LegalWebAgent: Empowering Access to Justice via LLM-Based Web Agents

Jinzhe Tan[1,*], Karim Benyekhlef[1]

[1]Cyberjustice Laboratory, Faculty of Law, University of Montreal, Canada

Abstract

Access to justice remains a global challenge, with many citizens still finding it difficult to seek help from the justice system when facing legal issues. Although the internet provides abundant legal information and services, navigating complex websites, understanding legal terminology, and filling out procedural forms continue to pose barriers to accessing justice. This paper introduces the `LegalWebAgent` framework that employs a web agent powered by multimodal large language models to bridge the gap in access to justice for ordinary citizens. The framework combines the natural language understanding capabilities of large language models with multimodal perception, enabling a complete process from user query to concrete action. It operates in three stages: the Ask Module understands user needs through natural language processing; the Browse Module autonomously navigates webpages, interacts with page elements (including forms and calendars), and extracts information from HTML structures and webpage screenshots; the Act Module synthesizes information for users or performs direct actions like form completion and schedule booking. To evaluate its effectiveness, we designed a benchmark test covering 15 real-world tasks, simulating typical legal service processes relevant to Québec civil law users, from problem identification to procedural operations. Evaluation results show `LegalWebAgent` achieved a peak success rate of 86.7%, with an average of 84.4% across all tested models, demonstrating high autonomy in complex real-world scenarios.

Keywords

Large Language Models, Multimodal LLMs, Access to Justice, Agentic AI, Web Agent, Legal Agent

1. Introduction

The significant gap between the public's legal needs and their ability to find legal information and solutions has created a "justice gap" in countries worldwide [1]. For ordinary citizens, the cost of hiring a lawyer is prohibitively expensive [2], forcing them to spend a great deal of time navigating a maze of government websites, legal statutes, and procedural forms on their own. Even so, they often find it difficult to search using correct legal terminology, locate relevant information on cluttered websites, or make critical errors when filling out online forms.

Existing legal tech tools, such as static frequently asked questions (FAQ) portals and simple rule-driven chatbots, seek to address this issue. They typically provide information in plain language[1], or offer simplified legal pathways and relevant cases [3, 4] to help reduce the user's cognitive load. However, as the legal domains covered by these websites and the volume of information they contain increase, the cognitive burden on users increases correspondingly. Furthermore, these tools still lack the ability to interact with the broader web ecosystem or perform actions on behalf of the user. This means that the most difficult and error-prone practical "operational" steps, such as submitting forms or scheduling appointments, are still left to the user.

The recent development in the field of multimodal large language models (MLLMs) has advanced the role of AI in building a general-purpose tool to enhance access to justice. The improved reasoning capabilities of LLMs have made it possible to build autonomous agents [5, 6]. `GPT-4o-vision` and more advanced LLMs demonstrate strong capabilities in natural language understanding and generation. Moreover, their multimodal understanding abilities allow for the simultaneous processing of inputs

To appear in AI for Access to Justice, Dispute Resolution, and Data Access Workshop(AI4A2J), 2025

*Corresponding author.

jinzhe.tan@umontreal.ca (J. Tan)

[1]For example, Québec's Éducaloi (https://educaloi.qc.ca/) provides legal information to the public in plain language.

other than text, such as images [7], making it feasible to build web agents that can comprehend both the text and visual elements of webpages [8]. In this study, we introduce the `LegalWebAgent` (see Figure 1) framework, a multimodal web agent designed to autonomously create plans based on a user's query to perform tasks such as web browsing, information gathering, and web interaction.

In the following sections, we review related work (Section 2), introduce the `LegalWebAgent` framework (Section 3), outline our experiment design (Section 4), and present results (Section 5). We then discuss key insights and current limitations (Section 6), and conclude.

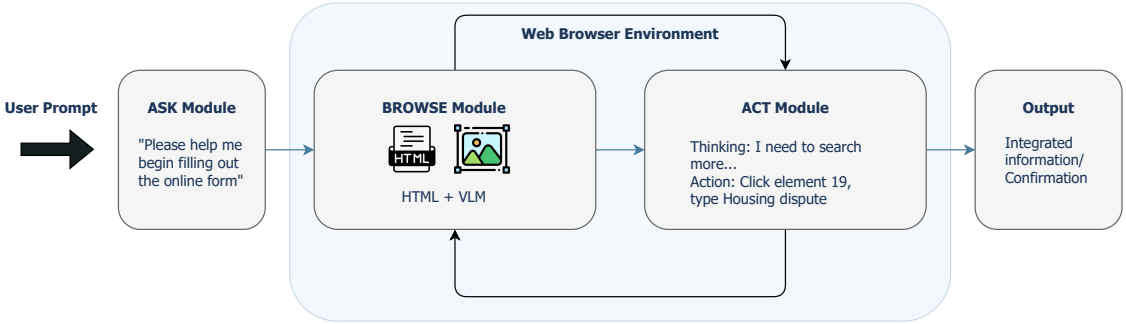


Figure 1: The overall workflow of `LegalWebAgent`. Given a user's query, `LegalWebAgent` formulates a plan, analyzes the webpage's HTML elements and screenshots, and determines the appropriate actions (such as clicks, scrolls, or inputs). After gathering the necessary information or completing the requested task, it generates a concise summary of the process and presents the results to the user.

