Dual-View Visual Contextualization for Web Navigation

Jihyung Kil Chan Hee Song Boyuan Zheng Xiang Deng Yu Su Wei-Lun Chao
The Ohio State University
{kil.5,song.1855,zheng.2372,deng.595,su.809,chao.209}@osu.edu

Abstract

*Automatic web navigation aims to build a web agent that can follow language instructions to execute complex and diverse tasks on real-world websites. Existing work primarily takes HTML documents as input, which define the contents and action spaces (*i.e.*, actionable elements and operations) of webpages. Nevertheless, HTML documents may not provide a clear task-related context for each element, making it hard to select the right (sequence of) actions. In this paper, we propose to contextualize HTML elements through their "dual views" in webpage screenshots: each HTML element has its corresponding bounding box and visual content in the screenshot. We build upon the insight*—web developers tend to arrange task-related elements nearby on webpages to enhance user experiences—*and propose to contextualize each element with its neighbor elements, using both textual and visual features. The resulting representations of HTML elements are more informative for the agent to take action. We validate our method on the recently released Mind2Web dataset, which features diverse navigation domains and tasks on real-world websites. Our method consistently outperforms the baseline in all the scenarios, including cross-task, cross-website, and cross-domain ones.*

1. Introduction

We study automatic web navigation with natural language instructions [8, 36]. This problem is crucial as it can potentially streamline and automate a wide range of tasks in our increasingly web-centric world, from online shopping to accessing information. Successfully solving this problem can also broadly advance artificial intelligence as it requires understanding and executing various tasks by interacting with dynamic and complex real-world (web) environments.

Existing work primarily takes HTML documents as the web agent's input [8, 10, 31], which define the meaning and layout of webpage content. Written partially in natural language, HTML documents enable the use of large language models (LLMs) [1, 4–6, 15, 29, 33, 34] to ground language instructions (*e.g.*, "Find one-way flights from New York to Toronto.") in web environments. Moreover, elements in HTML documents directly define the space of actions (*e.g.*, element "[button] Search" with operation "click"), preventing the agent from hallucinating infeasible actions.

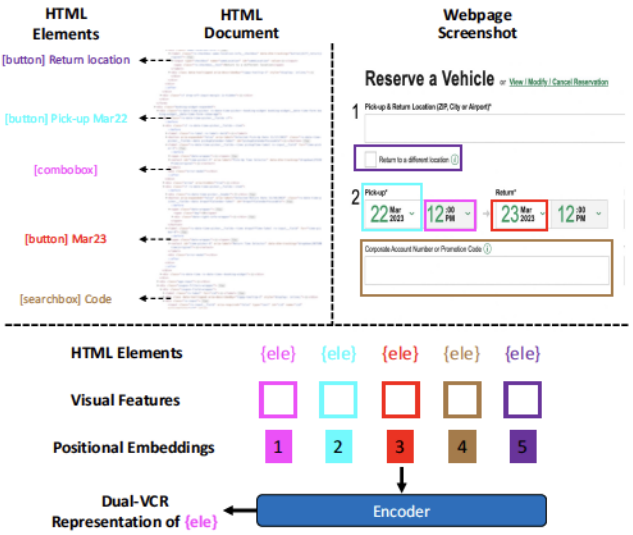

Figure 1. **Overview of our proposed Dual-View Contextualized Representation (DUAL-VCR).** HTML elements (*e.g.*, "[combobox]") may not have clear contexts for solving web navigation tasks (*e.g.*, "Find the lowest rent truck with a pick-up time at 11 am on March 27."). DUAL-VCR contextualizes each element with its neighbors in the screenshot (*e.g.*, "[button] Pick-up Mar22") to obtain more informative representations for decision-making.

With that being said, HTML documents may lack a clear task-related context for each element, impeding the agent from selecting the right (sequence of) actions to complete a task. HTML is quite flexible for web developers to arrange their code. Even semantically related elements, such as an actionable element (*e.g.*, "drop-down box") and its label element (*e.g.*, "Number of Passengers"), may not be located nearby in the document or the DOM tree. This problem also applies to elements relevant to solving a task. While LLMs may learn to capture the context, a raw HTML document of real-world webpages is often quite huge, consisting of tens of thousands of tokens, making it either infeasible or cost-prohibitive to be directly fed into LLMs [8, 10, 31].

In this paper, we propose to enhance the context of each HTML element by leveraging its "dual view" in the screenshot of the rendered webpage: many of the HTML elements (including the actionable ones) are visible in the screenshot and have their corresponding bounding boxes[1]. Taking the insight—*semantically related and task-related HTML elements are often located nearby on the webpage* to facilitate user experiences—we propose to contextualize each HTML element with its neighbors in the screenshot. Concretely, when encoding each HTML element, we 1) append its spatially adjacent elements with positional embeddings and 2) incorporate both the visual and textual features (Figure 1).

While simple, our method, which we name **Dual-View Contextualized Representation (DUAL-VCR)**, has several compelling properties that benefit web navigation fundamentally. First, **DUAL-VCR** uses the built-in feature of HTML documents to align textual and visual content, making it robust to complex and diverse websites. Second, **DUAL-VCR** effectively leverages visual cues on the webpages, which are designed to ease users' efforts in understanding and completing tasks. Specifically, **DUAL-VCR** connects *visually proximate elements that are often semantically related and task-related*, providing the agent with more explicit contexts to take not only individual actions but also the sequence of actions. Last but not least, **DUAL-VCR** can potentially be integrated into any web navigation algorithms that take HTML documents as input.

We validate DUAL-VCR on the Mind2Web dataset [8], the largest web navigation benchmark with over 2,000 tasks curated from 137 real-world websites across 31 domains, including restaurants, airlines, public services, etc. Concretely, we implement **DUAL-VCR** on top of the **MindAct** algorithm [8], which was proposed to tackle huge HTML documents. In short, at each action, MindAct first applies a small LM to rank each HTML element to shrink the document; it then uses an LLM to predict the action. We integrate DUAL-VCR into both steps to enhance the context for element ranking and decision-making. DUAL-VCR consistently improves MindAct across all three scenarios (cross-task, cross-website, and cross-domain), leading to a **3.7**% absolute gain on average over nine evaluation metrics. Moreover, DUAL-VCR notably outperforms baselines that use entire HTML documents or screenshots as input, offering significant advantages in computation and accuracy.

Our contributions are three-folded:

- We propose DUAL-VCR, a simple and effective dual-view representation of HTML elements for web navigation.
- DUAL-VCR consistently outperforms baselines on the real-world web navigation benchmark Mind2Web [8].
- We conduct comprehensive analyses to understand the effect of our design choices on web navigation performance.

[1]These bounding boxes can be directly inferred from the HTML document without the need to detect them.

2. Related Work

Web navigation datasets. Several prior studies [2, 16, 23, 32, 36] have introduced promising benchmarks for assessing agents in web navigation tasks. However, these benchmarks are often limited to a narrow range of website domains or confined to simplified simulated environments. For instance, MiniWob++ [16] and WebShop [36] collected a set of websites including daily tasks (*e.g.*, shopping), but each website only has fewer than fifty HTML elements on average. Some other studies [2, 23, 32] instead explored other domains, including mobile applications, but their action spaces are often simpler than web navigation. Recently, Mind2Web [8] released the first large-scale web navigation benchmark consisting of over 2K tasks from various real-world websites. This enables a comprehensive understanding of web agent's behaviors in "real-world" scenarios.

The use of HTML documents. Most earlier work [16, 18, 26, 36] focused on simple navigation scenarios like MiniWob++ [16]. Due to the brevity of its HTML documents, they input whole HTML documents into LLMs to complete the web navigation tasks. A few studies represented HTML documents in a more dense format. For instance, ASH [31] summarized the HTML document using LLMs with hierarchical prompting. DOM-Q-NET [18] leveraged a graph neural network to represent a document as a graph. For real-world web navigation (*e.g.*, Mind2Web), HTML documents are often overly lengthy and complex. Thus, recent studies [8–10] applied text-based filtering to first identify key HTML elements within the document and only used the selected elements to complete the task. While all these prior methods are promising, the HTML document alone may not provide a clear task-related context for each element, making it challenging to select the right actions. Our approach instead enhances the context of each HTML element based on their dual view in the screenshot.

