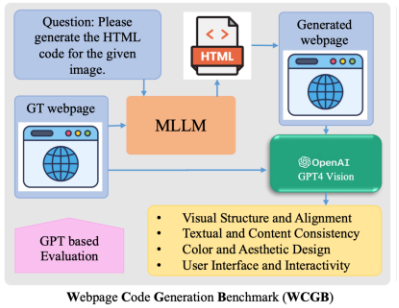
Interaction2Code: Benchmarking MLLM-based Interactive Webpage Code Generation from Interactive Prototyping

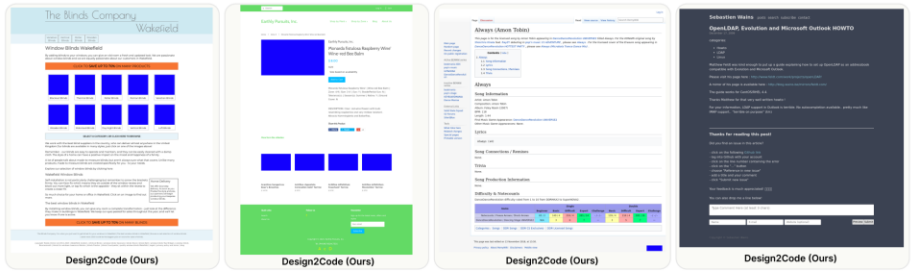
Jingyu Xiao, Yuxuan Wan, Yintong Huo, Zixin Wang, Xinyi Xu, Wenxuan Wang, Zhiyao Xu, Yuhang Wang, Michael R. Lyu

MLLMs have been widely used in UI2Code task

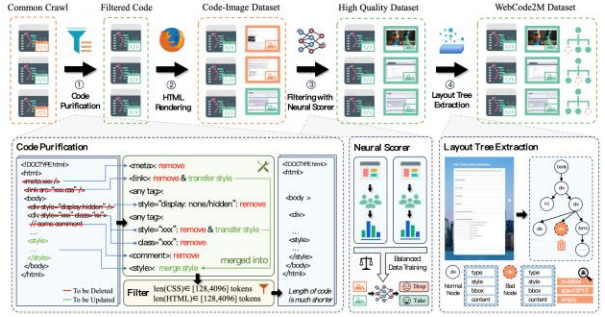
- Converting webpage designs into functional UI code is labor-intensive in web development.
- Multimodal Large Language Models (MLLMs) have demonstrated remarkable capabilities in visually rich code generation tasks.
- MLLM-based UI code generation: benchmarks and methods.



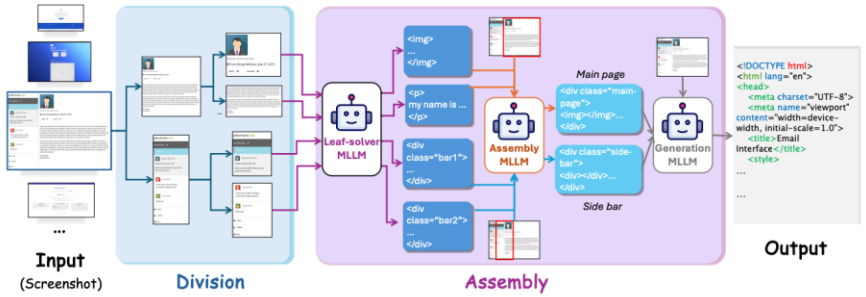
[NeurIPS 2024] Web2Code



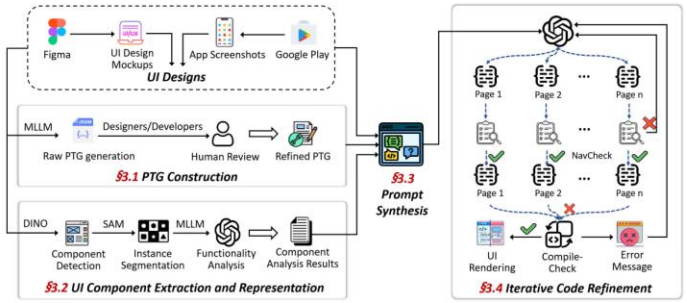
[NAACL 2025] Design2Code



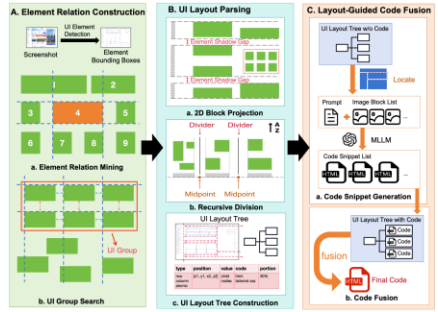
[WWW 2025] Webcode2M



[FSE 2025] DCGen



[FSE 2025] DeclarUI



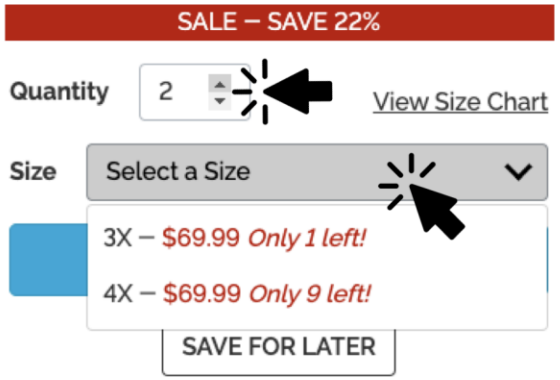
[ISSTA 2025] LayoutCoder

Existing Research Only Focuses the Static Webpage

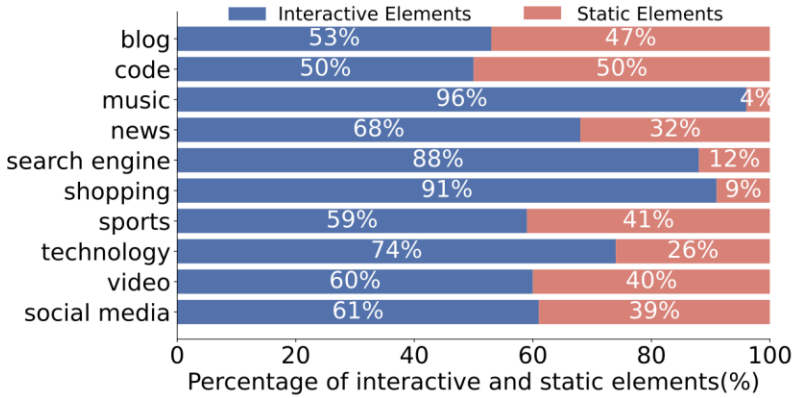
- Existing research ignores the dynamic interactive properties and functionality.