2. Related Work

2.1. AI for Access to Justice

The access to justice gap is a global issue that not only incurs monetary, temporal, and psychological costs [9, 10, 11, 12, 13] but also erodes public confidence in the justice system. AI technology has been widely applied to bridge this gap [14], with efforts ranging from providing legal information [15, 16] and assisting with form completion [17, 18] to online dispute resolution [19, 20, 21]. In addition to these primarily text-focused efforts, Vision-Large Language Models (VLLMs) have also undergone preliminary exploration in the legal domain in recent years [22]. Researchers have also explored integrating AI with information portals to help users map natural language descriptions to relevant legal issues [16, 23]. However, these systems are often built for specific contexts and thus lack generalizability. Furthermore, they are largely passive, requiring users to independently read, comprehend, and act upon the information provided.

2.2. Web Agents

The internet has long been one of the primary channels for users to obtain information. It can be viewed as a continuously updated real-time database. In the legal field, modifications to legal information, promulgation of new laws, and introduction of new cases are all published on the internet in a very timely manner. When needed, people can readily find the information they require online. Furthermore, tasks such as booking meetings, filling out online forms, and completing online questionnaires offer users the possibility of handling tasks remotely, saving them valuable time.

In practice, however, this process is far from straightforward: (1) users may encounter websites containing outdated or misleading information; (2) retrieving legal information often requires extensive cross-verification and synthesizing information from numerous web pages; (3) users may be unaware of the existence of certain authoritative resources; and (4) human cognitive limitations, along with states such as distraction and fatigue, can significantly hinder users' ability to navigate the web effectively [24].

The internet is designed for humans, enabling them to interact with the digital world through operations such as clicking, scrolling, and typing [25]. Creating web agents capable of simulating these operations is expected to significantly help reduce the sense of disorientation and inefficiency experienced during the aforementioned web browsing process.

Such web agents or web automation have been extensively explored for decades. Early research typically relied on structured data sources or site-specific wrappers to perform tasks such as flight booking or information scraping [26, 27, 28]. These methods required significant manual configuration for each website and were highly brittle, often breaking when sites changed [29, 30].

With the development of deep learning and reinforcement learning, more *generalized* web agents have become possible. Depending on the architecture, they range from agents trained purely on HTML files [31, 32, 33], to models that combine HTML and visual screenshots [34, 35], and even to agents that rely solely on visual input from screenshots [36]. These modern web agents are capable of helping users interact with real-world websites and perform everyday tasks [37].

3. `LegalWebAgent` Framework

`LegalWebAgent` is composed of three modules (Ask Module / Browse Module / Act Module) and works in concert with a web browsing environment. `LegalWebAgent` builds on top of the open-source `Browse-Use` framework [38], the core of which is `Playwright` [39], which enables reliable and fast web automation by providing a unified API for Chromium, Firefox, and WebKit. When a user command is received, `LegalWebAgent` launches a Chromium browser and then provides context to the LLM by collecting the webpage's HTML elements and a screenshot (see Figure 2). Based on the current state, `LegalWebAgent` analyzes the completion status of the previous objective, updates its memory, and formulates the next objective. According to the objective, `LegalWebAgent` proposes an action to be executed and performs the operation within the browser environment.