The use of webpage screenshots. Beyond using HTML documents, several studies [9, 11, 14, 16, 17, 21, 30, 36, 37] have explored the incorporation of screenshots for web navigation. Some of them [9, 11, 14, 16, 17, 37] utilized both screenshots and HTML documents to learn their joint representations during decision-making. Some others [3, 21, 30] solely relied on screenshots, bypassing the use of HTML documents. We note that all prior methods primarily focused on utilizing "whole" screenshots. In contrast, we shift the focus to neighboring elements within the screenshot, providing significant benefits in computation and accuracy.

3. Approach: DUAL-VCR

We introduce **Dual-View Contextualized Representation (DUAL-VCR)** for enhanced web navigation. To begin with, we provide a brief background about web navigation.

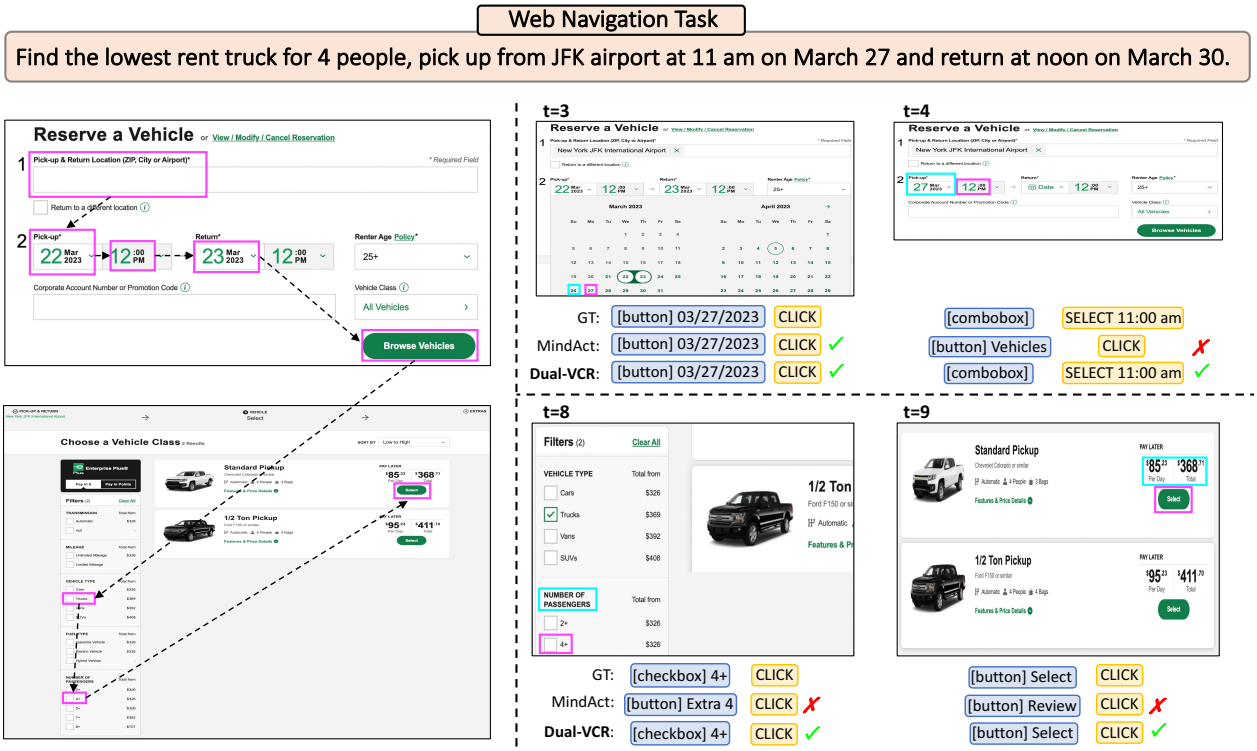

Figure 2. **Example of real-world web navigation. Top**: the web navigation task described in natural language. **Left**: the sequence of HTML elements (visualized on webpages, not HTML documents) to interact with to complete the task. We superimpose bounding boxes and arrows to locate the target elements and indicate their order. **Right**: the detail at each time step (we showed $t = \{3, 4, 8, 9\}$ for brevity). GT: ground-truth action (Element with Operation). We compare the predicted actions by MindAct [8] and our **DUAL-VCR**. The bounding box and bounding box indicate the target element and one of its neighbors encoded by **DUAL-VCR**. As shown, **DUAL-VCR** correctly predicts the elements and operations at "all" time steps, taking advantage of the much richer task-related dual-view context it encodes.

3.1. Background: web navigation

A web navigation task consists of a website S (*e.g.*, an airline website) and an instruction q ("Find one-way flights from New York to Toronto."). Given (S, q), a web agent f needs to decide and perform a sequence of actions $a = \{a_1, a_2, \cdots, a_t, \cdots\}$ on the website to complete the task. Figure 2 (left) gives an illustration.

At time step t, the website has an HTML document H_t, composed of a list of elements $H_t = \{e_{t,1}, e_{t,2}, \cdots, e_{t,N}\}$. These HTML elements jointly define 1) the layout and content on the rendered webpage I_t, and 2) the action space at time t: each candidate action is a pair of an actionable element (*e.g.*, "[textbox] To") and an operation (*e.g.*, "Type Toronto"). After taking action a_t, both the HTML document and webpage will be updated into (H_{t+1}, I_{t+1}). For example, clicking the "[checkbox] One way" on the airline webpage removes the "[textbox] Return date" from the webpage. Namely, the web environment is dynamic, and the agent must take this into account to decide its actions.

Because of the rich content in the HTML document H_t, existing work primarily takes it, together with the instruction q and the action history (*e.g.*, *Type New York in the From box*), as the agent's input at time t to decide the next action (*e.g.*, *Type Toronto in the To box*),

$$a_{t+1} = f(q, H_t, \{a_1, a_2, \cdots, a_t\}). \quad (1)$$

One excellent candidate for f is LLMs [1, 4–6, 15, 29, 33, 34], which have shown straggering sucesses in question answering [35] and logical reasoning [7]. For example, [16, 19] applied LLMs to simplified web navigation.

However, for real-world webpages that easily contain thousands of HTML elements (amounting to tens of thousands of tokens), directly applying LLMs is neither efficient nor effective. As such, recent work [8, 10, 31] employed a two-stage framework: first summarizing the HTML document and then predicting the action. For instance, given the instruction q and the action history at time t, the MindAct algorithm [8] first ranks each HTML element using a small LM. Only the top-K HTML elements are fed into an LLM to predict the next action. (See Figure 3 for an illustration.)

3.2. Context enhancement

We identify one critical pitfall in the two-stage framework. *Since HTML documents may not provide a clear context for each element, the element ranker and the subsequent action*

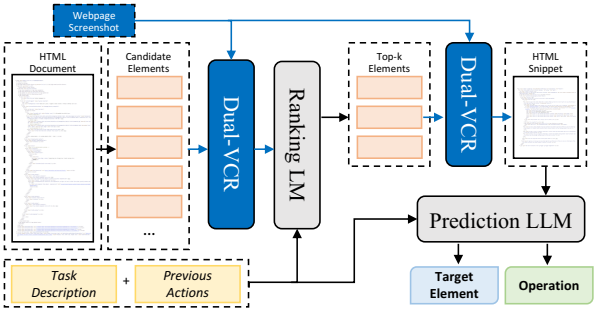

Figure 3. **The web navigation pipeline with DUAL-VCR,** built on top of the MindAct algorithm [8]. MindAct uses a small ranking LM to select candidate HTML elements and a prediction LLM to decide actions. Blocks and arrows in NavyBlue indicate the insertion of **DUAL-VCR** for enhanced element representations.

