast between gray text/icons and the background, ensuring readability. In the current design, gray text and icons lack visual emphasis due to low contrast with the background, hindering readability. To fix this, adjust the contrast between gray text/icons and the background, or introduce accent colors for better visual emphasis and readability.

The expected standard is that the design should be clear and easy to understand. In the current design, it is not clear what the user is supposed to do with the text field. The text "Add notes or details of period here" is not very helpful. To fix this, provide more guidance to the user. For example, the text field could be labeled "Notes" or "Details".

The expected standard is that the design should be visually appealing and easy to use. In the current design, the layout is not visually appealing. The elements are not well-organized, and there is too much white space. To fix this, improve the layout of the design. The elements should be organized in a more visually appealing way, and the amount of white space should be reduced.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the color scheme is not visually appealing. The use of a black background with white text is harsh on the eyes. To fix this, use a more visually appealing color scheme.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small. To fix this, we can increase the text font size.

Figure 5: Illustration of four example outputs from the pipeline. The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box. Helpful comments with reasonably accurate bounding boxes are highlighted in screen.

sourced from DesignVault⁴, and Figure 13 illustrates feedback for five modern websites from 2024,

⁴<https://designvault.io/>

972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025

The expected standard is that the design should match the importance of content to its visual prominence - make the most important information visually dominant. Use clear contrast to distinguish different levels of information. In the current design, the current background color creates low contrast with the foreground element, hindering readability for users. To fix this, consider using a lighter background for better contrast.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text is difficult to read and understand because it is too small and the background makes the foreground text difficult to read. To fix this, the designer should use a larger font size and choose a different font color or choose a different contrasting background.

The expected standard is that the design should be accessible to all users. In the current design, the design is not accessible to all users. The small font size and low contrast make it difficult for users with visual impairments to read the text. To fix this, the font size should be increased, the contrast should be improved, and the design should be tested with users with disabilities.

The expected standard is that the design should be easy to use and navigate. In the current design, the design is not easy to use or navigate. The buttons are small and difficult to tap, and the overall layout is confusing. To fix this, the buttons should be made larger and easier to tap, and the overall layout should be simplified.

The expected standard is that the design should use a consistent font throughout. In the current design, the design uses multiple fonts, which makes it look cluttered and unprofessional. To fix this, the design should use a single font throughout.

The expected standard is to have a prominent 'x' icon in the top-right corner of the pop-up window for easy closure, ensuring seamless user interaction. In the current design, the absence of a 'x' icon impedes user experience by complicating the closure of pop-up windows. To fix this, simply add a clear 'x' icon in the top-right corner of the pop-up window, enabling users to close it effortlessly.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the font is too small, making it difficult to read. To fix this, the designer should use a larger font size.

The expected standard is that the design should be visually appealing and easy to understand. In the current design, the color scheme, particularly the combination of yellow and white, might not be suitable for all users, especially those with visual impairments. To fix this, explore alternative color combinations that offer better contrast and are accessible to a wider range of users.

The expected standard is that the design should be visually appealing and easy to understand. In the current design, the dotted line above the image container appears visually distracting and doesn't contribute to the overall aesthetics or functionality. To fix this, remove the dotted line to create a cleaner and more visually appealing design.

The expected standard is that the page should have a next button on the page. In the current design, there is no button on the page for navigating further. To fix this, add a button for further navigation with mentioning words like (next or continue)

The expected standard is that the design should be visually appealing and easy to use. In the current design, the use of white space is excessive. To fix this, the amount of white space should be reduced and the elements should be arranged in a more organized manner.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, there are too many elements on the screen, which makes it difficult to focus on the important information. To fix this, the number of elements on the screen should be reduced and the most important information should be highlighted.

The expected standard is that the design should have a clear hierarchy of information. In the current design, the design does not have a clear hierarchy of information. All of the text is the same size and weight, which makes it difficult to scan and understand. To fix this, the design should use different font sizes and weights to create a clear hierarchy of information.

The expected standard is that the design should be visually appealing. In the current design, the design is not visually appealing. It is cluttered, overwhelming, and difficult to read. To fix this, the design should be simplified, the number of colors should be reduced, and a clear hierarchy of information should be created.

The expected standard is that the text and background colors used in the design should be complementary and easy to read. In the current design, texts are in grey color on a black background which are not making a good contrast. To fix this, change colors of texts and backgrounds to be more complementary to each other and make it easier to read.

The expected standard is that icons should indicate the state of the icons. In the current design, all icons are all gray which indicate that none of the icons are active. To fix this, change the color of the icons to blue that are in an active state.

The expected standard is that the design should effectively utilize white space and have an appealing visual presentation. In the current design, there are issues with the utilization of white space, resulting in an unappealing layout. To fix this, the design should undergo a thorough review to eliminate irrelevant blank space and ensure optimal utilization of white space for a more visually pleasing layout.

The expected standard is that the design should make the most important information visually dominant. In the current design, the back button size is small. To fix this, increase the back button size to make it visually prominent for the users.

The expected standard is that the text's visual treatment and formatting should make it easy to understand. In the current design, the text (Accessories) is too small and difficult to read. To fix this, increase font size and weight to make it easier to read.

The expected standard is that the design should be visually appealing and easy to understand. In the current design, the arrangement of elements within the layout appears cluttered and lacks a clear visual hierarchy. To fix this, reorganize the elements, ensuring ample spacing between them, to establish a clear visual hierarchy and improve readability.

