

VISUAL PROMPTING WITH ITERATIVE REFINEMENT FOR DESIGN CRITIQUE GENERATION

Anonymous authors

Paper under double-blind review

ABSTRACT

Feedback is essential in all design processes, such as user interface (UI) design. Automating design critiques can significantly enhance design workflow efficiency. Although existing vision language models (VLMs) 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 a multimodal iterative refinement and visual prompting framework 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 VLMs, which iteratively refine both the text output and bounding boxes (in a mutually conditioned manner), 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 its generalizability 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., 2024b; 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, which may be expensive and not always readily available. This makes automated critique very valuable, as it can provide instant feedback for designers to quickly iterate on (Duan et al., 2024b). Furthermore, automated design feedback can serve as a reward function for automated UI generation, which has started to gain traction (Zhao et al., 2021; Gajjar et al., 2021; Duan et al., 2024c).

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 explain and contextualize the feedback in a way that human designers can understand and act upon (Duan et al., 2024a) (Figure 1).

VLMs 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 VLMs appear to be readily usable for design critique, a multimodal task, there remains a significant gap in quality between the feedback generated by these VLMs compared to that of human design experts Duan et al. (2024a). In addition, VLMs often struggle to achieve

054 accurate visual grounding (Duan et al., 2024a; Dorkenwald et al., 2024), making it difficult for them
055 to mark relevant regions in the UI screenshots, which is crucial for contextualizing feedback for
056 designers (Duan et al., 2024a).

057 Recent advances in prompting techniques have improved both visual grounding and text generation
058 performance. For example, Fang et al. (2024) introduced a visual prompting technique that adds
059 visual markers to an image, which helps VLMs better ground objects. Separately, a method called
060 *iterative refinement* Madaan et al. (2023); Xu et al. (2024a) has been proposed for text-only tasks,
061 where an LLM’s output is repeatedly refined by itself or another model until the output is deemed
062 correct. *Iterative refinement* has been shown to improve performance for text-only tasks like code
063 optimization and machine translation. Building on these, we propose a novel multimodal framework
064 that combines both iterative refinement and visual prompting to generate UI design critiques (Figure
065 1). Our approach extends iterative refinement to multimodal tasks by jointly refining two coupled
066 outputs: (1) the design critique text and (2) corresponding bounding boxes that ground the critique in
067 the UI. Each refinement is conditioned on the other, i.e., text refinement is based on the grounding,
068 and grounding refinement leverages the current text. This creates a feedback loop that promotes
069 semantic and spatial alignment. To further improve grounding, we introduce visual prompting at
070 several stages of the pipeline, including the iterative refinement steps. This enhances bounding box
071 accuracy and, in turn, improves text accuracy. Our approach is implemented through an architecture
072 that coordinates multiple VLMs (Figure 2) and incorporates both novel prompting techniques and
073 established practices (Chen et al., 2024).

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

082 2 RELATED WORK

083 2.1 AUTOMATED UI DESIGN CRITIQUE WITH LLMs AND VLMs

084 Prior work has studied the capabilities of LLMs/VLMs for UI design critique. Duan et al. (2024b)
085 explored the performance of zero-shot (text-only) GPT-4 in critiquing UI mockups, using a JSON
086 representation of the UI. They identified gaps between the feedback capabilities of general-purpose
087 LLMs and human experts. To address this, they collected a dataset (UICrit) (Duan et al., 2024a) con-
088 sisting of human-annotated design critiques (grounded within UI screenshots via bounding boxes)
089 for UI screens that could be applied to train general-purpose VLMs. They also built a UI design
090 critique model that takes in a UI screenshot and outputs critiques grounded in screenshot regions.
091 Their method showed a significant improvement in VLM-generated feedback with just few-shot
092 sampling from UICrit, although the feedback quality still falls short of human experts. Similarly,
093 Wu et al. (2024) generated a synthetic dataset of UI design comments and trained a CLIP model
094 (Radford et al., 2021) to assess UI designs. We apply our approach to the design critique task, which
095 augments the method from Duan et al. (2024a) by incorporating mutually-conditioned iterative re-
096 finement of design comments and their corresponding bounding box positions on the UI screen.

097 2.2 PROMPTING LLMs WITH ITERATIVE REFINEMENT

098 Iterative refinement on LLM output has been explored in prior studies to improve LLM performance
099 for text-only tasks. Madaan et al. (2023) developed an approach called “SELF-REFINE”, where a
100 single LLM generates an initial output and then iteratively provides feedback on its own output and
101 revises the output based on the feedback. They applied this technique across a diverse set of tasks,
102 such as math reasoning and dialogue response, and found that SELF-REFINE resulted in a 20% av-
103 erage performance gain. Similarly, Zhou et al. (2023) utilized this iterative self-refinement technique
104 on long-horizon sequential task planning in robotics, leading to higher success rates. However, Xu
105 et al. (2024b) found that LLMs often exhibit *self-bias* (i.e., a tendency to favor its own generated
106 output).

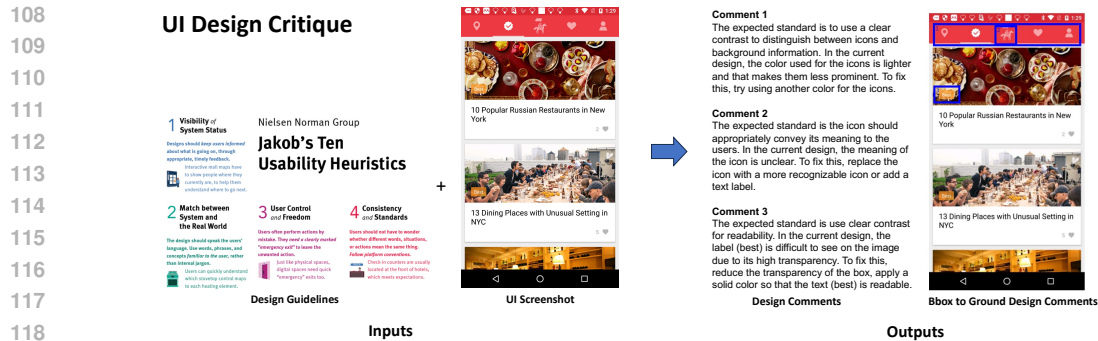


Figure 1: Illustration of the UI Design Critique Task, which takes in a UI screenshot and a set of design guidelines and outputs a list of design comments with corresponding bounding boxes (Bbox).

output) during self-refinement across a variety of tasks and languages. To account for this, they developed “LLMRefine” (Xu et al., 2024a), 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 conditionally refining both text and bounding boxes that associate the text with relevant regions in the image. Following the method in LLMRefine, we use separate VLMs for generation and refinement to prevent self-bias.

2.3 MULTIMODAL TASKS

Previous work has investigated a variety of grounded multimodal tasks using VLMs, where a VLM 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. Similarly, Bravo et al. (2023) introduced the Open Vocabulary Object and Attribute Detection task, which identifies and grounds both objects and their corresponding attributes in an image in an open vocabulary setting. In robotics, VLMs were used to help systems understand the physical world. Fang et al. (2024) introduced MOKA, which utilizes VLMs 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) evaluated VLMs in the LLM-as-a-judge paradigm (Gu et al., 2025) across three tasks: pairwise comparison, scoring, and ranking. They found that while VLMs excelled at pairwise comparison, they struggled with the other tasks. However, Liu et al. (2024) showed that providing few-shot examples can improve an LLM’s evaluation performance. Visual grounding is a vital component of our method, and we utilize visual prompting to enhance bounding box generation and refinement. We also utilize VLMs to validate output accuracy in our pipeline and use few-shot examples to enhance their evaluation accuracy.

3 TASK

UI design critique generation was introduced as a grounded multimodal task by Duan et al. (2024a). The model takes in a UI screenshot and a set of design guidelines and outputs a list of design critiques. Each design critique consists of 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 Figure 1.

