

From Design Critique to Editable HTML: LLM-Based Transformation of UI Feedback into Front-End Edits

Kevin Liu¹, Stephanie Huang²

¹Computer Science, University Of California, Irvine, Irvine, CA, USA

²Informatics (Human-Computer Interaction), University Of California, Irvine, Irvine, CA, USA

k.liu_uci@gmail.com

ABSTRACT

Natural-language design critique contains concrete instructions about spacing, hierarchy, component styling, content, and usability, yet front-end generation is usually evaluated as one-shot reconstruction. This study evaluates critique-conditioned HTML editing on the Apple ML-RLDF designer-feedback corpus. The analysis uses all 1,499 records across four feedback modalities: 152 comment-conditioned HTML pairs, 181 region-grounded critique pairs, 1,098 designer ranking pairs, and 68 direct designer edit pairs. The 333 critique-conditioned outputs are measured with normalized HTML and DOM-tag edit distance, visible-text preservation, class-level changes, structural similarity, pixel error, color similarity, edge overlap, and an eight-category detectable realization score. Ranking pairs train description-grouped pairwise preference models, with exact screenshot overlap removed before transfer scoring. Comment-conditioned pages changed more code than grounded pages (mean normalized distance 0.122 versus 0.090; $p = 0.013$), while their visual similarity remained comparable (mean SSIM 0.703 versus 0.720; $p = 0.380$). Direct designer edits produced a mean SSIM of 0.731, and SSIM did not differ across the three edit modalities (Kruskal-Wallis $p = 0.468$). Detectable realization averaged 0.705 for comments and 0.702 for grounded critique. Spacing and component requests were implemented most consistently, whereas hierarchy and accessibility requests were rarely accompanied by corresponding semantic changes. The strongest ranking model reached 53.7% description-grouped accuracy, and its transfer probabilities stayed near 0.5. The results show that critique-conditioned HTML editing can preserve page identity while making targeted source changes, but ranking-derived preference is too weak to serve as a stand-alone quality measure.

Keywords: *designer feedback; HTML editing; user interface generation; large language models; visual similarity; preference learning; front-end engineering*

1. Introduction

Generating interface code from language or images has progressed from specialized screenshot-to-code systems to large multimodal and code language models. pix2code framed graphical interface reconstruction as program generation [1], while RICO [2], Screen2Words [3], and WebSight [4] expanded the available supervision for learning visual structure and language-screen relationships. More recent systems combine code generation with compiler, rendering, and multimodal feedback. UICoder uses automated feedback to improve UI code generation [5], Design2Code evaluates real webpages with fine-grained visual and structural measures [6], and Ferret-UI strengthens grounded understanding of mobile screens [7]. Together, these studies show that executable source and rendered appearance are related but distinct outcomes.

Critique-conditioned editing poses a different problem from reconstruction. A designer rarely asks for a new page after every observation; instead, the designer points to weak hierarchy, excessive whitespace, an unclear label, or a misaligned control and expects the current page to remain recognizable after the change. The model must preserve the document, locate the relevant source region, choose an implementation strategy, and return complete executable HTML. A large source edit can produce a small pixel change, while a one-token utility-class change can move a component across the viewport. Evaluation therefore has to inspect both code and rendering.

The Apple ML-RLDF study collected ranking, commenting, region-grounded commenting, sketching, and direct editing feedback from professional designers [8]. Its accompanying repository exposes the data modalities and model resources used in the study [9]. The comment and grounded subsets contain original HTML, designer-authored critique, complete edited HTML, and rendered before-and-after screenshots. The ranking subset contains preferred and nonpreferred HTML-image pairs, while the direct edit subset contains screenshots before and after a designer changed the artifact. These complementary modalities make it possible to connect code difference, visual change, critique realization, and preference transfer in one analysis.

Preference learning is central to modern language-model alignment. Reinforcement learning from human feedback fits reward models from comparisons [10], while DPO [11] and ORPO [12] optimize preference objectives more directly. For interfaces, UIClip demonstrates that UI-specific visual-language learning can capture design quality and relevance [13], building on general contrastive image-text representation learning such as CLIP [14]. Pairwise labels, however, compress the reasons for a design judgment into a binary choice. Whether a modest set of designer rankings transfers to critique-driven page edits must therefore be tested rather than assumed.

This study addresses four questions. First, how much source code changes when a language model applies designer critique, and which edit mechanisms dominate? Second, how much does the rendered page change relative to direct designer edits? Third, which critique categories are accompanied by detectable source changes? Fourth, can models trained on ranking pairs distinguish the after state from the before state in critique-conditioned and direct edit pairs? The analysis covers every record in all four subsets, removes exact-image overlap before preference transfer, reports uncertainty intervals and nonparametric tests, and links aggregate results to concrete before-and-after pages. The contribution is an evaluation framework for editable HTML that keeps code locality, visual continuity, semantic realization, and learned preference as separate dimensions.

2. Literature Review

2.1 Screenshot-to-code and editable interface generation

Research on interface generation has moved from direct screenshot reconstruction toward systems that combine visual understanding, code synthesis, and iterative validation. Early reconstruction work demonstrated that a screenshot could be translated into a structured program [1]. Larger UI corpora and language-screen datasets then supported learning across diverse applications and descriptions [2]-[4]. Automated compiler and rendering feedback further shifted attention from merely producing plausible code to producing code that executes and visually matches its target [5], [6]. Grounded multimodal models add region-level understanding that is especially relevant when a critique refers to one control or component rather than the whole screen [7].

Sketch-to-prototype research extends the same progression to low-fidelity design input. Chen and Li evaluate the transformation of hand-drawn sketches into interactive HTML/CSS prototypes with both structural and visual consistency measures [17]. This line of work reinforces two requirements that are also central to critique-conditioned editing: the output must remain executable, and visual agreement alone cannot reveal whether the underlying structure is appropriate. Unlike sketch-to-code generation, however, critique-conditioned editing begins with an existing implementation and must preserve unaffected regions.

2.2 Automated critique and rationale-bearing feedback

Automated design evaluation increasingly uses natural-language critique rather than a single scalar score. UICrit provides expert critiques, region boxes, and design-quality ratings for mobile interfaces, supporting feedback generation and grounded evaluation [15]. LLM-as-Design-Critic similarly studies alignment between AI-generated feedback and human graphic-design judgment [16]. The Apple ML-RLDF corpus adds an important complementary perspective by recording how designers express comments, ground them to regions, rank alternatives, and directly edit artifacts [8]. Together, these datasets show that design supervision includes both a decision and a rationale.

Recent interface frameworks also make design reasoning inspectable through structured cards. Text-grounded design-rationale interfaces turn layout metadata into evidence-linked decision cards [18], while structured visual brief interfaces separate creative intent into designer-editable fields before production [19]. These systems are not direct HTML editors, but they support the same principle: intermediate design intent should remain visible enough to inspect, revise, and audit. For critique-conditioned code editing, this principle motivates category-level realization checks rather than treating the final screenshot as the only outcome.

2.3 Semantic quality, oversight, and preference

Semantic quality can also depend on content and interaction cues that are difficult to infer from global pixel similarity. Work on detecting and explaining dark patterns from interface microcopy shows that wording and labels can carry usability and ethical consequences even when layout is unchanged [20]. Privacy and data-integrity risk cards for LLM agents likewise emphasize explicit, human-readable oversight signals in interface design [21]. These studies support evaluating visible text, labels, interaction elements, and accessibility-related attributes alongside visual continuity.

Preference models remain useful because they summarize which alternative a designer selected, and modern alignment methods provide effective objectives for learning from comparisons [10]-[12]. UIClip demonstrates that large, UI-specific visual-language data can improve agreement with human design rankings [13]. At the same time, critique datasets reveal information that a pairwise label omits: the affected region, the reason for the judgment, and the intended correction [8], [15], [16]. The literature therefore points toward a multi-view evaluation in which preference is one signal among code change, visual effect, and semantic realization.