Figure 6: Illustration of four example outputs from the pipeline. The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box. Helpful comments with reasonably accurate bounding boxes are highlighted in screen.

also taken from Mobbin. In these figures, helpful comments with reasonably accurate bounding boxes are highlighted in green.

1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079



Figure 7: Illustration of four example outputs from the pipeline. The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box. Helpful comments with reasonably accurate bounding boxes are highlighted in screen.

The pipeline was able to provide helpful feedback with reasonably accurate bounding boxes for these out-of-domain UIs. It performed surprisingly well on the modern iOS UIs, with results comparable to those for the UIs from the test split of UICrit, as shown in Figures 5, 6, 7, and 8. Additionally, the pipeline even managed to generate helpful feedback and bounding boxes for websites. While design

1080 principles often overlap between mobile and web interfaces, their layouts and screenshot dimensions
1081 differ significantly. This suggests that the LLM was able to generalize and adapt its knowledge to
1082 generate and refine bounding boxes for website screenshots, despite only being trained with few-shot
1083 examples from mobile screenshots, which are very different.

1084 An interesting observation is that, since websites have more screen space, they are generally more
1085 complex and information-dense than mobile UIs (Gazzawe, 2017). We found one instance where
1086 the pipeline incorrectly flagged a relatively simple website as being too complex (i.e. having too
1087 many elements) in Figure 13 (second screen from the bottom), likely because it evaluated the com-
1088 plexity based on the mobile standards presented in the few-shot examples. However, the pipeline
1089 did correctly critique the bottom screenshot in the same figure for being overly complex, showing
1090 that it can appropriately identify this issue in some cases.

1091 A.4.3 REFINING BOUNDING BOXES

1092 While the bounding boxes could be improved qualitatively, as shown in Figures 5, 6, 7, and 8, there
1093 are straightforward approaches to easily improve the bounding box accuracy. For instance, the DOM
1094 tree representation of the UI contains the exact bounding boxes of UI elements and element groups.
1095 This information could be used to refine the output bounding boxes for the elements/groups dis-
1096 cussed in the critiques by finding the closest bounding box from the DOM tree via IoU comparison,
1097 distances between the bounding box centers and sizes, or utilizing an LLM for matching, as was
1098 done in Zhao et al. (2024). This DOM representation is available through the UI’s XML code, or the
1099 internal UI mockup representation available in design tools like Figma. In the case that the DOM
1100 tree is not available, we could use a screen object parser (Wu et al., 2021) to extract UI element and
1101 group locations from the screenshot.

1102 We demonstrate the results of using the DOM tree (taken from the XML-based Android View Hier-
1103 archy available in RICO(Deka et al., 2017)) to refine the pipeline’s bounding boxes for some of the
1104 UICrit UIs in Figures 5, 6, 7, and 8. As discussed above, we matched the generated bounding boxes
1105 with those from the UI elements and groups in the DOM tree via an IoU threshold. The results are
1106 shown in Figures 14 and 15 and illustrate that this simple refinement method significantly improves
1107 the bounding boxes. This step could potentially be applied at the end of the pipeline to clean up the
1108 generated bounding boxes.

1109 A.5 ANALYSIS OF ITERATIVE REFINEMENT

1110 Figure 16 illustrates an example of iterative bounding box refinement (conditioned on the comment)
1111 by BoxRefine, which terminates on a significantly more accurate bounding box. Figure 17 illus-
1112 trates an example of comment refinement (conditioned on the bounding box) by TextRefine, which
1113 terminates on an accurate comment on the poor layout of the region inside the bounding box.

1114 We calculated the average number of bounding box refinements, which were 1.25 for Gemini-1.5-
1115 pro and 0.88 for GPT-4o, as well as the average number of comment refinements, which were
1116 1.48 for Gemini-1.5-pro and 1.17 for GPT-4o. Additionally, we estimated the expected number of
1117 LLM calls required for a complete run of the pipeline, including the small fraction sent for further
1118 refinement by Validation. The expected number of calls is 7.16 for Gemini-1.5-pro and 6.70 for
1119 GPT-4o.

1120 A.6 HUMAN EVALUATION METHOD

1121 Figure 18 shows a snippet of the form used by human design experts to rate the quality of individual
1122 comments and rank the comment sets for the three different conditions.

1123 Given the limited availability of UI design experts and the extensive evaluation required per UI
1124 screen for a detailed comparison across the three conditions, only the Gemini-1.5-pro outputs for 33
1125 UI screenshots were rated. To better represent the UI design space in this sample, we maximized
1126 the diversity of the UI screenshots by randomly sampling an even number of UIs from each of the
1127 UI task categories identified by Duan et al. (2024b). We followed their method of clustering by task
1128 descriptions from UICrit to obtain the task clusters. These 33 UIs were split into 6 groups for rating,
1129 with three participants assigned to each group. The rating and ranking study took approximately
1130 1 hour. We recruited 18 design experts for this study. Five of the participants had 2-4 years of

1134 design experience, and the rest had 6-10 years. Their areas of design expertise include mobile, web,
 1135 interaction, and user experience research.

