# WebMap: Improving LLM Web Agents with Semantic Search for Relevant Web Pages

Michal Spiegel[1,2] and Aleš Horák[1]

[1] Masaryk University
[2] Kempelen Institute of Intelligent Technologies

**Abstract.** The development of autonomous intelligent agents capable of complex decision-making and task-solving within specific environments remains a key challenge in Artificial Intelligence. This work focuses on autonomous agents navigating websites and solving tasks on the web through a browser, with experiments conducted in both English and Czech to evaluate performance across different linguistic contexts. It highlights the limitations of current state-of-the-art web agents, which lack in-domain knowledge about the websites they operate on and often fail to navigate to critical resources. To address this, we propose WebMap, a novel approach that preprocesses the website in advance and retrieves only task-relevant web pages that serve as a shortcut to essential resources. Experimental validation on the WebArena dataset demonstrates that WebMap achieves a 15% relative gain in task completion over the current leading method.

**Keywords:** autonomous web navigation, autonomous LLM agents, LLM reasoning, HTML understanding, vector databases.

## 1 Introduction

One of the key challenges in Artificial Intelligence has long been developing autonomous intelligent agents capable of complex decision-making and task-solving within specific environments. An agent is a system that operates independently, making decisions and taking actions to achieve specific goals within its environment without direct human intervention. These agents can perceive their surroundings, reason about information, and execute actions autonomously.

This work will focus on autonomous agents navigating websites and solving tasks on the web. In this case, the agent operates on the Internet through the use of a browser. The emphasis will be on web agents designed to function within a single website rather than general-purpose agents that work across the entire web.

An intelligent agent could effectively serve as a natural language interface to these web-based systems. That would mean the user could perform any task just by issuing a natural language command. This would streamline the

A. Horák, P. Rychlý, A. Rambousek (eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2024, pp. 139–152, 2024. 

process without requiring prerequisite knowledge or training from the user's side, making the interaction with the system much more effective and accessible.

A major flaw hindering the performance of the current state-of-the-art web agents is the lack of any in-domain knowledge about the website and its contents in advance. Because the agent lacks a comprehensive understanding of the entire website, it has a tendency to abandon tasks prematurely or search and explore the web pages using a trial-and-error strategy, which is ineffective and wastes time and resources. This happens because the model has no contextual information about the whole website and its functionalities and only ever sees isolated web pages, lacking a bigger picture view of the whole system.

This work proposes WebMap, a novel approach that improves web agents by providing more in-domain knowledge. WebMap preprocesses the website in advance, digesting and processing its contents into a semantic map of the whole website, all of its web pages, and functionalities. When confronted with a new task it retrieves only task-relevant URLs that serve as a shortcut for the agent to only the relevant web pages. This allows the agent to skip much of the work spent searching and exploring the website and effectively makes the task significantly easier.

We experimentally validate the proposed method on a subset of tasks from the WebArena dataset [30]. WebMap achieves 15% more tasks in relative gain over the state-of-the-art baseline method [15] and shows significant promise for further improvement.

The key contributions of this work are as follows:

- **Design and implementation of WebMap**, a novel method that improves the accuracy and efficiency of autonomous agents on the web
- **Experimental evaluation and analysis** of the proposed method on the WebArena benchmark
- **Error analysis** of WebMap and baseline approaches on the WebArena benchmark, highlighting important challenges for future work

## 2 Related Work

Recent advancements in autonomous web navigation have produced state-of-the-art techniques focused on empowering agents to execute tasks on real-world websites using natural language instructions. WebLINX [19] supports conversational web navigation, enabling agents to follow complex, dialogue-based instructions effectively. Synapse [29] leverages state abstraction and exemplar prompting to filter out task-irrelevant information and enhance multi-step decision-making. WebVoyager [11] uses multimodal inputs, including images and HTML, which enriches contextual understanding and improves interaction with dynamic web content. AutoWebGLM [15] optimizes page-level comprehension by simplifying HTML structures and utilizing reinforcement learning to refine the agent's navigation strategy.

