EVALUATING THE EFFECTIVENESS OF LLMS IN PERFORMING BROWSER TASKS

AI6127: Deep Natural Language Processing

Adnan Azmat G2303265K

Bendale Aneesh Santosh G2303517J

Bugnot Reinelle Jan Cruz G2304329L

Chithra Ramesh Asswin G2302832A

van der Vliet Riemer G2304212K

23 April 2024

Abstract

In this project, we explore the effectiveness of Large Language Models (LLMs) in autonomously performing complex browser tasks across various domains. This is accomplished through two objectives: (1) using vanilla pre-trained LLMs available from hugging face, and (2) aligning a selected LLM for web browsing using different methods such as QLoRA and FewShot Learning. Through these strategies, we aim to evaluate the capabilities of Large Language Models in executing web-based actions. By comparing LLM performance with human operators and proposing algorithms for data collection and annotation, we aim to advance the field of intelligent automation in web navigation, addressing the growing need for enhanced user experience and accessibility.

1 Introduction

The alignment of Large Language Models (LLMs) has recently emerged as a critical area of research in the field of Artificial Intelligence. This project explores 2 common alignment techniques, QLoRA (Quantized Low Rank Adaptation) and Few-Shot Learning, which have demonstrated great potential in boosting the performance and versatility of LLMs across a range of applications. In this project, we have two objectives: (1) To evaluate the capabilities of vanilla open-source LLMs in executing web-based tasks within the WebArena environment, and (2) perform perform model alignment on a target LLM for web browsing task using QLoRA and Fewshot Learning. The WebArena environment, with its dynamic and complex nature, presents several challenges and opportunities for evaluating and enhancing the capabilities of LLMs. Through these experiments, we aim to gain insights into the effectiveness of these alignment techniques and identify areas for further improvement. The findings from this project can hopefully contribute to the ongoing efforts in the AI community to develop high-performing, aligned LLMs that can effectively and efficiently perform web-based tasks, thereby expanding the range of practical applications of these models.

2 Related Works

In the domain of controlling agents through natural language, significant progress has been made in addressing the functional correctness and adaptability in dynamic environments, yet challenges persist. Branavan et al. (2009) and Shi et al. (2017) [1] initially explored the constraints of static states that limit evaluating an agent's functionality and decision-making capabilities. These early studies paved the way for more sophisticated environments, as Liu et al. (2018) [2] began to address the simplification of real-world complexities that restricted task variety.

Recent advances have focused on enhancing agent autonomy and robustness in task execution within interactive digital environments. Nakano et al. (2021) [3] introduced WebGPT, which leverages web search results for answering questions, setting a precedent for integrating web interactions in agent tasks. This was further refined by Gur et al. (2023) [4], who developed a web agent capable of decomposing tasks into sub-tasks and synthesizing Javascript for executions, and Lee et al. (2023) [5] and Shaw et al. (2023) [6], who proposed using screenshots of web pages for action prediction instead of traditional text-based DOM trees. This evolution towards complex task handling and environmental interaction culminated in the development of WebArena by Zhou et al. (2023) [7], which presents a realistic web environment designed to challenge and enhance the capabilities of autonomous agents. **WebArena** not only supports complex task executions but also integrates advanced features like hierarchical planning, state tracking, error recovery, and a multi-modal interaction framework, offering a comprehensive platform for testing and improving agent interaction in a manner closely aligned with human web use.

3 The Environment: WebArena

The WebArena is a highly realistic and reproducible environment developed to facilitate the building and testing of autonomous agents. This environment is crucial because it addresses the gap between the simplified synthetic environments typically used in AI research and the complex, dynamic nature of real-world web interactions. Traditional evaluation environments for autonomous agents tend to oversimplify real-world scenarios, which can lead to a lack of task diversity and complexity. WebArena overcomes these limitations by providing an environment that includes fully functional websites from four common domains: e-commerce, social forum discussions, collaborative software development, and content management. A demonstration of the WebArena environment is shown in Fig. 1. These sites are mimicking their real-world counterparts, along with tools (e.g., maps) and external knowledge bases (e.g., user manuals) to enable human-like problem-solving. [7]