Benchmark	Real World	Failure Annotation	Interactive
WebSight [37]	✗	✗	✗
Vision2UI [25]	✓	✗	✗
Design2Code [22]	✓	✗	✗
DesignBench [26]	✓	✗	✗
Interaction2Code (Ours)	✓	✓	✓

- Interactive elements account for a large proportion of the webpage in real-world software practices..



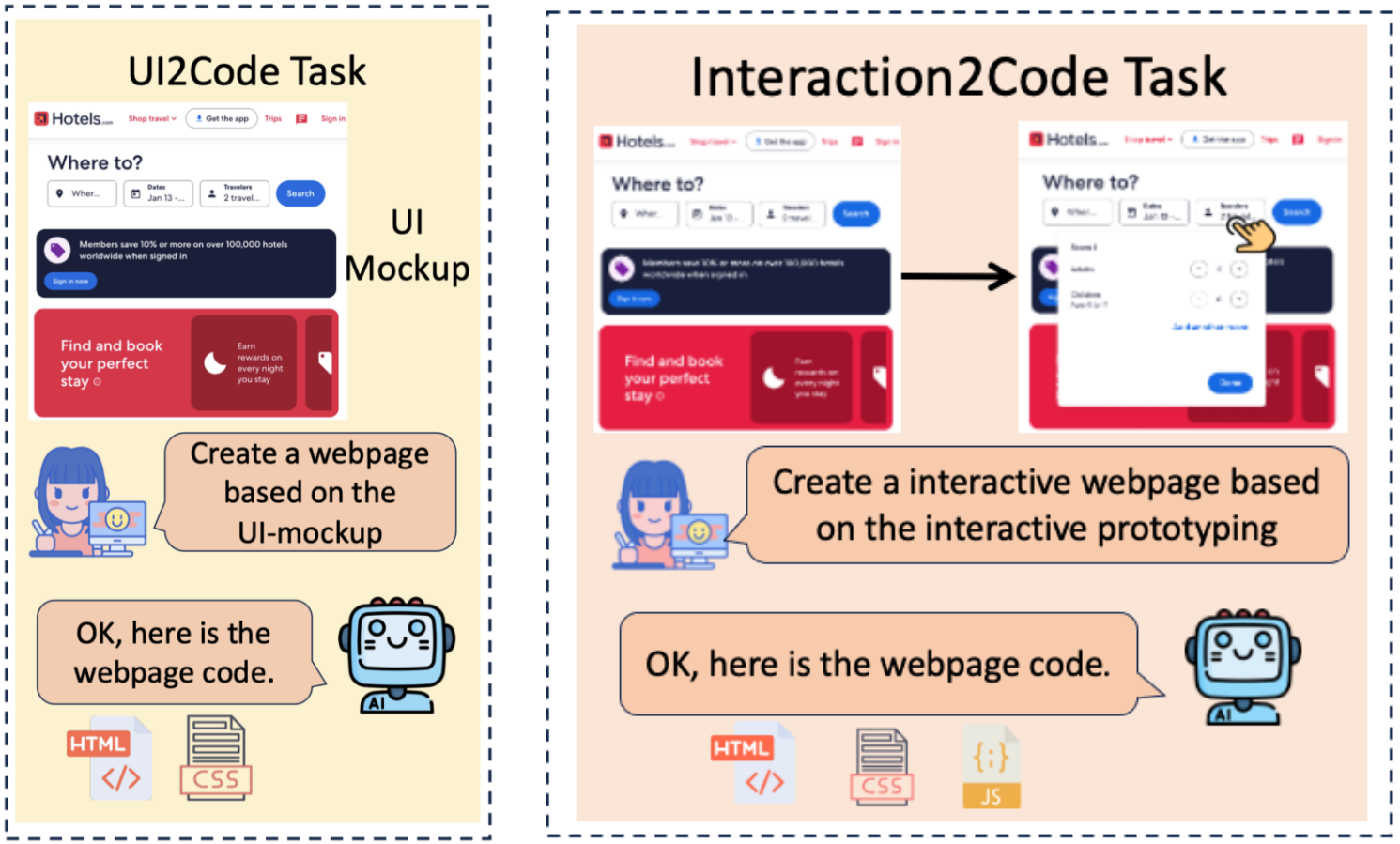
(a) Interactive elements.



(b) Interactive and static ratio.