As discussed earlier, this task is challenging because the model must understand and apply UI design principles to identify design issues in the screenshot. Furthermore, finding the exact region of the screen for a comment (i.e., the bounding box) is not always straightforward. For example, a com-

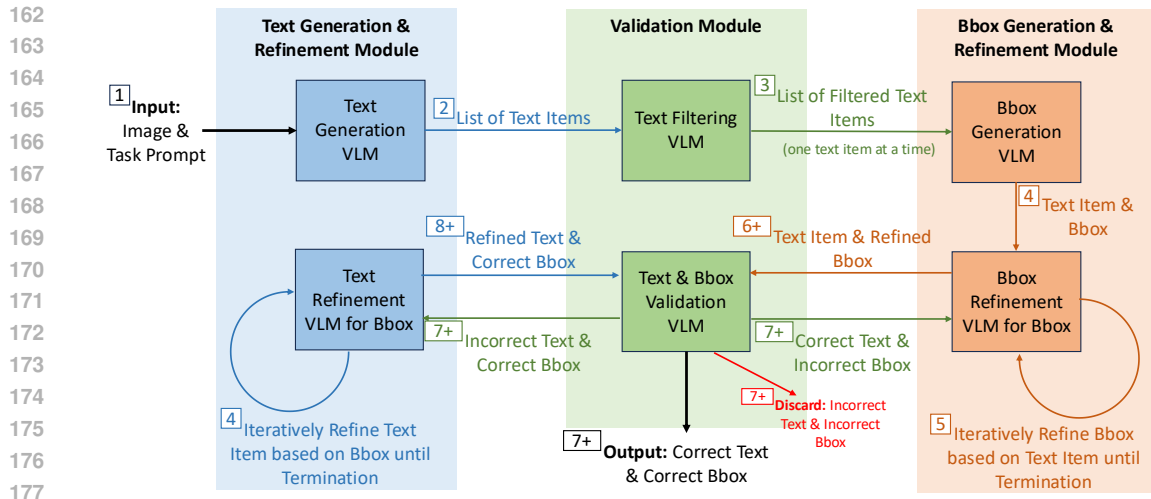


Figure 2: The figure illustrates our prompting pipeline, which takes an image and a task prompt as input and outputs text items with their corresponding bounding boxes on the image. The main inputs and outputs for each VLM are shown, and Section 4 details all the inputs, outputs, and few-shot examples for each VLM. Each input/output is numbered with their order of generation, and numbers with a '+' indicate multiple iterations of input/output.

ment might state that the text in the UI has poor contrast with the background, but not specify which text element is problematic, requiring the model to both identify the problematic elements and also determine their bounding box. While we focus 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 VLMs to generate UI design critiques. It consists of six distinct VLMs, 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 pipeline, showing the main inputs and outputs of each VLM, 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, each step’s VLM output 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 a Validation module (i.e., an LLM-as-a-judge) between the Text and Bounding Box modules to assess each module’s output, ensuring each refinement step is based on more accurate inputs. Additionally, each VLM 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 instruction prompts for relevant steps. The specific inputs, outputs, and few-shot examples for each VLM are detailed in the following sections, and the instructions prompt for each step can be found in Appendix A.4. Appendix A.6 presents a cost analysis of the pipeline and visualizations of bounding box and comment refinements.

Text Generation VLM (TextGen) The pipeline begins with the TextGen VLM 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, VLMs 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 VLM (TextFilter) To reduce the chance of bounding box generation being conditioned on incorrect text items (i.e., incorrect design comments), we utilize a separate VLM to assess each design comment’s accuracy, operating in an LLM-as-a-judge role. This TextFilter VLM takes a list of generated text items from TextGen and the image as input, and outputs a filtered list of text items it considers valid. Based on the findings of Liu et al. (2024), we incorporate few-shot examples for TextFilter. These examples are constructed by injecting invalid items into a correct list of text items, and using this augmented list as input and the original correct list as the expected output. These examples illustrate how to filter out invalid items.

Bounding Box Generation VLM (BoxGen) The BoxGen VLM generates bounding boxes based on the filtered text items from TextFilter. The VLM takes in one text item at a time, as well as the image, and predicts a relevant region on the image via bounding box coordinates. Following the visual prompting technique from Duan et al. (2024a), we augment the screenshot by adding coordinate markers along its edges (Figure 4) to help the VLM associate coordinates with specific regions in the screen.

Bounding Box Refinement VLM (BoxRefine) To avoid self-bias during iterative refinement (Xu et al., 2024b), we use a separate VLM to iteratively refine the bounding box from the previous step, conditioned on the filtered text item. The BoxRefine VLM takes in several inputs, as shown in Figure 4 (Appendix). 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 were 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 VLM assesses the quality of the current bounding box based on all these inputs. If the bounding box is deemed accurate by the BoxRefine VLM, the iterative refinement process terminates. Otherwise, the VLM returns the refined coordinates, which are then re-evaluated by the VLM. This process is repeated until the VLM either confirms the bounding box as correct or the maximum number of iterations is reached. Prior work (Madaan et al., 2023) has shown that the history of refinements provides helpful information. Thus, we include the history of the VLM’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 (Appendix) details our methods for bounding box perturbation and the generation of few-shot examples for bounding box refinement.

Text & Bounding Box Validation VLM (Validation) After determining the bounding box for the text item, we again apply the LLM-as-a-judge framework through the Validation VLM, which 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 VLM 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 their accuracy. Figure 2 illustrates how the output is handled for each case: If both the text and bounding box are deemed correct, the pair is returned. If either the text or bounding box is incorrect, the pair is sent for further refinement. If both are incorrect, the pair is discarded, and the pipeline moves on to the next filtered text item. Few-shot examples are generated differently to handle 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 case where both are incorrect.

Text Refinement VLM (TextRefine) The TextRefine VLM is used to refine incorrect text items, conditioned on bounding boxes that correctly identify relevant regions in the image, as determined by the Validation VLM. This iterative refinement process mirrors the bounding box refinement procedure. The VLM 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 the correct text, which simulates the refinement process. The refined text item and bounding box are then returned to the Validation VLM.

5 EXPERIMENTS

5.1 DATASET AND BASELINE

We used the UICrit dataset¹ (Duan et al., 2024a) 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 the screen’s purpose. 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. (2024a) in the task prompt, and evaluated the model’s output against the comments and bounding boxes for the screenshot 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 evaluation. The few-shot sampling method used at each step is detailed in A.3.1.

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

5.2 IMPACT OF VISUAL PROMPTING AND 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 VLMs: 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 VLMs (BoxGen and BoxRefine), and BoxRefine was only used for the conditions with iterative refinement.

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., 2024a). 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 VLMs, indicating that iterative visual prompting significantly enhances bounding box estimation.

5.3 PIPELINE ABLATION AND QUALITATIVE ANALYSIS

Table 2 presents the results of the ablation study for UI design critique for both VLMs, 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 (Reimers & Gurevych, 2019) 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 pipeline step incrementally improved comment similarity and estimated IoU for both VLMs. While GPT-4o and Gemini-1.5-pro showed similar comment similarity, GPT-4o achieved a higher estimated average IoU, likely due to its nearly three-fold larger parameter count. The complete

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

Table 2: Ablation study of our UI design critique pipeline. IR stands for Iterative Refinement. We combine the Validation step’s results with subsequent refinements for bounding boxes and text, as these refinements apply to a much smaller subset of pairs flagged as having incorrect bounding boxes or text during Validation. We also include results from the Baseline 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., 2024a)	0.651	0.680	0.176	0.257
Finetuned Llama-3.2 11b	0.842		0.230	

pipeline also outperforms the baseline in both comment similarity and estimated IoU. 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.

While finetuned Llama-3.2 had higher comment similarity, it generated a very limited set of critiques, whereas our pipeline generated a considerably more diverse set of comments. To quantify diversity, we embedded design comments using sentenceBERT (Reimers & Gurevych, 2019) and computed the average distance ($1 - \text{cosine similarity}$) from each comment to the centroid, following the metric by Cox et al. (2021). The pipeline’s diversity score (0.670) was considerably higher than finetuned Llama’s (0.302). Example outputs and detailed qualitative analyses are provided in Appendix A.5.1. Appendix A.5.3 discusses simple ways to further refine the pipeline’s output, such as using a third-party UI element detector (Xie et al., 2020) to clean up the bounding boxes.