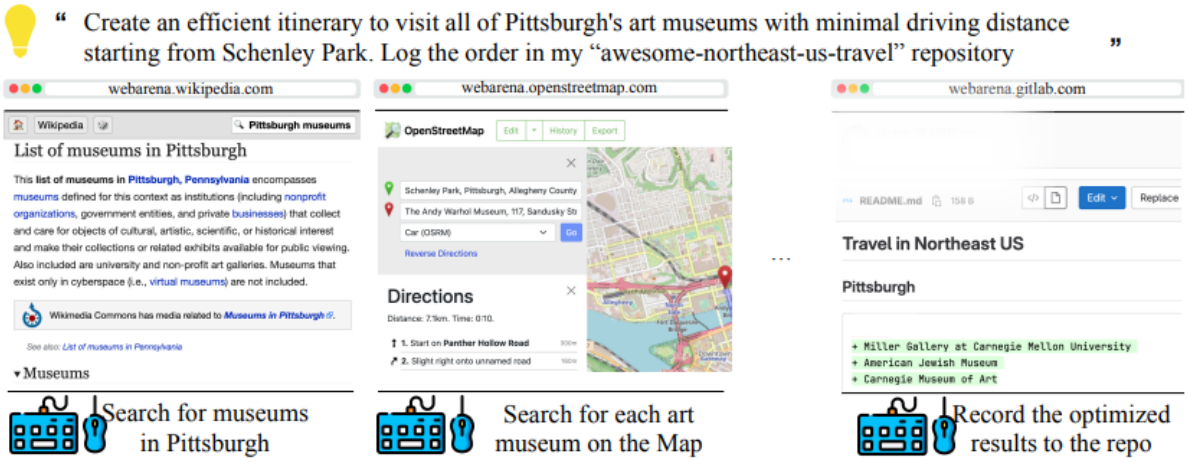

Figure 1: A high-level task that can be fully executed in WebArena. [7]

Even the best-performing agent, based on GPT-4, achieved an end-to-end task success rate of only 14.41%, which is significantly lower than human performance at 78.24%. This gap underscores the need to develop more robust and effective autonomous agents further. The results from WebArena serve as a benchmark for measuring progress in this area. [7].

4 Methodology

4.1 Training data

The first main challenge of this project is obtaining data for training/fine-tuning. WebArena provides 812 testing data points which we can already use for direct inferencing to benchmark our vanilla models' performance with the models presented in the original WebArena paper. However, WebArena does not provide training data instances that we need for LLM finetuning. Hence, the first main step for the project is to find external usable data instances for fine-tuning.

In this regard, we found **Mind2Web**, which encompasses over 2,000 open-ended tasks derived from 137 websites across 31 diverse domains, making it the first of its kind to offer a holistic and challenging landscape for building generalist web agents. One of the dataset's key strengths is its use of real-world websites–similar to WebArena, providing models with the challenge of navigating the complexities and nuances of actual web environments [8].

Figure 2 illustrates the data conversion process we implemented at a high level. The raw HTML pages sourced

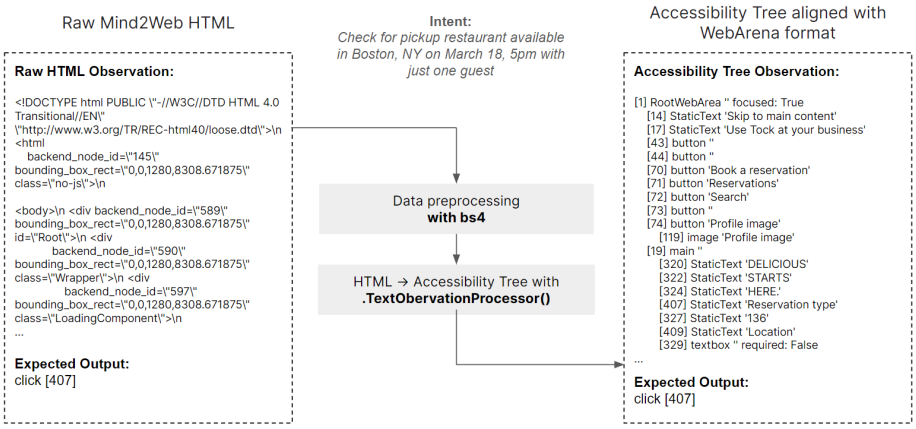

Figure 2: Data Conversion from Mind2Web to WebArena format

from Mind2Web, each containing specific user intents and corresponding observations. Obtained through by tracing subjects actions in performing such intent. These pages contained multiple sequential steps necessary to fulfill each intent. To align with our objectives of organizing data around overarching intents and elemental steps, we transformed these HTML pages into a more structured format similar to the observation format used in The WebArena.