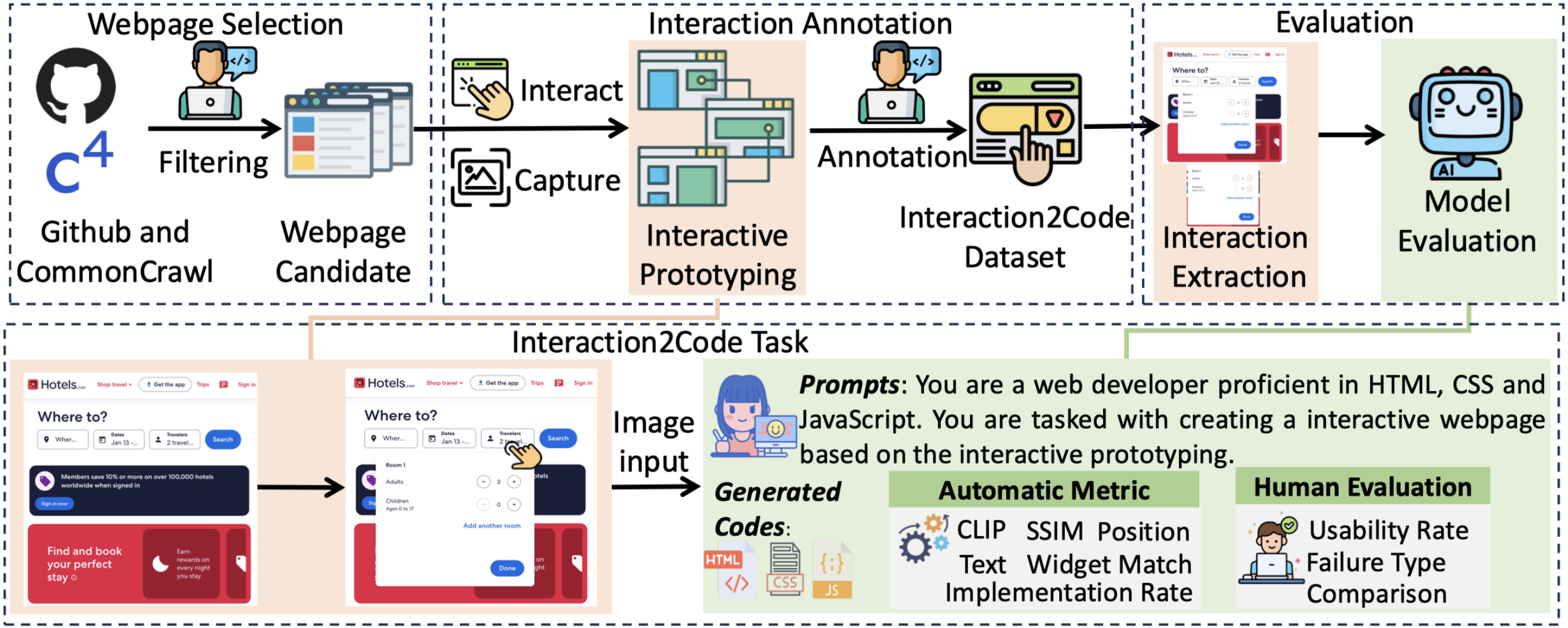
Task Definition

- UI2Code: it takes the static UI-Mockup S as input and generates a static webpage.
- **Interaction2Code**: it takes the interactive prototyping as input and generates an interactive webpage.



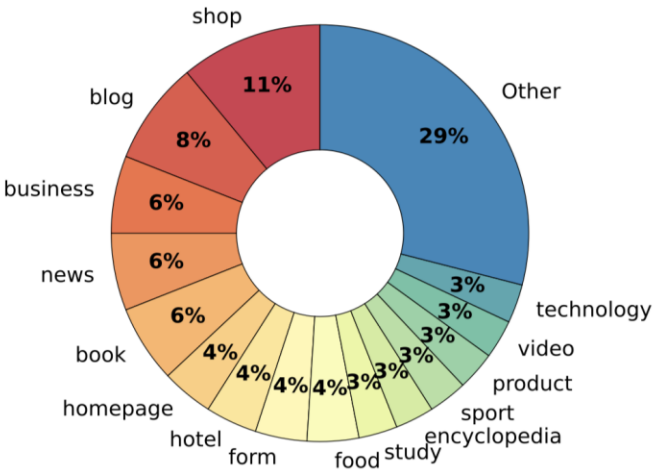
Interactive Prototyping allows users to simulate interactions and navigate through the interface to test usability and functionality before full development. An interactive behavior is represented as an interactive prototype $IP = \{ S_o , S_I \}$, where S_o is the UIMockup of original webpage and S_I is the UI-Mockup after the interaction I.

Benchmark Construction

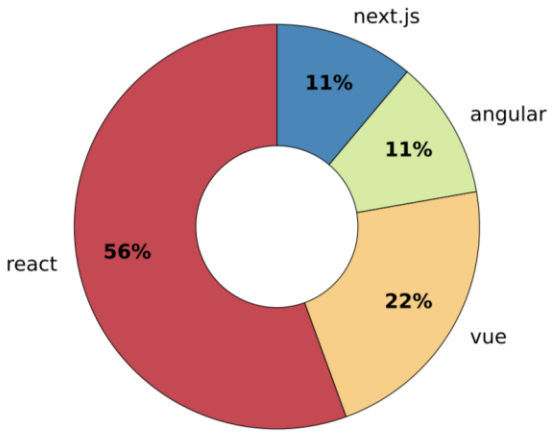


Data Statistics and Diversity

- Topic and Framework Distribution. Our benchmark covers a diverse range of web topics with more than 15 types, including business, shop, technology, entertainment, and so on. It includes mainstream front-end open source frameworks such as react, next.js, vue, and angular.
- Interaction Type Distributions: 23 tag categories and 8 visual categories.



(a) Topic.



(b) Framework.

Fig. 4. Topic and framework distribution.

Tag Categories				Visual Categories	
Element	Number	Element	Number	Type	Number
button	235	summary	15	text	162
input	52	form	13	new component	161
span	37	detail	12	color	85
link	36	video	11	position	45
select	35	area	9	switch	41
textarea	35	output	9	new page	37
option	31	datalist	8	new window	34
iframe	28	dialog	6	size	20
text	24	audio	5	-	-
progress	22	template	3	-	-
image	21	table	1	-	-
label	16	-	-	-	-

Metrics

- Visual Similarity: we use CLIP score to measure the visual similarity.
- Structure Similarity. SSIM (Structural Similarity Index Measure) score is applied to calculate the structure similarity.
- Text Similarity. We apply OCR tools to recognize the text in the webpages, and then use the BLEU score [45] to measure the text similarity between the two webpages.
- Widget Match. Widget match measures the widget-level consistency between the original UI and the generated UI. We calculate the Widget Similarity (WS) and Widget Match Rate (WMS) based on the method proposed by GUIPilot [46].
- Position Similarity. The position similarity between original interaction Io and generated interaction Ig is defined as
- Implement Rate (IR) measures the ratio of interactions successfully implemented by MLLM. An interaction is considered implemented if detectable by webdriver, and unimplemented otherwise.

$$IR = \frac{N(implemented)}{N(implemented)+N(unimplemented)}$$

- Usability Rate (UR). Human annotators are asked to interact with the generated webpage and judge the usability.

$$UR = \frac{N(usable)}{N(usable)+N(unusable)}$$

Metrics

- **Visual Metrics** (**Full page and Interaction Part**): CLIP, SSIM, Text (Bleu Score), Position and Widget Match (GUIPilot [1])
- **Functional Metrics** (**Interaction Part**): Implementation Rate, Usability Rate

Automatic Metric

CLIP SSIM Position
Text Widget Match
Implementation Rate

Human Evaluation

Usability Rate
Failure Type
Comparison

$$IR = \frac{N(implemented)}{N(implemented)+N(unimplemented)}$$

$$UR = \frac{N(usable)}{N(usable)+N(unusable)}$$

[1] Liu, Ruofan, et al. “GUIPilot: A Consistency-Based Mobile GUI Testing Approach for Detecting Application-Specific Bugs.” *ISSTA 2025*

Model Performance: Automatic Metrics

Limitation 1: The performance of the interactive part is lower than that of the full page, which is caused by the fact that the MLLMs do not pay attention to the interaction part enough.