3. Materials and Methods

3.1 Dataset and integrity checks

The corpus contains four Hugging Face Arrow datasets. The comment subset has 152 rows with original and edited HTML, paired screenshots, and critique text. The grounded critique subset contains the same fields for 181 rows, with feedback tied to selected interface regions. The ranking subset has 1,098 rows of designer-chosen and designer-rejected screenshots with their HTML. The direct edit subset has 68 screenshot pairs in which the chosen image is the designer-produced result and the rejected image is the initial page. Every row entered the analysis; no sampling step was applied. Table 1 summarizes the four subsets and their image counts.

All image byte streams were decoded, converted to RGB, and checked for dimensions, format, and exact SHA-256 identity. No invalid images were found. The archive contains 2,998 image instances representing 1,792 unique image hashes. Most screenshots are 512 by 1,108 pixels. Eight ranking images and three grounded images differ by one to three vertical pixels, while nineteen direct edit images have shorter heights. Images with nonstandard dimensions were resized to 512 by 1,108 only for metric and feature extraction; stored files were not altered. Exact hashing identified 101 images shared by the ranking and comment subsets, 118 shared by ranking and grounded critique, and thirteen shared by the two critique subsets. No direct edit screenshot overlapped another subset. Figure 1 visualizes the record and image distribution, and Table 2 maps each feedback modality to its experimental role.

Table 1. Dataset composition and image integrity

Subset	Rows	Images	Designers	Descriptions	Unique images	Duplicate instances
Comment	152	304	21	114	293	11
Grounded critique	181	362	21	118	350	12
Ranking	1098	2196	21	215	1239	957
Direct edit	68	136	10	58	133	3

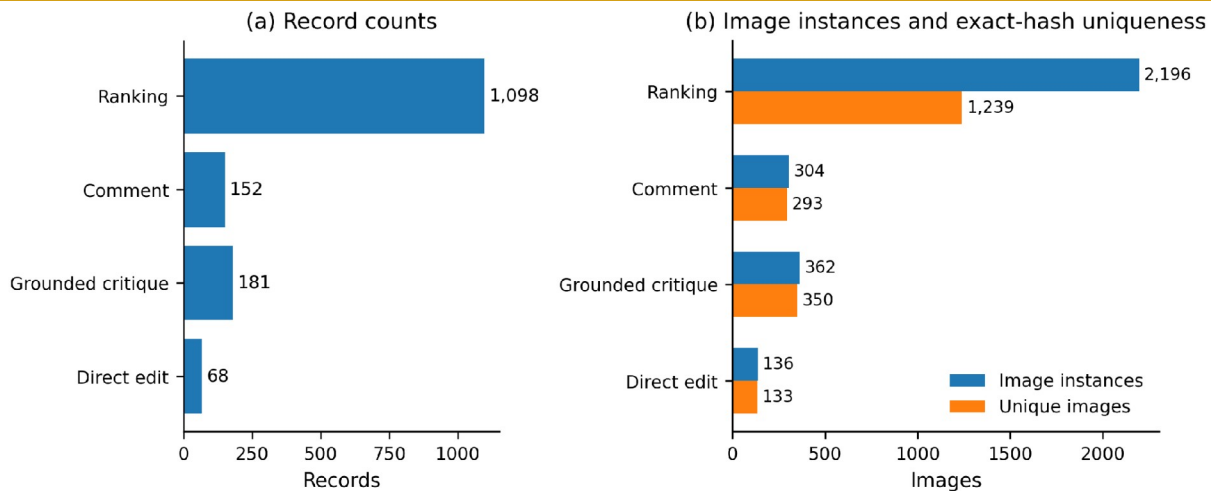


Figure 1. Dataset composition. The ranking subset supplies most records and repeated image instances; the three edit modalities provide paired before-and-after screenshots.

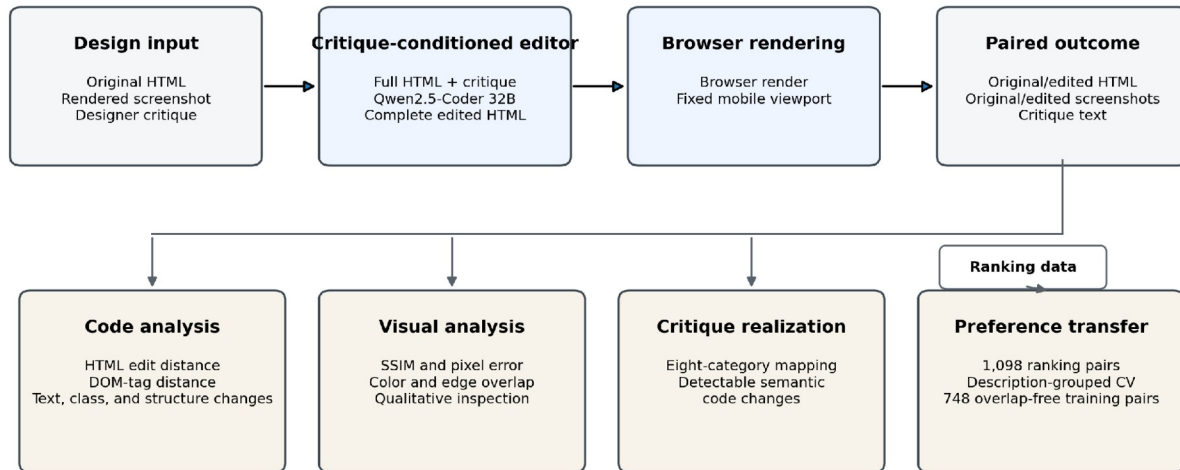
Table 2. Role of each feedback modality in the experiments

Subset	Input evidence	Observed outcome	Experimental use
Comment	Critique + original HTML	Complete edited HTML and screenshot	Code, visual, realization, preference transfer
Grounded critique	Region-linked critique + original HTML	Complete edited HTML and screenshot	Code, visual, realization, preference transfer
Ranking	Two HTML-render pairs	Designer choice	Grouped preference-model training and validation
Direct edit	Initial design screenshot	Designer-produced screenshot	Human visual-change benchmark and visual preference transfer

3.2 Critique-conditioned HTML transformation

For the comment and grounded subsets, Qwen2.5-Coder 32B [22] receives the existing page source together with a prompt derived from designer comments and returns complete HTML. The rendered output is stored beside the source and critique, enabling deterministic comparison across all 333 complete transformations.

Grounded annotations contain repeated comment and code-region blocks. The parser extracted each comment block, removed the associated code excerpt from category analysis, and joined the designer comments for the row. Ordinary comment annotations were split into nonempty lines after list markers were removed. This procedure kept critique language separate from source tokens. Figure 2 summarizes the path from critique and HTML to code, visual, realization, and preference analyses.



Direct designer edits provide a screenshot-only human benchmark for visual-change metrics.

Figure 2. Experimental pipeline. Code and visual measures are computed on every critique-conditioned pair; ranking data train separate preference-transfer models after exact-image overlap is removed.

3.3 HTML and content measures

HTML was parsed with BeautifulSoup using the lxml parser. Comments, scripts, and noscript elements were removed from the analysis tree. The body was serialized into a canonical sequence containing opening and closing tags, sorted attributes, and normalized visible-text tokens. Normalized Levenshtein distance [23] was computed as edit operations divided by the longer token sequence. A second distance used only body tag tokens and therefore measured structural change with less sensitivity to text and utility classes. Full tree-edit algorithms such as Zhang and Shasha [24] preserve parent-child structure explicitly; the tag-sequence measure was selected because the pages mix long class lists with shallow mobile layouts and the analysis focuses on relative edit magnitude across paired documents.