5.4 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. (2024a), 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. (2024a), 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 ensures a more rigorous and standardized approach to evaluate bounding box accuracy and a more focused evaluation on comment quality. See Appendix A.7 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 the 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.,

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., 2024a)	0.45	2.3	0.423
Our Pipeline	0.47	2.0	0.451
Human	0.56	1.7	N/A

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 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

1971) among the participants was 0.22 for comment quality and 0.29 for comment set ranking, indicating fair agreement.

Across all 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 variability in dataset quality, potentially due to UICrit’s annotators’ limited design experience (Duan et al., 2024a). 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, closing the gap from human performance by 50%. 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 tasks that output visually grounded text. To assess if its performance enhancement generalizes to other tasks, we evaluate our pipeline on an existing vision-language benchmark: Open Vocabulary Object and Attribute Detection Bravo et al. (2023).

6.1 OPEN VOCABULARY OBJECT & ATTRIBUTE DETECTION

Open vocabulary object and attribute detection 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 (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.3.2 describes the fewshot sampling strategy in more detail.

²<https://ovad-benchmark.github.io/>

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 5.1, as it can also be applied to this task. We followed the evaluation method from 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 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 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 potentially be applied to enhance other grounded multimodal 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 experts. Despite its performance gap with human critique (which is expensive to acquire), our pipeline’s generalizability and consistent improvements over baselines, achieved solely through prompting without parameter tuning nor third-party models, demonstrate its potential as a versatile and resource-efficient solution for enhancing VLM performance across various tasks and domains.

A reason for the performance gap could be that the VLM-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. This 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 through visual prompting and conditional iterative refinement of both text and bounding boxes. Our approach outperformed the baseline in human evaluations, generating higher quality comments with more accurate grounding. We also 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. Despite its limitations, our method offers a versatile and resource-efficient solution for improving VLM performance across various tasks and domains.

REFERENCES

- 486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
- María A. Bravo, Sudhanshu Mittal, Simon Ging, and Thomas Brox. Open-vocabulary attribute detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7041–7050, June 2023.
- Dongping Chen, Ruoxi Chen, Shilin Zhang, Yinuo Liu, Yaochen Wang, Huichi Zhou, Qihui Zhang, Pan Zhou, Yao Wan, and Lichao Sun. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark, 2024.
- Samuel Rhys Cox, Yunlong Wang, Ashraf Abdul, Christian von der Weth, and Brian Y. Lim. Directed diversity: Leveraging language embedding distances for collective creativity in crowd ideation. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445782. URL <https://doi.org/10.1145/3411764.3445782>.
- Michael Dorckenwald, Nimrod Barazani, Cees G. M. Snoek, and Yuki M. Asano. Pin: Positional insert unlocks object localisation abilities in vlms, 2024. URL <https://arxiv.org/abs/2402.08657>.
- Peitong Duan, Chin-Yi Cheng, Gang Li, Bjoern Hartmann, and Yang Li. Uicrit: Enhancing automated design evaluation with a ui critique dataset. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, UIST '24, New York, NY, USA, 2024a. Association for Computing Machinery. ISBN 9798400706288. doi: 10.1145/3654777.3676381. URL <https://doi.org/10.1145/3654777.3676381>.
- Peitong Duan, Jeremy Warner, Yang Li, and Bjoern Hartmann. Generating automatic feedback on ui mockups with large language models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, New York, NY, USA, 2024b. Association for Computing Machinery. ISBN 9798400703300. doi: 10.1145/3613904.3642782. URL <https://doi.org/10.1145/3613904.3642782>.
- Shiyu Duan, Runsheng Zhang, Mengmeng Chen, Ziyi Wang, and Shixiao Wang. Efficient and aesthetic ui design with a deep learning-based interface generation tree algorithm. *ArXiv*, abs/2410.17586, 2024c. URL <https://api.semanticscholar.org/CorpusID:273532777>.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, and et. al. The llama 3 herd of models, 2024. URL <https://arxiv.org/abs/2407.21783>.
- Kuan Fang, Fangchen Liu, Pieter Abbeel, and Sergey Levine. MOKA: Open-World Robotic Manipulation through Mark-Based Visual Prompting. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.062.
- J.L. Fleiss et al. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.
- Nishit Gajjar, Vinoth Pandian Sermuga Pandian, Sarah Suleri, and Matthias Jarke. Akin: Generating ui wireframes from ui design patterns using deep learning. In *Companion Proceedings of the 26th International Conference on Intelligent User Interfaces*, IUI '21 Companion, pp. 40–42, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380188. doi: 10.1145/3397482.3450727. URL <https://doi.org/10.1145/3397482.3450727>.
- Foziah Gazzawe. Comparison of websites and mobile applications for e-learning. 07 2017.
- Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A survey on llm-as-a-judge, 2025. URL <https://arxiv.org/abs/2411.15594>.

- 540 Jan Hartmann, Alistair Sutcliffe, and Antonella De Angeli. Towards a theory of user judgment of
541 aesthetics and user interface quality. *ACM Trans. Comput.-Hum. Interact.*, 15, 11 2008. doi:
542 10.1145/1460355.1460357.
- 543
544 Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish
545 Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *The*
546 *Eleventh International Conference on Learning Representations, 2023*.
- 547
548 Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr
549 Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet,
550 Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision – ECCV 2014*, pp.
551 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1.
- 552
553 Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei
554 Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for
555 open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.
- 556
557 Yuxuan Liu, Tianchi Yang, Shaohan Huang, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng,
558 Feng Sun, and Qi Zhang. Calibrating LLM-based evaluator. In Nicoletta Calzolari, Min-Yen
559 Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings*
560 *of the 2024 Joint International Conference on Computational Linguistics, Language Resources*
561 *and Evaluation (LREC-COLING 2024)*, pp. 2638–2656, Torino, Italia, May 2024. ELRA and
562 ICCL.
- 563
564 Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri
565 Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad
566 Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-
567 refine: Iterative refinement with self-feedback. In A. Oh, T. Naumann, A. Globerson, K. Saenko,
568 M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36,
569 pp. 46534–46594. Curran Associates, Inc., 2023.
- 570
571 Jakob Nielsen and Rolf Molich. Heuristic evaluation of user interfaces. In *Proceedings of the*
572 *SIGCHI Conference on Human Factors in Computing Systems, CHI '90*, pp. 249–256, New York,
573 NY, USA, 1990. Association for Computing Machinery. ISBN 0201509326. doi: 10.1145/97243.
574 97281.
- 575
576 OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, and Lama Ahmad et. al. Gpt-4 technical
577 report, 2024. URL <https://arxiv.org/abs/2303.08774>.
- 578
579 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agar-
580 wal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya
581 Sutskever. Learning transferable visual models from natural language supervision. In Marina
582 Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine*
583 *Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR,
584 18–24 Jul 2021.
- 585
586 Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-
587 networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*
588 *Processing*. Association for Computational Linguistics, 11 2019. URL <https://arxiv.org/abs/1908.10084>.
- 589
590 Shreya Shankar, J. D. Zamfirescu-Pereira, Björn Hartmann, Aditya G. Parameswaran, and Ian
591 Arawjo. Who validates the validators? aligning llm-assisted evaluation of llm outputs with human
592 preferences, 2024.
- 593
594 Debbie Stone, Caroline Jarrett, Mark Woodroffe, and Shailey Minocha. *User interface design and*
595 *evaluation*. Elsevier, 2005.
- 596
597 Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, and et. al. Gemini 1.5:
598 Unlocking multimodal understanding across millions of tokens of context, 2024.