1137 A.7 ALGORITHMS FOR GENERATING FEW-SHOT EXAMPLES FOR BOUNDING BOX
 1138 REFINEMENT

1139 Algorithm 1 details the steps for generating the few-shot refinement examples for a selected bound-
 1140 ing box. The few-shot generation algorithm entails perturbing the bounding box coordinates by
 1141 decreasing amounts and adding the perturbations to the list of few-shot examples. The algorithm for
 1142 perturbing a bounding box is also shown in Algorithm 1.
 1143

1144 **Algorithm 1** Generate Bounding Box Refinement Few-shot Examples

1146 **Require:** the bounding box to be perturbed $input_bbox$, the fraction that the bounding box’s coord-
 1147 inates will be perturbed $perturb_frac$

1148 **Ensure:** The coordinates of $input_bbox$ perturbed by $perturb_frac$

```

1149 1: function GENERATE_PERTURB( $input\_bbox$ ,  $perturb\_frac$ )
1150 2:   Compute  $left\_margin$ ,  $right\_margin$ ,  $top\_margin$ ,  $bottom\_margin$ 
1151 3:    $all\_perturbed \leftarrow []$ 
1152 4:   for  $x\_perturb$  in  $[-perturb\_frac \times left\_margin, perturb\_frac \times right\_margin]$  do
1153 5:     for  $y\_perturb$  in  $[-perturb\_frac \times top\_margin, perturb\_frac \times bottom\_margin]$  do
1154 6:       Update bounding box location based on  $x\_perturb$ ,  $y\_perturb$ 
1155 7:       Add perturbed bounding box to  $all\_perturbed$ 
1156 8:     end for
1157 9:   end for
1158 10:   $final\_perturbed \leftarrow []$ 
1159 11:  Compute  $width$  and  $height$  of the input bounding box
1160 12:  for each  $perturbed\_bbox$  in  $all\_perturbed$  do
1161 13:    for  $width\_fraction$  in  $[-perturb\_frac, perturb\_frac]$  do
1162 14:      for  $height\_fraction$  in  $[-perturb\_frac, perturb\_frac]$  do
1163 15:        Update bounding box size based on  $width\_fraction$  and  $height\_fraction$ 
1164 16:      end for
1165 17:    end for
1166 18:  end for
1167 19:   $filtered\_perturbed \leftarrow remove\_invalid\_perturbed\_bbox(final\_perturbed, input\_bbox)$ 
1168 20:   $final\_bbox \leftarrow random.choice(filtered\_perturbed)$ 
1169 21:  return  $final\_bbox$ 
1170 22: end function

```

1171 **Require:** Bounding box $bbox$, maximum number of perturbations of $bbox$ in the list of fewshot
 1172 refinement examples $max_num_perturb$

1173 **Ensure:** A list of bounding boxes coordinates that are perturbed versions of $bbox$ in decreasing
 1174 amounts, where $bbox$ is the last item in the list.

```

1175 23: function GENERATE_PERTURBED_FEWSHOT_EXAMPLES( $bbox$ ,  $max\_num\_perturb$ )
1176 24:   $perturb\_options \leftarrow LIST(range(max\_num\_perturb + 1))$ 
1177 25:   $num\_perturb \leftarrow RANDOM.CHOICE(perturb\_options)$ 
1178 26:   $perturb\_list \leftarrow []$ 
1179 27:  for  $j \leftarrow num\_perturb$  to 1 do
1180 28:     $perturb\_frac \leftarrow j/max\_num\_perturb$ 
1181 29:     $output\_bbox \leftarrow GENERATE\_PERTURB(bbox, perturb\_frac)$ 
1182 30:     $perturb\_list.append(output\_bbox)$ 
1183 31:  end for
1184 32:   $perturb\_list.append(bbox)$ 
1185 33:  return  $perturb\_list$ 
1186 34: end function

```

1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241

The expected standard is that the page should have a back button. In the current design, there is no button on the page for going to the previous page. To fix this, add a button for going back to the previous page.

The expected standard is that the design should use a clear color scheme. In the current design, the 'FINISH' button uses the same red color as the header, which can be confusing for users. To fix this, the 'FINISH' button should be a different color that contrasts well with the header, such as white or black.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the color scheme is not appealing. To fix this, use a more visually appealing color scheme.

The expected standard is that the text's visual treatment and formatting should make it easy to understand. In the current design, the texts (Place Added) are too small and difficult to read. To fix this, increase font size and weight to make it prominent visible.

The expected standard is that the text and background colors used in the design should be complementary and easy to read. In the current design, texts are in white color on red background which are difficult to read. To fix this, change colors to be more complementary to each other (texts in dark colors) to make it easier to read.

The expected standard is that the page should have a back button. In the current design, there is no button on the page for going to the previous page. To fix this, add a button for going back to the previous page.

The expected standard is that the design should be consistent with the rest of the app. In the current design, the color scheme and font are different from the rest of the app. To fix this, use the same color scheme and font as the rest of the app.

The expected standard is that the design should be accessible to all users. In the current design, the contrast between the text and the background is not sufficient for users with visual impairments. To fix this, increase the contrast between the text and the background.

The expected standard is that the design should be documented so that others can understand how it works and how to use it. In the current design, it is not clear whether the design is documented. To fix this, document the design so that others can understand how it works and how to use it.

The expected standard is that the design should be tested with users to ensure that it is usable and meets their needs. In the current design, it is not clear whether the design has been tested with users. To fix this, test the design with users to ensure that it is usable and meets their needs.

The expected standard is that the design should be easy to understand. In the current design, it is not clear what the user is supposed to do next. To fix this, add a clear call to action, such as a button that says 'Next' or 'Continue.'

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the line is unnecessary. To fix this, remove the line.