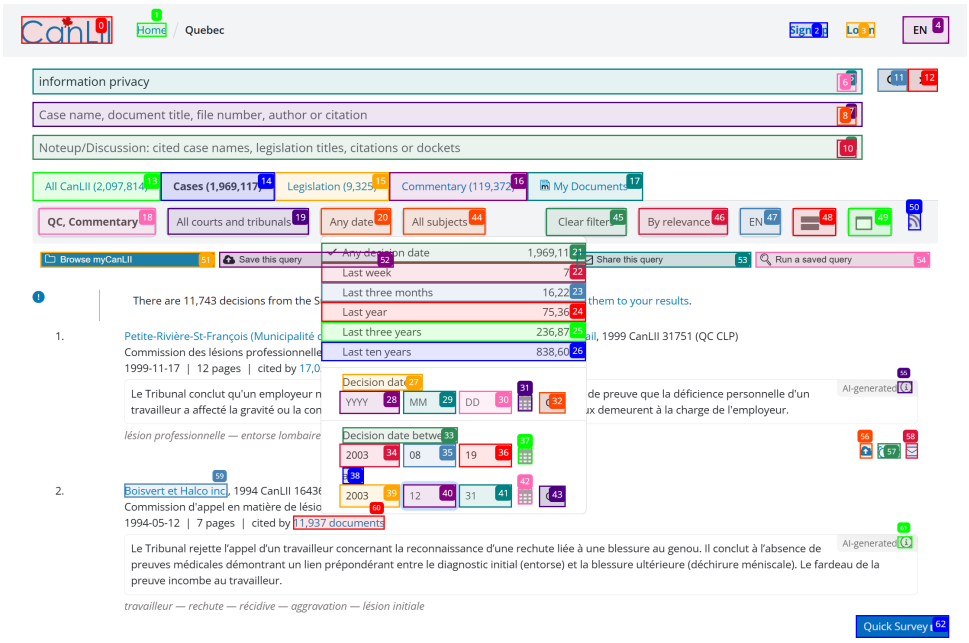


Figure 2: Example of a webpage screenshot provided to the web agent. By reading the HTML file, the application automatically adds borders to interactive elements on the webpage and labels them with numbered tags in the upper-right corner.

3.1. Ask Module

When the user provides a prompt (e.g., "Find the rental dispute department closest to H3A0G4."), the Ask module performs the following steps: (1) It uses an LLM to parse the user's intent. In this example, the module may recognize that the user needs legal information on landlord-tenant disputes in Québec, as well as the address and phone number of the relevant tribunal. (2) Based on the parsed intent, the Ask module generates a web navigation plan. For instance, the plan might be:

1. Search for rental dispute resolution services;
2. The user's postal code is H3A0G4, focus the search on Montreal.

3.2. Browse Module

The Browse module receives the current state of the webpage at each execution step and decides the next action to take. Because modern webpages often contain a large number of elements and complex layouts, we adopt a multimodal perception mechanism to improve the robustness of the Browse module.

HTML Analysis. The module parses the HTML Document Object Model (DOM) of the page, identifies candidate interactive elements (e.g., `<a>` links, `<button>` elements, form `<input>` fields), and extracts textual content. Using the method introduced in [40], all candidate elements are automatically highlighted with bounding boxes, and a numbered tag is added in the upper-right corner of each box (see Figure 2).

Visual Analysis. At the same time, the tagged page screenshot is passed to a VLLM. Since relying solely on DOM information can be misleading or incomplete (for instance, menus that appear only on hover), the visual input helps the LLM better handle different types of web pages.

After extracting webpage information, the Browse module must choose an action from a predefined action library. We define a set of primitive actions inspired by human browser interactions: `click(element)`, `input(text, field)`, `scroll(direction)`, `wait(seconds)`, etc.

3.3. Act Module

The Act module executes the actions provided by the Browse module in a real browser environment. If the action involves a click or text input, it performs the corresponding operation on the page and then returns control, along with the updated page state, back to the Browse module. This loop continues until either the predefined goal is achieved or the maximum number of execution steps is reached.

If the user requests an explanation or information, once the browsing stage has collected the relevant text, the Act module calls the LLM to generate a concise and user-friendly answer.

If the task objective involves performing an operation, the Act module verifies the completion of each sub-operation (e.g., filling in forms, uploading files, booking an appointment) and proceeds until the final submission. Upon completion, it outputs a confirmation message to the user (see Figure 3), such as:

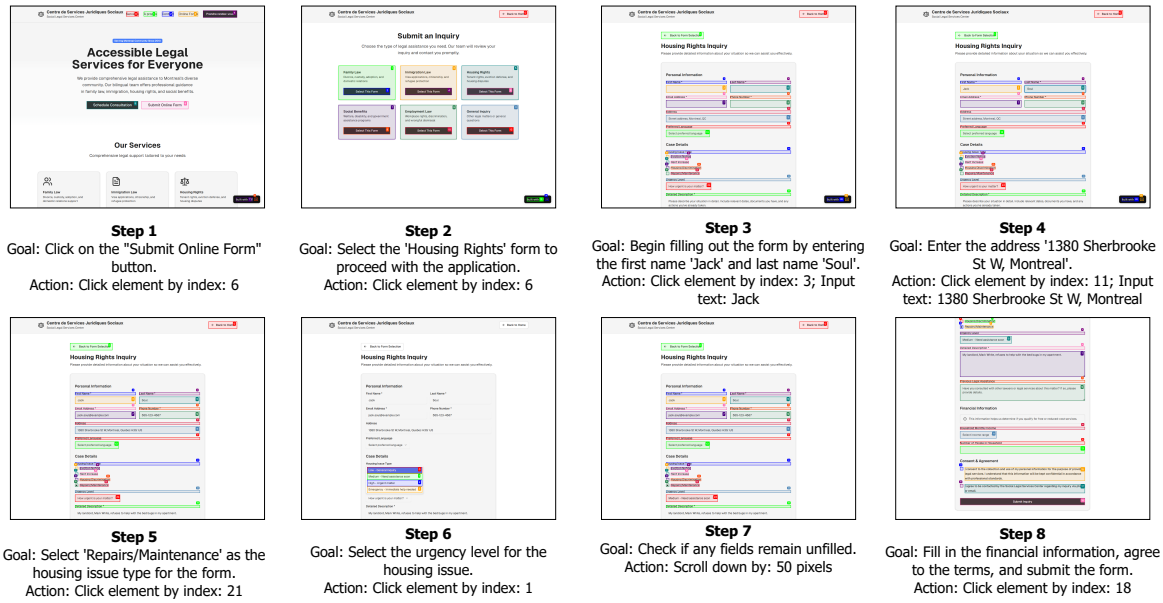
> Form submitted successfully. Your confirmation number is 123-456.

4. Experiment Design

4.1. Data Collection

To evaluate the performance of `LegalWebAgent` on legal web tasks, we designed a preliminary benchmark suite comprising 15 tasks (described in Table 1). These tasks were selected from the domain of Québec Civil Law and are designed to simulate real user queries or goals. The tasks are structured around three key stages of an ordinary citizen's legal journey:

Information Gathering. In this stage, the user is just starting to understand their legal problem. The tasks involve obtaining basic explanations of legal rights or concepts. For example, the user inquires



Step 1
Goal: Click on the "Submit Online Form" button.
Action: Click element by index: 6

Step 2
Goal: Select the 'Housing Rights' form to proceed with the application.
Action: Click element by index: 6

Step 3
Goal: Begin filling out the form by entering the first name 'Jack' and last name 'Soul'.
Action: Click element by index: 3; Input text: Jack

Step 4
Goal: Enter the address '1380 Sherbrooke St W, Montreal'.
Action: Click element by index: 11; Input text: 1380 Sherbrooke St W, Montreal

Step 5
Goal: Select 'Repairs/Maintenance' as the housing issue type for the form.
Action: Click element by index: 21

Step 6
Goal: Select the urgency level for the housing issue.
Action: Click element by index: 1

Step 7
Goal: Check if any fields remain unfilled.
Action: Scroll down by: 50 pixels

Step 8
Goal: Fill in the financial information, agree to the terms, and submit the form.
Action: Click element by index: 18

Figure 3: A demonstration of how `LegalWebAgent` navigates the website and completes an online form task through a series of actions on the `legal-agent-sandbox` we created.

about "my landlord raised the rent without prior notice, saying I have to move out next month if I don't pay the new rent. Can he do that?" This type of query primarily involves information retrieval and comprehension. `LegalWebAgent` must locate authoritative sources (such as government websites or legal information platforms) to find relevant resources and explain them to users in plain English.

Resource Finding. In this stage, the user is already aware of their general problem and is seeking specific resources or contact information. Task S2-LSA-01 asks, "Where is the nearest rental dispute office to my postal code H3A0G4?" This requires not only finding information about rental dispute offices but also locating the nearest one based on the user's location in Québec. Task S2-CS-01 is, "Search CanLii for a judgment date between August 19, 2023, and August 31, 2023, and a judgment length of exactly 23 pages." This task tests the agent's database retrieval capabilities. These tasks involve complex, multi-page information retrieval but do not include form submission.

Action Taking. This stage contains the most complex, multi-step tasks, requiring the agent to perform concrete actions on web pages. Task S3-OFC-01 is, "Fill out the online form on `legal-agent-sandbox`." This simulates the scenario of a user submitting a claim or application through an official court web form. The agent must navigate a multi-page form interface, input predefined user information, interact with complex components (such as a calendar), and finally submit the application. These tasks significantly challenge the agent's autonomous navigation and form-handling capabilities, especially when dealing with dynamic elements such as date pickers or multi-step processes.

All tasks were executed on live or staged websites. For the first two stages, we require the agent to interact with actual public websites. For the third stage, which involves web form filling and online appointment scheduling, we built a `legal-agent-sandbox`[2] to avoid wasting public resources. This sandbox is deployed on a public domain and mimics the design of real websites, allowing us to test the agent's action-taking capabilities in a controlled environment.