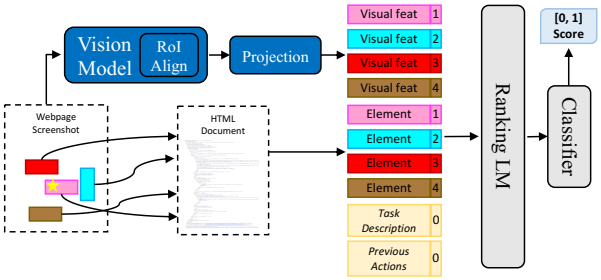

Figure 4. **DUAL-VCR-enhanced element ranker.**. We contextualize the candidate element (denoted by $\star$) with its neighbors in the screenshot, using both the visual features (by [20]) and textual features (extracted from the HTML document). Positional embeddings are added to specify neighbor elements, learning their spatial relationships and pairing the textual features with visual features. This dual-view contextualized representation is used to rank the candidate element, measuring its relevance to the current task.

predictor may not perform as effectively as expected. Figure 1 illustrates one such issue: the element "[combobox]" should be paired with "[button] Pick-up Mar22" to fully describe its role, *i.e.*, time for pick-up. However, these two elements are not necessarily nearby in the HTML document.

To resolve this issue, we propose to leverage the "dual view" of each HTML element $e_{t,n} \in H_t$ in the rendered webpage I_t to enhance its context. In essence, many HTML elements (including the actionable ones) are visible in I_t. Further, their visual location (*e.g.*, bounding boxes) can be inferred from HTML documents. Since a webpage (specifically, its screenshot) is designed for users to interact with the website visually, we hypothesize that incorporating the visual cues into HTML element representations would benefit the web agent in understanding and completing tasks.

To this end, we propose **Dual-View Contextualized Representation (DUAL-VCR)**. In the screenshot view, we identify the bounding box of each HTML element using a web automation testing tool[2]. Taking the insight—web developers tend to arrange semantically relevant and task-related elements in proximity to each other on the screenshot to enhance user experiences—we contextualize each element with its "visual" neighbors. Concretely, we calculate the center points of all elements using their bounding boxes and measure their pairwise distances. For each *candidate* element to be ranked by MindAct, we search for the closest M elements to form its context jointly.

We consider both the visual and textual information to encode the candidate element and its visual neighbors. We extract each element's visual feature using the Pix2Struct Vision Transformer (ViT) [20], which is pre-trained on webpage screenshots. Specifically, we input the whole screenshot I_t into the ViT and apply ROI Align [12, 24] on top of the output embeddings to obtain the feature vector corresponding to each element's bounding box. In the HTML document view, we extract each element's corresponding "HTML text" following MindAct [8].

[2] https://playwright.dev/

3.3. DUAL-VCR-enhanced element ranker

In MindAct, a small ranking LM is built to predict each element's importance for action prediction. At each time step, the ranking LM takes the element's HTML text tokens, the task description q, and the previous actions as input.

We propose to expand the ranking LM to integrate 1) both visual features and textual features and 2) both the candidate element and its neighbor elements. (See Figure 4 for an illustration.) We make the following design choices. To align the visual embedding and textual embedding, we follow the recent practice of vision-and-language models (*e.g.*, BLIP-2 [22], LLaVA [28], LLaVA-1.5 [27]) to learn a linear projection layer to project ViT visual features into the same dimensionality as the token embeddings in the ranking LM. To pair each of the projected visual vectors with its corresponding text tokens and specify each neighbor element in the context, we add positional encoding. Concretely, we sort the neighbors based on their spatial distances from the candidate element and add a learnable positional embedding (unique for each rank) to the neighbor element's visual and text token embeddings. These positionally encoded visual and text token embeddings (of the candidate and the neighbor elements) are fed into the ranking LM; the projected visual features are prepended to the text embeddings, serving as soft visual prompts. In training, we only learn the linear projection layer, the positional embeddings, and the LM while keeping the ViT frozen. This training scheme has been shown to effectively enhance the alignment between vision and language components and improve the pre-trained LM's adaptability to downstream tasks. Please see more details in the supplementary materials.

3.4. DUAL-VCR-enhanced action predictor

After obtaining the top-K elements from the ranker (§3.3), MindAct combines them into an HTML snippet as the input

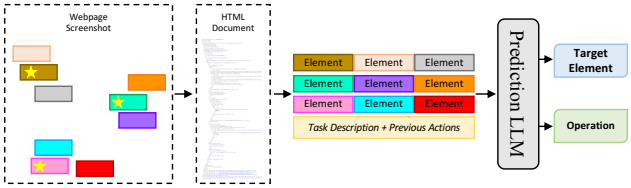

Figure 5. **DUAL-VCR-enhanced action predictor**. Given the top-K candidate elements (three in the figure, marked with $\star$), DUAL-VCR appends each with its neighbor elements. The resulting HTML snippet, together with the task description and previous actions, is then fed into an LLM for predicting the next action.

to LLMs. The objective is to predict the action for the current time step, including the target element (*e.g.*, "[textbox] To") and its associated operation (*e.g.*, "Type Toronto"). Specifically, MindAct converts the target element prediction problem into multiple-choice question-answering.

We apply DUAL-VCR to contextualize each of the answer candidates. Similarly to §3.3, we find the M closest neighbors for each candidate element on the screenshot. We then append the HTML text tokens of these M neighbors to the candidate element; we add specific tokens to separate between elements. Figure 5 gives an illustration. Please see the supplementary material for more details.

3.5. Why DUAL-VCR?

DUAL-VCR leverages and encodes visual cues on the webpage, offering valuable contexts for the HTML elements in element ranking and action prediction. We show two cases.

First, as shown in Figure 1, some HTML elements (*e.g.*, "[combobox]") are quite generic and must be paired with spatially nearby elements (*e.g.*, "[button] Pick-up Mar22") to specify their meanings (*i.e.*, time for pick-up). Similar examples can be found in Figure 2. At $t = 8$, there are two seemingly similar candidates "[checkbox] 4+" and "[button] Extra 4". Nevertheless, the former is spatially closer to the element "Number of passengers", indicating its relatedness to the task "... truck for 4 people ..." (see the top of Figure 2). At $t = 9$, two identical "[button] Select" elements exist. The only way to differentiate them is through their visual neighbors: one is associated with a lower price than the other. Our DUAL-VCR offers an explicit way to enforce these spatial contexts in the screenshots.

Second, as shown in the left panel of Figure 2, consecutive steps to solve a task often involve spatially nearby elements. Completing one step thus introduces a prior that its nearby elements may be the next to take action upon. As both the ranking LM and prediction LLM take the task description q, *past actions*, and our DUAL-VCR representation as input, the models could potentially capture such prior information to increase the success rate for the following action. For example, at $t = 4$, DUAL-VCR successfully takes the action "Select 11:30 am", likely attributing to its capability to recognize that the previously completed task was the spatially nearby "Select 03/27/2023".

4. Experimental Results

Dataset. We validate DUAL-VCR on Mind2Web [8], a comprehensive benchmark for real-world web navigation. Unlike other benchmarks based on simulated websites with only a few HTML elements, Mind2Web uses over 100 real-world websites with thousands of HTML elements. Concretely, they provide over 2K open-ended tasks collected from 137 real-world websites across 31 different domains, including travel, shopping, public service, etc (Table 1). Please see more details in the supplementary material.

Evaluation Tasks. Followed by Mind2Web [8], we evaluate models at three different test splits. In **Cross-Domain**, we evaluate the model's generalizability to a new domain where it has not seen any websites or tasks associated with that domain during training. This split contains 912 tasks in total. In **Cross-Website** (177 tasks), while the model is not exposed to test websites, it is trained on websites from the same domain and potentially with similar tasks. This configuration enables us to evaluate the model's capacity to adapt to entirely new websites within familiar domains and tasks. Similar to the conventional training/test split, **Cross-Task** (252 tasks) randomly splits 20% of the data as a test set, regardless of the domains and the websites. Please see the supplementary material for more details.