Visible-text preservation was measured by Jaccard similarity between lowercase alphanumeric token sets. Class additions and deletions were computed as multiset differences across class attributes. Boolean indicators recorded whether the ordered visible text changed, whether the tag sequence changed, and whether embedded style-block text changed. These measures distinguish a source-preserving utility-class edit from a change that inserts elements, modifies page copy, or rewrites CSS. Table 3 summarizes the primary measures and their interpretation.

Table 3. Primary measures and interpretation

Measure	Representation	Interpretation
HTML edit distance	Canonical body tokens	0 = identical; larger = more source change
DOM-tag distance	Opening/closing tag sequence	0 = same structural tag sequence
Text Jaccard	Unique visible word tokens	1 = complete token-set preservation
Class delta	Tailwind/HTML class multisets	Counts additions, deletions, and changed tokens
SSIM	Downsampled grayscale renders	1 = identical local structure

Measure	Representation	Interpretation
Pixel MAE	Normalized RGB absolute error	0 = identical pixels
Color cosine	HSV histograms	1 = identical color distribution
Edge IoU	Canny edge masks	1 = identical edge support
Realization rate	Requested vs. detectable change categories	1 = all detected categories changed
Preference probability	Pairwise ranking model	>0.5 favors the after state

3.4 Visual comparison

Visual metrics were computed after each screenshot was resized to the common portrait canvas. SSIM [25] was evaluated on 256 by 554 grayscale images to retain the page aspect ratio while controlling computational cost. Pixel mean absolute error used RGB values divided by 255. Hue, saturation, and value histograms were normalized and compared by cosine similarity. Canny edge maps [27], implemented through OpenCV [26], were compared by intersection over union. These measures cover local luminance structure, global color distribution, absolute pixel change, and component boundaries. The 68 direct designer pairs were processed with the same image pipeline and provide a designer-created reference distribution for visual change.

3.5 Critique taxonomy and detectable realization

Designer comments were mapped to eight nonexclusive categories using fixed regular-expression dictionaries: spacing/layout, typography, color/contrast, hierarchy/emphasis, component styling, content/labeling, navigation/interaction, and accessibility/usability. The dictionaries were specified before aggregate outcomes were examined. A requested category was marked as detectably realized when the edited HTML contained a corresponding semantic change. Examples include spacing utility changes for layout, text or font utilities for typography, color utilities for contrast, heading or emphasis utilities for hierarchy, text changes for content, interactive-element or state-class changes for navigation, and changes to ARIA, role, alt, or tabindex attributes for accessibility. Structural changes also counted toward layout and component styling. The row-level rate is the number of requested categories with matching detectable changes divided by the number of requested categories. Rows without a dictionary match were excluded only from this rate, not from other metrics. Table 4 lists the critique cues and source-level evidence used for each category.

Table 4. Critique categories and source-level detection rules

Category	Critique cues	Detectable implementation evidence
Spacing/layout	padding, margin, align, position, grid, width	spacing utilities or structural change
Typography	font, size, weight, heading, readability	font/text/leading/tracking utilities
Color/contrast	color, contrast, background, named colors	background, text, border, fill, stroke utilities
Hierarchy/emphasis	hierarchy, prominent, primary, emphasis	heading-count or emphasis-utility change
Component styling	button, card, icon, border, image, field	component utilities, structure, interactive count
Content/labeling	label, copy, wording, missing, information	visible-text sequence change
Navigation/interaction	menu, tab, click, hover, scroll, select	interactive element or state-class change
Accessibility/usability	accessibility, target size, difficult, visibility	ARIA, role, alt, or tabindex change

3.6 Ranking-based preference models

Each ranking row was represented as the feature difference between the designer-chosen page and the rejected page. Symmetric training duplicated every difference with reversed sign and the opposite label, following the pairwise logic of Bradley-Terry models [28]. Four feature sets were tested. The 142-dimensional global visual descriptor contains RGB and HSV moments, luminance entropy, bright and dark area ratios, edge density, gradient statistics, coarse spatial luminance, symmetry measures, and normalized color histograms. The 910-dimensional spatial descriptor adds pooled luminance, saturation, and edge maps. The 170-dimensional HTML descriptor contains document length, tag counts, tree depth, attributes, utility-class prefix counts, and the eighty most frequent class tokens. Fusion concatenates spatial visual and HTML features.

Logistic regression, random forest, extremely randomized trees, and histogram gradient boosting were evaluated. Three-fold GroupKFold partitioning grouped rows by normalized page description, so versions of the same requested application did not span train and test folds. Accuracy, area under the ROC curve, Brier score, and log loss were computed from out-of-fold probabilities using scikit-learn [29]. The best visual-only model was used for all three edit modalities because direct edit rows have no HTML. The best model overall was also applied to the two HTML-bearing critique modalities.

Preference transfer used stricter training data than cross-validation. Every ranking pair was removed when either screenshot exactly matched a screenshot in the comment or grounded evaluation sets. This exclusion removed 350 pairs and left 748 overlap-free training pairs. The fitted model returned the probability that the after state would be selected over the before state, which was used to test whether ranking preferences transferred to edit pairs.

3.7 Statistical analysis

Means and 95% confidence intervals used 3,000 bootstrap resamples following the nonparametric bootstrap framework [30]. Differences between comment and grounded means used 5,000 independent bootstrap resamples. Mann-Whitney tests [31] compared the two critique modalities, and Kruskal-Wallis tests compared image metrics across comment, grounded, and direct edit pairs. Wilcoxon signed-rank tests [32] evaluated whether transferred preference probabilities differed from 0.5 after centering, and exact binomial tests evaluated win counts. Spearman correlation [33] measured the monotonic association between HTML edit distance and visual change. SciPy supplied the statistical routines [34]. All reported group comparisons are two-sided, alpha was 0.05, and the fixed random seed was 20250303.

4. Results

4.1 Corpus coverage and edit completeness

The integrity scan confirmed all 1,499 records and all 2,998 image instances. The four subsets contained 21, 21, 21, and 10 participant identifiers for comment, grounded critique, ranking, and direct edit, respectively. Exact duplicates were concentrated in the ranking subset because the same design participated in multiple comparisons. The complete critique evaluation comprised 333 HTML pairs and 666 screenshots; the direct designer benchmark comprised 68 screenshot pairs. Three critique pairs had zero normalized HTML distance, twenty-nine had SSIM above 0.99, and 235 preserved the complete visible-text token set. Thus almost every critique generated a detectable source difference, while most edits retained page wording.

4.2 HTML edit magnitude and anatomy

Comment-conditioned HTML had a mean normalized edit distance of 0.122 (95% CI [0.103, 0.143]), compared with 0.090 [0.078, 0.103] for grounded critique. The mean difference was 0.032, with a bootstrap interval of [0.009, 0.056], Mann-Whitney $p = 0.013$, and Cliff's delta = 0.157. Grounding therefore corresponded to smaller source edits. DOM-tag distance was also lower in the grounded subset (0.057 versus 0.075), but the distributional difference was not significant ($p = 0.319$). Table 5 summarizes the main outcomes by modality.

Visible content was strongly preserved: mean text Jaccard was 0.958 for comments and 0.971 for grounded critique, with a median of 1.000 in both groups. Structure changed in 57.9% and 55.8% of rows, while visible text changed in 41.4% and 30.9%. Utility classes were the dominant edit mechanism. At least one class was added in 94.1% of comment rows and 93.9% of grounded rows. Comment rows changed an average of 24.0 class tokens, compared with 18.9 for grounded rows. Embedded style blocks changed in only six grounded rows and no comment row, showing that the transformations relied primarily on existing utility-class conventions rather than rewriting CSS blocks. Table 6 details this edit anatomy.