Model	Prompt	Full Page					Interaction Part				
		CLIP	SSIM	Text	WS	WMR	CLIP	SSIM	Text	Position	IR
Qwen2.5-vl-3B-instruct	Direct	**0.3220**	**0.1932**	**0.1510**	**0.0203**	**0.3462**	**0.2100**	**0.1531**	0.0415	**0.2090**	**0.3449**
	CoT	0.2031	0.1085	0.0800	0.0185	0.3302	0.1219	0.0894	0.0352	0.1212	0.1979
	Mark	0.2752	0.1503	0.1200	0.0190	0.3418	0.1706	0.1188	**0.0514**	0.1706	0.2647
	Average	0.2668	0.1507	0.1170	0.0193	0.3394	0.1675	0.1204	0.0427	0.1669	0.2692
Qwen2.5-vl-7B-instruct	Direct	0.4169	0.2886	0.2519	0.0222	0.3887	0.3230	0.2177	0.0952	0.2529	0.4786
	CoT	0.3895	0.2529	0.2207	**0.0286**	**0.3909**	0.2806	0.1981	0.0744	0.2259	0.4305
	Mark	**0.4586**	**0.3282**	**0.2703**	0.0234	0.3666	**0.3541**	**0.2468**	**0.1348**	**0.2798**	**0.5267**
	Average	0.4217	0.2899	0.2477	0.0247	0.3821	0.3192	0.2209	0.1015	0.2529	0.4786
Qwen2.5-vl-72B-instruct	Direct	0.6430	0.4234	0.4197	0.0371	0.4285	0.4624	0.3207	0.2450	0.3950	0.6524
	CoT	0.6335	**0.4785**	0.4585	0.0369	0.4395	**0.5090**	**0.3692**	0.2376	0.4385	**0.7380**
	Mark	**0.6954**	0.4569	**0.4586**	**0.0401**	**0.4430**	0.4992	0.3621	**0.2995**	**0.4541**	0.7112
	Average	0.6573	0.4529	0.4456	0.0380	0.4370	0.4902	0.3507	0.2607	0.4292	0.7005
Gemini-1.5-flash	Direct	0.5967	0.4526	0.4749	0.0330	**0.4964**	0.4737	0.3616	0.2809	0.4320	0.6738
	CoT	0.6166	0.4810	0.4775	**0.0332**	0.4783	0.5093	0.3854	0.3217	0.4511	0.7112
	Mark	**0.6321**	**0.4946**	**0.4878**	0.0322	0.4826	**0.5194**	**0.3898**	**0.3454**	**0.4612**	**0.7326**
	Average	0.6151	0.4761	0.4801	0.0328	0.4858	0.5008	0.3789	0.3160	0.4481	0.7059
GPT-4o	Direct	0.7114	0.5277	**0.5147**	**0.0440**	0.4645	0.5605	0.4149	0.3590	0.4888	0.7754
	CoT	0.6905	0.4962	0.4761	0.0435	**0.4674**	0.5234	0.4013	0.3663	0.4668	0.7273
	Mark	**0.7160**	**0.5539**	0.5112	0.0414	0.4418	**0.5955**	**0.4488**	**0.4474**	**0.5225**	**0.8128**
	Average	0.7059	0.5259	0.5007	0.0430	0.4579	0.5598	0.4217	0.3909	0.4927	0.7718
Claude-3.5-Sonnet	Direct	0.7172	**0.5318**	**0.6003**	**0.0533**	**0.5284**	0.5674	0.4209	0.3833	0.5123	0.7914
	CoT	0.6961	0.5110	0.5603	0.0451	0.5099	0.5606	0.4005	0.3662	0.5085	0.7727
	Mark	**0.7258**	0.5299	0.5899	0.0486	0.5111	**0.5944**	**0.4282**	**0.4319**	**0.5149**	**0.7968**
	Average	0.7130	0.5242	0.5835	0.0490	0.5165	0.5742	0.4165	0.3938	0.5119	0.7870

Model Performance: Automatic Metrics

Improvement 1: Interactive element highlighting. We propose Chain-of-Thought (CoT) and Mark prompts to force models to focus on the interaction.