- 594 Chunxiao Wang, Shuai Lu, Hongzhong Chen, Ziwei Li, and Borong Lin. Effectiveness of one-click
595 feedback of building energy efficiency in supporting early-stage architecture design: An exper-
596 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/science/article/pii/S0360132321001876>.
599
- 600 Jason Wu, Xiaoyi Zhang, Jeff Nichols, and Jeffrey P Bigham. Screen parsing: Towards reverse
601 engineering of ui models from screenshots. In *The 34th Annual ACM Symposium on User In-*
602 *terface Software and Technology*, UIST '21, pp. 470–483, New York, NY, USA, 2021. Association
603 for Computing Machinery. ISBN 9781450386357. doi: 10.1145/3472749.3474763. URL
604 <https://doi.org/10.1145/3472749.3474763>.
605
- 606 Jason Wu, Yi-Hao Peng, Xin Yue Amanda Li, Amanda Swearngin, Jeffrey P Bigham, and Jeffrey
607 Nichols. Uiclip: A data-driven model for assessing user interface design. UIST '24, New York,
608 NY, USA, 2024. Association for Computing Machinery. ISBN 9798400706288. doi: 10.1145/
609 3654777.3676408. URL <https://doi.org/10.1145/3654777.3676408>.
610
- 611 Mulong Xie, Sidong Feng, Zhenchang Xing, Jieshan Chen, and Chunyang Chen. Uied: a hy-
612 brid tool for gui element detection. In *Proceedings of the 28th ACM Joint Meeting on Euro-*
613 *pean Software Engineering Conference and Symposium on the Foundations of Software Engi-*
614 *neering*, ESEC/FSE 2020, pp. 1655–1659, New York, NY, USA, 2020. Association for Com-
615 puting Machinery. ISBN 9781450370431. doi: 10.1145/3368089.3417940. URL <https://doi.org/10.1145/3368089.3417940>.
616
- 617 Anbang Xu, Shih-Wen Huang, and Brian Bailey. Voyant: generating structured feedback on vi-
618 sual designs using a crowd of non-experts. In *Proceedings of the 17th ACM Conference on*
619 *Computer Supported Cooperative Work & Social Computing*, CSCW '14, pp. 1433–1444, New
620 York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450325400. doi:
621 10.1145/2531602.2531604. URL <https://doi.org/10.1145/2531602.2531604>.
622
- 623 Wenda Xu, Daniel Deutsch, Mara Finkelstein, Juraj Juraska, Biao Zhang, Zhongtao Liu,
624 William Yang Wang, Lei Li, and Markus Freitag. LLMRefine: Pinpointing and refining large
625 language models via fine-grained actionable feedback. In Kevin Duh, Helena Gomez, and
626 Steven Bethard (eds.), *Findings of the Association for Computational Linguistics: NAACL*
627 *2024*, Mexico City, Mexico, June 2024a. Association for Computational Linguistics. doi:
628 10.18653/v1/2024.findings-naacl.92.
629
- 630 Wenda Xu, Guanglei Zhu, Xuandong Zhao, Liangming Pan, Lei Li, and William Wang. Pride and
631 prejudice: LLM amplifies self-bias in self-refinement. In Lun-Wei Ku, Andre Martins, and Vivek
632 Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational*
633 *Linguistics (Volume 1: Long Papers)*, pp. 15474–15492, Bangkok, Thailand, August 2024b. As-
634 sociation for Computational Linguistics.
635
- 636 Haoyu Zhao, Wenhong Ge, and Yingcong Chen. Llm-optic: Unveiling the capabilities of large
637 language models for universal visual grounding, 2024. URL <https://arxiv.org/abs/2405.17104>.
638
- 639 Tianming Zhao, Chunyang Chen, Yuanning Liu, and Xiaodong Zhu. guigan: Learning to gen-
640 erate gui designs using generative adversarial networks. In *Proceedings of the 43rd Internat-*
641 *ional Conference on Software Engineering*, ICSE '21, pp. 748–760. IEEE Press, 2021. ISBN
642 9781450390859. doi: 10.1109/ICSE43902.2021.00074. URL <https://doi.org/10.1109/ICSE43902.2021.00074>.
643
644
- 645 Zhehua Zhou, Jiayang Song, Kunpeng Yao, Zhan Shu, and Lei Ma. Isr-llm: Iterative self-refined
646 large language model for long-horizon sequential task planning. *2024 IEEE International Con-*
647 *ference on Robotics and Automation (ICRA)*, pp. 2081–2088, 2023.

702 ment, multiple invalid comments were selected, following the process described earlier, and then
 703 sorted by increasing cosine similarity to simulate the comment refinement process.

704
 705 For bounding box refinement, we considered another technique to generate fewshot examples. This
 706 technique involves selecting the first bounding box location based on visual similarity of the region it
 707 contains in the fewshot UI to that of the region contained by the input bounding box proposal of the
 708 input screenshot. This bounding box is then gradually moved closer to the ground truth bounding
 709 box for the fewshot UI to simulate the refinement process. However, we found that the simpler
 710 approach of randomly perturbing the bounding box actually gave better results (IoU 0.357 (random
 711 perturbation, from Table 1) vs 0.333 (visual similarity match)).

712 A.3.2 OPEN VOCABULARY OBJECT AND ATTRIBUTE DETECTION

713
 714 For text generation (i.e., category and attributes) and filtering, we sampled images based on the
 715 semantic similarity of their CLIP embeddings. Negative text samples for the filtering step were
 716 generated by sampling irrelevant text from other images. For bounding box generation, refinement,
 717 and subsequent steps applied to individual text items, we sampled few-shot examples by selecting
 718 the most semantically similar text items and their corresponding bounding boxes from a split of their
 719 annotated dataset. We used sentenceBERT (Reimers & Gurevych, 2019) to embed the text items for
 720 similarity ranking. For validation, invalid text examples were perturbed by randomly swapping the
 721 category or attributes, or by deleting or adding attributes. Similarly, for text refinement, few-shot
 722 examples were generated by perturbing the text in decreasing amounts.

723 A.4 INSTRUCTIONS PROMPTS FOR PIPELINE

724
 725 We provide the instructions prompt for each step of the pipeline for the UI Critique Task.

726
 727 Text Generation: For these sets of guidelines: [Guidelines]. Please find
 728 all the guideline violations in the UI provided. For violation found,
 729 please provide an explanation that includes these three things: 1.
 730 the expected standard (i.e. what good design should look like), 2.
 731 the gap between the current design and the expected standard (i.e.
 732 the critique for the design), and 3. how to fix the issue in the
 733 current design. For formatting each violation, please include these
 734 three things in separate sentences. For the expected standard (#1),
 735 start the sentence with 'The expected standard is that...'. For the
 736 gap (#2), start the sentence with 'In the current design, ...', and
 737 for how to fix the design (#3), start the sentence with 'To fix this
 738 ...'. Please end each violation explanation with two newline
 739 characters (\n\n). Please be specific in your violation explanations,
 740 making sure to refer to specific UI elements and groups in the UI.
 741 After determining all guideline violations, please also share any
 742 other design feedback you have for the UI and follow the same format
 743 of providing the expected standard, the critique for the design, and
 744 how to fix the issue. We will provide N examples of a UI screenshot
 745 and a set of valid design comments. Please learn how to give valid
 746 design comments from these examples and apply this knowledge to
 747 determine valid design comments for the last UI. Please be specific
 748 in your comments, referring to specific UI elements by their text
 749 label or icon, like in the examples provided. Also, please do not
 750 return any comments regarding user testing nor adherence to platform
 751 standards.

752
 753 Text Filtering: For the provided UI and a list corresponding design
 754 comments, please filter out the incorrect design comments and return
 755 a list tuples. Each tuple contains its index i in the list, followed
 by True or False. The tuple would contain True if the design comment
 at index i in the input list is a valid design comment, and False if
 the design comment at index i is an invalid comment. Please analyze
 the UI screenshot to determine whether or not each design comment is
 valid. We will give N examples, where each UI screenshot is followed
 by a list of its corresponding design comments and an output list of
 tuples, where each tuple contains the list index and True/False

756 indicating the validity of the design comment at that index. Please
757 learn from these examples, analyzing the UI screenshot to see why
758 each comment was considered valid or invalid. Finally, we will give a
759 UI screenshot, followed by its corresponding design comments. Please
760 output a list of tuples consisting of the comment's list index and
761 an indication of each comment's validity, like in the provided
762 examples. Please output False for the design comment if it is about
763 consistency with the brand, user testing, or adherence to platform
764 standards. Please only output this list of tuples and nothing else.