The expected standard is to have a noticeable clickable element with clear contrast. In the current design, the element on the screen appears non-clickable and seamlessly integrated with the background, giving users the impression that it is merely a static part of the display rather than an interactive component. To fix this, add a border or make a tile, so it doesn't blend in with the background.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small. To fix this, increase the font size of the text.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the text 'OR' is unnecessary. To fix this, remove the text 'OR'.

The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the text 'Old School Login' is not visually prominent. To fix this, enlarge the text and use a different font color or make it bold.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the line is unnecessary. To fix this, remove the line.

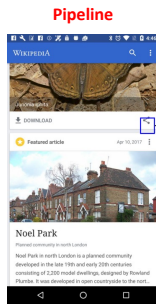
The expected standard is that the g+ icon and text should be easily distinguishable. In the current design, the button's icon and text lack clear differentiation, causing potential confusion for users. To fix this, enhance visual separation between the icon and text, create contrast for clarity, and align them properly.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the text 'OR' is unnecessary. To fix this, remove the text 'OR'.

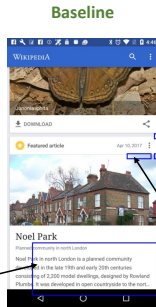
The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the text 'Old School Login' is not visually prominent. To fix this, enlarge the text and use a different font color or make it bold.

Figure 8: Illustration of outputs from the pipeline and baseline, highlighting two cases where our pipeline outperformed the baseline. The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box (some comments have the same bounding box). Both the pipeline and baseline begin with the TextGen module, so we used the same initial comments from TextGen for both conditions for easier comparison. In the top example, the pipeline produced more accurate bounding boxes, eliminated several generic and unhelpful comments, and refined an inaccurate comment (red) into a more accurate one (green). In the bottom example, the pipeline produced more considerably more accurate bounding boxes, and eliminated an invalid comment (red).

1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295



The expected standard is that the design should be consistent throughout. In the current design, the share icon is not consistent. To fix this, we can use the same share icon.

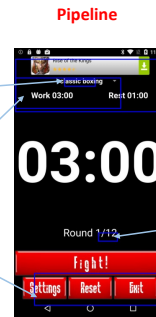


The expected standard is that the design should be consistent throughout. In the current design, the share icon is not consistent. To fix this, we can use the same share icon.

The expected standard is that the design should be consistent throughout. In the current design, the three dot icon is not consistent. To fix this, we can use the same three dot icon.

The expected standard is that the design should be consistent throughout. In the current design, the date format is not consistent. To fix this, we can use the same date format.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small. To fix this, we can increase the text font size.



The expected standard is that the design should be visually appealing and easy to use. In the current design, the font size of the text "Round 1/12" is too small. To fix this, the designer should increase the font size of the text "Round 1/12".

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small and the background makes the foreground text difficult to read. To fix this, we can increase the text font size and choose a different contrasting background.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small and the background makes the foreground text difficult to read. To fix this, we can increase the text font size and choose a different contrasting background.

The expected standard is that the design should use as few elements as possible to achieve its goals. Each visual element should contribute to the overall message; all non-essential elements should be omitted. In the current design, there are too many elements on the screen, making it difficult to focus on any one thing. To fix this, the designer should remove any unnecessary elements from the screen.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the font size of the text "Round 1/12" is too small. To fix this, the designer should increase the font size of the text "Round 1/12".



The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the text "classic boxing" is not visually prominent. To fix this, we can increase the font size of the text "classic boxing".

The expected standard is that the design should be visually appealing and easy to use. In the current design, the text is not aligned properly. To fix this, the designer should align the text to the center.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the spacing between the elements is not consistent. To fix this, the designer should use a consistent spacing between the elements.

The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the text within the highlighted buttons lacks visual prominence. To fix this, we can increase the text font size.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the buttons are too close to each other. To fix this, the designer should add more space between the buttons.

The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the download button is not visually prominent. To fix this, we can enlarge the download button.

The expected standard is that the design should use as few elements as possible to achieve its goals. Each visual element should contribute to the overall message; all non-essential elements should be omitted. In the current design, there are too many elements on the screen, making it difficult to focus on any one thing. To fix this, the designer should remove any unnecessary elements from the screen.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the font size of the text "Round 1/12" is too small. To fix this, the designer should increase the font size of the text "Round 1/12".

Figure 9: Illustration of outputs from the pipeline and baseline, highlighting two cases where the baseline outperformed our pipeline. The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box (some comments have the same bounding box). Both the pipeline and baseline begin with the TextGen module, so we used the same initial comments from TextGen for both conditions for easier comparison. For the top example, while a lot of the comments from the baseline are inaccurate, the pipeline eliminated the only correct comment (green) and only kept an invalid comment (red), though its bounding box is considerably more accurate. In the bottom example, the pipeline removed two valid comments (green) and some invalid ones, and also made the bounding box around the comment regarding the red buttons less accurate.

1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349

Pipeline

The expected standard is that the page should have a back button. In the current design, there is no button on the page for going to the previous page. To fix this, add a button for going back to the previous page.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the color scheme is not appealing. To fix this, use a more visually appealing color scheme.

The expected standard is that the text's visual treatment and formatting should make it easy to understand. In the current design, the texts (Place Added) are too small and difficult to read. To fix this, increase font size and weight to make it prominent visible.

The expected standard is that the text and background colors used in the design should be complementary and easy to read. In the current design, texts are in white color on red background which are difficult to read. To fix this, change colors to be more complementary to each other (texts in dark colors) to make it easier to read.