Table 5. Main outcomes by edit modality; values are means with 95% confidence intervals where shown

Modality	n	HTML distance	DOM-tag distance	Text Jaccard	SSIM	Realization
Comment	152	0.122 [0.103, 0.143]	0.075	0.958	0.703 [0.677, 0.730]	0.705 [0.649, 0.759]
Grounded critique	181	0.090 [0.078, 0.103]	0.057	0.971	0.720 [0.695, 0.745]	0.702 [0.647, 0.755]
Direct edit	68	—	—	—	0.731 [0.690, 0.771]	—

Table 6. HTML edit anatomy

Edit property	Comment	Grounded critique
Structure changed	88 (57.9%)	101 (55.8%)
Visible text changed	63 (41.4%)	56 (30.9%)
Style block changed	0 (0.0%)	6 (3.3%)
Any class addition	143 (94.1%)	170 (93.9%)
Any class deletion	125 (82.2%)	130 (71.8%)
Mean class additions	14.7	11.7
Mean class deletions	9.3	7.1
Mean changed class tokens	24.0	18.9

4.3 Visual continuity and the direct designer benchmark

Mean SSIM was 0.703 for comments, 0.720 for grounded critique, and 0.731 for direct designer edits. The omnibus Kruskal-Wallis test did not detect a difference among the three distributions ($p = 0.468$). Pairwise tests were also nonsignificant: comment versus grounded $p = 0.380$, comment versus direct edit $p = 0.241$, and grounded versus direct edit $p = 0.659$. Pixel MAE means were 0.075, 0.072, and 0.072. Figure 3 compares the group-level outcomes, and Table 7 reports the corresponding tests.

HTML distance and visual change were positively associated but far from interchangeable. For comment rows, Spearman ρ was 0.297 ($p = 0.0002$); for grounded rows, ρ was 0.207 ($p = 0.0051$). Figure 4 shows substantial variation around both fitted lines. Several small source edits produced large visual shifts, and several extensive class changes preserved most of the rendering. A front-end evaluation based only on source difference would therefore mischaracterize a meaningful portion of the transformations.

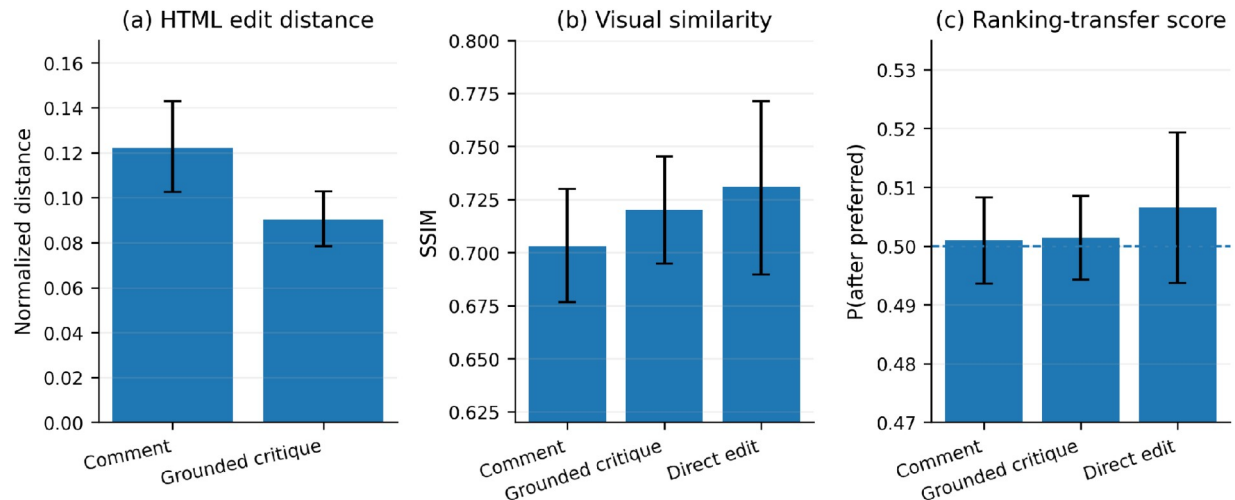


Figure 3. Main outcomes. Error bars show bootstrap 95% confidence intervals. Ranking-transfer probabilities use the overlap-free visual model; the dashed line marks a neutral probability of 0.5.

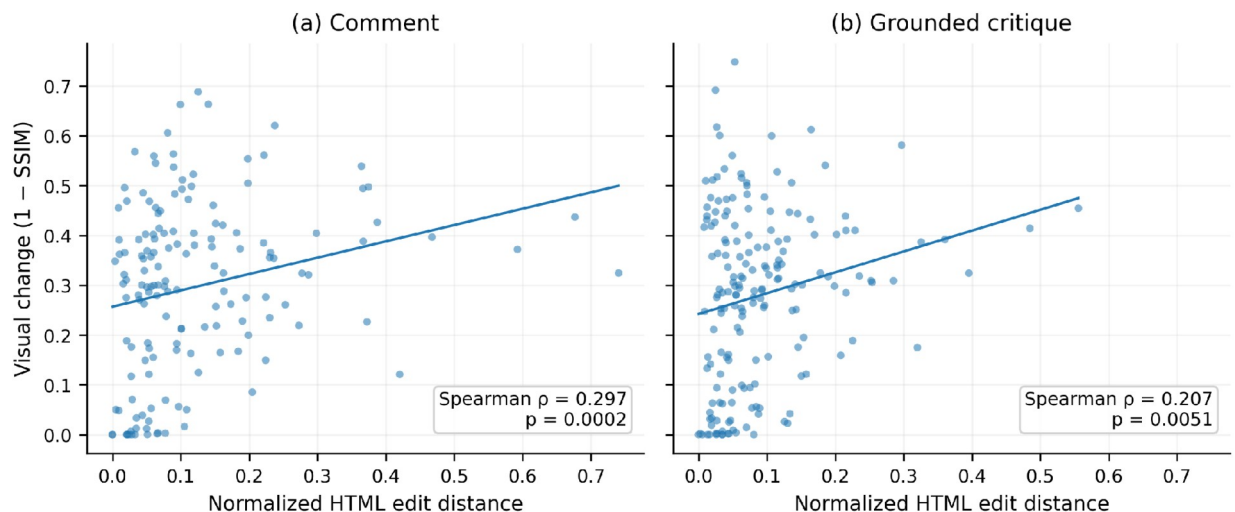


Figure 4. Relationship between normalized HTML edit distance and visual change. HTML and pixels are significantly associated, but the weak-to-moderate correlations leave substantial pair-level dispersion.

Table 7. Key distributional tests and code-visual associations

Comparison	Outcome	Difference/association	95% CI	p	Cliff delta
Comment vs. grounded	HTML distance	0.032	[0.009, 0.056]	0.013	0.157
Comment vs. grounded	DOM-tag distance	0.018	[-0.003, 0.041]	0.319	0.061
Comment vs. grounded	SSIM	-0.017	[-0.055, 0.021]	0.380	-0.056

Comparison	Outcome	Difference/association	95% CI	p	Cliff delta
Comment vs. grounded	Realization rate	0.003	[-0.076, 0.080]	0.828	-0.014
All three modalities	SSIM	—	—	0.468	—
Comment	HTML distance vs. 1-SSIM	rho = 0.297	—	0.0002	—
Grounded critique	HTML distance vs. 1-SSIM	rho = 0.207	—	0.0051	—

4.4 Critique content and detectable realization

Comment rows contained an average of 2.94 detectable issue categories and 29.9 critique words, compared with 2.52 categories and 22.3 words for grounded critique. Component styling and spacing/layout dominated both modalities. In comments, component styling appeared in 63.8% of rows and spacing/layout in 59.9%. In grounded critique, the corresponding prevalences were 53.0% and 52.5%. Accessibility/usability was least frequent, at 5.9% and 5.0%. Grounding reduced the frequency of content/labeling comments from 33.6% to 22.1% and color/contrast comments from 26.3% to 18.8%. Figure 5 and Table 8 summarize category prevalence and associated edit distance.