4.2. Models

Given the high intelligence requirements of the web agent applications, we selected and evaluated three leading large language models: two vision-language models and one text-only model (to assess performance in an HTML-based plain-text context). We include one text-only model to measure how

[2] https://legal-agent-sandbox.vercel.app/

much performance depends on visual perception. Specifically, we tested OpenAI's `GPT-4o`, Anthropic's `Claude-Sonnet-4-20250514`, and DeepSeek's `DeepSeek-v3.1`.

4.3. Experiments

In the experiment, each model was given the same prompt and run at a temperature of 0.6. Once the prompt was sent to the model, no further intervention was made. After task completion, we scored whether the task was successful. A task was marked as successful if the model correctly fulfilled the request (e.g., providing accurate legal information, finding the correct institutions and locations, or submitting a form as instructed). It was marked as a failure if the model provided incorrect information, failed to retrieve useful information, or was unable to complete the required web interactions. We recorded the success/failure status, the number of reasoning/interaction steps, the time to completion, and the tokens consumed for each model on every task.

Table 1
`LegalWebAgent` benchmark overview.

Stage	Category	Count	Query Description Example
Information Gathering	Vague Inquiry	2	User inquires about the legality of a landlord raising rent without prior notice and threatening eviction for non-payment of the new amount.
	Consumer Dispute	2	User asks for recourse options after an expensive online purchase broke after one week and the seller refused to provide a refund.
Resource Finding	Complex Search	2	User requests a search for a legal case on CanLii using highly specific criteria, including jurisdiction (Québec), legal field, a narrow judgment date range, and exact document length.
	Locating Authority	3	User needs to identify the correct government department for a rental dispute and find the location and phone number of the nearest office based on their postal code.
	Legal Aid	3	User is looking for free or low-cost community legal services specializing in family law near downtown Montreal.
Action Taking	Form Completion	1	User requests assistance in filling out an online legal application form with their personal details (name, address) and case description (landlord-tenant issue).
	Appointment Booking	2	User instructs the system to schedule a legal consultation for them on a specific website for a given date.

5. Results

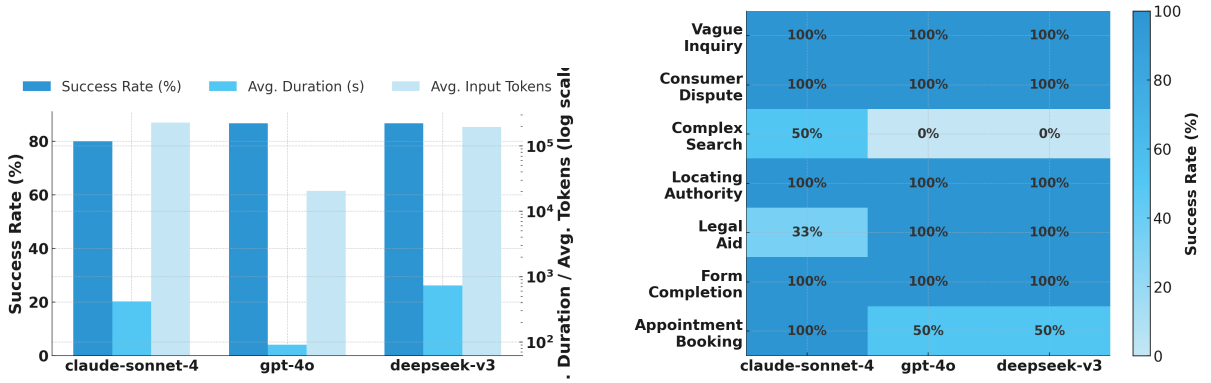
Table 2 and Figure 4 (a) show the models' overall task success and efficiency. OpenAI `GPT-4o` and `DeepSeek-v3` achieved the highest success rate at 86.7% (13 out of 15 tasks each), while `Claude-Sonnet-4`, known for higher reasoning ability, completed 80.0% (12/15 tasks). Notably, `GPT-4o` was by far the most efficient, with an average run time of only about 90 seconds per task and approximately 20k tokens consumed, compared to Claude's 416 seconds and 227k tokens. `DeepSeek-v3`'s runtime and token consumption are also an order of magnitude higher than `GPT-4o`'s (averaging 730 seconds and consuming 195k tokens). Claude-Sonnet-4 tends to generate extremely detailed and lengthy outputs while exploring more pages (resulting in high token consumption), yet this does not translate into higher success rates. In fact, Claude's comprehensive approach sometimes leads to diminishing returns, as it fails on tasks other models accomplish with fewer steps.

Figure 4 (b) provides a heatmap of success rates across the seven task categories (see Table 1 for task breakdown). We observe 100% success across all models for most information-seeking tasks, including vague rights inquiries (S1-VRI) and consumer disputes (S1-CDD). All models also succeeded in locating authorities (S2-LSA) and completing the complex online form (S3-OFC) on our sandbox site.