Evaluation Metrics. We use the Mind2Web's official metrics. The ranker performance is measured by **Recall@K**, where K is the number of top HTML candidate elements. **Element Accuracy** (Ele. Acc) compares the selected element with the ground-truth elements. **Operation F1** (Op. F1) calculates the token-level F1 score for the predicted operation. **Step Success Rate** (Step SR) measures the success of each step; A step is considered successful only if both the selected element and the predicted operation are correct. For each step, they provide previous "ground-truth" actions with the assumption that the model successfully completes all previous steps.

Baselines. DUAL-VCR is based on MindAct [8], which has a ranking LM and a prediction LLM. Our main baselines are thus its ranker and action predictor, denoted by **MINDACT$_{\text{RANK}}$** and **MINDACT$_{\text{PRED}}$**. MINDACT$_{\text{RANK}}$ uses DeBERTa$_{\text{base}}$ [13], a small encoder-only LM to rank elements. For action prediction, MINDACT$_{\text{PRED}}$ uses Flan-T5$_{\text{base}}$ [6], an instruction fine-tuned LLM.

Our Models. Aligned with MindAct, we use the same DeBERTa$_{\text{base}}$ [13] / Flan-T5$_{\text{base}}$ [6] for our ranker / action predictor, repsectively. For visual features extraction, we utilize Pix2Struct [20]'s ViT (pre-trained on screenshots) as the visual backbone and apply ROI Align [12] on the element's region. We use two linear layers to project visual

Dataset	# Websites	Website Type	# Tasks	Avg # HTML	
				Elements	Tokens
MiniWoB++ [16]	100	Simplified	100	28	500
Mind2Web [8]	137	Real-world	2,350	1,135	44,402

Table 1. **Statistics of Mind2Web [8].** Min2Web, the largest web navigation benchmark, collects real-world websites across various domains. The significant volume of content on the webpage (*e.g.*, an average of 1K/44K HTML elements/tokens) poses challenges for LLMs in both computational and learning aspects.

Ranker	Recall			
	@1	@5	@10	@50
MINDACT$_{\text{RANK}}$	25.4	61.0	73.5	88.9
DUAL-VCR$_{\text{VNEI-TXT}}$	37.3	70.8	79.3	89.2
DUAL-VCR$_{\text{VIS}}$	37.1	70.2	79.2	89.1
DUAL-VCR$_{\text{VNEI-TXT+VIS}}$	**38.4**	**71.6**	**79.7**	**90.1**

Table 2. **Ranking performance.** Visual neighbors' HTML text (DUAL-VCR$_{\text{VNEI-TXT}}$) consistently outperforms MINDACT$_{\text{RANK}}$. Moreover, DUAL-VCR$_{\text{VNEI-TXT+VIS}}$, using both visual neighbors' HTML text and visual features, performs best, showing the strength of dual-view contextualization in element ranking.

features into textual embedding space. Please see the supplementary materials for details on the model training.

Notation of DUAL-VCR. DUAL-VCR has several variations to understand the effect of each of its components in detail. We denote them as follows:

- **DUAL-VCR$_{\text{VIS}}$**: Ranker w/ candidate's visual features.
- **DUAL-VCR$_{\text{VNEI-TXT}}$**: Ranker w/ neighbors' HTML text.
- **DUAL-VCR$_{\text{VNEI-TXT+VIS}}$**: Ranker w/ candidate's visual features and its neighbors' visual features and HTML text.
- **DUAL-VCR$_{\text{PRED}}$**: Action predictor w/ neighbors' HTML text.

4.1. Effectinvess of DUAL-VCR

The main goal of our experiments is to show that our dual-view contexutalization is beneficial in (i) finding promising top-K candidates from entire HTML documents (*i.e.*, ranking peformance), and (ii) predicting the action, including both element selection and operation prediction.

Ranking performance. Table 2 summarizes the ranking results across different top-K candidate elements. First, we see that incorporating the visual neighbor elements' HTML text (DUAL-VCR$_{\text{VNEI-TXT}}$) consistently and significantly outperforms MINDACT$_{\text{RANK}}$ on all Recall@Ks (*e.g.*, 37.3% vs. 25.4% on Recall@1, 79.3% vs. 73.5% on Recall@10), suggesting that contextualizing the element with its neighbors indeed helps find the target element. Second, the candidate element's visual features (DUAL-VCR$_{\text{VIS}}$) lead to notable improvements over MINDACT$_{\text{RANK}}$ (*e.g.*, 70.2% vs. 61.0% on Recall@5). This implies that the visual features offer additional context in differentiating HTML elements, compared to using only its HTML text. Lastly, DUAL-VCR$_{\text{VNEI-TXT+VIS}}$ achieves a further boost by leveraging both visual neighbors' HTML text and visual features (*e.g.*, 38.4%/90.1% on Recall@1/@50).

Action prediction performance. Table 3 shows the results of action prediction. Compared to the baseline (the combination of MINDACT$_{\text{RANK}}$ and MINDACT$_{\text{PRED}}$), using the visual neighbors' HTML texts (DUAL-VCR$_{\text{VNEI-TXT}}$ $\rightarrow$ DUAL-VCR$_{\text{PRED}}$) notably improves across all metrics. For instance, we achieve gains of 3.4% on Step SR in Cross-Task, 1.3% on Ele. Acc in Cross-Webiste, and 6.3% on Op. F1 in Cross-Domain. These consistent improvements demonstrate the advantages of incorporating visual neighbor information during the model's decision-making process. Moreover, aligning with the ranking result, integrating the visual neighbors' visual features into the ranker (DUAL-VCR$_{\text{VNEI-TXT+VIS}}$) shows its effectiveness in action prediction as well. Concretely, it achieves the best performance on all nine metrics, along with a 5% maximum gain on each type of metric against the baseline (*e.g.*, Ele. Acc: 47.0% vs. 42.0% on Cross-Task, Op. F1: 72.0% vs. 67.0% on Cross-Website, Step SR: 46.0% vs. 41.1% on Cross-Task).

4.2. Analysis

We aim to understand DUAL-VCR in detail. We show a) a more in-depth analysis of the main table, b) the interaction between the ranker and the action predictor, c) its effectiveness compared to whole input data and random elements, and d) the effect of different sizes of visual neighbors.

Detailed ablation. Table 4 provides more details about the main table to better understand the impact of each component in DUAL-VCR. First, we keep the action predictor as MINDACT$_{\text{PRED}}$ and focus on the pure effects of our rankers on the action prediction task (*i.e.*, 1st to 4th rows). We see that incorporating the candidate element's visual features (DUAL-VCR$_{\text{VIS}}$) achieves a slight but significant improvement over MINDACT$_{\text{RANK}}$ across all metrics (*e.g.*, 42.5% vs. 42.0% on Ele. Acc). Furthermore, our ranker with the visual neighbors' HTML text (DUAL-VCR$_{\text{VNEI-TXT}}$) outperforms MINDACT$_{\text{RANK}}$ by a notable margin of +2.6%/+0.8%/+2.1% on Ele. Acc/Op. F1/Step SR, respectively. Besides, DUAL-VCR$_{\text{VNEI-TXT+VIS}}$, which encodes the visual neighbors' visual features, further improves the model's decision-making ability (*e.g.*, 46.0% vs. 44.6% on Ele. Acc). In short, we consistently demonstrate the effectiveness of each component in our ranker.