Detectable realization averaged 0.705 for comments and 0.702 for grounded critique; the difference was 0.003 ($p = 0.828$). Spacing/layout had the highest realization rate, 97.8% for comments and 92.6% for grounded critique. Component styling reached 71.1% and 70.8%. Typography was detected in 84.4% of comment rows but 56.8% of grounded rows. Content/labeling reached 49.0% and 37.5%. Hierarchy/emphasis and accessibility/usability were substantially lower than the other categories, indicating that broad design-language requests were less often matched by heading, emphasis, or semantic-attribute changes. Table 9 reports realization and visual similarity by category.

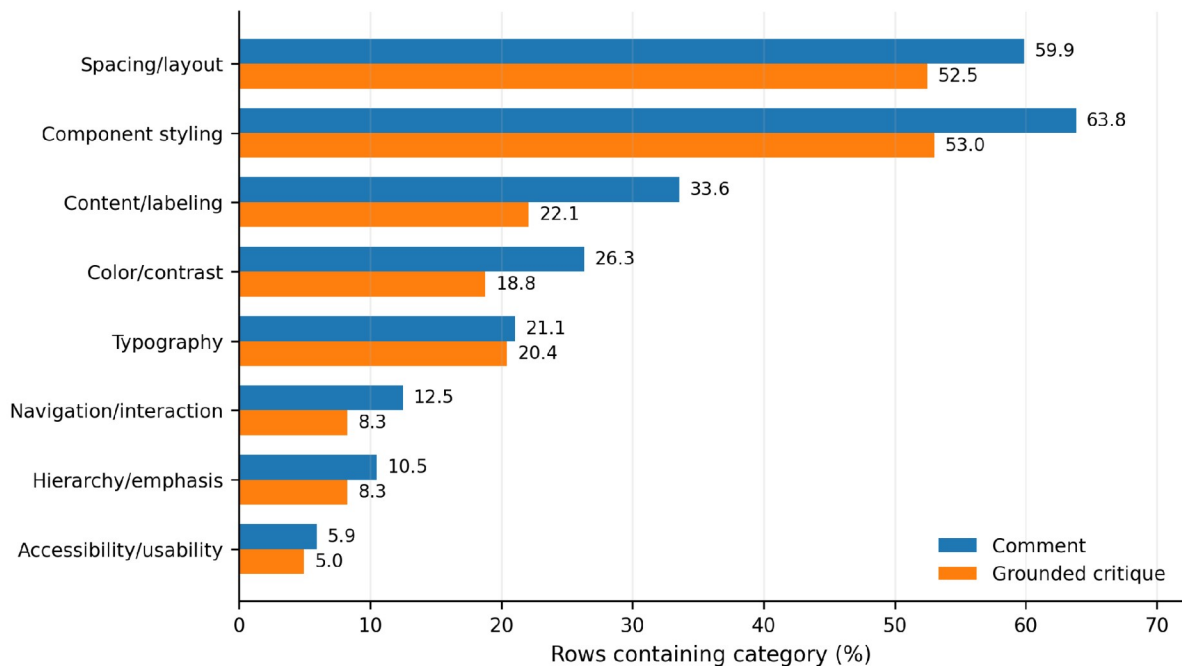


Figure 5. Critique-category prevalence. Categories are nonexclusive, so percentages sum to more than 100%.

Table 8. Critique prevalence and mean HTML edit distance

Category	Comment n (%)	Grounded n (%)	Comment distance	Grounded distance
Spacing/layout	91 (59.9%)	95 (52.5%)	0.115	0.086
Component styling	97 (63.8%)	96 (53.0%)	0.115	0.097
Content/labeling	51 (33.6%)	40 (22.1%)	0.125	0.125
Color/contrast	40 (26.3%)	34 (18.8%)	0.105	0.077
Typography	32 (21.1%)	37 (20.4%)	0.112	0.074
Navigation/interaction	19 (12.5%)	15 (8.3%)	0.125	0.107
Hierarchy/emphasis	16 (10.5%)	15 (8.3%)	0.122	0.175
Accessibility/usability	9 (5.9%)	9 (5.0%)	0.176	0.084

Table 9. Detectable realization and mean visual similarity by category

Category	Comment realization	Grounded realization	Comment SSIM	Grounded SSIM
Spacing/layout	97.8%	92.6%	0.697	0.707
Component styling	71.1%	70.8%	0.711	0.721
Content/labeling	49.0%	37.5%	0.691	0.723
Color/contrast	75.0%	67.6%	0.733	0.753
Typography	84.4%	56.8%	0.671	0.679
Navigation/interaction	57.9%	40.0%	0.707	0.757
Hierarchy/emphasis	6.2%	13.3%	0.741	0.672
Accessibility/usability	0.0%	11.1%	0.661	0.599

4.5 Critique complexity

Edit magnitude increased with the number of issue categories in a critique. For comments, mean HTML distance rose from 0.051 for one detected issue to 0.128 for two or three and 0.142 for four or more. Grounded critique followed the same pattern, rising from 0.051 to 0.096 and 0.119. Visual similarity generally decreased as critiques became more complex. Comment SSIM fell from 0.836 for one issue to 0.678 for two or three; grounded SSIM fell from 0.773 to 0.715 and 0.674. At the same time, detectable realization rose with issue count, reaching 0.726 for complex comments and 0.773 for complex grounded critique. Figure 6 and Table 10 show these tradeoffs.

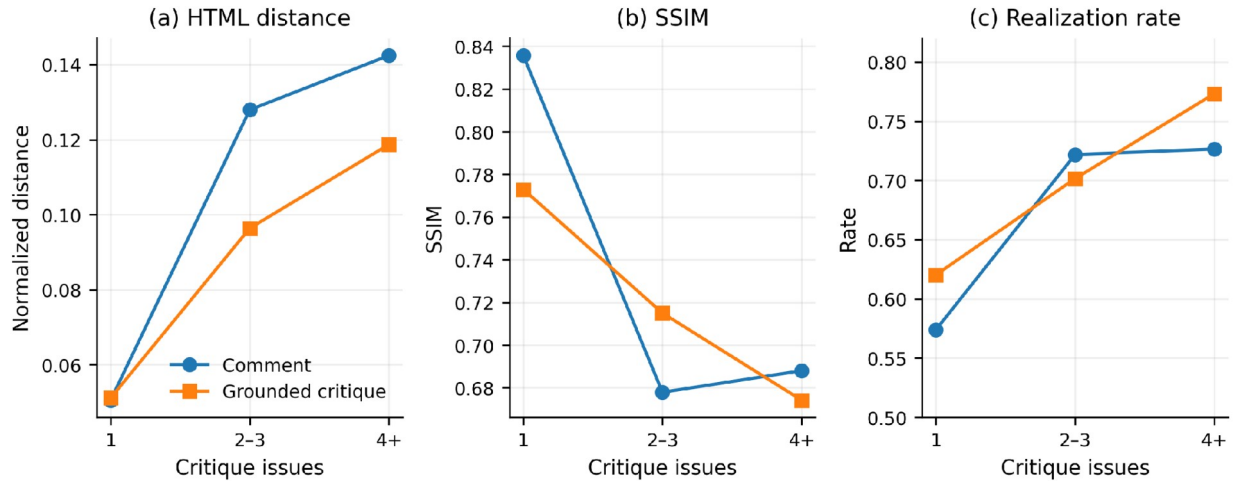


Figure 6. Outcomes by number of critique issues. More complex critiques produced larger HTML changes, lower visual similarity, and higher detectable realization.