The expected standard is that the design should use a clear color scheme. In the current design, the 'FINISH' button uses the same red color as the header, which can be confusing for users. To fix this, the 'FINISH' button should be a different color that contrasts well with the header, such as white or black.

Finetuned Model

The expected standard is that the text should be easy to read and respect rules of typography. In the current design, the text "You've Added Amanda Park, Washington" is difficult to read because it is too small and there is not enough contrast between the text and the background. To fix this, the text should be increased in size and the contrast between the text and the background should be increased.

Pipeline

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the line is unnecessary. To fix this, remove the line.

The expected standard is to have a noticeable clickable element with clear contrast. In the current design, the element on the screen appears non-clickable and seamlessly integrated with the background, giving users the impression that it is merely a static part of the display rather than an interactive component. To fix this, add a border or make a tile, so it doesn't blend in with the background.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small. To fix this, increase the font size of the text.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the text "OR" is unnecessary. To fix this, remove the text "OR".

The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the text "Old School Login" is not visually prominent. To fix this, enlarge the text and use a different font color or make it bold.

Finetuned Model

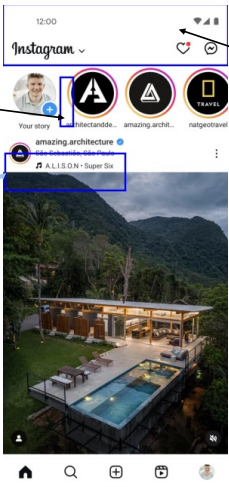
The expected standard is that the design should be visually appealing and easy to use. In the current design, the design is not visually appealing and is difficult to use. To fix this, the design should be made more visually appealing and easier to use.

The expected standard is that the text should be easy to read and respect rules of typography. In the current design, the text is difficult to read because it is too small and there is not enough contrast between the text and the background. To fix this, the text should be increased in size and the contrast between the text and the background should be increased.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, there are too many elements on the screen, which makes it difficult to focus on the important information. To fix this, the number of elements on the screen should be reduced and the most important information should be emphasized.

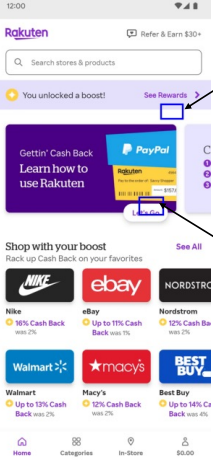
Figure 10: Illustration of outputs from the pipeline and finetuned Llama-3.2 11b. The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box (some comments have the same bounding box). The fine-tuned model produces a limited range of critiques, some of which are inaccurate (red), though the bounding boxes are generally accurate. In contrast, the pipeline generates a significantly more diverse set of critiques, and its bounding boxes are tighter but generally less accurate.

1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403



The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, there are two icons for adding stories. To fix this, remove one of the icons to keep the design simple.


The expected standard is that the text's visual treatment and formatting should make it easy to understand. In the current design, the text is too small and difficult to read. To fix this, increase font size and weight to make it easier to read.



The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, there are too many icons on the top of the screen. To fix this, remove unnecessary icons to keep the UI design simple so that users should not get distracted.

The expected standard is that the design should match the importance of content to its visual prominence - make the most important information visually dominant. In the current design, the "See Rewards" button is not visually prominent. To fix this, increase the font size and weight of the "See Rewards" button.

The expected standard is that the design should match the importance of content to its visual prominence - make the most important information visually dominant. In the current design, the "Let's Go" button is not visually prominent. To fix this, increase the font size and weight of the "Let's Go" button.




The expected standard is that the design should be visually appealing and easy to use. In the current design, the layout is cluttered and difficult to follow. To fix this, the layout should be simplified and the elements should be rearranged.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the color scheme is not visually appealing. To fix this, the color scheme should be changed to a more visually appealing color scheme.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the bell icon at the top right of the screen is not necessary and does not contribute to the overall message of the app. To fix this, the bell icon should be removed.

The expected standard is that the design should be visually appealing and easy to use. In the current design, the upload icon at the top right of the screen is not visually appealing and it is not clear what it is used for. To fix this, the upload icon should be replaced with a more visually appealing and informative icon.



The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the background makes the foreground text difficult to read. To fix this, we can choose a different contrasting background.

Figure 11: Example design feedback and bounding boxes generated by our pipeline for four modern Android UIs (from 2024). These UIs are out-of-domain inputs, as we used fewshot examples from only UICrit, which consists of older UIs (from 2014). The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box. Helpful comments with reasonably accurate bounding boxes are highlighted in screen.

1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457

The expected standard is that the design should be easy to understand and use. In the current design, the layout is cluttered and difficult to understand, and the information is not organized in a logical order. To fix this, the layout should be rearranged and made more intuitive, and the information should be organized in a more logical order.

The expected standard is that interactive elements should be visually distinct and afford interaction. In the current design, the "Cancel" button lacks visual prominence as it blends with the background. To fix this, provide a background color or border to the button to clearly distinguish it from the surrounding text.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small and the background makes the foreground text difficult to read. To fix this, we can increase the text font size and choose a different contrasting background.

The expected standard is to make the most important information visually dominant. In the current design, the text inside the button is not visually prominent. To fix this, we can increase the text font size.

The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the top navigation bar has too many icons, which can be overwhelming for users. To fix this, prioritize the most frequently used icons and consider grouping or hiding less important ones behind a menu.

The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the month labels (FEB, JAN) are large and visually heavy, drawing unnecessary attention. To fix this, reduce the size and visual weight of the month labels, making them less prominent.