Although each of these approaches innovates in areas like HTML simplification, multimodal input integration, and multi-step instruction-following, they all lack a persistent memory of a website's overarching structure. Consequently, these agents operate on isolated individual pages, relying on trial-and-error navigation rather than drawing on in-domain knowledge that could enable more streamlined, expert-level task completion.

By contrast, WebMap introduces a novel solution through the construction of a semantic map of the entire website, providing agents with comprehensive in-domain knowledge of its structure, contents, and functionalities. This preprocessed map allows WebMap to bypass trial-and-error navigation, enabling agents to access only task-relevant sections of a website. As a result, WebMap significantly enhances navigational efficiency and task success, marking a major step forward in autonomous web navigation.

## 3 WebMap - Converting Website HTML into a Vector Database

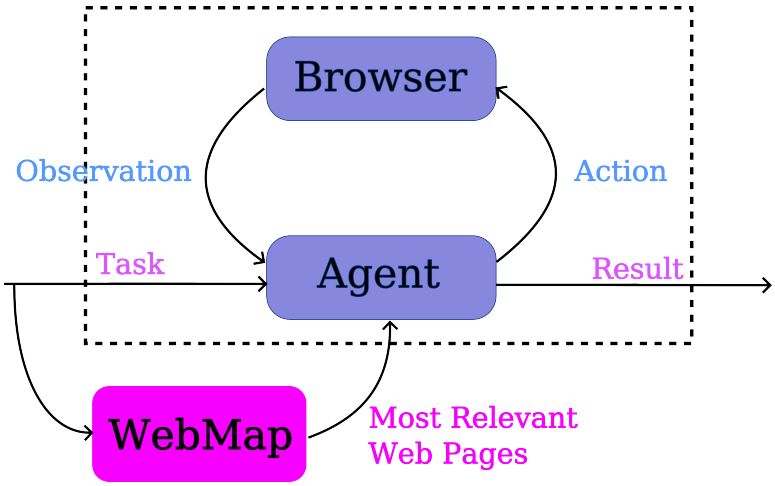


Fig. 1: The high-level workflow of the proposed solution. It takes the user task description as input and returns the most relevant web pages regarding the given task description.

Fig. 1 describes the high-level workflow of the proposed method. WebMap is completely separate and independent from the agent or the environment implementation. It works as a vector database storing representations of the individual web pages (URLs). For each new incoming task description for the agent, the task description is also used to search in the vector database for the most similar, most relevant web pages in regard to this task. Each web page is represented by its unique URL. The most relevant web pages are then passed to the agent which can use this information to start from a better position or

perform a more detailed analysis of the contents of the most relevant web pages to perform a more effective search through the possible web pages than would be possible without WebMap.

The four main components, which are outlined in the Fig. 2, are

- **Scraping** - this component is responsible for obtaining the raw, unprocessed web pages (HTML)
- **Preprocessing** - HTML web pages contain redundant elements, requiring cleaning and processing to achieve better representation.
- **Transforming data into a database** - to ensure effective search, it's necessary to convert them into representations that preserve the semantics and store these in a database that enables effective retrieval based on similarity or relevance