This suggests that current LLM agents are highly capable in straightforward information retrieval and form-filling scenarios in the legal domain.

The Complex Search (S2-CS) tasks had the lowest success rates: one CanLII case query (S2-CS-01), inspired by [24], that is very difficult to search for but very easy to verify, was not solved by any model (0% success). The other, a statistical query (S2-CS-02), was solved only by `Claude` (50% success vs. 0% for `GPT-4o` and `DeepSeek-v3` in this category). These complex searches require combining multiple filters or conditions on the website and involve extensive browsing to find the correct answer. Most failures were due to the models failing to correctly apply multiple filters simultaneously or abandoning the search prematurely.

Another challenging area was the online appointment booking system. In this task category, Claude-Sonnet-4 successfully completed both scheduling tasks (100%), whereas `GPT-4o` and `DeepSeek-v3` each completed only one. Specifically, in task S3-OAB-02, which involves booking via an external Microsoft Forms page, both `GPT-4o` and `DeepSeek-v3` struggled. `DeepSeek-v3` failed after 50 repetitive steps on the booking interface, while `Claude` completed the form and secured the appointment in just 15 steps. This discrepancy suggests that despite its slower execution and higher cost, `Claude`'s more comprehensive planning strategy yields superior performance in handling highly complex, multi-step interactions.



(a) Overall performance comparison across models. The left y-axis shows the success rate (%) for each model, while the right y-axis (log-scale) compares average duration (s) and average input tokens. `GPT-4o` achieves the highest efficiency (shortest duration, lowest token use), while `Claude-Sonnet-4` produces the most thorough outputs but at a higher computational cost.

(b) Heatmap of model success rates by task category. Darker cells indicate higher success rates. Most categories achieve 100% success across all models except Complex Search (low performance across the board) and Appointment Booking, where `GPT-4o` and `DeepSeek-v3` underperform relative to `Claude-Sonnet-4`.

Figure 4: Overall performance comparison and task-wise success rates of the evaluated models.

Table 2
Overall model comparison (success, efficiency, and input cost).

Model	Success Rate	Successful Tasks (n)	Avg. Duration (s)	Avg. Tokens
`Claude-Sonnet-4`	80.0%	12	416.32	227,594
`GPT-4o`	86.7%	13	90.9	20,514
`DeepSeek-v3`	86.7%	13	730.0	195,519

6. Discussion

Our initial evaluation suggests that web agents powered by large language models can autonomously perform information gathering, resource discovery, and take action to promote access to justice.

`LegalWebAgent` demonstrated strong performance on our tasks, achieving high success rates in answering legal questions, locating government agencies, and even completing online form submissions. However, we also observed challenges during experimentation that would arise when deploying such agents in real-world scenarios.

6.1. Deep or Success?

We observed that tasks requiring deep website navigation (many clicks or layers of pages) had a higher failure probability. Advanced models like `Claude-Sonnet-4` tended to "dig deeper" into sites, which sometimes helped but often led to getting sidetracked or timing out. Excessive exploration can cause the agent to lose context or encounter dead ends (e.g., endlessly following related links). An effective agent must balance thoroughness with focus. Future agents might benefit from adaptive search depth control, stopping when sufficient information is found rather than exhaustively clicking every link.

6.2. Web Design for AI Agents

Our findings suggest that today's websites, designed for human users, can inadvertently confuse or hinder AI agents. For example, pop-ups like cookie consent or privacy policy notices are easy for humans to handle but can disrupt an agent's understanding. Similarly, information hidden within complex menus or behind interactive components is difficult for web agents to extract. To improve agent performance (and perhaps human usability too), web developers could consider offering "AI-friendly" design, such as simplified HTML views, Markdown or JSON endpoints, or well-labeled content, tailored for autonomous agents.

6.3. Token Efficiency

The difference in token consumption across different models for the same task highlights the importance of optimizing how agents process web content. `Claude-Sonnet-4` tends to load entire pages or even multiple pages in detail, leading to an exceptionally large context window. This not only impacts agent execution speed but may also trigger LLM context-length limitations. We can reduce token consumption by optimizing the content extracted by agents, which is important for cost reduction and efficiency improvement in web agents as well as their real-world deployment.

6.4. Limitations

This research is a preliminary exploration of web agents for promoting access to justice, tested on a limited dataset (15 tasks) focused on Québec's legal context. While our dataset was designed to reflect genuine user needs, broader evaluation is required for more accurate results. Future work should expand the testing scope to include more diverse legal scenarios, such as different jurisdictions, various legal websites, and a wider array of legal problem types.