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

784 **Bounding Box Refinement:** You will be refining bounding boxes for a given
785 UI screenshot and design comment. The bounding box will enclose a
786 relevant region in the screenshot that is discussed in the design
787 comment. You will be given a proposed bounding box candidate and will
788 evaluate whether or not this bounding box accurately encloses the
789 region in the screenshot that is discussed in the comment. The
790 proposed bounding box coordinates, in the format of (left_coordinate,
791 top_coordinate, right_coordinate, bottom_coordinate) and is
792 displayed as a blue box in the screenshot patch that is also provided
793 , with some additional margin around the blue bounding box. Please
794 reflect on whether or not this bounding box is accurate and look
795 closely at the UI elements contained in the blue bounding box to
796 judge its accuracy and relevance to the design comment. If the
797 bounding box is not accurate, please output a new bounding box that
798 you think is accurate in the format of (left_coordinate,
799 top_coordinate, right_coordinate, bottom_coordinate), where each
800 coordinate is determined from the coordinate axes along the edge of
801 the UI screenshot provided earlier. Please make sure the new bounding
802 box you output is accurate, and refer to the coordinate axes along
803 the edge of the zoomed-in screenshot patch and the entire screenshot
804 (provided earlier) to determine the bounding box coordinates. If the
805 bounding box is accurate, please output 'BOUNDING BOX IS ACCURATE,
806 PLEASE TERMINATE'. Please only output either the updated bounding or
807 'BOUNDING BOX IS ACCURATE, PLEASE TERMINATE' and nothing else. We
808 will provide N examples of bounding box refinements for a given
809 design comment, UI screenshot, and bounding box candidate. Please
learn how to accurately refine bounding boxes for the design comment
in the UI screenshot based on these examples. We will provide a final
design comment, UI screenshot, and bounding box candidate; please
apply what you have learned from the examples to accurately refine
the bounding box candidate for this final design comment, UI
screenshot, and the zoomed in patch showing the bounding box
candidate.

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

840 Text Refinement: You will be refining the design comment for a specific
841 region in a UI screenshot. You will be given a UI screenshot, a
842 zoomed-in patch of the screenshot with a blue box containing the
843 region of interest, and a design comment for the UI region inside the
844 blue box. Please evaluate whether or not the design comment
845 accurately describes the design issue for the UI region inside the
846 blue box. If the design comment is accurate, please output 'COMMENT
847 IS ACCURATE, PLEASE TERMINATE'. If the design comment is not accurate
848 , please refine the design comment to the accurate and output this
849 accurate design comment for the region of interest, following the
850 same format as the input design comment. We will provide N examples
851 of bounding box refinements for each UI screenshot, region of
852 interest, and design comment candidate for the region of interest.
853 Please learn how to accurately refine the design comment for the
854 region of interest in the UI screenshot based on these examples. We
855 will provide a final UI screenshot, region of interest, and design
856 comment candidate for the region of interest; please apply what you
857 have learned from the examples to accurately refine design comment
858 candidate for this final UI screenshot and region of interest. Please
859 only output the refined comment or 'COMMENT IS ACCURATE, PLEASE
860 TERMINATE' and nothing else.

858
859
860
861
862
863

864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917

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 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 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 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 and group related elements together.

The expected standard is that the design should use a consistent and unified color scheme. In the current design, the color scheme is inconsistent and distracting, which makes it difficult for users to focus on the content. To fix this, the designer should use a more consistent and unified color scheme. The different elements of the interface should use the same colors, or at least colors that are complementary to each other. This will help users to focus on the content more easily.

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 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 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. 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.

A.5 QUALITATIVE ANALYSIS

A.5.1 QUALITATIVE ANALYSIS OF OUTPUTS FROM PIPELINE, BASELINE, AND FINETUNED VLM

We qualitatively analyzed the outputs from our pipeline, baseline, and finetuned Llama-3.2 11b. Figures 5, 6, and 7 illustrate the design feedback and corresponding bounding boxes generated by our

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 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 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.

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.

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.

pipeline (using Gemini-1.5-pro) for a diverse set of 12 UIs. Figure 8 presents two examples where our pipeline outperformed the baseline, and Figure 9 contains two examples where the baseline performed better. To enable easier comparison between the two conditions, we used the same set

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 be visually appealing and easy to use. In the current design, the text is difficult to read because of the background image. To fix this, the designer should use a solid background color or a background image that does not interfere with the text.

The expected standard is that the design should be appropriate for the target audience. In the current design, the design is not appropriate for the target audience, with elements and styles that are not relevant to the target audience. To fix this, the design should be made more appropriate for the target audience by using elements and styles that are relevant to the target audience.

The expected standard is that the design should be visually appealing and engaging. In the current design, the design is not visually appealing, with a poor choice of background image and color scheme. To fix this, the design should be improved by choosing a more visually appealing background image and color scheme.

The expected standard is that the design should be accessible to all users. In the current design, the design is not accessible to users with visual impairments. The contrast between the text and the background is not sufficient for users with low vision. To fix this, the designer should increase the contrast between the text and the background. The designer could also use a larger font size for the text.

The expected standard is that the design should use color and contrast to create a visual hierarchy and guide the user's eye to the most important elements. In the current design, the color contrast is minimal, making it difficult for users to distinguish between different elements and understand the hierarchy of information. To fix this, the designer should use a wider range of colors and contrasts to create a more visually appealing and easy-to-understand interface.

The expected standard is that the design should match the importance of content to its visual prominence. In the current design, the "One Sms" title is not much larger than the other elements on the page, even though it is more important than the other elements. To fix this, the "One Sms" title should be increased in size so that it is more prominent.

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, text (To the world you may be just one person...) is in white color on pink background which are difficult to read. To fix this, change colors to be more complementary to each other (text in dark colors) to make it easier to read.

The expected standard is that the design should provide clear visual feedback to indicate when an input field is active or selected. In the current design, the input fields lack clear visual feedback when they are active or selected. To fix this, provide visual feedback, such as a change in border color or background color, to indicate when an input field is active or selected.

The expected standard is that the design should provide appropriate error handling and validation for user input. In the current design, there are no visible mechanisms for error handling or validation of user input. To fix this, implement error handling and validation to provide users with feedback on incorrect or missing information, ensuring data integrity and a smooth user experience.

The expected standard is that the design should provide a clear call to action to guide users towards the next step. In the current design, there is no clear call to action after users have entered their information. To fix this, add a button or link with a clear call to action, such as "Submit" or "Next," to guide users towards the next step in the process.

The expected standard is that the text should have enough contrast against the background. In the current design, the text does not have enough contrast against the background, making it difficult to read. To fix this, increase the contrast between the text and the background.

The expected standard is that the text should have enough contrast against the background. In the current design, the text does not have enough contrast against the background, making it difficult to read. To fix this, increase the contrast between the text and the background.

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 at the top that are not necessary and do not contribute to the overall message of the design. To fix this, remove the icons at the top.

The expected standard is that the design should be visually appealing and engaging. In the current design, the design is bland and uninspired. The use of a single color for the background and the lack of any imagery makes the design feel flat and uninteresting. To fix this, the designer could add a gradient to the background or use an image that is relevant to the app's purpose.

The expected standard is that the design should be easy to use and understand. In the current design, it is not clear what the user is supposed to do next. The "TAP TO BEGIN" text at the bottom of the screen is small and easy to miss. To fix this, the designer could make the "TAP TO BEGIN" text larger and more prominent. The designer could also add an animation to the check mark to draw the user's attention to it.

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 and group related elements together.

The expected standard is that the design should use a clear and concise language for labels and instructions. In the current design, the label "Home Address (PO Boxes not accepted)" is a bit lengthy and could be more concise. To fix this, shorten the label to "Home Address" and provide information about PO Boxes not being accepted elsewhere, such as in a tooltip or help text.