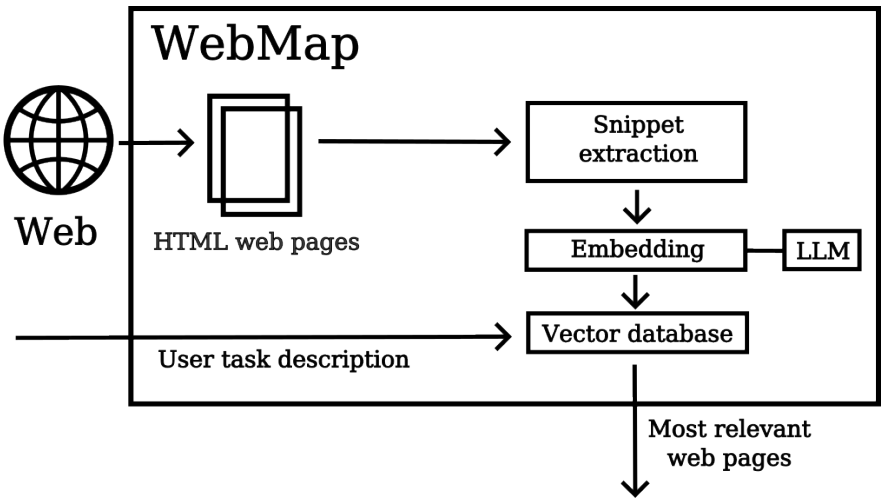


Fig. 2: WebMap and details its main components. WebMap first crawls the specified domain and retrieves a list of HTML web pages. Each HTML web page is then split into smaller chunks using snippet extraction and embedded using an LLM into a representation in vector space. The representations are used for indexing the snippets in a vector database. Given a user task description, this database is used to retrieve the most relevant web pages for completing the task.

The goal of WebMap is to convert raw data into a searchable database of web pages, allowing for efficient retrieval based on task descriptions. To achieve this, we represent text in vector space using state-of-the-art LLMs, which can encode semantic information. Preprocessing removes redundant elements like CSS styling, metadata, and unnecessary structural information to ensure efficient transformation and precise similarity search.

### 3.1 Preprocessing HTML Web Pages

Preprocessing HTML is crucial to reduce data volume and make similarity-based embedding with LLMs feasible, as typical web pages exceed model token limits. To streamline content, we:

- Remove non-semantic elements (e.g., metadata, stylesheets, scripts).
- Simplify structure by replacing tags (e.g., div, span) with inner content.
- Retain only visible attributes (e.g., type, placeholder).

**Extracting Short and Meaningful Snippets** We extract smaller snippets around salient elements, such as buttons and input fields, to simplify long HTML pages, adding surrounding context to clarify their function. [3]

### 3.2 Transforming Data into a Vector Database

After snippet extraction, we store them in a vector database to support similarity searches by converting snippets into vector representations for distance-based metrics like Euclidean and cosine similarity. The following sections detail embedding creation and vector database search.

**HTML Snippet Embeddings** Pre-trained LLMs generate high-dimensional vector embeddings for text, though they are not optimized for HTML, potentially making computation less efficient. While simpler methods like TF-IDF encode shallow semantics, specialized HTML models like DOM-LM [9], LayoutLM [28], MarkupLM [17] or HTLM [1] generally perform suboptimally against general LLMs [10]. HTML cleaning further simplifies text structure. We evaluate top models from the MTEB benchmark [23]:

- SFR-Embedding-Mistral [22]
- text-embedding-gecko [16]
- gte-large-en-v1.5 [18]
- LLM2Vec-Meta-Llama-3-8B-Instruct-mntp-supervised [5]

To see a full evaluation and comparison of different embedding models and approaches, see Appendix C. Based on our evaluation, SFR-Embedding-Mistral with instruction turns out to be the most suitable model for this task.

## 4 Experimental Results & Discussion

This section presents the experimental results of the proposed WebMap method, assessing various implementations (e.g., different embedding techniques). Experiments were conducted in both English and Czech to test the method's effectiveness across linguistic contexts. However, due to the lack of existing Czech

[3] Inspired by [10]

datasets for web navigation evaluation, primary results focus on a widely used English dataset to maintain comparability with established research and benchmarks. Notably, this study also evaluated WebMap using Masaryk University's Information System (IS MU), a complex platform that supports a range of academic and administrative tasks and offers structured, semantically rich content in both English and Czech. Qualitative evaluations on the Czech IS MU interface, using multimodal versions of the pre-trained models, demonstrated correlations with human judgments comparable to those achieved on the English WebArena dataset.

### 4.1 WebArena Evaluation Environment and Dataset

Among recent datasets, WebArena uniquely offers live, self-hostable websites, enabling the mining of in-domain information. It includes about 800 complex, realistic tasks, highlighting the practical applicability of WebMap and addressing the challenges posed by real-world website complexity.