Table 10. Outcomes by critique complexity

Modality	Issue categories	n	HTML distance	SSIM	Realization
Comment	1	21	0.051	0.836	0.574
Comment	2–3	80	0.128	0.678	0.722
Comment	4+	51	0.142	0.688	0.726
Grounded critique	1	45	0.051	0.773	0.620
Grounded critique	2–3	95	0.096	0.715	0.702
Grounded critique	4+	41	0.119	0.674	0.773

4.6 Preference-model validation and transfer

Ranking prediction was difficult under description-grouped evaluation. The strongest model, histogram gradient boosting on HTML structural features, achieved 0.537 accuracy (95% CI [0.507, 0.566]) and AUC 0.531. HTML logistic regression achieved 0.531 accuracy, HTML random forest 0.529, and global visual random forest 0.529. Several alternatives were indistinguishable from chance, and no confidence interval established a large margin above 0.5. Figure 7 and Table 11 compare the leading configurations.

After exact-image overlap was removed, the global visual random forest produced mean after-preference probabilities of 0.501 for comment edits, 0.501 for grounded edits, and 0.507 for direct designer edits. None differed from 0.5 by a Wilcoxon test, and their predicted win rates were 49.3%, 48.1%, and 44.1%. The strongest HTML model gave 0.486 for comments and 0.493 for grounded critique. The comment probability was slightly below 0.5 ($p = 0.010$), while the grounded result was not different from 0.5 ($p = 0.255$). Because the cross-validated ranking models were weak, the near-neutral transfer results in Table 12 show that ranking data did not provide a dependable direct score of critique-conditioned edit quality.

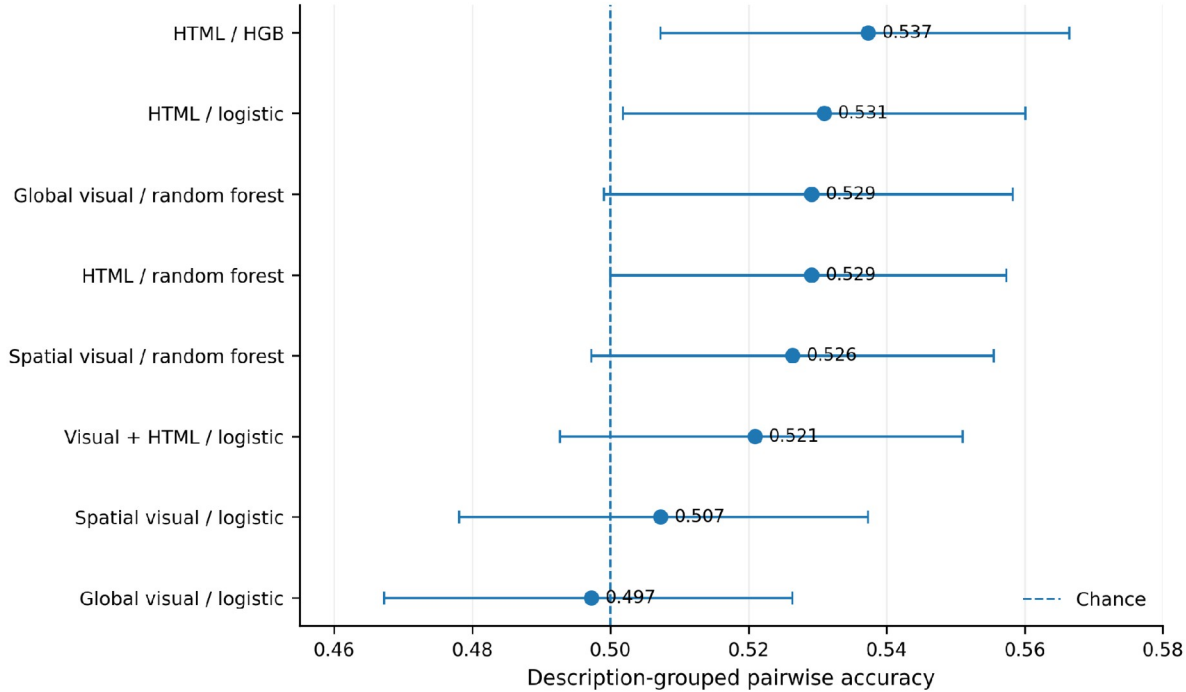


Figure 7. Description-grouped pairwise accuracy for preference models. Intervals are bootstrap 95% confidence intervals; the dashed line marks chance accuracy.

Table 11. Description-grouped preference-model performance

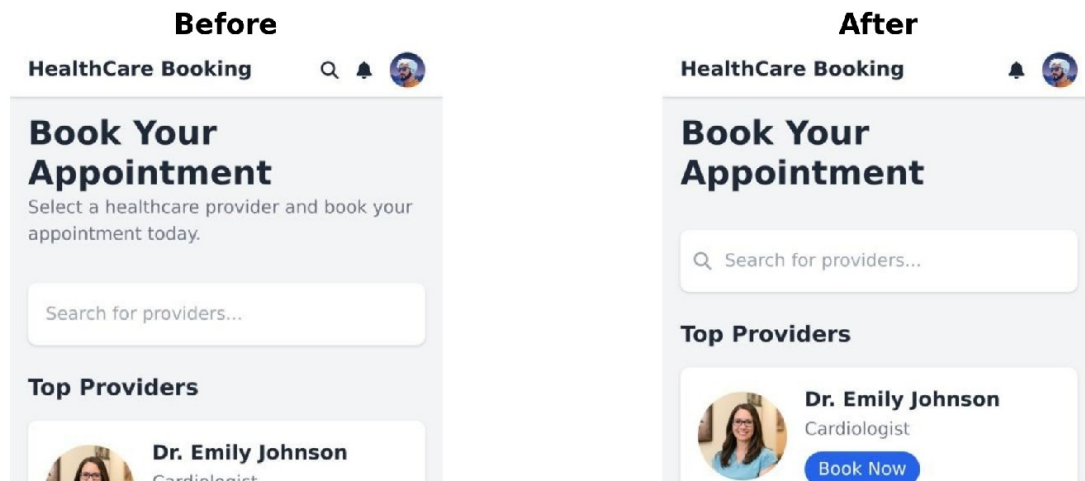
Feature/model	Accuracy	95% CI	AUC	Brier	Log loss
HTML / histogram gradient boosting	0.537	[0.507, 0.566]	0.531	0.265	0.727
HTML / logistic regression	0.531	[0.502, 0.560]	0.531	0.299	0.850
HTML / random forest	0.529	[0.500, 0.557]	0.547	0.248	0.690
Global visual / random forest	0.529	[0.499, 0.558]	0.535	0.251	0.695
Spatial visual / random forest	0.526	[0.497, 0.556]	0.534	0.250	0.693
Visual + HTML / logistic regression	0.521	[0.493, 0.551]	0.530	0.373	1.407
Spatial visual / logistic regression	0.507	[0.478, 0.537]	0.522	0.361	1.303
Global visual / logistic regression	0.497	[0.467, 0.526]	0.502	0.285	0.796

Table 12. Overlap-free ranking-transfer results

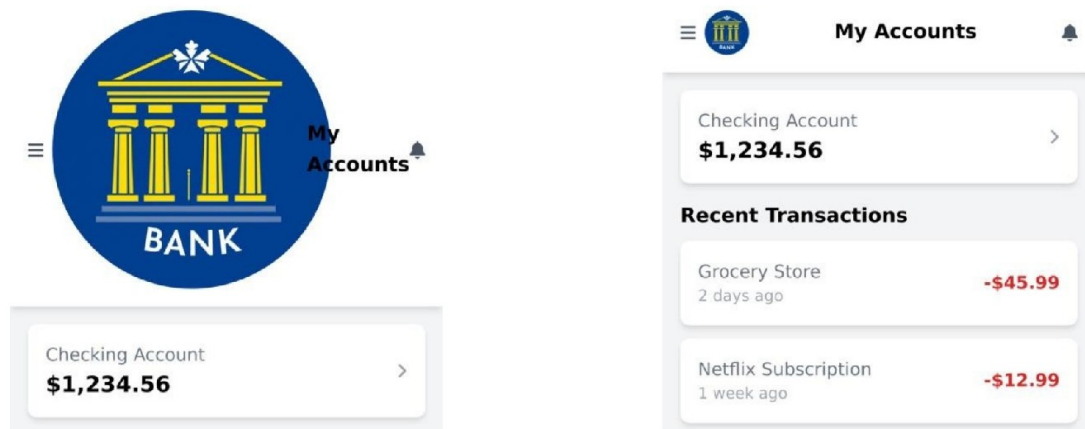
Modality	Scorer	n	Mean P(after)	95% CI	Win rate	Wilcoxon p	Binomial p
Comment	Visual	152	0.501	[0.494, 0.508]	49.3%	0.746	0.935
Grounded critique	Visual	181	0.501	[0.494, 0.509]	48.1%	0.970	0.656
Direct edit	Visual	68	0.507	[0.494, 0.519]	44.1%	0.460	0.396
Comment	HTML	152	0.486	[0.476, 0.496]	47.4%	0.010	0.570
Grounded critique	HTML	181	0.493	[0.483, 0.502]	51.4%	0.255	0.766