The expected standard is that the design should use a consistent font size and weight for all labels. In the current design, the labels for the input fields use different font sizes and weights, creating visual inconsistency. To fix this, use a consistent font size and weight for all labels to improve readability and visual appeal.

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.

of initial comments from the TextGen module, as both the pipeline and baseline begin with this module.

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

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

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

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

A.5.2 QUALITATIVE ANALYSIS OF PIPELINE OUTPUTS FOR OUT OF DOMAIN UIs

Since UICrit consists of older UIs (from 2014) Duan et al. (2024a), we evaluated the pipeline’s performance to determine whether it generalizes to modern UIs and other out-of-domain UIs, such as websites, using only few-shot examples selected from UICrit. Figure 11 displays the generated feedback for four modern Android UIs (the few-shot examples from UICrit are also Android UIs) from 2024, taken from Mobbin³. Figure 12 presents feedback for four modern iOS UIs from 2024, sourced from DesignVault⁴, and Figure 13 illustrates feedback for five modern websites from 2024, also taken from Mobbin. In these figures, helpful comments with reasonably accurate bounding boxes are highlighted in green.

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

An interesting observation is that, since websites have more screen space, they are generally more complex and information-dense than mobile UIs (Gazzawe, 2017). We found one instance where the pipeline incorrectly flagged a relatively simple website as being too complex (i.e. having too many elements) in Figure 13 (second screen from the bottom), likely because it evaluated the com-

³<https://mobbin.com/>

⁴<https://designvault.io/>

plexity based on the mobile standards presented in the few-shot examples. However, the pipeline did correctly critique the bottom screenshot in the same figure for being overly complex, showing that it can appropriately identify this issue in some cases.

A.5.3 REFINING BOUNDING BOXES

While the bounding boxes could be improved qualitatively, as shown in Figures 5, 6, 7, and 8, there are straightforward approaches to easily improve the bounding box accuracy. For instance, the DOM tree representation of the UI contains the exact bounding boxes of UI elements and element groups. The DOM tree is available through the UI’s XML code, or the internal UI mockup representation available in design tools like Figma. If the DOM tree is not available, we could use a screen object parser (Wu et al., 2021; Xie et al., 2020) to extract the exact bounding boxes of UI elements and groups from the screenshot. These exact bounding boxes could then be used to refine the output bounding boxes for the critiques by updating them to precisely contain the exact bounding boxes of the closest UI elements or groups. This matching could be achieved through methods like IoU comparison, computing distances between the bounding box centers and sizes, or utilizing an VLM for matching, as was done by Zhao et al. (2024).

We demonstrate the refinement results from using the UI screenshot element detector created by Xie et al. (2020), which extracts the exact bounding boxes of all UI elements, for a few of the UICrit UIs in Figures 5, 6, 7, and 8. We refined the generated bounding boxes to precisely contain the closest exact UI element bounding boxes, determined via an IoU threshold. Since only the exact bounding boxes for UI elements were available, output bounding boxes intended for UI groups may be refined to include the exact bounding boxes of multiple UI elements. The refined results, shown in Figures 14 and 15, demonstrate that this simple refinement approach, which requires only the UI screenshot as input and no additional data (e.g., the DOM tree), significantly improves bounding box accuracy. This step could potentially be applied at the end of the pipeline to further clean up the generated bounding boxes.

A.6 ANALYSIS OF ITERATIVE REFINEMENT AND PIPELINE COST ANALYSIS

Figure 16 illustrates an example of iterative bounding box refinement (conditioned on the comment) by BoxRefine, which gradually improves the bounding box and terminates on a significantly more accurate bounding box. Figure 17 illustrates an example of comment refinement (conditioned on the bounding box) by TextRefine, which terminates on an accurate comment on the poor layout of the region inside the bounding box.

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

A.7 HUMAN EVALUATION METHOD

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

Given the limited availability of UI design experts and the extensive evaluation required per UI screen for a detailed comparison across the three conditions, only the Gemini-1.5-pro outputs for 33 UI screenshots were rated. To better represent the UI design space in this sample, we maximized the diversity of the UI screenshots by randomly sampling an even number of UIs from each of the UI task categories identified by Duan et al. (2024a). We followed their method of clustering by task descriptions from UICrit to obtain the task clusters. These 33 UIs were split into 6 groups for rating, with three participants assigned to each group. The rating and ranking study took approximately 1 hour. We recruited 18 design experts for this study. Five of the participants had 2-4 years of design experience, and the rest had 6-10 years. Their areas of design expertise include mobile, web, interaction, and user experience research.

1134 A.8 ALGORITHMS FOR GENERATING FEW-SHOT EXAMPLES FOR BOUNDING BOX
 1135 REFINEMENT
 1136

1137 Algorithm 1 details the steps for generating the few-shot refinement examples for a selected bound-
 1138 ing box. The few-shot generation algorithm entails perturbing the bounding box coordinates by
 1139 decreasing amounts and adding the perturbations to the list of few-shot examples. The algorithm for
 1140 perturbing a bounding box is also shown in Algorithm 1.

1141 **Algorithm 1** Generate Bounding Box Refinement Few-shot Examples
 1142

1143 **Require:** the bounding box to be perturbed *input_bbox*, the fraction that the bounding box’s coord-
 1144 inates will be perturbed *perturb_frac*

1145 **Ensure:** The coordinates of *input_bbox* perturbed by *perturb_frac*
 1146 1: **function** GENERATE_PERTURB(*input_bbox*, *perturb_frac*)
 1147 2: Compute *left_margin*, *right_margin*, *top_margin*, *bottom_margin*
 1148 3: *all_perturbed* \leftarrow []
 1149 4: **for** *x_perturb* in $[-perturb_frac \times left_margin, perturb_frac \times right_margin]$ **do**
 1150 5: **for** *y_perturb* in $[-perturb_frac \times top_margin, perturb_frac \times bottom_margin]$ **do**
 1151 6: Update bounding box location based on *x_perturb*, *y_perturb*
 1152 7: Add perturbed bounding box to *all_perturbed*
 1153 8: **end for**
 1154 9: **end for**
 1155 10: *final_perturbed* \leftarrow []
 1156 11: Compute *width* and *height* of the input bounding box
 1157 12: **for each** *perturbed_bbox* in *all_perturbed* **do**
 1158 13: **for** *width_fraction* in $[-perturb_frac, perturb_frac]$ **do**
 1159 14: **for** *height_fraction* in $[-perturb_frac, perturb_frac]$ **do**
 1160 15: Update bounding box size based on *width_fraction* and *height_fraction*
 1161 16: **end for**
 1162 17: **end for**
 1163 18: **end for**
 1164 19: *filtered_perturbed* \leftarrow *remove_invalid_perturbed_bbox*(*final_perturbed*, *input_bbox*)
 1165 20: *final_bbox* \leftarrow *random.choice*(*filtered_perturbed*)
 1166 21: **return** *final_bbox*
 1167 22: **end function**

1168 **Require:** Bounding box *bbox*, maximum number of perturbations of *bbox* in the list of fewshot
 1169 refinement examples *max_num_perturb*

1170 **Ensure:** A list of bounding boxes coordinates that are perturbed versions of *bbox* in decreasing
 1171 amounts, where *bbox* is the last item in the list.

1172 23: **function** GENERATE_PERTURBED_FEWSHOT_EXAMPLES(*bbox*, *max_num_perturb*)
 1173 24: *perturb_options* \leftarrow LIST(range(*max_num_perturb* + 1))
 1174 25: *num_perturb* \leftarrow RANDOM_CHOICE(*perturb_options*)
 1175 26: *perturb_list* \leftarrow []
 1176 27: **for** *j* \leftarrow *num_perturb* **to** 1 **do**
 1177 28: *perturb_frac* \leftarrow *j*/*max_num_perturb*
 1178 29: *output_bbox* \leftarrow GENERATE_PERTURB(*bbox*, *perturb_frac*)
 1179 30: *perturb_list.append*(*output_bbox*)
 1180 31: **end for**
 1181 32: *perturb_list.append*(*bbox*)
 1182 33: **return** *perturb_list*
 1183 34: **end function**

1184
 1185
 1186
 1187

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

Top Screen: 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 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 design has too many icons that are not necessary. To fix this, remove unnecessary icon

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 meet the needs of the business. In the current design, it is not clear whether the design meets the needs of the business. To fix this, ensure that the design meets the needs of the business.