This is done through Playwright to host the websites and using BeautifulSoup (bs4) to perform some intial HTML processing. Notably, these raw pages presented considerable messiness compared to the pristine webpages utilized by WebArena for their processes, prompting reflection on the representativeness of the latter for real-world web content. Following this preprocessing, we employed the TextObservationProcessor (TextObervationProcessor sic.) from WebArena's environment to convert the HTML pages into accessibility trees, mirroring the methodology utilized by WebArena. This step facilitated the transformation of HTML elements into structured observations, a critical component for our dataset's utility.

Consequently, our dataset's features (X) comprised the cleaned observations in accessibility tree format, with the corresponding actions (y) represented the elemental steps necessary to fulfill the user intent.

4.2 Model

Model Name	# Parameters	Context Window	Features
Meta-Llama-3-8B-Instruct	8 billion	8,000	Optimized for dialogue, scalable on consumer hardware, CO2 neutral certification
Meta-Llama-3-70B-Instruct	70 billion	8,000	Enhanced inference capabilities, suitable for commercial and research use in English
Mixtral-8×7B-Instruct-v0.1	46.7 billion	128K	Likely focuses on scalability and fine-tuning for specific tasks

Table 1: Comparison of different Vanilla models used in our experiments

The Table 1 provides a overview of the vanilla models used in our evaluation experiments (objective 1). Here, we elaborate on the primary characteristics that differentiate these models in terms of scale, training, and application focus. All models employ a blend of supervised fine-tuning (SFT) and reinforcement learning with human feedback

(RLHF), with the Meta Llama series also utilizing Grouped-Query Attention (GQA) to enhance model response times and scalability across different computational setups.

For the model alignment experiments (objective 2), we primarily utilized **Mistral-7B-Instruct-v0.2** a state-of-the-art Large Language Model (LLM) with 32k context window, known for its ability to understand and generate human-like text based on given instructions.

4.3 Model Alignment

4.3.1 Supervised Fine-Tuning (SFT)

Fine-tuning large language models (LLMs) such as the Mistral model with techniques like QLoRa (Quantized Low-Rank Adaptations) is a highly effective method for adapting pre-trained models to specific tasks while controlling the computational overhead typically associated with such large models. LLMs like Mistral are often massive, with billions of parameters. Fine-tuning all these parameters on specific tasks can be computationally expensive and memory-intensive. QLoRA, like standard LoRA, introduces a low-rank matrix to each of the transformer layers in the model (specifically to projection matrices such as Q, K, V in attention mechanisms). This matrix is much smaller than the original weights and focuses the learning on a subset of adaptable parameters, thus reducing the number of trainable parameters and saving computational resources. What makes QLoRA different from standard LoRA is that it employs quantization strategies (e.g., 4-bit quantization) to reduce the memory footprint further. This quantization not only helps in reducing the size of the model but also speeds up the computation, especially on compatible hardware that can efficiently handle low-precision arithmetic.

Furthermore, while pre-trained models are generalized across a wide range of topics and styles, QLoRA allows for task-specific adaptations without overwhelming the model's pre-learned knowledge. By adjusting only a small subset of parameters, QLoRA ensures that the model retains its broad capabilities while becoming more efficient at a specific task.

The fine-tuning process using QLoRA involves several steps, from initializing the low-rank matrices to training them along with the original model parameters. The algorithm below illustrates the pseudocode for using QLoRA for Supervised Fine-tuning (SFT).

Algorithm 1 Fine-tuning an LLM with QLoRA

```
Initialize model with pre-trained weights
Load training and validation datasets
Initialize QLoRA matrices R_Q, R_K, R_V with small random values
Set learning rates for QLoRA parameters higher than for pre-trained parameters
for each epoch do
    for each batch in training data do
        Update model using both pre-trained and QLoRA parameters
        Apply dropout to QLoRA matrices to prevent overfitting
        Calculate loss and backpropagate
    end for
    Evaluate model on validation dataset
    Adjust learning rates if necessary
end for
Save fine-tuned model
```

For simplicity, in this project, we opted to use most of the default hyperparameters used in the original QLoRA fine-tuning process.

4.3.2 Few-Shot Learning

Few-shot learning is a machine learning technique that is used in learning when a very limited amount of data is available. Few-shot learning aims to build models that can generalize well from only a few training examples.