The expected standard is that the design should use the positioning of elements relative to each other to deliberately achieve an active or restive appearance. In the current design, the "N SERIES" label below the "YOU" movie poster is small and difficult to read. To fix this, increase the size and contrast of the "N SERIES" label or consider a different visual treatment to improve its visibility.

The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the movie titles ("YOU", "That '90s Show") are very large and overwhelm other content. To fix this, reduce the size of the movie titles slightly to create a more balanced visual hierarchy.

The expected standard is that the design should use the positioning of elements relative to each other to deliberately achieve an active or restive appearance. In the current design, the layout feels somewhat cluttered and lacks a clear visual flow. To fix this, introduce more white space between elements and sections to improve visual breathing room and create a more organized layout.

The expected standard is that the layout should be organized and easy to understand. In the current design, the layout is cluttered and difficult to understand. To fix this, the designer should use a more organized layout. (Bounding Box is around the entire screenshot)

Figure 12: Example design feedback and bounding boxes generated by our pipeline for four modern iOS UIs (from 2024). These UIs are out-of-domain inputs, as we used fewshot examples from only UICrit, which consists of older UIs (from 2014). The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box. Helpful comments with reasonably accurate bounding boxes are highlighted in screen.

1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511

The expected standard is to make the most important information visually dominant. In the current design, the icons are not visually prominent. To fix this, we can enlarge the icons

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small. To fix this, we can increase the text font size.

The expected standard is that the design should use clear and concise language. In the current design, some of the language used is jargonistic and could be difficult for users to understand. To fix this, the designer should use clear and concise language that is easy for the user to understand.

The expected standard is that the design should use a consistent visual hierarchy to distinguish between different levels of importance in the content. In the current design, the visual hierarchy is not clear, and it is difficult to distinguish between different levels of importance in the content. To fix this, use a consistent visual hierarchy to distinguish between different levels of importance in the content.

The expected standard is that the design should use as few elements as possible to achieve its goals; all non-essential elements should be omitted. In the current design, the top navigation bar has too many options, which might overwhelm the user. To fix this, try removing content that does not help convey the primary message

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size for "Per user / month, billed yearly" is small and difficult to read. To fix this, we can increase the text font size

The expected standard is that the design should use as few elements as possible to achieve its goals, and each visual element should contribute to the overall message. In the current design, there are too many elements on the page and it is difficult to focus on the most important information. To fix this, the design should be simplified and the number of elements should be reduced.

The expected standard is that the design should be consistent in its use of typography, color, and layout. In the current design, the use of typography, color, and layout is inconsistent. To fix this, the designer should create a style guide that defines the typography, color palette, and layout grid for the design.

The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text is too small and difficult to read. To fix this, increase the font size.

The expected standard is that the design should make the most important information visually dominant. In the current design, the button "Talk to Sales" is not visually prominent. To fix this, we can enlarge the button "Talk to Sales".

The expected standard is that the design should use a clear and easy-to-understand visual hierarchy to help users navigate the interface. In the current design, the visual hierarchy is unclear and confusing. To fix this, the designer should use a more distinct visual hierarchy to emphasize the most important elements of the interface. This can be done by using larger font sizes, bolder colors, or more white space.

The expected standard is that the design should be visually appealing and engaging. In the current design, the design is cluttered and overwhelming. To fix this, the designer should simplify the design and use more white space.

Figure 13: Example design feedback and bounding boxes generated by our pipeline for five modern websites (from 2024). These websites are out-of-domain inputs, as we used fewshot examples from only UICrit, which consists of older mobile UIs (from 2014) that differ significantly from modern websites. The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box. Helpful comments with reasonably accurate bounding boxes are highlighted in screen.