Second, conversely, we fix the ranker and examine the benefit of encoding visual neighbors' HTML text features into the action predictor (DUAL-VCR$_{\text{PRED}}$). Compared to MINDACT$_{\text{PRED}}$, DUAL-VCR$_{\text{PRED}}$ achieves consistent gains across all rankers. For instance, MINDACT$_{\text{RANK}}$ $\rightarrow$ DUAL-VCR$_{\text{PRED}}$ outperforms MINDACT$_{\text{RANK}}$ $\rightarrow$ MINDACT$_{\text{PRED}}$

Ranker	Action Predictor	Cross-Task			Cross-Website			Cross-Domain		
		Ele. Acc	Op. F1	Step SR	Ele. Acc	Op. F1	Step SR	Ele. Acc	Op. F1	Step SR
MINDACT$_{\text{RANK}}$	MINDACT$_{\text{PRED}}$	42.0	74.9	41.1	30.7	67.0	30.0	31.5	66.6	31.0
DUAL-VCR$_{\text{VNEI-TXT}}$	DUAL-VCR$_{\text{PRED}}$	45.3	78.4	44.5	32.0	71.5	31.5	32.4	72.9	32.0
DUAL-VCR$_{\text{VNEI-TXT+VIS}}$		**47.0**	**78.7**	**46.0**	**32.7**	**72.0**	**32.5**	**33.2**	**73.3**	**32.5**

Table 3. **Results of action prediction.** Our DUAL-VCR$_{\text{VNEI-TXT}}$ → DUAL-VCR$_{\text{PRED}}$, leveraging visual neighbors' HTML text information, notably improves over the baseline (MINDACT$_{\text{RANK}}$ →MINDACT$_{\text{PRED}}$) on all nine metrics. Adding visual neighbors' visual features (DUAL-VCR$_{\text{VNEI-TXT+VIS}}$) leads to further improvements, highlighting the benefit of dual-view context on real-world web navigation.

Ranker	Action Predictor	Cross-Task		
		Ele. Acc	Op. F1	Step SR
MINDACT$_{\text{RANK}}$	MINDACT$_{\text{PRED}}$	42.0	74.9	41.1
DUAL-VCR$_{\text{VIS}}$		42.5	75.1	41.5
DUAL-VCR$_{\text{VNEI-TXT}}$		44.6	75.7	43.2
DUAL-VCR$_{\text{VNEI-TXT+VIS}}$		46.0	78.6	44.8
MINDACT$_{\text{RANK}}$	DUAL-VCR$_{\text{PRED}}$	44.4	75.2	43.1
DUAL-VCR$_{\text{VIS}}$		44.6	76.8	43.8
DUAL-VCR$_{\text{VNEI-TXT}}$		45.3	78.4	44.5
DUAL-VCR$_{\text{VNEI-TXT+VIS}}$		**47.0**	**78.7**	**46.0**

Table 4. **Ablation studies** for validating the importance of each component in DUAL-VCR. See §4.2 for a detailed discussion.

(*e.g.*, 44.4% vs. 42.0% on Ele. Acc). Similarly, when fixing the ranker with DUAL-VCR$_{\text{VNEI-TXT+VIS}}$, DUAL-VCR$_{\text{PRED}}$ improves over MINDACT$_{\text{PRED}}$ (*e.g.*, 46.0% vs. 44.8% on Step SR). This shows directly encoding the visual neighbor's HTML text into the action predictor is beneficial.

Finally, DUAL-VCR$_{\text{VNEI-TXT+VIS}}$ and DUAL-VCR$_{\text{PRED}}$ are complementary; we achieve the best performance across all metrics when leveraging both (*e.g.*, 47.0%/78.7%/46.0% on Ele. Acc/Op. F1/Step SR). Please see more ablation studies in the supplementary materials.

Ranker-action predictor relationship. We analyze the relationship between the ranker and the action predictor in Table 5. We observe a linear connection between the two. Concertely, improving the ranker (*e.g.*, 25.4% vs. 37.3% on Recall@1) correlates with improved action prediction results (*e.g.*, 24.0% vs. 35.5% on Ele. Acc). Aligned with results in §4.2, this again highlights the importance of improving the model's ranking ability in web navigation.

Comparison to whole input data. Since HTML documents contain a significant amount of content, such as thousands of HTML elements, conducting experiments with whole data is computationally challenging. Nevertheless, we do our best to report the associated results on Table 6 to give more context on the effect of DUAL-VCR. First, instead of asking the ranker to prune HTML documents, we directly pass the whole HTML documents into the action predictor (WHOLEHTML$_{\text{PRED}}$). We see that WHOLEHTML$_{\text{PRED}}$ performs notably less against the baseline (MINDACT$_{\text{PRED}}$) (*i.e.*, 38.6% vs. 42.0% on Ele. Acc). We attribute this to the difficulty of finding the target element among *all thousands* of elements. In contrast, our DUAL-VCR$_{\text{PRED}}$ achieves a much better result (*i.e.*, 44.4%) with significantly less amount of input elements.

Second, DUAL-VCR outperforms the utilization of whole images. We first use the entire image for the ranker (WHOLEIMAGE$_{\text{RANK}}$). To extract the image features, we use the same procedure mentioned in §3.2, except for providing the region of the whole image instead of that of specific elements. We then use these whole image features, along with the same HTML text input used in MINDACT$_{\text{PRED}}$, to train WHOLEIMAGE$_{\text{RANK}}$. Although the entire image features are shown effective over the baseline (*i.e.*, 43.9% vs. 42.0%), it performs notably less than our approach using the *visual neigbhor*'s visual information (*i.e.*, 46.0% of DUAL-VCR$_{\text{VNEI-TXT+VIS}}$). In addition, we conducted a study applying the whole image to the action predictor. Specifically, similar to recent vision-and-language models [22, 27, 28], we extract whole image features using fine-tuned ViT [20] and prepend them to the top-50 candidate elements extracted from MINDACT$_{\text{RANK}}$ as the input to the LLM (Flan-T5$_{\text{base}}$ [6]). Similar to the result of WHOLEIMAGE$_{\text{RANK}}$, this action predictor (WHOLEIMAGE$_{\text{PRED}}$) performs worse than DUAL-VCR$_{\text{PRED}}$, which only uses *visual neighbors*' HTML text. Overall, this highlights the advantages of our approach in terms of computational efficiency and performance. See additional results in the supplementary materials.

Visual neighbors offer meaningful contexts. We examine whether visual neighbors provide meaningful context for element ranking and action prediction. To assess this, we compare visual neighboring elements with random elements (Table 7). Specifically, We randomly select (five) elements from HTML documents and use them to train either the ranker or the action predictor. While our ranker (*e.g.*, DUAL-VCR$_{\text{VNEI-TXT}}$) notably improves the ranking performance over MINDACT$_{\text{RANK}}$ (*e.g.*, 89.2% vs. 88.9%), the "random" ranker performs less than MINDACT$_{\text{RANK}}$ (*e.g.*, 86.7% vs. 88.9%). This, in turn, leads to a significant performance drop in the action prediction (*e.g.*, 42.0% vs. 40.6% on Ele. Acc). Similarly, compared to the MINDACT$_{\text{PRED}}$, including random elements in the ac-

Ranker	Action Predictor	Top-1			Top-5			Top-10			Top-50		
		Recall	Ele. Acc	Op. F1	Recall	Ele. Acc	Op. F1	Recall	Ele. Acc	Op. F1	Recall	Ele. Acc	Op. F1
MINDACT$_{\text{RANK}}$	MINDACT$_{\text{PRED}}$	25.4	24.0	23.7	61.0	39.2	52.1	73.5	41.4	62.8	88.9	42.0	74.9
DUAL-VCR$_{\text{VNEI-TXT}}$		**37.3**	**35.5**	**33.5**	**70.8**	**43.1**	**54.1**	**79.3**	**43.9**	**63.0**	**89.2**	**44.6**	**75.7**

Table 5. **Relationship between ranker and action predictor on Cross-Task.** The ranker has a linear correlation with the action predictor, suggesting the importance of improving its ranking capabilities for decision-making.

Ranker	Action Predictor	Cross-Task Ele. Acc
MINDACT$_{\text{RANK}}$	MINDACT$_{\text{PRED}}$	42.0
	WHOLEIMAGE$_{\text{PRED}}$	43.6
	DUAL-VCR$_{\text{PRED}}$	44.4
WHOLEIMAGE$_{\text{RANK}}$	MINDACT$_{\text{PRED}}$	43.9
DUAL-VCR$_{\text{VNEI-TXT}}$		44.6
DUAL-VCR$_{\text{VNEI-TXT+VIS}}$		**46.0**
-	WHOLEHTML$_{\text{PRED}}$	38.6

Table 6. **Visual neighbor vs. whole input data.** Using visual neighbors notably outperforms the use of whole data, offering advantages regarding computational efficiency and performance.