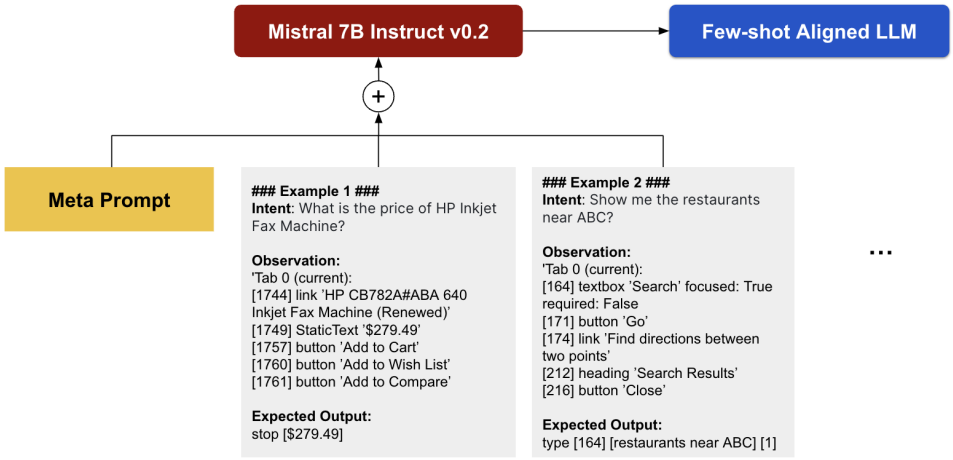

Figure 3: Few Shot- learning pipeline

In the visual workflow depicted in Figure 3, the process of few-shot learning is initiated with the construction of a meta prompt, serving as the blueprint for instructing the large language model. This guide comprises structured examples that illustrate the tasks the model is expected to perform. For instance, one example directs the model to discern the price of an HP Inkjet Fax Machine from a web page, while another instructs it to locate restaurants near a specific point of interest, "ABC". These examples not only outline the user's objectives but also detail the observable elements on the web page, providing a context for the model to understand and respond appropriately.

Upon absorbing the information from the meta prompt, the language model, "**Mistral 7B Instruct v0.2**", transitions into a few-shot learning phase. It tailors its capabilities to align with the task profiles presented in the examples. In essence, the model is fine-tuned in real-time to adapt to the nature of these web navigation tasks. It proceeds to generate outputs that align with the user's intent and the web page's details, encapsulating the task completion in the precise format showcased within the meta prompt examples. This represents the model's adaptability and its ability to grasp and perform web-based tasks with minimal prior examples, showcasing the efficiency and effectiveness of few-shot learning.

5 Evaluation Metrics

In this section, we will discuss the evaluation metrics that assess the performance of autonomous agents in complex, web-based tasks. These metrics are essential for measuring both the accuracy of the agent's actions and their effectiveness in achieving the desired outcomes within our simulated web environment.

Functional Correctness: Checks if the agent's actions lead to expected outcomes by comparing against pre-set expectations.

Task Success Rate: Measures the agent's ability to complete tasks from start to end, especially in tasks requiring multiple, sequential interactions.

Scoring Functions: Tailors the evaluation to the context of the task:

- **Exact Match:** Requires the agent's response to match a predefined answer precisely, used when exactness is critical.
- **Must Include:** Offers flexibility, ensuring the agent's response, while possibly varying in format, contains all necessary information.
- **Fuzzy Match:** Allows for variations in phrasing, assessing if the agent's response is semantically equivalent to what's expected.