1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565

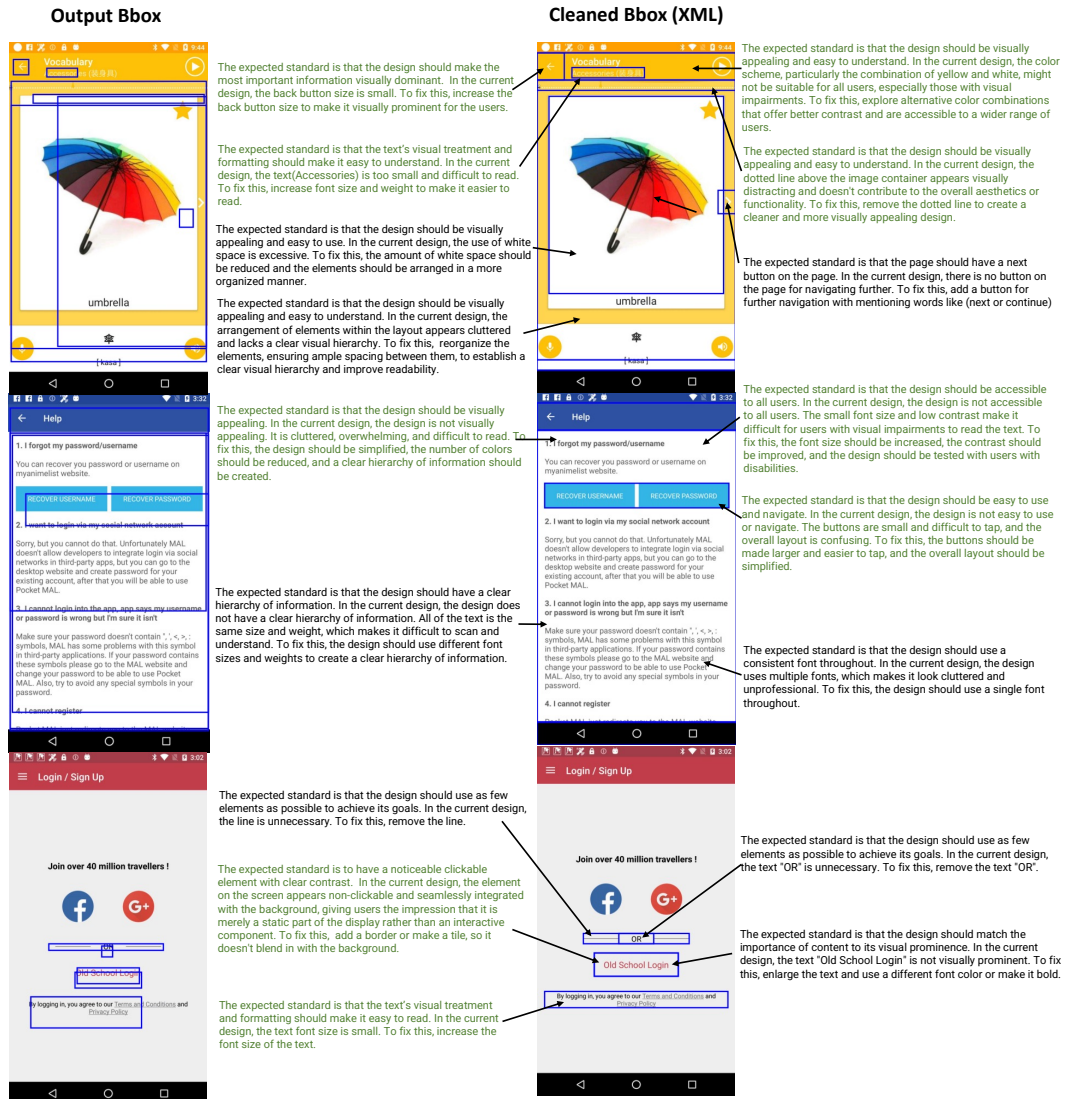


Figure 14: Side by side comparison of the bounding boxes generated by the pipeline (“Output Bbox”) and the output bounding boxes after refinement using a simple method that locates the nearest elements and groups from the DOM tree based on an IoU threshold (“Cleaned Bbox (XML)”). This refinement approach significantly improves the quality of the generated bounding boxes.

1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619

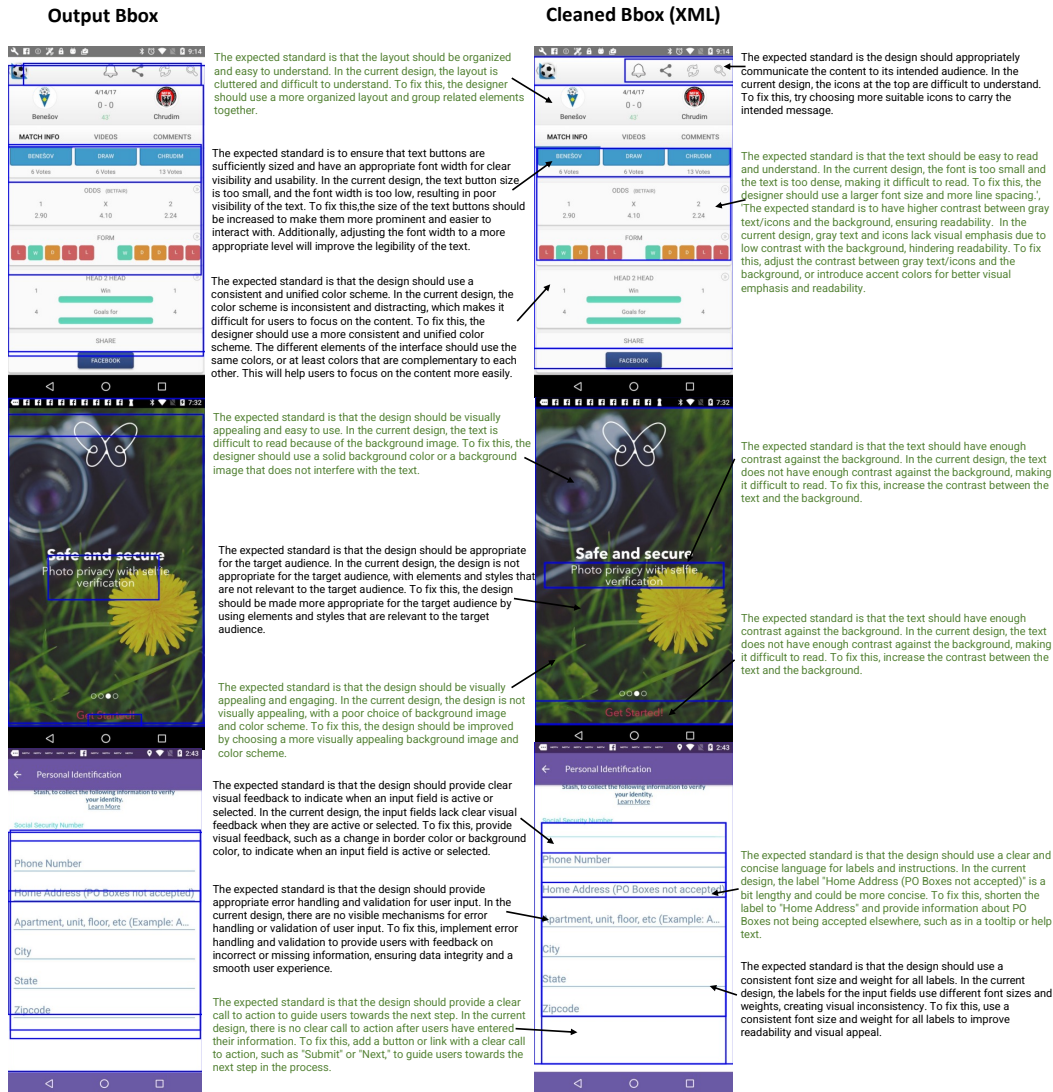
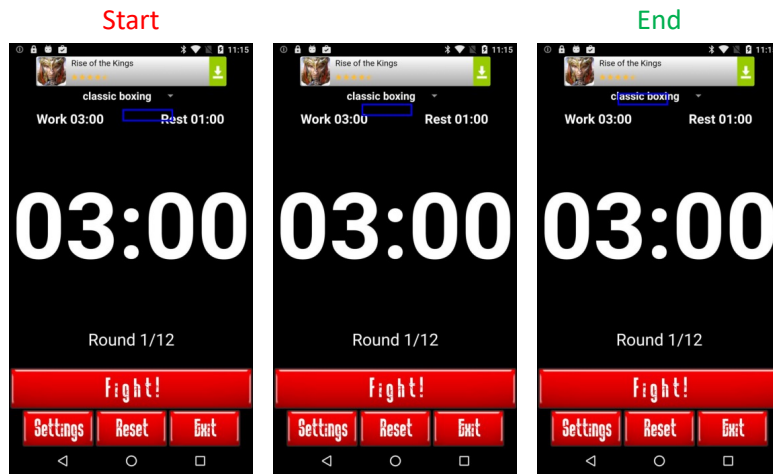