A further limitation is the framework's dependence on large proprietary language models, which may incur excessive operational costs. Investigating smaller or open-source models and enhancing agent efficiency are therefore worthwhile research avenues. Agent safety is another important concern. For instance, [41] suggests web agents might be more vulnerable to jailbreaking, and [42] discusses how developers could add a "toxic" page, visible only to agents, to manipulate their behavior. This indicates that the present study is experimental in nature and requires critical robustness testing and security reviews before practical deployment.

7. Conclusion

The `LegalWebAgent` framework showcases the potential of multimodal LLM-based agents in bridging the "justice gap." By combining MLLMs with web navigation and action execution, such agents can empower individuals to find legal information and complete online procedures with minimal manual

effort. Our model comparison results indicate that while current state-of-the-art LLMs can reliably handle most tasks, they still require fine-tuning to effectively manage complex searches and intricate web forms. There remains significant room for improvement in agent response speed, reliability, and web content parsing. We believe that by automating web browsing processes, web agents based on MLLMs have strong potential to enhance access to justice.

Declaration on Generative AI

During the preparation of this work, the authors used large language model tools (such as GPT-style assistants) for grammar and spelling checking, wording suggestions, and minor text refinement. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] R. Child, The global justice gap, Critical Review of International Social and Political Philosophy 19 (2016) 574–590.

[2] G. K. Hadfield, The cost of law: Promoting access to justice through the (un) corporate practice of law, International Review of Law and Economics 38 (2014) 43–63.

[3] H. Westermann, K. Benyekhlef, Justicebot: A methodology for building augmented intelligence tools for laypeople to increase access to justice, in: Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law, 2023, pp. 351–360.

[4] S. Janatian, H. Westermann, J. Tan, J. Savelka, K. Benyekhlef, From text to structure: Using large language models to support the development of legal expert systems, in: Legal Knowledge and Information Systems, IOS Press, 2023, pp. 167–176.

[5] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, et al., A survey on large language model based autonomous agents, Frontiers of Computer Science 18 (2024) 186345.

[6] L. Ning, Z. Liang, Z. Jiang, H. Qu, Y. Ding, W. Fan, X.-y. Wei, S. Lin, H. Liu, P. S. Yu, et al., A survey of webagents: Towards next-generation ai agents for web automation with large foundation models, in: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2, 2025, pp. 6140–6150.

[7] J. Wu, W. Gan, Z. Chen, S. Wan, P. S. Yu, Multimodal large language models: A survey, in: 2023 IEEE International Conference on Big Data (BigData), IEEE, 2023, pp. 2247–2256.

[8] B. Zheng, B. Gou, J. Kil, H. Sun, Y. Su, Gpt-4v (ision) is a generalist web agent, if grounded, arXiv preprint arXiv:2401.01614 (2024).

[9] T. C. Farrow, What is access to justice, Osgoode Hall LJ 51 (2013) 957.

[10] A. Currie, The legal problems of everyday life, in: Access to justice, volume 12, Emerald Group Publishing Limited, 2009, pp. 1–41.

[11] N. Semple, The cost of seeking civil justice in canada, Can. B. Rev. 93 (2015) 639.

[12] T. C. Farrow, A. Currie, N. Aylwin, L. Jacobs, D. Northrup, L. Moore, Everyday legal problems and the cost of justice in Canada: Overview report, Technical Report, York University, 2016.

[13] Y. Cannon, Unmet legal needs as health injustice, U. Rich. L. Rev. 56 (2021) 801.

[14] H. Westermann, Using artificial intelligence to increase access to justice, 2023. PhD thesis, University of Montreal.

[15] J. Tan, H. Westermann, K. Benyekhlef, Chatgpt as an artificial lawyer?, in: AI4AJ@ ICAIL, 2023, pp. 1–8.

[16] H. Westermann, S. Meeùs, M. Godet, A. C. Troussel, J. Tan, J. Savelka, K. Benyekhlef, Bridging the gap: Mapping layperson narratives to legal issues with language models., in: ASAIL@ ICAIL, 2023, pp. 37–48.