## 5 Baseline Agent

For a more credible evaluation and comparison, it is necessary to establish a baseline agent. We evaluate multiple different approaches. However, since only Gemini 1.5 Pro reaches non-zero accuracy, we do not provide any explicit evaluation results for these approaches. Notably, we have experimented with the following approaches as baseline agents:

- Heuristic rule-based agents (e.g. text similarity)
- LLM-based approaches
  - Using in-context learning
  - Different agent architectures
    - Recursively Criticizes and Improves (RCI) [13]
    - Chain-of-Thought (CoT) [27]
    - AutoWebGLM [15]
  - Different underlying LLMs
    - T5-based encoder-decoder models (FLAN-T5 [8], CoT fine-tuned variants [14])
    - Open-source decoder-only models (Vicuna [7], Mistral [12], Llama 2 [26], Llama 3 [2], web navigation fine-tuned variants [19])
    - Multimodal models (e.g. Fuyu [4], Gemini Pro Vision [3])
    - Gemini 1.5 Pro [3]

The only agent that was able to achieve moderate success on the WebArena dataset was a combination of AutoWebGLM, Gemini 1.5 Pro, few-shot learning and Chain-of-Thought prompting. The agent achieved an accuracy rate of 15% which is comparable to the best-performing WebArena evaluated GPT4-based agent which achieved 14% accuracy on the dataset. Therefore, the baseline agent is composed as follows:

- **AutoWebGLM** provides the agent with a simplified form of HTML as an observation space and defines the action space
- **Gemini 1.5 Pro** acts as the reasoning engine for input processing, planning and action prediction
- **In-context examples** are given to the model in the prompt to improve downstream performance
- **Chain-of-Thought** prompting [27] is used to improve reasoning abilities

### 5.1 End-to-end Evaluation of the Best-performing Strategy

To comprehensively evaluate the whole method, we perform end-to-end evaluation of WebMap on a subset of tasks from the WebArena dataset. The Table 1 shows results of multiple experiments done with different base models and compared to their version using WebMap. The results show that WebMap consistently improves the base models with a mean approximate relative gain of 30% over just using base models.

Table 1: The performance of WebMap on multiple base models.

| Agent | Accuracy | Relative gain |
|---|---|---|
| AutowebGLM+Gemini | 17% | – |
| AutowebGLM+Gemini+WebMap | 20% | 15% |

## 6 Conclusion

Autonomous web navigation promises a revolution in human-computer interaction, providing more accessible user interfaces and streamlining complex processes with the use of intelligent agents. However, current autonomous web agents substantially fall behind in performance and effectiveness. Using in-context We analyse the current state-of-the-art (SOTA) methods of autonomous web navigation and find that in a significant number of cases, the agents completely fail to navigate to relevant web pages and resources important for completing the task on the web. This is confirmed by an error analysis of a SOTA agent on a subset of tasks from the WebArena dataset (Table 2).

This work explores a problem of effectively representing website contents to enable fast and accurate search for relevant resources. We propose WebMap, a solution for processing website contents into an information retrieval system that enables effective search for relevant web pages based on a user task description. The main problem lies in effective processing and understanding of HTML. We design the method, experimentally evaluate multiple alternatives and provide experimental validation on the WebArena evaluation dataset. WebMap achieves improved accuracy of over 15% in relative gain over state-of-the-art

baseline agent on real-world complex web navigation tasks from the WebArena dataset.

WebMap significantly impacts the current state of research on autonomous web navigation agents by proving the importance of modeling in-domain information as highlighted in the provided error analysis, including but not limited to representing website contents, standard interfaces, or usage of functionalities.

**Acknowledgements.** Computational resources were provided by the e-INFRA CZ project (ID:90254), supported by the Ministry of Education, Youth and Sports of the Czech Republic.