Chain-of-Thought (CoT) Prompt

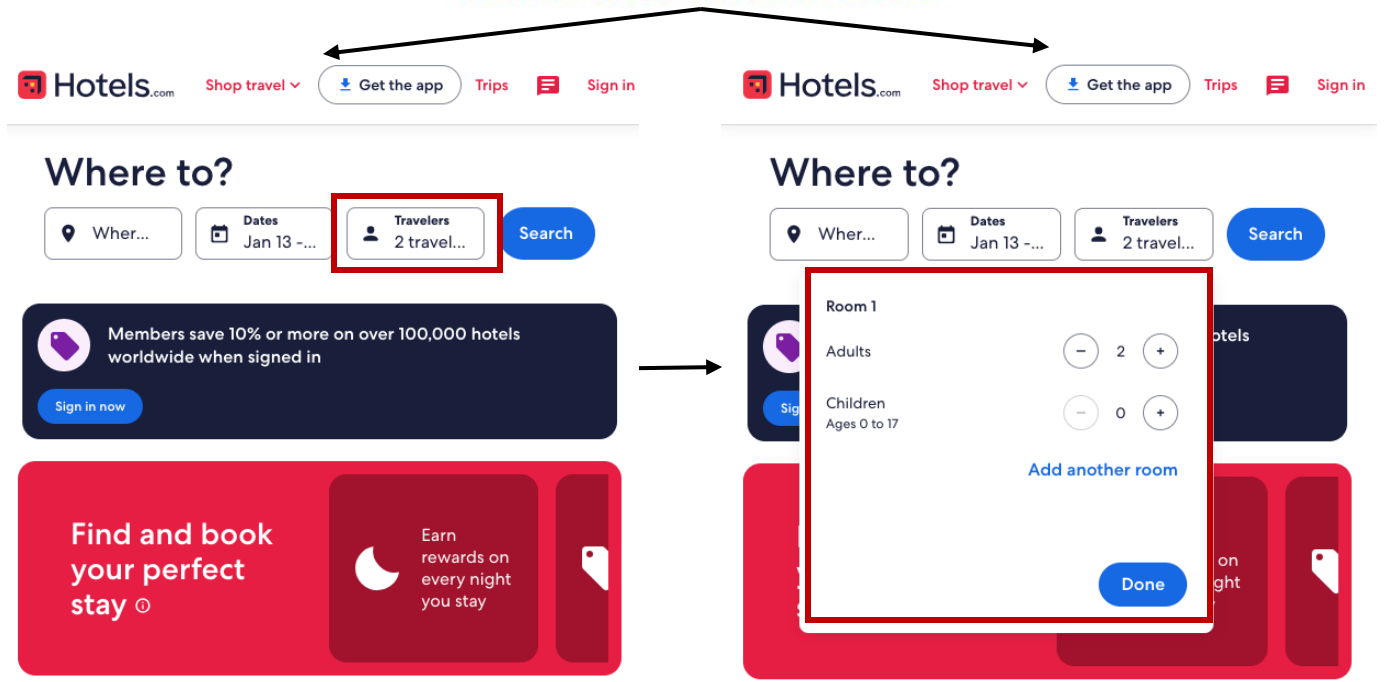
[Instruction]
[Requirements]
[CoT]
You should think step by step:
Step 1: Understand the interaction effects by analyzing the difference between the first and other screenshots.
Step 2: Locate interactive elements.
Step 3: Implement the interaction: the interaction function should cause the difference you analyze in Step 1 and be implemented the interactive element you locate from step 2.

Combine HTML, CSS and JavaScript codes into one file and respond the codes only:

Mark Prompt

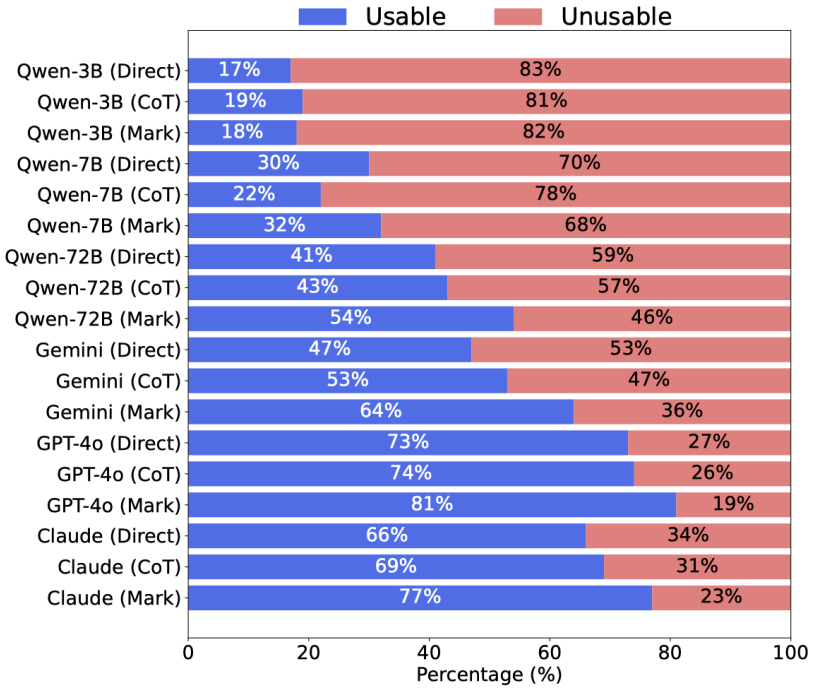
[Instruction]
[Requirements]
[Mark]
In the first screenshot, the interactive elements are highlighted with red bounding boxes. In other screenshots, the interaction effects are highlighted with red bounding boxes. Pay attention to the position of the red bounding boxes, which mark the position of interaction. But do not generate the red bounding box, which is just used for marking the interaction area.

Combine HTML, CSS and JavaScript codes into one file and respond the codes only:

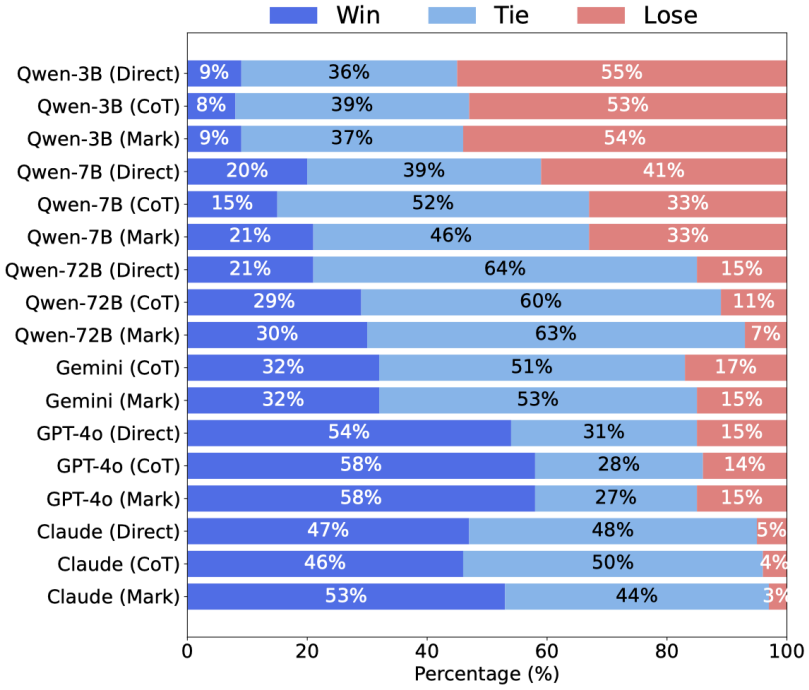


Model Performance: Human Evaluation

- These trend of human evaluation results are consistent with automatic metrics, confirming the validity of our automatic evaluation metrics.



(a) Usability evaluation.



(b) Pairwise comparision.

Fig. 6. Human evaluation, a higher usable rate indicates better functionality and a higher win rate indicates better quality.