4.7 Qualitative inspection

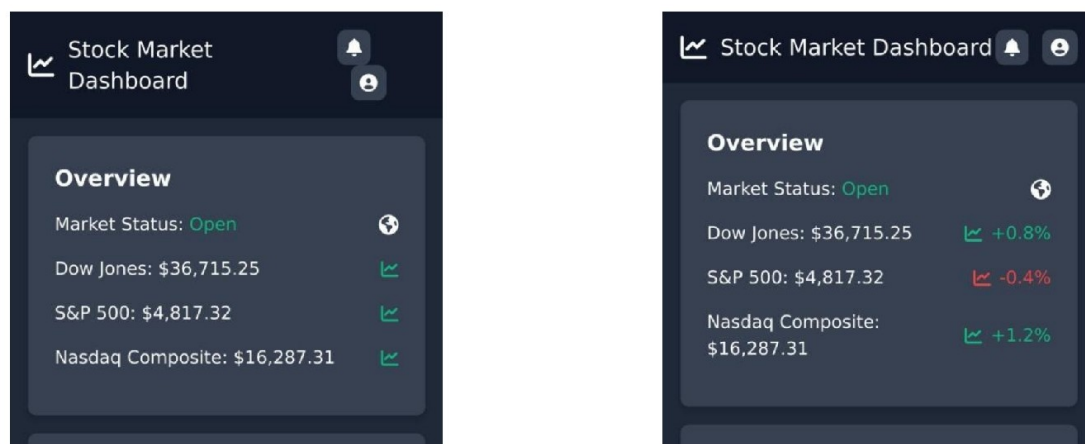
Figure 8 shows three representative transformations. In the appointment page, the search icon moved from the top navigation into the search field and vertical spacing was redistributed; the visible text remained unchanged while the HTML distance was 0.119 and SSIM was 0.596. In the banking page, a dominant logo was reduced and account information moved upward, producing a substantial visual correction from a small normalized source distance of 0.011. The market dashboard aligned the header controls and added directional indicators, but its lower chart content weakened. These cases show why a single similarity metric is insufficient: a low source distance can correct a major layout fault, and a locally successful edit can coexist with a regression elsewhere on the page.



Comment: place the search icon inside the field and redistribute the vertical layout.



Grounded critique: reduce the logo, move the menu, and consolidate account information.



Grounded critique: align header icons and add market direction; the chart content regressed.

Figure 8. Before-and-after pages. The first two rows show targeted layout improvements; the third shows successful local alignment alongside a content regression.

5. Discussion

The experiments establish that critique-conditioned HTML editing is predominantly a source-preserving operation. Mean normalized HTML distance remained near one tenth, median text Jaccard was perfect, and utility-class changes appeared in more than 93% of rows. This pattern matches the practical structure of Tailwind-style interface code: a small set of class replacements can adjust spacing, alignment, typography, color, and component geometry without replacing the document. The grounded subset changed fewer source tokens than the ordinary comment subset while maintaining comparable SSIM and realization. Region-linked feedback therefore concentrated edits without reducing measurable implementation coverage.

Visual continuity is not the same as minimal code change. The direct designer benchmark had an SSIM distribution similar to both model-driven modalities, indicating that a moderate visual shift is compatible with legitimate design work. At the same time, the modest code-visual correlations demonstrate that source difference cannot stand in for visual effect. This extends the evaluation logic of screenshot-to-code benchmarks such as Design2Code [6]. Reconstruction benchmarks ask whether generated code matches a target image; critique-conditioned editing additionally asks whether unaffected content was preserved and collateral change was limited. Both code and render evidence are necessary.

The category results separate straightforward implementation requests from abstract design judgments. Spacing and component styling map directly to utility classes and DOM structure, so their high realization rates are expected. Typography and color also have recognizable source signatures. Hierarchy and accessibility are harder because they can require coordinated changes in scale, whitespace, grouping, content order, target size, labeling, or interaction behavior. Ferret-UI's emphasis on grounded screen understanding [7], Screen2Words' connection between screens and language [3], and UICrit's region-linked critiques [15] all suggest richer evaluators that combine localization, semantics, and interaction state. Source rules alone are not sufficient for these categories.

The ranking experiment provides an important negative result. The strongest description-grouped model reached only 53.7% accuracy, and overlap-free transfer probabilities clustered around 0.5. Pairwise preference methods are powerful when labels are consistent and the representation captures the deciding features [10]-[12], but design choices can depend on task-specific rationale, personal taste, and local defects that disappear in global descriptors. UIClip shows the benefit of large-scale UI-specific learning [13], while critique-oriented work shows the value of retaining the reason and location associated with a judgment [8], [15], [16]. A practical training pipeline should preserve both pairwise choices and critique rationales rather than using one as a substitute for the other.

For front-end systems, the results imply a multi-objective editing loop. The model should first localize the critique, generate the smallest coherent source change, render the page, and evaluate both requested and unaffected regions. A code-distance budget can discourage whole-page rewrites, while SSIM or localized visual comparison can detect broad drift. Semantic checks should verify content preservation, headings, labels, interactive controls, and accessibility attributes. When a request spans several issue categories, the system should decompose it into explicit checks rather than rely on one final quality score. The complexity results in Figure 6 show that broader critiques already produce larger edits and lower visual similarity; explicit decomposition would make those tradeoffs visible to the designer.

Several boundaries define the interpretation of the results. The pages are synthetic mobile interfaces rendered at a fixed portrait viewport, so the findings do not measure responsive behavior across screen sizes. Direct designer rows contain screenshots but not edited HTML, which limits code-level comparison with that modality. The realization taxonomy uses lexical cues and observable HTML changes; it captures consistent implementation evidence but not every valid design response. Static screenshots do not test event handlers, keyboard flow, focus order, or data-dependent states. The ranking models use compact engineered descriptors rather than a large UI foundation model. These boundaries explain why semantic realization and predicted preference remain separate outcomes.

The corpus also contains repeated images in the ranking subset and exact overlap with critique subsets. Hash-based exclusion was therefore essential. Exact-image overlap would let a preference model associate memorized screenshots with labels and inflate transfer performance. Grouping folds by normalized description blocked a second leakage path created by pages from the same application request sharing layout and content patterns. These controls produced conservative but interpretable preference results and should be standard in UI datasets where multiple variants of the same brief are compared.

6. Conclusions

A full evaluation of the Apple ML-RLDF corpus shows that language-model editing of existing UI code generally preserves page identity while applying measurable source changes. Comment-conditioned transformations changed more HTML than region-grounded transformations, yet both maintained mean SSIM above 0.70 and achieved a detectable realization rate near 0.70. Their visual-change distributions did not differ from direct designer edits. Utility-class changes dominated, visible text was usually preserved, and code distance correlated only modestly with visual change. Spacing and component requests were implemented consistently, while hierarchy and accessibility requests remained difficult to verify

from source changes alone. Designer ranking labels produced only weak grouped prediction and no dependable transfer advantage for the after state. Effective evaluation of critique-driven front-end editing therefore requires separate evidence for code locality, rendered continuity, semantic realization, and human preference.