Figure 15: Side by side comparison of the bounding boxes generated by the pipeline (“Output Bbox”) and the output bounding boxes after refinement using a simple method that locates the nearest elements and groups from the DOM tree based on an IoU threshold (“Cleaned Bbox (XML)”). This refinement approach significantly improves the quality of the generated bounding boxes.

1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673



Comment: The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the text "classic boxing" is not visually prominent. To fix this, we can increase the font size of the text "classic boxing".

Figure 16: An example of iterative bounding box refinement, with the comment it is conditioned on displayed on the right. The bounding box in the first screenshot ('Start') is the output from BoxGen. The refinement process progressively improves the bounding box, terminating on a significantly more accurate bounding box ('End').

Start

The expected standard is that design should convey a clear message. In the current design, it does not provide enough information to the users to understand what the app itself is all about. To fix this, redesign it by adding additional information with features to communicate the content to its intended users.

End

The expected standard is that the design should be consistent throughout the app. In the current design, the "Ringtone" section and the "Message Notification Sounds" section are not consistent with each other. The "Ringtone" section has a dropdown menu, while the "Message Notification Sounds" section does not. To fix this, the designer should make the two sections consistent with each other. For example, both sections could have dropdown menus.

The expected standard is that the design should be easy to understand and use. In the current design, the layout of the notification settings is confusing and difficult to follow. To fix this, the designer should reorganize the layout to make it more intuitive.

Screenshot

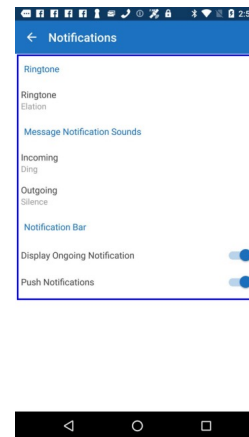


Figure 17: An example of iterative comment refinement, with the bounding box it is conditioned on displayed on the right. The first comment ('Start') was classified as incorrect by the Validation but has an accurate corresponding bounding box. The refinement process progressively improves the comment, terminating with an accurate comment on the poor layout of the region in the bounding box. ('End').

1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727

Comment Set Ranking

Set A (All Comments)

1. The expected standard is that the text's visual treatment and formatting should make it easy to read. In the current design, the text font size is small and the background makes the foreground text difficult to read. To fix this, we can increase the text font size and choose a different contrasting background.

Set B (All Comments)

1. The expected standard is to have high contrast and a visually appealing background that complements the design's overall aesthetic. In the current design, the black background lacks visual appeal. To fix this, consider a lighter background or a textured black option to improve contrast and visual interest.

Set C (All Comments)

1. The expected standard is that the design should use as few elements as possible to achieve its goals. Each visual element should contribute to the overall message; all non-essential elements should be omitted. In the current design, there are too many elements on the screen, making it difficult to focus on any one thing. To fix this, the designer should remove any unnecessary elements from the screen.

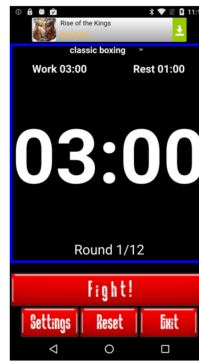
Please rank each set of comments, as a whole, based on their overall quality. Please rank them in decreasing quality.

	1	2	3
Set A	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Set B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Set C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Individual Comment Rating

Set C Comment (10 of 15)

The expected standard is that the design should be visually appealing and easy to use. In the current design, the text is not aligned properly. To fix this, the designer should align the text to the center.



- Invalid
- Partially Valid
- Valid

Figure 18: The form used for individual comment quality rating and comment set ranking.