Failure Type Analysis

- 10 types of Failure modes: the most critical failures are “Interactive element missing”, “Wrong function”, “No interaction” and “Effect on wrong element”.

Failure Object	Failure Type	Content	Function	User Experience	Usability Rate
Interactive element	(a) Interactive element missing	🔴	🔴	🔴	0%
	(b) No interaction	🟡	🔴	🔴	6.93%
	(c) Wrong interactive element	🟡	🟡	🔴	92.31%
	(d) Wrong type of interactive element	🟡	🟡	🔴	96.82%
	(e) Wrong position of interactive element	🟡	🟡	🔴	98.41%
Interaction effects	(f) Wrong position after interaction	🟡	🟡	🔴	96.17%
	(g) Wrong type of interaction effects	🟡	🟡	🔴	57.14%
	(h) Effect on wrong element	🟡	🟡	🔴	44.44%
	(i) Partial Implementation	🟡	🟡	🔴	89.20%
	(j) Wrong function	🔴	🔴	🔴	0%

Failure Type Analysis

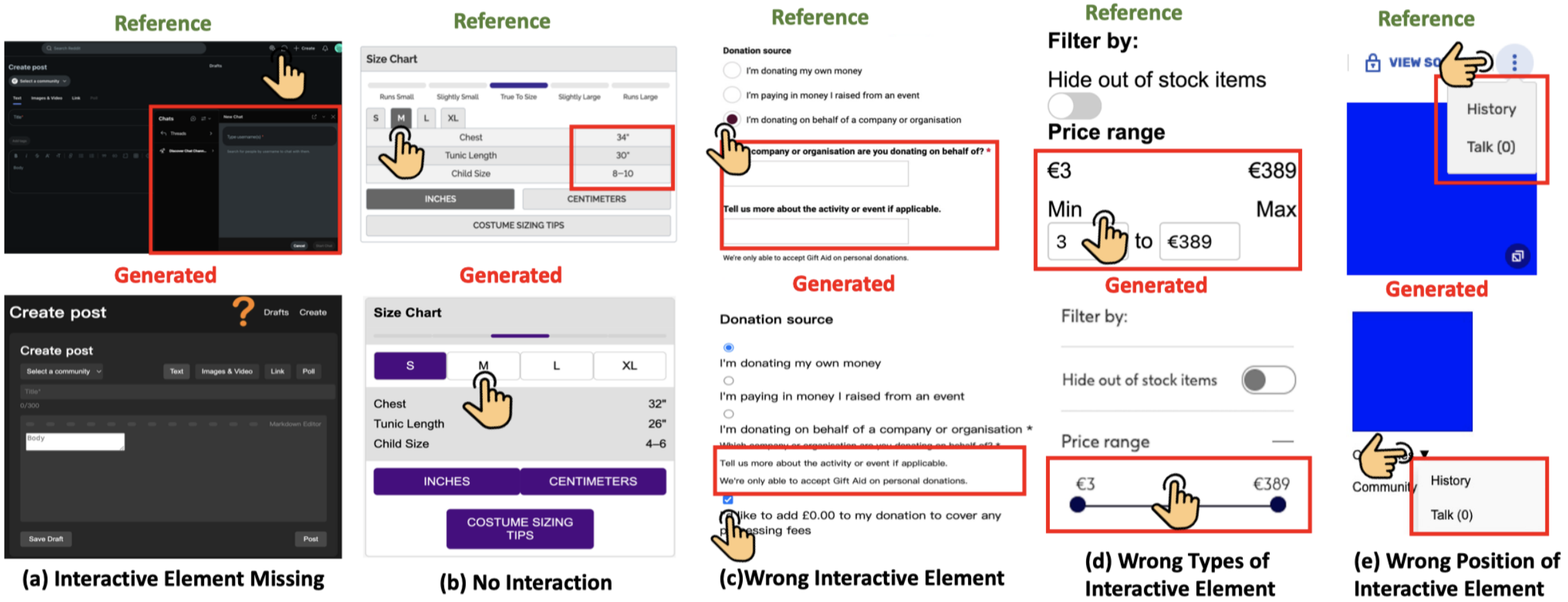
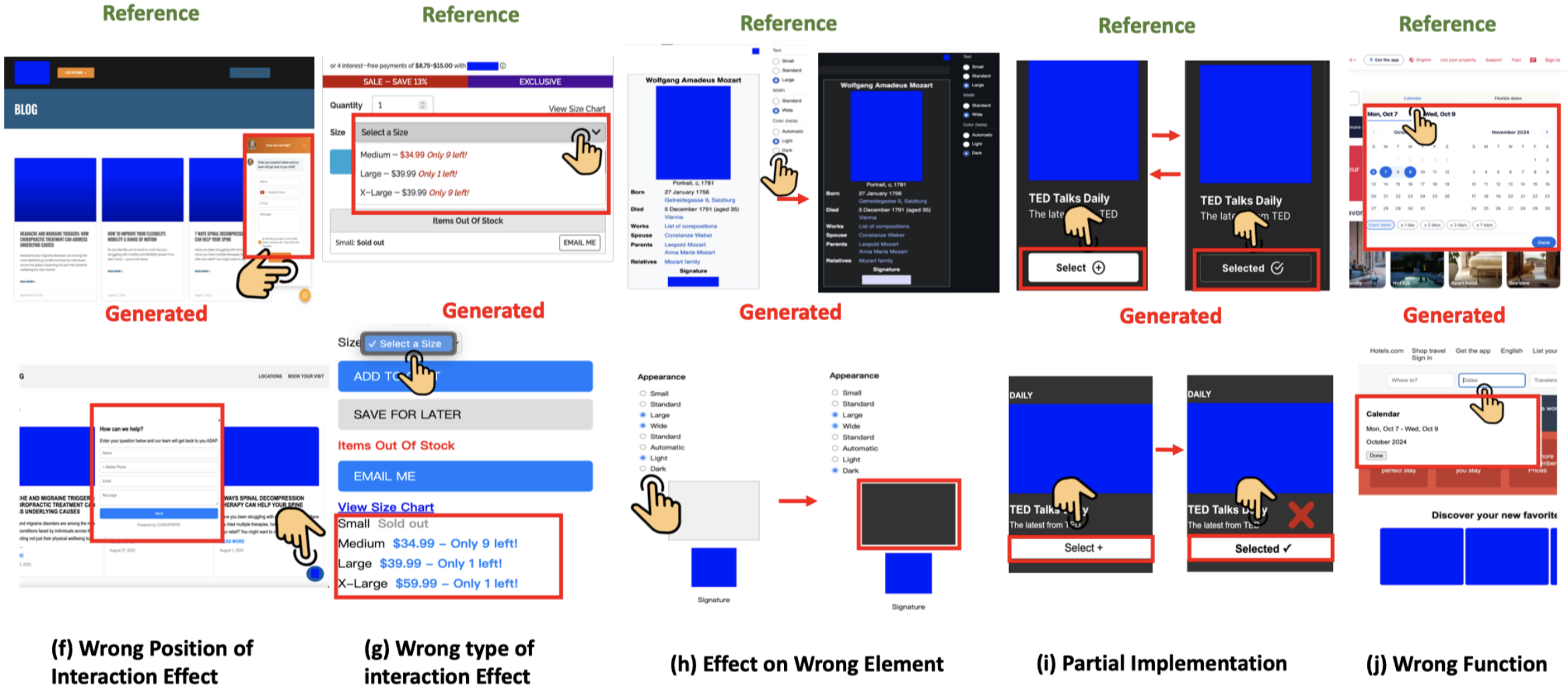