The expected standard is that the design should be responsive and work well on all devices. In the current design, the layout is not responsive and does not work well on all devices. To fix this, make the layout responsive so that it works well on all devices.

Bottom Screen: 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.

Bottom Screen: Baseline

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.

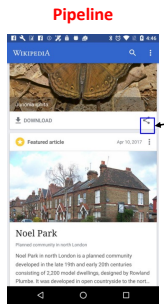
The expected standard is that the design should use as few elements as possible to achieve its goals. In the current design, the text "OR" is unnecessary. To fix this, remove the text "OR".

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.

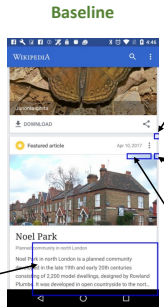
The expected standard is to have a noticeable clickable element with clear contrast. In the current design, the element on the screen appears non-clickable and seamlessly integrated with the background, giving users the impression that it is merely a static part of the display rather than an interactive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

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.

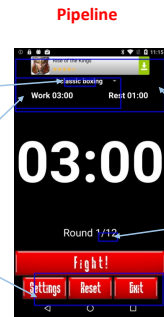


The expected standard is that the design should be consistent throughout. In the current design, the share icon is not consistent. To fix this, we can use the same share icon.

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

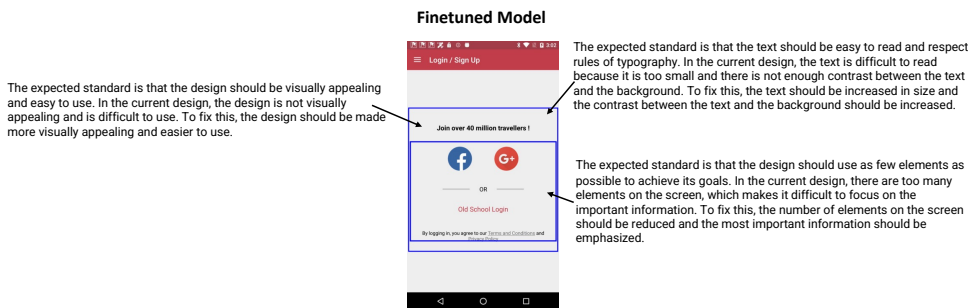
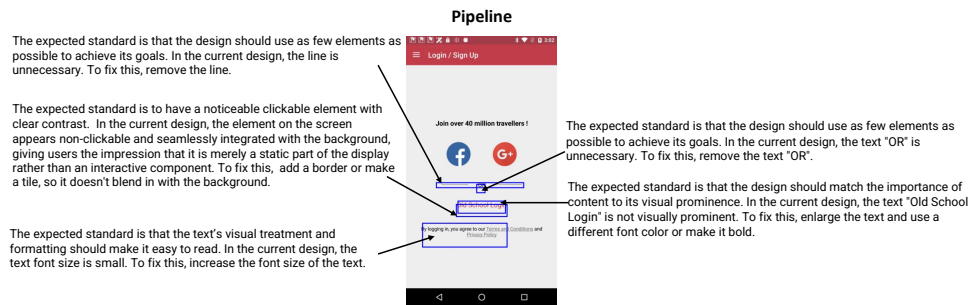
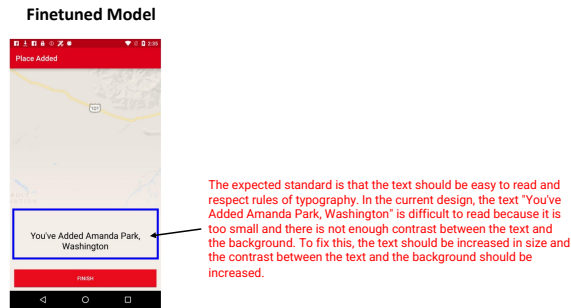
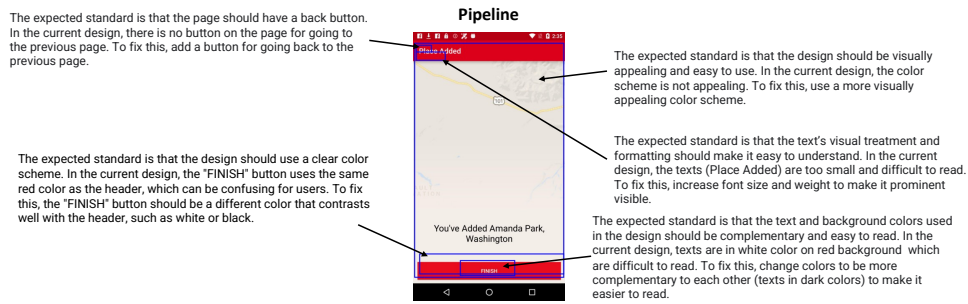


Figure 10: Illustration of outputs from the pipeline and finetuned Llama-3.2 11b. The screenshots are marked with the output bounding boxes, and the generated comments are shown, each pointing to its corresponding bounding box (some comments have the same bounding box). The fine-tuned model produces 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. Each visual element should contribute to the overall message; 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