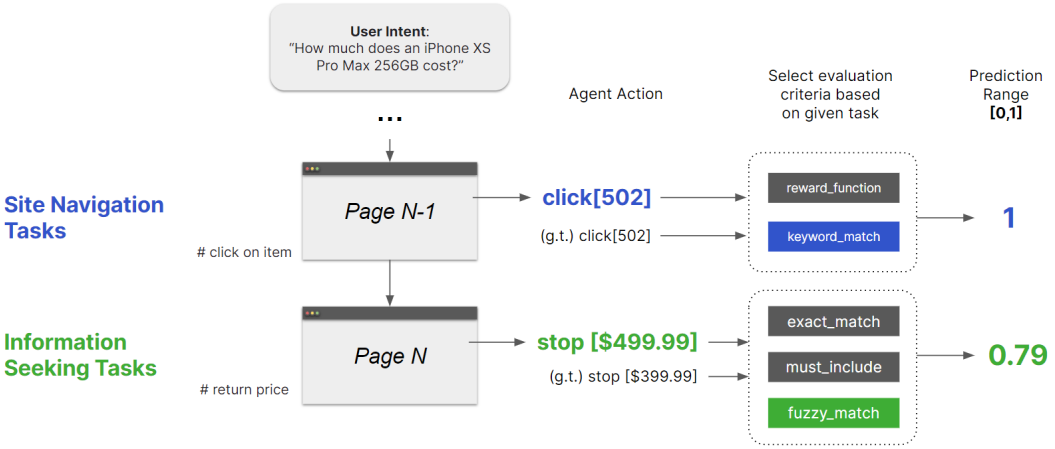

Figure 4: Illustrative example of Evaluation Criteria

For example, as shown in Figure 4 an agent tasked with finding an iPhone's price might navigate correctly but report an incorrect price. If the expected price is $399.99 and the agent reports $499.99, a Fuzzy Match score of 0.79 is assigned, indicating the agent was close but not accurate. These metrics work in unison, offering a comprehensive evaluation of the agent's effectiveness in web-based tasks, from following precise instructions to understanding and conveying the right semantic content.

6 Experiments & Results

	Pass	Fail	Err. Instances	SR*
Mixtral-8x7B-Instruct-v0.1	11	61	740	15.3%
Meta-Llama-3-70B-Instruct	3	19	790	13.6%
Meta-Llama-3-8B-Instruct	1	42	769	2.38%
GPT4**	-	-	0	14.4%
Human**	-	-	0	78.2%

Table 2: LLM Inference Results across different Vanilla Models
* Varying number of successful instances per model adds bias to the success rate (SR) metric
** Results pulled directly from the WebArena paper (arXiv:2307.13854) for reference

6.1 Running the Webarena

Test a variety of vanilla LLMs, and evaluate their capacity in performing web browser tasks from a given User intent and Observation space data. Where we can see in the results in table 1, we can see two noteworthy points.

	Zero-shot	3-shot learning	Ground Truth
Experiment 1	The user's objective is to 'Look for an available full-time job in USA finance, save all the accountant jobs'. Since the user's last action was 'unknown', the first action should be to navigate to the job search page. "go_forward"	CLICK [1064]	CLICK [209]
Experiment 2	"hover [id of the price element] type [tab] press [Enter]"	CLICK [13]	CLICK [5054]

Table 3: Few-shot Learning vs Unaligned Model Sample Outputs

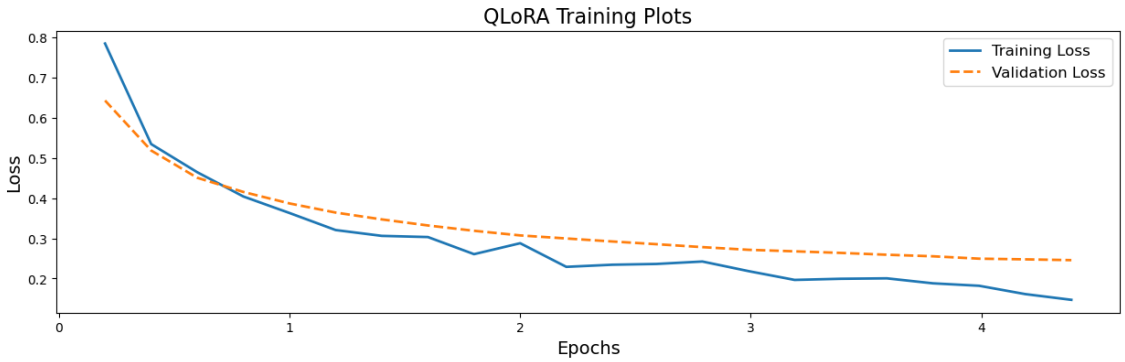

Figure 5: QLoRA Training plots

First of all we can see that many of the tests fail. This illustrates a key problem with The Webarena, where the majority of the trials fail, not due to the model, but due to the websites which The Webarena fails to host.

Second to this point is that these vanilla models do not perform very well. Reaching only 15% in the 'Mixtral-7x7B-Intstruct-v0.1'. It is important to realise that the best score on the complete, which means all instances are non err. , is only 14.4% by GPT4. Which implies that this task, which is end to end, is very complex and requires a high intelligence. To further illustrate this point we can look at the SR from the human test subjects, that can on average only end to end complete 78.2% of the tasks. [7].