Fig. 7. Failure on interactive elements.

Failure Type Analysis



(f) Wrong Position of Interaction Effect

(g) Wrong type of interaction Effect

(h) Effect on Wrong Element

(i) Partial Implementation

(j) Wrong Function

Fig. 8. Failure of interaction effects.

Failure Type Analysis

Limitation 2: MLLMs are prone to make 10 types of failure. The main failure modes are “No interaction”, “Partial implementation”, “Interactive element missing”, and “Wrong function”.

Improvement 2: Failure-aware Prompt (FAP). Based on failure types, we propose FAP to stimulate the self-criticism ability of MLLMs, thereby avoiding problems that may occur in the Interaction-to-Code task.

Failure-aware Prompt (FAP)

[Instruction]
[Requirements]
[Failure Type Example]
There are ten types of errors you should avoid:
(a) Interactive element missing: MLLMs do not generate interactive elements...
(b) No interaction: There is no interaction in the generated webpage...
(c) Wrong interactive element: MLLMs implement the interactive function on the wrong element...
(d) (e) (f) (g) (h) (i) (j)...

Combine HTML, CSS and JavaScript codes into one file and respond the codes only:

Model	Method	CLIP	SSIM	Text	Position
Gemini 1.5-flash	Direct	0.5403	0.4494	0.3602	0.5802
	FAP	0.5886	0.4584	0.4394	0.6032
	Δ	↑ 0.0483	↑ 0.0090	↑ 0.0792	↑ 0.0230
GPT 4o	Direct	0.5700	0.4891	0.3652	0.5803
	FAP	0.6072	0.5405	0.4580	0.6452
	Δ	↑ 0.0372	↑ 0.0514	↑ 0.0928	↑ 0.0649
Claude 3.5 Sonnet	Direct	0.4582	0.3771	0.3086	0.4927
	FAP	0.4921	0.4035	0.3822	0.5154
	Δ	↑ 0.0339	↑ 0.0264	↑ 0.0736	↑ 0.0227
Qwen2.5 vl-72B instruct	Direct	0.4741	0.3612	0.3275	0.5022
	FAP	0.5144	0.3750	0.3286	0.5376
	Δ	↑ 0.0403	↑ 0.0138	↑ 0.0011	↑ 0.0354

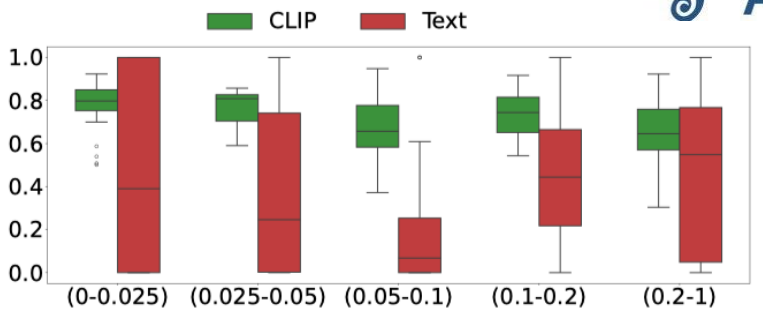
Visual Saliency vs Performance

- **Visual Saliency:** The proportion of interactive area to the total UI area.

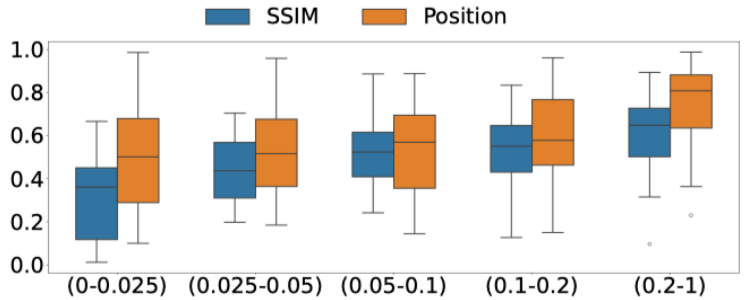
$$VS(I) = \frac{area(I)}{area(S_I)}$$

Limitation 3: The group with lower visual saliency has lower SSIM and position similarity.

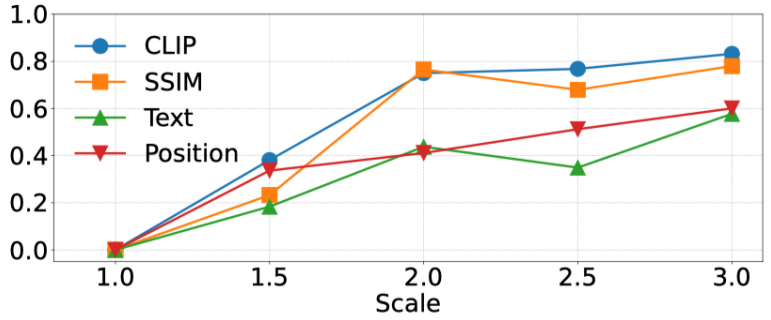
Improvement 3: Visual Saliency Enhancement (VSE). By cropping the image to increase the proportion of the interactive part, the model can better perceive the interaction area.