Ranker	Recall @50	Action Predictor	Cross-Task Ele. Acc	Cross-Task Op. F1
MINDACT$_{\text{RANK}}$	88.9	MINDACT$_{\text{PRED}}$	42.0	74.9
		RANDOM$_{\text{PRED}}$	41.5	73.6
		DUAL-VCR$_{\text{PRED}}$	44.4	75.2
RANDOM$_{\text{RANK}}$	86.7	MINDACT$_{\text{PRED}}$	40.6	72.0
DUAL-VCR$_{\text{VNEI-TXT}}$	**89.2**		**44.6**	**75.7**

Table 7. **Visual neighbors vs. random elements.** Visual neighbors provide meaningful contexts for web navigation, notably outperforming elements randomly extracted from HTML documents.

Ranker			Cross-Task	
Method	# neighbors	Recall@50	Ele. Acc	Op. F1
DUAL-VCR$_{\text{VIS}}$	0	89.1	42.5	75.1
DUAL-VCR$_{\text{VNEI-TXT+VIS}}$	3	89.7	45.5	77.3
	5	**90.1**	**46.0**	**78.6**
	10	89.5	45.2	77.0

Table 8. **Effects of the number of neighbors on ranker.** Choosing the right size of visual neighbors is important for element ranking, and the size of five is found to be most effective for Mind2Web [8]. We fix the action predictor with MINDACT$_{\text{PRED}}$.

tion predictor hurts the action prediction performance (*e.g.*, 74.9% vs. 73.6 on Op. F1) while visual neighbors are beneficial (*e.g.*, 75.2%). In sum, we empirically demonstrate the benefits of context in visual neighbors for web navigation.

Effects of the number of visual neighbors. We ablate the impact of varying sizes of visual neighbors, starting with Table 8, which shows its effect on the ranker while maintaining the same action predictor (MINDACT$_{\text{PRED}}$). We observe a linear correlation between the size of visual neighbors and their ranking/action prediction performance. For instance, increasing the size of neighbors up to five shows consistent improvements (*e.g.*, 89.1%→90.1% on Recall@50 and 75.1%→78.6% on Op. F1). However, considering too many neighbors (*e.g.*, the size of ten) hurts the performance. For example, increasing the size from five to ten decreases the element accuracy from 46.0% to 45.2%. We also see a similar pattern when ablating the effect of the visual neighbor size on the action predictor (Table 9). Concretely, while keeping the same ranker (DUAL-VCR$_{\text{VNEI-TXT+VIS}}$), the action performance increases up to the size of five (*e.g.*, 46.0%→47.0% on Ele. Acc) but decreases when the size becomes ten (*e.g.*, 46.2% on Ele. Acc). Overall, this suggests that choosing an appropriate number of neighbors is necessary for both element ranking and action prediction.

Action Predictor		Cross-Task	
Method	# neighbors	Ele. Acc	Op. F1
MINDACT$_{\text{PRED}}$	0	46.0	78.6
DUAL-VCR$_{\text{PRED}}$	3	46.4	78.7
	5	**47.0**	**78.7**
	10	46.2	78.6

Table 9. **Effects of the number of neighbors on action predictor.** Similar to Table 8, the size of five is most appropriate for the action prediction. We use DUAL-VCR$_{\text{VNEI-TXT+VIS}}$ for the ranker.

5. Conclusion

We introduce DUAL-VCR to effectively represent HTML elements for web navigation. DUAL-VCR contextualizes each element with its visual neighbor elements, leveraging both textual and visual features. DUAL-VCR consistently improves real-world web navigation in the Mind2Web benchmark, supported by comprehensive analyses.

Acknowledgments

This research is supported in part by grants from the National Science Foundation (IIS-2107077, OAC-2112606) and ARL W911NF2220144. We are thankful for the generous support of the computational resources by the Ohio Supercomputer Center.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *In NeurIPS*, 2020. 1, 3

[2] Andrea Burns, Deniz Arsan, Sanjna Agrawal, Ranjitha Kumar, Kate Saenko, and Bryan A Plummer. A dataset for interactive vision-language navigation with unknown command feasibility. In *ECCV*, 2022. 2, 13

[3] Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. *arXiv preprint arXiv:2401.10935*, 2024. 2, 13

[4] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. https://lmsys.org/blog/2023-03-30-vicuna/, 2023. 1, 3

[5] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[6] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022. 1, 3, 5, 7, 11

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 3

[8] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *In NeurIPS*, 2023. 1, 2, 3, 4, 5, 6, 8, 10, 11, 12

[9] Hiroki Furuta, Ofir Nachum, Kuang-Huei Lee, Yutaka Matsuo, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. *In ICLR*, 2024. 2, 13

[10] Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. *In ICLR*, 2024. 1, 2, 3

[11] Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models. *arXiv preprint arXiv:2401.13919*, 2024. 2, 13

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 4, 5, 10

[13] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *In ICLR*, 2021. 5, 11

[14] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. *arXiv preprint arXiv:2312.08914*, 2023. 2, 13

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

[16] Peter C. Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Alex Goldin, Adam Santoro, and Timothy P. Lillicrap. A data-driven approach for learning to control computers. In *ICML*, 2022. 2, 3, 6, 11, 12, 13

[17] Taichi Iki and Akiko Aizawa. Do berts learn to use browser user interface? exploring multi-step tasks with unified vision-and-language berts. *arXiv preprint arXiv:2203.07828*, 2022. 2

[18] Sheng Jia, Jamie Kiros, and Jimmy Ba. Dom-q-net: Grounded rl on structured language. *In ICLR*, 2019. 2, 13

[19] Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. *In NeurIPS*, 2023. 3

[20] Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *ICML*, 2023. 4, 5, 7, 10, 12, 13

[21] Gang Li and Yang Li. Spotlight: Mobile ui understanding using vision-language models with a focus. *In ICLR*, 2023. 2

[22] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023. 4, 7, 10

[23] Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, and Jason Baldridge. Mapping natural language instructions to mobile ui action sequences. *In ACL*, 2020. 2, 13

[24] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. In *ECCV*, 2022. 4

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 12, 13

[26] Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. *In ICLR*, 2018. 2, 13

[27] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023. 4, 7, 10

[28] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *In NeurIPS*, 2023. 4, 7

[29] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1, 3, 13

[30] Peter Shaw, Mandar Joshi, James Cohan, Jonathan Berant, Panupong Pasupat, Hexiang Hu, Urvashi Khandelwal, Kenton Lee, and Kristina Toutanova. From pixels to ui actions: Learning to follow instructions via graphical user interfaces. *In NeurIPS*, 2023. 2, 13

[31] Abishek Sridhar, Robert Lo, Frank F Xu, Hao Zhu, and Shuyan Zhou. Hierarchical prompting assists large language model on web navigation. *In EMNLP*, 2023. 1, 2, 3

[32] Liangtai Sun, Xingyu Chen, Lu Chen, Tianle Dai, Zichen Zhu, and Kai Yu. Meta-gui: Towards multi-modal conversational agents on mobile gui. *In EMNLP*, 2022. 2, 13

[33] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023. 1, 3

[34] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 1, 3

[35] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *In EMNLP*, 2022. 3

[36] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *In NeurIPS*, 2022. 1, 2, 11, 12, 13

[37] Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*, 2024. 2, 13

Appendices

In this supplementary material, we provide details omitted in the main text.

- Appendix A: Model implementation & training details (cf. §3.3, §3.4, and §4 of the main text).
- Appendix B: Dataset details (cf. §4 of the main text).
- Appendix C: Additional experiments (cf. §4.2 of the main text).

A. Model implementation & training details

As mentioned in §1 of the main text, we implement DUAL-VCR on top of MindAct algorithm [8]. We exactly follow its implementation[3] but provide the details for reference.