## References

1. Aghajanyan, A., Okhonko, D., Lewis, M., Joshi, M., Xu, H., Ghosh, G., Zettlemoyer, L.: HTLM: Hyper-Text Pre-Training and Prompting of Language Models (Jul 2021), http://arxiv.org/abs/2107.06955, arXiv:2107.06955 [cs]
2. AI@Meta: Llama 3 model card (2024)
3. et al., P.G.: Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context (2024), https://arxiv.org/abs/2403.05530
4. Bavishi, R., Elsen, E., Hawthorne, C., Nye, M., Odena, A., Somani, A., Taşırlar, S.: Introducing our multimodal models (2023), https://www.adept.ai/blog/fuyu-8b
5. BehnamGhader, P., Adlakha, V., Mosbach, M., Bahdanau, D., Chapados, N., Reddy, S.: LLM2Vec: Large Language Models Are Secretly Powerful Text Encoders (Apr 2024), http://arxiv.org/abs/2404.05961, arXiv:2404.05961 [cs]
6. Chen, Q., Geng, X., Rosset, C., Buractaon, C., Lu, J., Shen, T., Zhou, K., Xiong, C., Gong, Y., Bennett, P., Craswell, N., Xie, X., Yang, F., Tower, B., Rao, N., Dong, A., Jiang, W., Liu, Z., Li, M., Liu, C., Li, Z., Majumder, R., Neville, J., Oakley, A., Risvik, K.M., Simhadri, H.V., Varma, M., Wang, Y., Yang, L., Yang, M., Zhang, C.: MS MARCO Web Search: a Large-scale Information-rich Web Dataset with Millions of Real Click Labels. In: Companion Proceedings of the ACM on Web Conference 2024. pp. 292–301 (May 2024). https://doi.org/10.1145/3589335.3648327, http://arxiv.org/abs/2405.07526, arXiv:2405.07526 [cs]
7. Chiang, W.L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J.E., Stoica, I., Xing, E.P.: Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality (March 2023), https://lmsys.org/blog/2023-03-30-vicuna/
8. Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S.S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E.H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q.V., Wei, J.: Scaling Instruction-Finetuned Language Models (Dec 2022). https://doi.org/10.48550/arXiv.2210.11416, http://arxiv.org/abs/2210.11416, arXiv:2210.11416 [cs]
9. Deng, X., Shiralkar, P., Lockard, C., Huang, B., Sun, H.: DOM-LM: Learning Generalizable Representations for HTML Documents (Jan 2022), http://arxiv.org/abs/2201.10608, arXiv:2201.10608 [cs]

10. Gur, I., Nachum, O., Miao, Y., Safdari, M., Huang, A., Chowdhery, A., Narang, S., Fiedel, N., Faust, A.: Understanding HTML with Large Language Models (May 2023). https://doi.org/10.48550/arXiv.2210.03945, http://arxiv.org/abs/2210.03945, arXiv:2210.03945 [cs]
11. He, H., Yao, W., Ma, K., Yu, W., Dai, Y., Zhang, H., Lan, Z., Yu, D.: WebVoyager: Building an End-to-End Web Agent with Large Multimodal Models (Feb 2024), http://arxiv.org/abs/2401.13919, arXiv:2401.13919 [cs]
12. Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., Casas, D.d.l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M.A., Stock, P., Scao, T.L., Lavril, T., Wang, T., Lacroix, T., Sayed, W.E.: Mistral 7B (Oct 2023). https://doi.org/10.48550/arXiv.2310.06825, http://arxiv.org/abs/2310.06825, arXiv:2310.06825 [cs]
13. Kim, G., Baldi, P., McAleer, S.: Language Models can Solve Computer Tasks (Jun 2023), http://arxiv.org/abs/2303.17491, arXiv:2303.17491 [cs]
14. Kim, S., Joo, S.J., Kim, D., Jang, J., Ye, S., Shin, J., Seo, M.: The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning. arXiv preprint arXiv:2305.14045 (2023)
15. Lai, H., Liu, X., Iong, I.L., Yao, S., Chen, Y., Shen, P., Yu, H., Zhang, H., Zhang, X., Dong, Y., Tang, J.: AutoWebGLM: Bootstrap And Reinforce A Large Language Model-based Web Navigating Agent (Apr 2024). https://doi.org/10.48550/arXiv.2404.03648, http://arxiv.org/abs/2404.03648, arXiv:2404.03648 [cs]
16. Lee, J., Dai, Z., Ren, X., Chen, B., Cer, D., Cole, J.R., Hui, K., Boratko, M., Kapadia, R., Ding, W., Luan, Y., Duddu, S.M.K., Abrego, G.H., Shi, W., Gupta, N., Kusupati, A., Jain, P., Jonnalagadda, S.R., Chang, M.W., Naim, I.: Gecko: Versatile Text Embeddings Distilled from Large Language Models (Mar 2024). https://doi.org/10.48550/arXiv.2403.20327, http://arxiv.org/abs/2403.20327, arXiv:2403.20327 [cs]
17. Li, J., Xu, Y., Cui, L., Wei, F.: MarkupLM: Pre-training of text and markup language for visually rich document understanding. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 6078–6087. Association for Computational Linguistics, Dublin, Ireland (May 2022). https://doi.org/10.18653/v1/2022.acl-long.420, https://aclanthology.org/2022.acl-long.420
18. Li, Z., Zhang, X., Zhang, Y., Long, D., Xie, P., Zhang, M.: Towards General Text Embeddings with Multi-stage Contrastive Learning (Aug 2023). https://doi.org/10.48550/arXiv.2308.03281, http://arxiv.org/abs/2308.03281, arXiv:2308.03281 [cs]
19. Lù, X.H., Kasner, Z., Reddy, S.: WebLINX: Real-World Website Navigation with Multi-Turn Dialogue (Feb 2024). https://doi.org/10.48550/arXiv.2402.05930, http://arxiv.org/abs/2402.05930, arXiv:2402.05930 [cs]
20. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
21. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval, chap. Evaluation in information retrieval. Cambridge University Press (2008)
22. Meng, R., Liu, Y., Joty, S.R., Xiong, C., Zhou, Y., Yavuz, S.: Sfr-embedding-mistral:enhance text retrieval with transfer learning. Salesforce AI Research Blog (2024), https://blog.salesforceairesearch.com/sfr-embedded-mistral/

