Dual-View Visual Contextualization for Web Navigation (Supplementary Material)

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

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 [4]. We exactly follow its implementation[1] 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,

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

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) [13], 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 Pix2Struct$_{\text{base}}$ and apply RoIAlign [8] 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[2]. 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 [14], LLaVA-1.5 [18]), 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, $H_{M_{e_t}}$ 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

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

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

textual embeddings, serving as soft visual prompts,

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

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$_{\text{base}}$ [9], 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 [4] 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"),

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

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,

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

Training Details. We again adopt the configuration from MindAct. We train Flan-T5$_{\text{base}}$ [3], 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 [4] 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 [11, 23] limited to simulated environments, Mind2Web instead focuses on real-world environments (Table A). 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 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:

Dataset	# Domains	# Websites	Website Type	# Tasks	Avg # Actions	Avg # HTML	
						Elements	Tokens
MiniWoB++ [11]	-	100	Simplified	100	3.6	28	500
Mind2Web [4]	31	137	Real-world	2,350	7.3	1,135	44,402

Table A. **Detailed Statistics of Mind2Web [4].** Min2Web is the first real-world web navigation benchmark, collecting over 100 real-world websites across various domains. Unlike previous benchmarks [11, 23], 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$_{\text{RANK}}$		51.4	75.6	48.7
		54.2	79.5	50.9

Table B. **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.

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 B, 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 C). 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 C). Specifically, we train a ranker (WHOLEVISTOK$_{\text{RANK}}$) using *all visual tokens* extracted from the whole image based on the Pix2Struct ViT [13]. 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 C. **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 D 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 [13]'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 [16], 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 observe 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

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 D. **Effects of different types of pre-trained visual features.** The pre-trained screenshot visual features [13] are more beneficial on the downstream web navigation than those extracted from ViT pre-trained on natural images of COCO [16].

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 [1, 11, 12, 15, 17, 20–23] have explored web navigation but mainly worked on *simplified* websites [11, 23], 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 [2, 5–7, 10, 24] 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 [19]) 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.

References

[1] 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. 4

[2] 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. 4

[3] 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. 2

[4] 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

[5] 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. 4

[6] 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.

[7] 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. 4

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1

[9] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *In ICLR*, 2021. 2

[10] 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. 4

[11] 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, 4

[12] Sheng Jia, Jamie Kiros, and Jimmy Ba. Dom-q-net: Grounded rl on structured language. *In ICLR*, 2019. 4

[13] 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. 1, 3, 4

[14] 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. 1

[15] Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, and Jason Baldridge. Mapping natural language instructions to mobile ui action sequences. *In ACL*, 2020. 4

[16] 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. 3, 4

[17] Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. *In ICLR*, 2018. 4

[18] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023. 1

[19] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 4

[20] 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. 4

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

[22] 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.

[23] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *In NeurIPS*, 2022. 2, 3, 4

[24] 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. 4