A.1. DUAL-VCR-enhanced element ranker

MindAct utilizes a small ranking LM to measure the importance of each element e_t for action prediction. Concretely, at each time step t, the ranking LM takes the element's HTML text tokens h_{e_t}, the task description q, and the previous actions $\{a_1, a_2, \cdots, a_{t-1}\}$ as input and outputs its importance,

$$s_{e_t} = f(q, h_{e_t}, \{a_1, a_2, \cdots, a_{t-1}\}) \quad (2)$$

DUAL-VCR aims to expand this ranking LM to integrate (i) each element's visual features and textual features and (ii) both the candidate element and its neighbor elements. (See Figure 4 of the main text for an illustration.)

Integrating visual and textual features. We first extract each element's visual features from the Pix2Struct Vision Transformer (ViT) [20], pre-trained on webpage screenshots. Concretely, Pix2Struct learns rich representations of webpages by asking to predict an HTML-based parse from a masked screenshot. We input the whole screenshot I_t to $\text{Pix2Struct}_{\text{base}}$ and apply RoIAlign [12] on its output embeddings to obtain the element's visual features v_{e_t} based on its bounding box. On the HTML document side, we extract the element's HTML text h_{e_t}, using the triplet of its ID, HTML text, and bounding box provided in the HTML document.

Intergrating visual neighbor elements. Based on our key insight on webpages—web developers tend to arrange semantically relevant and task-related elements in proximity to each other on the screenshot to enhance user experiences—we contextualize each element e_t with its "visual" neighboring elements M_{e_t}. We measure the center points of all elements in the screenshot using their bounding boxes and calculate their pairwise Euclidean distances[4]. For each *candidate* element to be ranked by MindAct, we search for the closest M elements to form its context jointly.

Aligning visual and textual embedding spaces. After obtaining each element's visual features v_{e_t} and textual features h_{e_t}, we align them in the same embedding space. Following the recent practice of vision-and-language models (*e.g.*, BLIP-2 [22], LLaVA-1.5 [27]), we apply two linear projection layers W to map visual features into the textual embedding space. We then introduce a learnable positional embedding to (i) pair each projected visual feature u_{e_t} with its associated text tokens h_{e_t} and (ii) encode the relative distance between the candidate element e_t and its neighboring elements M_{e_t}. Concretely, we add the same positional embedding p_{e_t} to the candidate element's (projected) visual feature u_{e_t} and textual feature h_{e_t}. Besides, we sort the neighbors M_{e_t} based on their spatial distances from the candidate element e_t. We then encode the relative positional embedding $p_{m^k_{e_t}}$ (based on the spatial distance from the candidate) to each neighbor element's visual features $u_{m^k_{e_t}}$ and corresponding text tokens $h_{m^k_{e_t}}$. We denote the set of the neighbors' visual features by $U_{M_{e_t}}$. Similarly,

[3] https://github.com/OSU-NLP-Group/Mind2Web

[4] https://scikit-learn.org

$H_{M_{e_t}}$ and $P_{M_{e_t}}$ represent the set of their textual features and that of their positional embeddings, respectively. These positionally encoded visual and textual token embeddings (of the candidate and the neighbor elements) are passed into the ranking LM f; the visual features are prepended to the textual embeddings, serving as soft visual prompts,

$$s_{e_t} = f(q, R_{e_t}, \{a_1, a_2, \cdots, a_{t-1}\}),$$
$$R_{e_t} = [u_{e_t} + p_{e_t}; U_{M_{e_t}} + P_{M_{e_t}}; h_{e_t} + p_{e_t}; H_{M_{e_t}} + P_{M_{e_t}}] \quad (3)$$

Training Details. In training, we only learn the projection layer W, the positional embeddings P, and the ranking LM f while keeping the ViT frozen. For the ranking LM, we use DeBERTa$_{\text{base}}$ [13], a small encoder-only LM. We exactly follow the configuration of MindAct. Specifically, we train the LM (together with a linear classifier) with a batch size of 32 and a learning rate of 3e-5 for 5 epochs. The LM outputs the element's importance score through a sigmoid activation function. The score is optimized with a binary cross-entropy loss, where the ground-truth element serves as a positive example, and elements randomly sampled from the webpage are considered negative examples. The LM is trained on a single Nvidia A6000 48GB GPU. During inference, we score all candidate elements in the webpage and select top-K elements for the action predictor.

A.2. DUAL-VCR-enhanced action predictor

Due to the high computational cost of directly passing an entire HTML document into LLMs, MindAct [8] restricts its input to only the top-K candidate elements selected from the ranking LM. Concretely, MindAct combines the selected elements into an HTML snippet H_t and feeds it into an LLM g, along with the task description q ("Find one-way flights from New York to Toronto.") and the previous actions $\{a_1, a_2, \cdots, a_{t-1}\}$ ("Type New York in the From box"). At each time step t, the objective is to predict an action a_t, composing of the target element e_t (*e.g.*, "[textbox] To") and its associated operation o_t (*e.g.*, "Type Toronto"),

$$a_t = g(q, H_t, \{a_1, a_2, \cdots, a_{t-1}\}),$$
$$a_t : \{e_t, o_t\} \quad (4)$$

We note that MindAct converts the target element prediction problem into multiple-choice question-answering. Instead of directly generating the target element, they split top-K candidates into multiple clusters of five element options (including the "None" option) and ask the LLM to pick one element from each cluster. If more than one element is selected, they form a new group with the chosen ones and iterate this process until a single element is selected.

The action predictor of DUAL-VCR takes the same input as MindAct, except for appending each candidate element with its neighboring elements. We generate an HTML snippet S_t based on the top-K candidate elements and their adjacent elements, and input the snippet (with the task description and the previous actions) to the LLM g and predict the action a_t,

$$a_t = g(q, S_t, \{a_1, a_2, \cdots, a_{t-1}\}) \quad (5)$$

Training Details. We again adopt the configuration from MindAct. We train Flan-T5$_{\text{base}}$ [6], an instruction fine-tuned encoder-decoder LLM, with a batch size of 32 and a learning rate of 5e-5 for 5 epochs. We optimize its parameters with the language modeling loss on a single Nvidia A6000 48GB GPU.

B. Dataset Details

Mind2Web [8] recently proposed the first real-world web navigation benchmark, consisting of over 2,000 open-ended tasks from more than 100 real-world websites. They collect the websites across 31 diverse domains, including travel, shopping, entertainment, public service, etc. Unlike other existing benchmarks [16, 36] limited to simulated environments, Mind2Web instead focuses on real-world environments (Table 10). For instance, Mind2Web provides real-world websites with rich content, including thousands of HTML elements, tens of thousands of HTML tokens, and 7.3 web-related actions per task on average.

Data Collection. Given a real-world website (*e.g.*, an airline website), Mind2Web first asks annotators to write open-ended realistic tasks (*e.g.*, "Find one-way flights from New York to Toronto.") relevant to the website. The workers are then required to complete the defined task with a sequence of actions. Specifically, each action is composed of element selection and operation selection. The annotators should first find an element (*e.g.*, "[textbox] From") relevant to the task on the webpage and perform an operation (*e.g.*, "Type New York") on the element.

Dataset Split. The Mind2Web dataset provides a training split with 1,009 real-world tasks collected from 73 websites. Each task consists of a sequence of action samples. In total, there exist 7,775 samples in the training split. Mind2Web evaluates a web agent on three different test splits. **Test$_{\text{Cross-Domain}}$** measures the agent's generalizability to a new domain where it has not seen any websites or tasks associated with that domain during training. The split contains 912 tasks with 5,911 samples from 73 real-world websites. In **Test$_{\text{Cross-Website}}$**, while the agent is not exposed to test websites, it is trained on websites from the same domain and potentially with similar tasks. This configuration enables us to evaluate the agent's capacity to adapt to entirely new websites within familiar domains and tasks. This split consists of 177 tasks, along with 1,373 samples obtained from 10 websites. **Cross-Task** is a conventional test

Dataset	# Domains	# Websites	Website Type	# Tasks	Avg # Actions	Avg # HTML	
						Elements	Tokens
MiniWoB++ [16]	-	100	Simplified	100	3.6	28	500
Mind2Web [8]	31	137	Real-world	2,350	7.3	1,135	44,402

Table 10. **Detailed Statistics of Mind2Web [8].** Min2Web is the first real-world web navigation benchmark, collecting over 100 real-world websites across various domains. Unlike previous benchmarks [16, 36], Mind2Web provides an extensive amount of real-world webpage content, including over 1K/44K HTML elements/tokens on average.