6.2 Model Alignment

Test alignment strategies to align an LLM on web browser tasks on the specified environment and compare its performance against vanilla LLMs. Where we have tried to align 'Mistral-7B-Instruct-v0.2' in two different manners. Both through QLoRA and FewShot. The results can be seen in table 3. It is worth noting that the Zero-shot did not manage to change the output to close to the predetermined format. However, as can be seen in the same table, the 3-shot learning did manage to predict the correct action type consistently. However, the value of these actions is not consistent.

Regarding the QLoRA, the training did manage to converge, as we can see in figure 5. Both the evaluation and training loss converge at a healthy rate; however, due to a limit in computing power, we could not go through the entire five epochs and had to cap the training prematurely. Furthermore, due to the difficulties when working with The Webarena, no locally hosted model could be used, and with this, the aligned model could not be compared.

7 Conclusion

To conclude this report, we would like to point out three key takeaways from this topic. First of all, we would like to point out the difficulty in obtaining valuable training data within this sphere. Where the training data needs to contain three distinct features: the observation(s), the action(s), and the intent. Humans perform all. With this, we would like to call for a simple browser extension that mines this data from users.

Secondly, from the data mentioned previously, even the most powerful model cannot actually reliably operate the web. GPT4, to date, is the most capable model, but it only scored a small percentage of the score comparable to humans. With many LLMs outperforming human experts within their own field in many different problem-solving tasks, it begs the question of why the operation of the web is so tremendously difficult.

The final thing we would like to point out is regarding The Webarena, which tries to mimic the "wild web" by providing live and hosted mirrors of different websites. note that data is hard to come by note that LLMs are clearly incapable enough to parse the website might not actually be that good of a proxy of the web. Many of the websites are much cleaner than those you might find in the web. Although Webarena is the best dynamic environment to date, there are many improvements still to be made to further evaluate agents' web-browsing capabilities.

8 Contributions

Our team worked coherently. Everyone took part in the discussions actively and all of us showed great impact in the project to make it successul.

- Aneesh: First to introduce the idea; Run Llama on WebArena; Finetuning Mistral 7B; ataset;
- Rein: Fewshot Learning Experiments, Powerpoin and Report
- Riemer: Prepared Mind2Web dataset, Recorded pre entation video,
- Asswin: Report & lignment Techniques;
- Adnan: Initial Webarena code setup; Running vanilla LLMs on WebArena usig different va

Our code can be found on GitHub in the following url: `https://github.com/adnan-azmat/WebArenaNLP`

9 References

References

[1] S. Branavan, H. Chen, L. Zettlemoyer, and R. Barzilay, "Reinforcement learning for mapping instructions to actions," *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 82–90, 2009. [Online]. Available: https://aclanthology.org/P09-1010

[2] E. Z. Liu, K. Guu, P. Pasupat, T. Shi, and P. Liang, "Reinforcement learning on web interfaces using workflow-guided exploration," *6th International Conference on Learning Representations, ICLR 2018*, 2018. [Online]. Available: https://openreview.net/forum?id=ryTp3f-0-

[3] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders *et al.*, "Webgpt: Browser-assisted question-answering with human feedback," *arXiv preprint arXiv:2112.09332*, 2021.

[4] I. Gur, H. Furuta, A. Huang, M. Safdari, Y. Matsuo, D. Eck, and A. Faust, "A real-world webagent with planning, long context understanding, and program synthesis," *arXiv preprint arXiv:2307.12856*, 2023.

[5] K. Lee, M. Joshi, I. R. Turc, H. Hu, F. Liu, J. M. Eisenschlos, U. Khandelwal, P. Shaw, M.-W. Chang, and K. Toutanova, "Pix2struct: Screenshot parsing as pretraining for visual language understanding," *International Conference on Machine Learning*, pp. 18 893–18 912, 2023.

[6] P. Shaw, M. Joshi, J. Cohan, J. Berant, P. Pasupat, H. Hu, U. Khandelwal, K. Lee, and K. Toutanova, "From pixels to ui actions: Learning to follow instructions via graphical user interfaces," *arXiv preprint arXiv:2306.00245*, 2023.

[7] S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, T. Ou, Y. Bisk, D. Fried, U. Alon, and G. Neubig, "WebArena: A Realistic Web Environment for Building Autonomous Agents." [Online]. Available: http://arxiv.org/abs/2307.13854

[8] X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, and Y. Su, "Mind2Web: Towards a Generalist Agent for the Web."