(b) CLIP and Text.



(c) SSIM and Position.



Inputs Modality vs Performance

- We conduct experiments on GPT-4o using 10 randomly selected webpages from failure cases. Human annotators provide textual descriptions for each interaction (e.g., "clicking the login button triggers a new window with two input boxes").

Limitation 4: Visual-only (V) and text-only (T) inputs exhibits unsatisfactory performance.

Improvement 4: Visual and Textual Description Combination. Combined visual and textual inputs can optimize performance.

TABLE VIII
PERFORMANCE OF GPT-4O WITH DIFFERENT MODALITY INPUTS. **BOLD VALUES** ARE THE BEST PERFORMANCE AND UNDERLINED VALUES ARE THE SECOND-BEST PERFORMANCE.

Prompt	Modality	CLIP	SSIM	Text	Position
Direct	V	0.3737	0.1793	0.2539	0.3951
	T	0.4174	0.4067	0.2316	0.4293
	V+T	**0.6735**	**0.5612**	**0.3919**	**0.7157**
CoT	V	0.3871	0.3101	0.2433	0.4461
	T	0.5579	0.1828	0.3045	0.5465
	V+T	**0.6440**	**0.4800**	**0.4287**	**0.7080**
Mark	V	0.5015	**0.4520**	0.3389	0.5025
	T	0.4613	0.4454	0.2805	0.4810
	V+T	**0.6923**	0.4336	**0.4248**	**0.7469**

Ablation Study and Case Study

- **Ablation Study**: Interactive Element Highlighting (IEH), Failure-aware Prompt (FAP), Visual Saliency Enhancement (VSE) and Textual Description (TD)
- **Case Study**: four interactive webpage development tasks are assigned to four programmers with similar front-end development experience.

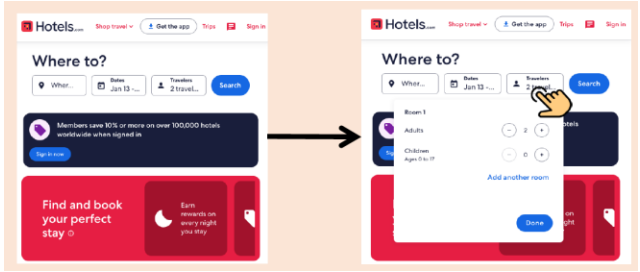
IEH	FAP	VSE	TD	CLIP	SSIM	Text	Position
✗	✗	✗	✗	0.5700	0.4891	0.3652	0.5802
✓	✗	✗	✗	0.5968	0.5456	0.4508	0.6408
✗	✓	✗	✗	0.6072	0.5405	0.4580	0.6452
✗	✗	✓	✗	0.6328	0.5723	0.4629	0.6874
✗	✗	✗	✓	0.6547	0.5681	0.4732	0.7024
✗	✓	✓	✓	0.6584	0.5938	0.4852	0.7027
✓	✗	✓	✓	0.6627	0.5987	0.4764	0.7128
✓	✓	✗	✓	0.6737	0.6136	0.4828	0.7269
✓	✓	✓	✗	0.6532	0.5823	0.4621	0.6987
✓	✓	✓	✓	**0.6937**	**0.6251**	**0.5187**	**0.7371**

Task	Time (w.)	Time (w/o)	Similarity (w.)	Similarity (w/o)
Task 1	323s	487s	5	4
Task 2	182s	319s	4	4
Task 3	125s	189s	5	4
Task 4	486s	657s	4	3.5
Average	**279s**	**413s**	**4.5**	**3.875**

Conclusions

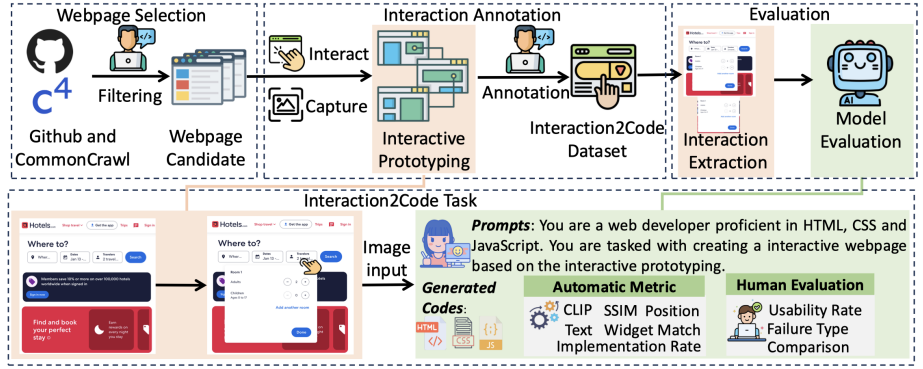
Task Formulation

We are the first to formulate the Interaction-to-Code task and present a systematic study on the code generation capabilities of MLLMs for dynamic interaction of webpages.



Benchmark Construction

We build the first real-world webpage interaction datasets Interaction2Code containing 127 webpages and 374 interactions, spanning 15 webpage topics and 31 interaction types.



Key Findings

(1) MLLMs struggle to generate interactive part compared with full static webpage generation.
(2) MLLMs are prone to make 10 types of failures.
(3) MLLMs perform poorly on interactions that are not visually obvious.
(4) Single visual modality description is not enough for MLLMs to understand the interaction.

Improvements

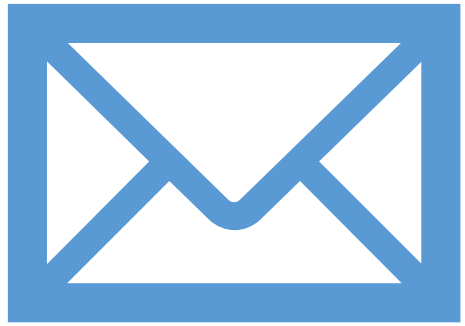
(1) Interactive Element Highlighting (IEH).

(2) Failure-aware Prompting (FAP).

(3) Visual saliency enhancement (VSE).

(4) Visual and =Textual Description Combination (TD).

Thank you!



- **Speaker: Jingyu Xiao**
- **Codes: https://github.com/WebPAI/Interaction2Code**
- **Homepage**: **https://whalexiao.github.io/**
- **Email**: **jyxiao@link.cuhk.edu.hk**