23. Muennighoff, N., Tazi, N., Magne, L., Reimers, N.: MTEB: Massive Text Embedding Benchmark (Mar 2023). https://doi.org/10.48550/arXiv.2210.07316, http://arxiv.org/abs/2210.07316, arXiv:2210.07316 [cs]
24. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. Journal of Machine Learning Research **21**(140), 1–67 (2020), http://jmlr.org/papers/v21/20-074.html
25. Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., Gurevych, I.: BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models (Oct 2021), http://arxiv.org/abs/2104.08663, arXiv:2104.08663 [cs]
26. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C.C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P.S., Lachaux, M.A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E.M., Subramanian, R., Tan, X.E., Tang, B., Taylor, R., Williams, A., Kuan, J.X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., Scialom, T.: Llama 2: Open Foundation and Fine-Tuned Chat Models (Jul 2023). https://doi.org/10.48550/arXiv.2307.09288, http://arxiv.org/abs/2307.09288, arXiv:2307.09288 [cs]
27. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E.H., Le, Q.V., Zhou, D.: Chain-of-thought prompting elicits reasoning in large language models. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. NIPS '22, Curran Associates Inc., Red Hook, NY, USA (2024)
28. Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M.: LayoutLM: Pre-training of Text and Layout for Document Image Understanding. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1192–1200 (Aug 2020). https://doi.org/10.1145/3394486.3403172, http://arxiv.org/abs/1912.13318, arXiv:1912.13318 [cs]
29. Zheng, L., Wang, R., Wang, X., An, B.: Synapse: Trajectory-as-Exemplar Prompting with Memory for Computer Control (Jan 2024). https://doi.org/10.48550/arXiv.2306.07863, http://arxiv.org/abs/2306.07863, arXiv:2306.07863 [cs]
30. Zhou, S., Xu, F.F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., Fried, D., Alon, U., Neubig, G.: WebArena: A Realistic Web Environment for Building Autonomous Agents (Oct 2023), http://arxiv.org/abs/2307.13854, arXiv:2307.13854 [cs]

## A Error Analysis

Despite the encouraging results, experimental evaluation of WebMap still shows a considerable error margin. For this reason, we decide to conduct an error analysis to understand the underlying issues and identify the main problems. The error analysis of both the baseline method and WebMap method with human annotations from the experiments in Section B can be seen in Table 2 and 3. This error analysis has been manually evaluated on the results of

the evaluation of the baseline agent and WebMap with human annotations discussed in Section B. The agents were evaluated on a sample of 63 tasks from the WebArena dataset. The occurrences do not sum up to 100% because many of these errors happen simultaneously in the same tasks, and it is not possible to evaluate certain tasks with only one type of error. The log files from both the evaluation and the error analysis are provided for inspection in attachments.