References

- [1] Jason Kuhn, Yushan Chen, and Evelyn Chan, "AI-Driven Mobile UI Pattern Recognition and Design Topic Mining on RICO: Semantic Clustering and Screenshot-Based Topic Classification", *JACS*, vol. 4, no. 5, pp. 67–83, May 2024, doi: 10.69987/JACS.2024.40506.
- [2] B. Deka, Z. Huang, C. Franzen, J. Hibsman, D. Afegan, Y. Li, J. Nichols, and R. Kumar, "Rico: A Mobile App Dataset for Building Data-Driven Design Applications," in *Proc. ACM UIST*, pp. 845–854, 2017, doi: 10.1145/3126594.3126651.
- [3] B. Wang, G. Li, X. Zhou, Z. Chen, T. Grossman, and Y. Li, "Screen2Words: Automatic Mobile UI Summarization with Multimodal Learning," in *Proc. ACM UIST*, 2021, doi: 10.1145/3472749.3474765.
- [4] H. Laurençon, L. Tronchon, and V. Sanh, "Unlocking the Conversion of Web Screenshots into HTML Code with the WebSight Dataset," *arXiv:2403.09029*, 2024.
- [5] J. Wu, E. Schoop, A. Leung, T. Barik, J. P. Bigham, and J. Nichols, "UICoder: Finetuning Large Language Models to Generate User Interface Code through Automated Feedback," *arXiv:2406.07739*, 2024.
- [6] C. Si, Y. Zhang, R. Li, Z. Yang, R. Liu, and D. Yang, "Design2Code: Benchmarking Multimodal Code Generation for Automated Front-End Engineering," in *Proc. NAACL-HLT*, pp. 3956–3974, 2025, doi: 10.18653/v1/2025.naacl-long.199.
- [7] Yushan Chen and Evelyn Chan, "Multimodal UI Representation Learning: Ablation of Screenshot, Wireframe, and View-Hierarchy Proxies on an Uploaded 168-Screen Dataset", *JACS*, vol. 3, no. 1, pp. 1–15, Jan. 2023, doi: 10.69987/JACS.2023.30101.
- [8] J. Wu, A. Swearngin, A. K. Vajjala, A. Leung, J. Nichols, and T. Barik, "Improving User Interface Generation Models from Designer Feedback," in *Proc. ACM CHI Conf. Human Factors in Computing Systems*, 2026, doi: 10.1145/3772318.3791567.
- [9] Apple, "Reinforcement Learning from Designer Feedback: Improving User Interface Generation Models from Designer Feedback," *GitHub repository*, 2026.
- [10] L. Ouyang et al., "Training Language Models to Follow Instructions with Human Feedback," in *Advances in Neural Information Processing Systems*, vol. 35, pp. 27730–27744, 2022.
- [11] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct Preference Optimization: Your Language Model Is Secretly a Reward Model," in *Advances in Neural Information Processing Systems*, vol. 36, pp. 53728–53741, 2023.
- [12] J. Hong, N. Lee, and J. Thorne, "ORPO: Monolithic Preference Optimization without Reference Model," in *Proc. EMNLP*, pp. 11170–11189, 2024.
- [13] J. Wu, Y.-H. Peng, X. Y. A. Li, A. Swearngin, J. P. Bigham, and J. Nichols, "UIClip: A Data-Driven Model for Assessing User Interface Design," in *Proc. ACM UIST*, Art. 45, 2024, doi: 10.1145/3654777.3676408.
- [14] A. Radford et al., "Learning Transferable Visual Models from Natural Language Supervision," in *Proc. ICML*, pp. 8748–8763, 2021.
- [15] P. Duan, C.-Y. Chen, G. Li, B. Hartmann, and Y. Li, "UICrit: Enhancing Automated Design Evaluation with a UI Critique Dataset," in *Proc. 37th Annu. ACM Symp. User Interface Softw. Technol.*, Art. 46, 2024, doi: 10.1145/3654777.3676381.
- [16] Y. Li, S. Lu, and L. Zhao, "LLM-as-Design-Critic: Aligning AI-Generated UI Feedback with Human Graphic Design Judgment," *Int. J. Graph. Des.*, vol. 3, no. 1, pp. 196–215, May 2025, doi: 10.51903/ijgd.v3i1.3661.
- [17] Y. Chen and M. Li, "From Hand-Drawn Sketches to Interactive Web Prototypes: A Reproducible Vision-Language Approach with Structural and Visual Consistency Evaluation," *J. Technol. Informatics Eng.*, vol. 4, no. 2, pp. 364–384, Aug. 2025, doi: 10.51903/jtie.v4i2.490.
- [18] Q. Wu, S. Meng, and J. Zhao, "Text-Grounded LLM-Assisted Design Rationale Interfaces: Turning Advertising Layout Metadata into Explainable UI/UX Decision Cards," *Int. J. Graph. Des.*, vol. 3, no. 1, pp. 216–240, May 2025, doi: 10.51903/ijgd.v3i1.3713.
- [19] J. Mu, Y. Lu, and E. Hwang, "Structured Visual Brief Interfaces for Advertising Design: A UI/UX Framework for Turning Creative Intentions into Designer-Editable Graphic Design Cards," *Int. J. Graph. Des.*, vol. 4, no. 1, pp. 192–208, Apr. 2026, doi: 10.51903/ijgd.v4i1.3702.
- [20] H. Xu, Y. Chen, and A. Med, "Automatic Detection and Explanation of Dark Patterns from Interface Microcopy: Empirical Comparison of BERT-Style Encoders, RoBERTa-Style Encoders, and LLM-Style Decoders on the ec-darkpattern Dataset," *J. Technol. Informatics Eng.*, vol. 4, no. 3, pp. 590–612, Dec. 2025, doi: 10.51903/jtie.v4i3.491.

- [21] W. Su, H. Rao, and E. Ma, "Privacy and Data-Integrity Risk Cards for LLM Agents: A UI/UX Design Framework for Secure Human Oversight under Prompt-Injection Attacks," *Int. J. Graph. Des.*, vol. 4, no. 1, pp. 186-191, Apr. 2026, doi: 10.51903/ijgd.v4i1.3699.
- [22] B. Hui et al., "Qwen2.5-Coder Technical Report," arXiv:2409.12186, 2024.
- [23] V. I. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals," *Sov. Phys. Dokl.*, vol. 10, no. 8, pp. 707-710, 1966.
- [24] K. Zhang and D. Shasha, "Simple Fast Algorithms for the Editing Distance between Trees and Related Problems," *SIAM J. Comput.*, vol. 18, no. 6, pp. 1245-1262, 1989.
- [25] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600-612, 2004.
- [26] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.
- [27] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679-698, 1986.
- [28] R. A. Bradley and M. E. Terry, "Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons," *Biometrika*, vol. 39, nos. 3-4, pp. 324-345, 1952.
- [29] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825-2830, 2011.
- [30] B. Efron, "Bootstrap Methods: Another Look at the Jackknife," *Ann. Statist.*, vol. 7, no. 1, pp. 1-26, 1979.
- [31] H. B. Mann and D. R. Whitney, "On a Test of Whether One of Two Random Variables Is Stochastically Larger than the Other," *Ann. Math. Statist.*, vol. 18, no. 1, pp. 50-60, 1947.
- [32] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80-83, 1945.
- [33] C. Spearman, "The Proof and Measurement of Association between Two Things," *Amer. J. Psychol.*, vol. 15, no. 1, pp. 72-101, 1904.
- [34] P. Virtanen et al., "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261-272, 2020.