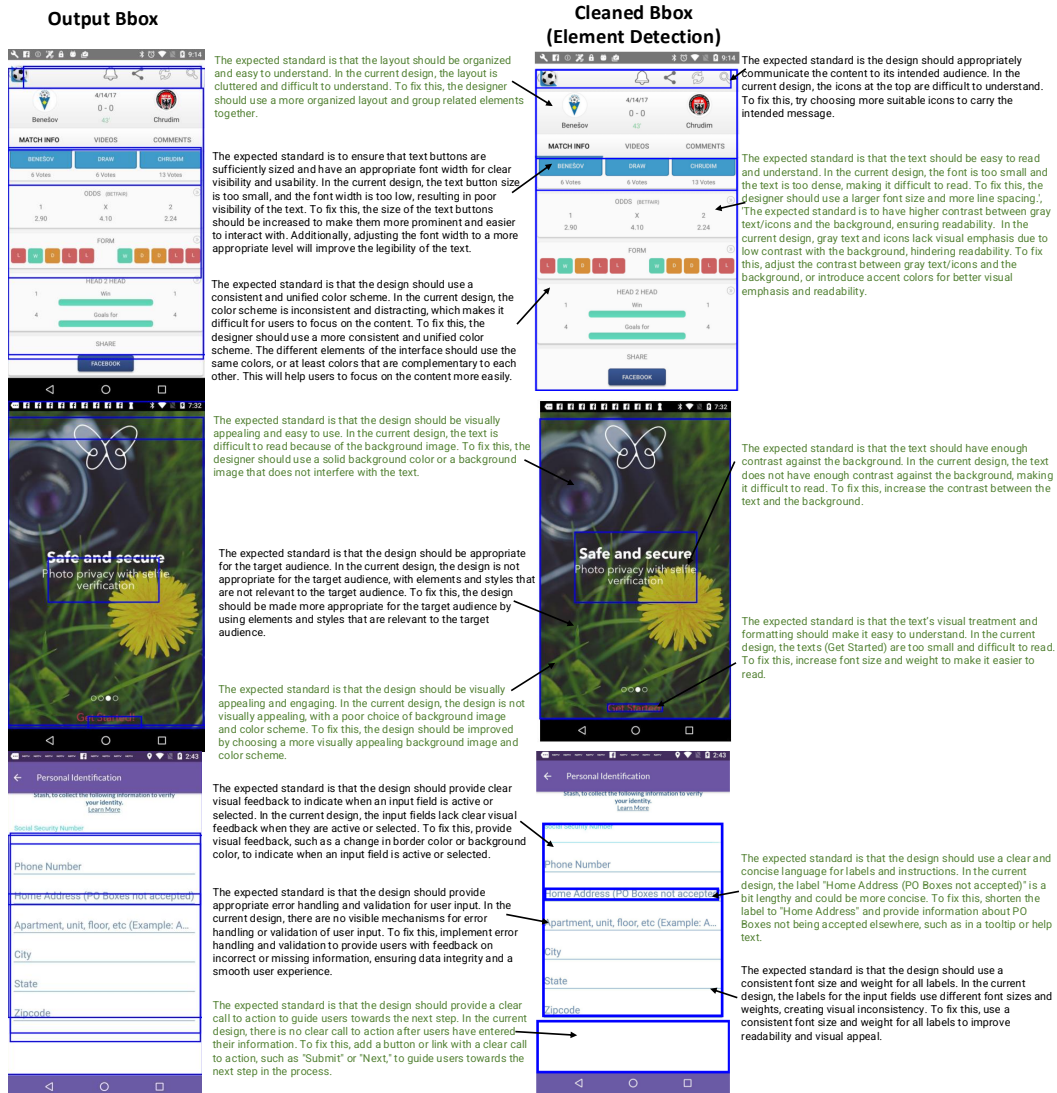


Figure 14: Side by side comparison of the bounding boxes generated by the pipeline (“Output Bbox”) and refined bounding boxes (“Cleaned Bbox (Element Detection)”), which were adjusted using the exact bounding boxes of the nearest UI elements. The exact bounding boxes were computed using a UI element detector (Xie et al., 2020), and the nearest elements were determined based on an IoU threshold with the output bounding box. 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

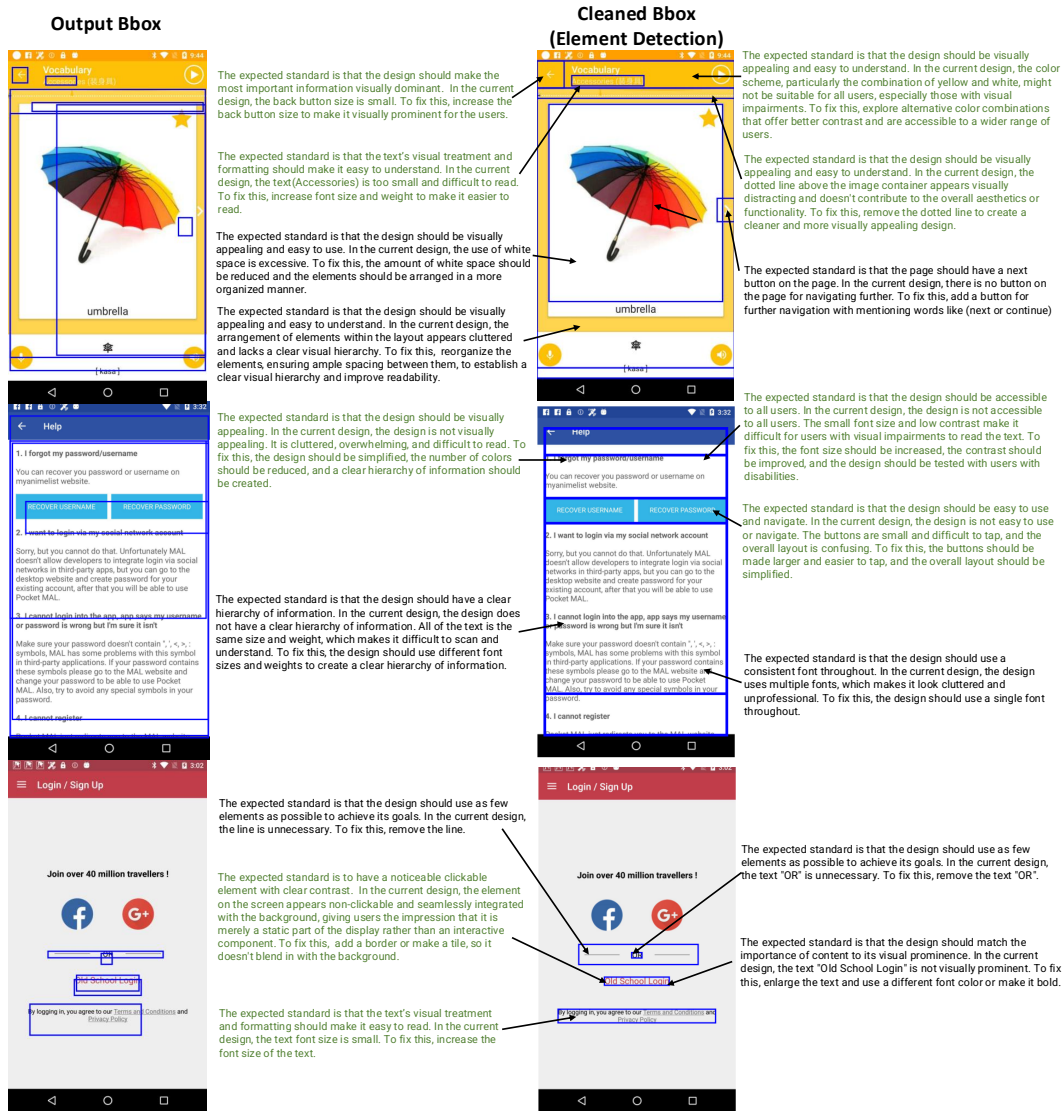
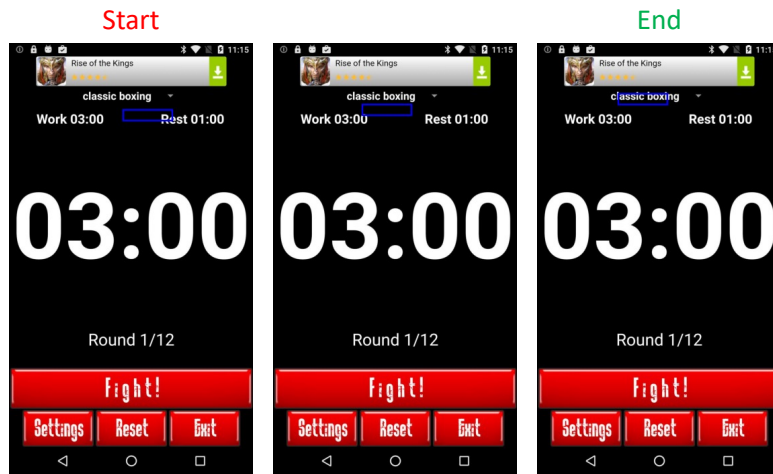


Figure 15: Side by side comparison of the bounding boxes generated by the pipeline (“Output Bbox”) and refined bounding boxes (“Cleaned Bbox (Element Detection)”), which were adjusted using the exact bounding boxes of the nearest UI elements. The exact bounding boxes were computed using a UI element detector (Xie et al., 2020), and the nearest elements were determined based on an IoU threshold with the output bounding box. 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

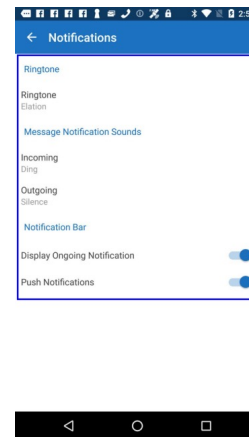


Figure 17: An example of iterative comment refinement, with the bounding box it is conditioned on displayed on the right. The first comment ('Start') was classified as incorrect by the Validation VLM but has a bounding box that accurately encloses a problematic region in the UI. The refinement process progressively improves the comment, terminating with a comment that correctly points out 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.

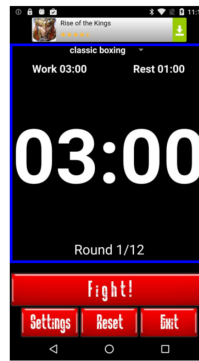
Please rank each set of comments, as a whole, based on their overall quality. Please rank them in decreasing quality.

	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.