Ranker	Action Predictor	Cross-Task		
		Ele. Acc	Op. F1	Step SR
MINDACT$_{\text{RANK}}$	MINDACT$_{\text{PRED-LARGE}}$	51.4	75.6	48.7
	DUAL-VCR$_{\text{PRED-LARGE}}$	54.2	79.5	50.9

Table 11. **DUAL-VCR with a larger predictor.** We increase the size of the predictor from Flan-T5$_{\text{base}}$ to Flan-T5$_{\text{large}}$. Even with the larger predictor, DUAL-VCR notably outperforms the baseline, showing the complementarity of DUAL-VCR and LLMs.

split, which is the random 20% of the dataset. The split has 252 tasks with 2,094 samples from 69 websites.

Task Details. The Mind2Web task consists of a sequence of actions, each comprising a pair of an actionable HTML element (*e.g.*, "[textbox] To") and an operation (*e.g.*, "Type Toronto"). Mind2Web provides three common operations: Click, Type, and Select. For Type and Select operations, an additional argument (*e.g.*, "Toronto") is required.

C. Additional Experiments

More powerful action predictor. We scale up the predictor from Flan-T5$_{\text{base}}$ to Flan-T5$_{\text{large}}$ to check whether our visual neighbors are still beneficial with the larger model. As shown in Table 11, DUAL-VCR still achieves notable gains, suggesting the complementary capabilities of LLMs and our visual neighbors.

Neighbors from an HTML tree. An HTML document can be represented as a DOM tree, a hierarchical tree of HTML objects (*e.g.*, Element: <head>). Thus, we can also extract each element's neighbors from the HTML tree. We compare the tree-based neighbors with our neighbors obtained from the screenshot (Table 12). Our visual neighbors (DUAL-VCR$_{\text{PRED}}$) significantly outperform those defined by the HTML tree (HTMLTREENEI$_{\text{PRED}}$), suggesting that visual-spatial context is more beneficial.

Ranker with whole visual tokens. In §4.2 of the main text, we show that DUAL-VCR (*i.e.*, the use of visual neighbors) is more effective than the use of the entire image for web navigation (*e.g.*, DUAL-VCR$_{\text{PRED}}$ vs. WHOLEIMAGE$_{\text{PRED}}$, DUAL-VCR$_{\text{VNEI-TXT+VIS}}$ vs. WHOLEIMAGE$_{\text{RANK}}$). To further substantiate the efficacy of DUAL-VCR over using the whole image, we conduct additional experiments (Table 12). Specifically, we train a ranker (WHOLEVISTOK$_{\text{RANK}}$) using *all visual tokens* extracted from the whole image based on the Pix2Struct ViT [20]. Like the previous results in the main text, WHOLEVISTOK$_{\text{RANK}}$ outperforms the baseline (*e.g.*, 44.1% vs. 42.0%), suggesting the benefit of utilizing the entire image. However, WHOLEVISTOK$_{\text{RANK}}$ falls short of DUAL-VCR$_{\text{VNEI-TXT+VIS}}$ (46.0%), which uses significantly fewer inputs (*i.e.*, only neighboring elements). This again supports the advantages of DUAL-VCR over the whole image regarding computational efficiency and performance.

Ranker	Action Predictor	Cross-Task Ele. Acc
MINDACT$_{\text{RANK}}$	MINDACT$_{\text{PRED}}$	42.0
	WHOLEIMAGE$_{\text{PRED}}$	43.6
	HTMLTREENEI$_{\text{PRED}}$	43.8
	DUAL-VCR$_{\text{PRED}}$	44.4
WHOLEIMAGE$_{\text{RANK}}$	MINDACT$_{\text{PRED}}$	43.9
WHOLEVISTOK$_{\text{RANK}}$		44.1
DUAL-VCR$_{\text{VNEI-TXT}}$		44.6
DUAL-VCR$_{\text{VNEI-TXT+VIS}}$		**46.0**
-	WHOLEHTML$_{\text{PRED}}$	38.6

Table 12. **Additional results for Table 6 in the main text.** Our neighbors defined by a screenshot (DUAL-VCR$_{\text{PRED}}$) notably outperform the neighbors defined by an HTML tree (HTMLTREENEI$_{\text{PRED}}$). Moreover, DUAL-VCR$_{\text{VNEI-TXT+VIS}}$ is significantly better than WHOLEVISTOK$_{\text{RANK}}$, which uses all visual tokens of the entire image. This again highlights the benefit of DUAL-VCR in both computational efficiency and performance.

Type of pre-trained visual features. Table 13 summarizes the importance of the type of pre-trained visual features on web navigation. As discussed in §3.2 of the main text, to train the ranker, we extract the element's visual features using Pix2Struct [20]'s VIT, pre-trained on webpage screenshots. We investigate if these pre-trained "screenshot" visual features (DUAL-VCR$_{\text{VNEI-TXT+VIS-WEB}}$) indeed contain meaningful HTML context for downstream web navigation tasks. Concretely, we compare them with features extracted from ViT pre-trained on COCO [25], an object recognition benchmark containing common objects in "natural images". We denote a ranker using the COCO visual features by DUAL-VCR$_{\text{VNEI-TXT+VIS-COCO}}$. We first ob-

Ranker	Cross-Task		
	Ele. Acc	Op. F1	Step SR
DUAL-VCR$_{\text{VNEI-TXT}}$	44.6	75.7	43.2
DUAL-VCR$_{\text{VNEI-TXT+VIS-COCO}}$	45.2	76.3	43.4
DUAL-VCR$_{\text{VNEI-TXT+VIS-WEB}}$	**46.0**	**78.6**	**44.8**

Table 13. **Effects of different types of pre-trained visual features.** The pre-trained screenshot visual features [20] are more beneficial on the downstream web navigation than those extracted from ViT pre-trained on natural images of COCO [25].

serve that DUAL-VCR$_{\text{VNEI-TXT+VIS-COCO}}$ outperforms DUAL-VCR$_{\text{VNEI-TXT}}$ that only leverages elements' HTML text features to train the ranker (*e.g.*, 45.2% vs. 44.6% on Ele. Acc). This implies that even if visual features are from a different domain (*i.e.*, natural images), incorporating them is still helpful in web navigation tasks. However, compared to DUAL-VCR$_{\text{VNEI-TXT+VIS-WEB}}$, which uses both HTML visual and textual features, DUAL-VCR$_{\text{VNEI-TXT+VIS-COCO}}$ performs less (*e.g.*, 46.0% vs. 45.2% on Ele. Acc). This highlights that the pre-trained "screenshot" visual features indeed contain HTML-related context, which benefits more in completing the downstream web navigation tasks.

Existing/Concurrent Works. A number of previous studies [2, 16, 18, 23, 26, 30–32, 36] have explored web navigation but mainly worked on *simplified* websites [16, 36], which deviate from the focus of our study. Our attention is instead directed towards *real-world* scenarios involving various real-world websites with extensive raw HTML documents (*e.g.*, Mind2Web). We have identified a few *concurrent* works [3, 9–11, 14, 37] exploring Mind2Web, but they mostly focus on (i) large-scale pre-training, requiring substantial amounts of pre-training HTML data, or (ii) evaluating the potential of recent vision-and-language models (*e.g.*, GPT4-V [29]) as a web agent. As their codes or pre-training datasets have not been released yet, replicating their work would be prohibitively costly. We thus do not consider them in our studies.