We identified the following major categories of errors:

- **Controls Understanding** captures all the situations in which the agent successfully navigates to the correct page, understands the task and what is needed to do, but ultimately fails to operate the needed controls correctly. For example, agents often struggle with formatting dates correctly and often produce the incorrect format or hallucinate invalid dates. Agents often forget to perform the last step to submit, show, or confirm selected settings.
- **Incorrect Web Page** is the second most predominant issue. This type of errors is what WebMap tries to solve. Agents have a tendency to navigate incorrectly and do not even find the correct resource.
- **Observation Understanding** describes a situation when the agents needs to extract some information from the observation (e.g. from the HTML), the information is perfectly visible, but the agents fails to recognize it
- **Planning and Reasoning Errors** often occur together often on tasks that require long-horizon planning, the agent forgets a vital step, makes an obvious reasoning error
- **Hallucinations** happen when the agent just straightforward outputs nonsense that just looks good, e.g., says it has successfully finished when it has not done any action at all
- **Evaluation Dataset Fault and not enough information in the observation** are both errors that are not directly the fault of the agent, e.g. there is a mistake in the dataset, in the observation
- **Fail to recover from an error** happens when the agent tries a strategy, the strategy fails and it just gives up, the inability to step back, rethink the problem and try again.

Table 2: The error analysis of the baseline AutowebGLM using Gemini 1.5 Pro.

| Type of Error | Absolute Count | Percentage (%) |
|---|---|---|
| Controls Understanding | 24 | 30 |
| Incorrect Web Page | 18 | 23 |
| Observation Understanding | 16 | 20 |
| Reasoning Error | 10 | 13 |
| Planning Error | 5 | 6 |
| Hallucinations | 4 | 5 |
| Evaluation Dataset Fault | 1 | 1 |
| Not enough information in the observation | 1 | 1 |
| Fail to recover from an error | 1 | 1 |

Table 3: The error analysis of AutowebGLM using Gemini 1.5 Pro together with human-annotated WebMap (Section B)

| Type of Error | Absolute Count | Percentage (%) |
|---|---|---|
| Controls Understanding | 31 | 40 |
| Incorrect Web Page | 0 | 0 |
| Observation Understanding | 10 | 13 |
| Reasoning Error | 6 | 8 |
| Planning Error | 7 | 9 |
| Hallucinations | 2 | 3 |
| Evaluation Dataset Fault | 3 | 4 |
| Not enough information in the observation | 1 | 1 |
| Fail to recover from an error | 7 | 9 |

The error analysis highlights the issue of control understanding that causes problems in 30-46% of tasks. Agents often have problems with operating more complex system interfaces that require multiple steps before the result is shown. They often struggle with correctly formatting dates. Similarly, they have problems with understanding simple protocols for working with certain interfaces, e.g., they often forget last steps such as submitting or confirming changes. At the same time, they often lack a good understanding of the observation. They often incorrectly reason about the observation and miss key information in the observation. We observe that in non-trivial amount cases, the HTML simply does not provide enough information to make the correct decision, even for a human.

## B Evaluation using human annotations

Evaluating WebMap solely on its end-to-end performance in web navigation tasks (e.g., WebArena datasets) limits the ability to analyze its individual components. Thus, it is essential to separately evaluate the information retrieval task.

While there are various datasets for benchmarking information retrieval techniques [25], including those for question answering and web search [6], searching web pages based on task descriptions presents unique challenges. Although web search is similar, differences in query formats and document types complicate direct comparisons. Therefore, we manually annotated a subset of tasks from the WebArena dataset with relevant URLs that align with WebMap's output. This human-annotated gold standard serves two main purposes:

- **Determine Optimal Performance Increase:** Evaluate whether the benefits of improving this method outweigh the challenges.
- **Benchmark Automatic HTML Processing Techniques:** Comparing different HTML preprocessing strategies requires a gold standard for effective evaluation.