[17] Q. Steenhuis, B. Willey, D. Colarusso, Beyond readability with ratemypdf: A combined rule-based and machine learning approach to improving court forms, in: Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law, 2023, pp. 287–296.
[18] H. Westermann, Dallma: Semi-structured legal reasoning and drafting with large language models, in: 2nd Workshop on Generative AI and Law, 2024.
[19] K. Branting, S. McLeod, S. Howell, B. Weiss, B. Profitt, J. Tanner, I. Gross, D. Shin, A computational model of facilitation in online dispute resolution, Artificial Intelligence and Law 31 (2023) 465–490.
[20] H. Westermann, J. Savelka, K. Benyekhlef, Llmediator: Gpt-4 assisted online dispute resolution, arXiv preprint arXiv:2307.16732 (2023).
[21] J. Tan, H. Westermann, N. R. Pottanigari, J. Šavelka, S. Meeùs, M. Godet, K. Benyekhlef, Robots in the middle: Evaluating llms in dispute resolution, arXiv preprint arXiv:2410.07053 (2024).
[22] H. Westermann, J. Savelka, Analyzing images of legal documents: Toward multi-modal llms for access to justice, arXiv preprint arXiv:2412.15260 (2024).
[23] Q. Steenhuis, H. Westermann, Getting in the door: Streamlining intake in civil legal services with large language models, arXiv preprint arXiv:2410.03762 (2024).
[24] J. Wei, Z. Sun, S. Papay, S. McKinney, J. Han, I. Fulford, H. W. Chung, A. T. Passos, W. Fedus, A. Glaese, Browsecomp: A simple yet challenging benchmark for browsing agents, arXiv preprint arXiv:2504.12516 (2025).
[25] B. Gou, R. Wang, B. Zheng, Y. Xie, C. Chang, Y. Shu, H. Sun, Y. Su, Navigating the digital world as humans do: Universal visual grounding for gui agents, arXiv preprint arXiv:2410.05243 (2024).
[26] J. Hammer, H. Garcia-Molina, J. Cho, R. Aranha, A. Crespo, Extracting Semistructured Information from the Web., Technical Report, Stanford InfoLab, 1997.
[27] R. B. Doorenbos, O. Etzioni, D. S. Weld, A scalable comparison-shopping agent for the world-wide web, in: Proceedings of the first international conference on Autonomous agents, 1997, pp. 39–48.
[28] N. Kushmerick, Wrapper induction for information extraction, University of Washington, 1997.
[29] K. Lerman, S. N. Minton, C. A. Knoblock, Wrapper maintenance: A machine learning approach, Journal of Artificial Intelligence Research 18 (2003) 149–181.
[30] I. Muslea, S. Minton, C. Knoblock, A hierarchical approach to wrapper induction, in: Proceedings of the third annual conference on Autonomous Agents, 1999, pp. 190–197.
[31] T. Shi, A. Karpathy, L. Fan, J. Hernandez, P. Liang, World of bits: An open-domain platform for web-based agents, in: International Conference on Machine Learning, PMLR, 2017, pp. 3135–3144.
[32] E. Z. Liu, K. Guu, P. Pasupat, T. Shi, P. Liang, Reinforcement learning on web interfaces using workflow-guided exploration, arXiv preprint arXiv:1802.08802 (2018).
[33] N. Xu, S. Masling, M. Du, G. Campagna, L. Heck, J. Landay, M. S. Lam, Grounding open-domain instructions to automate web support tasks, arXiv preprint arXiv:2103.16057 (2021).
[34] X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, Y. Su, Mind2web: Towards a generalist agent for the web, Advances in Neural Information Processing Systems 36 (2023) 28091–28114.
[35] X. H. Lù, Z. Kasner, S. Reddy, Weblinx: Real-world website navigation with multi-turn dialogue, arXiv preprint arXiv:2402.05930 (2024).
[36] W. Chen, J. Cui, J. Hu, Y. Qin, J. Fang, Y. Zhao, C. Wang, J. Liu, G. Chen, Y. Huo, et al., Guicourse: From general vision language models to versatile gui agents, arXiv preprint arXiv:2406.11317 (2024).
[37] V. Pahuja, Y. Lu, C. Rosset, B. Gou, A. Mitra, S. Whitehead, Y. Su, A. Awadallah, Explorer: Scaling exploration-driven web trajectory synthesis for multimodal web agents, arXiv preprint arXiv:2502.11357 (2025).
[38] M. Müller, G. Žunič, Browser use: Enable ai to control your browser, 2024. URL: https://github.com/browser-use/browser-use.
[39] Microsoft, Playwright, 2025. URL: https://github.com/microsoft/playwright, accessed: 2025-09-10.
[40] J. Yang, H. Zhang, F. Li, X. Zou, C. Li, J. Gao, Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v, arXiv preprint arXiv:2310.11441 (2023).
[41] P. Kumar, E. Lau, S. Vijayakumar, T. Trinh, S. R. Team, E. Chang, V. Robinson, S. Hendryx, S. Zhou,

M. Fredrikson, et al., Refusal-trained llms are easily jailbroken as browser agents, arXiv preprint arXiv:2410.13886 (2024).

[42] S. Zychlinski, A whole new world: Creating a parallel-poisoned web only ai-agents can see, arXiv preprint arXiv:2509.00124 (2025).