We manually annotate 184 tasks from the WebArena evaluation dataset. Each annotation represents the URL of the most relevant web page, the best starting point for successfully solving the current task.

### B.1 Determining the Best Possible Performance Increase

Implementing WebMap presents challenges in HTML understanding and web page information retrieval. Before allocating resources, it's essential to assess its potential for enhancing end-to-end performance in web navigation tasks. To this end, we designed an experiment comparing a baseline AutowebGLM agent with a WebMap agent, using human annotations instead of automated HTML processing.

The baseline agent, based on AutoWebGLM, incorporates a few in-context examples and chain-of-thought prompting, utilizing the Gemini 1.5 Pro LLM. The WebMap agent differs only in that it begins tasks at the URLs specified in the human annotations and is prompted that the current web page is most relevant. This experiment evaluates a subset of 63 tasks from the WebArena dataset, chosen due to computational constraints. The results will gauge WebMap's feasibility in improving task completion.

Table 4: Performance of WebMap using human annotations instead of automatic processing strategies.

| Agent | Accuracy | Relative Gain |
|---|---|---|
| AutowebGLM+Gemini | 17% | – |
| AutowebGLM+Gemini+WebMap (using human annotations) | 23% | $\sim$ 30% |

Table 4 shows that the agent guided by human annotations completed 6

### B.2 Automatic HTML Content Processing Strategies

Section B.4 details design decisions regarding specific WebMap components that require experimental evaluation, including:

- **Embedding Function:** This computes HTML snippet representations. Research indicates conventional LLMs (e.g., T5 [24]) outperform specialized models (e.g., MarkupLM [17]) [10]. The best-performing embedding models are selected based on the Massive Text Embedding Benchmark (MTEB) [23] and evaluated against the human-annotated gold standard.
- **Hyperparameters for Snippet Extraction:** A hyperparameter search is necessary to identify optimal settings, such as the maximum length and tree height of HTML snippets. The impact of snippet length and nested elements on retrieval performance remains unclear, as LLMs were not primarily trained on HTML modeling. Experimental evaluation against the gold standard is the best approach to address this.

### B.3 Evaluation Methodology

To evaluate automatic processing strategies, we use the Mean Reciprocal Rank (MRR) metric, referencing a human-annotated gold standard. This choice is based on WebMap's role as an information retrieval system, which retrieves the most relevant web pages based on user task descriptions [20, p. 1]. Effectiveness is traditionally measured against a gold standard that defines the relevance of documents for specific queries [21]. In this case, each task description corresponds to a single relevant web page, simplifying the relevance assessment.

Selecting appropriate metrics is crucial, as traditional measures like precision and recall are less informative when only one relevant document exists. Since WebMap outputs a ranked list of web pages, the effectiveness is best captured by evaluating the rank of the human-annotated gold standard in this list. MRR is a suitable choice, as it assesses the mean rank of the first relevant document.

### B.4 Effective Representation of HTML Snippets Using LLMs

This section examines methods for representing HTML snippets in vector space, crucial for WebMap's retrieval performance. While specialized architectures often struggle with HTML [10], conventional LLMs (e.g., T5 [24]) show promise in handling complex structures. We evaluate a subset of embedding techniques using the MRR methodology, with results shown in Table 4.

Based on evaluations from the Massive Text Embedding Benchmark (MTEB) [23], we selected techniques from relevant categories like *Overall* and *Retrieval*. The preliminary results suggest that SFR-Embedding-Mistral and gte-large-en-v1.5 show the most potential, with MRRs indicating the gold standard typically ranks within the top 3. However, high standard deviation points to performance instability, prompting further optimization of snippet extraction hyperparameters.

### B.5 Embedding with Instruction

Some embedding models, like LLM2Vec [5], suggest adding instructions to the input text to enhance performance. Testing this with the top-performing SFR-Embedding-Mistral, we found that including an instruction significantly improved the MRR to 0.63. However, this also increased computation time due to the added prompt length.