An Intelligent System Based on LLMs for Automated Healthcare Data Collection

Nathália C. O. C. Guimarães, Felipe C. B. Mello, Talita J. Fernandes, Luís F. H. Campelo, João G. M. G. Teodoro, Felipe A. L. Soares

Department of Computer Science
Pontifícia Universidade Católica de Minas Gerais (PUC Minas)
Belo Horizonte – MG – Brazil

{nathaliacocguimaraes,felipecbmello,talitajusto15}@gmail.com

{luisfhcampelo,jgmgteodoro}@gmail.com,felipesoares@pucminas.br

Abstract. *The Internet contains a large volume of important information for the health sector, spread across several websites. The excess of data and the variety of sources make it difficult to efficiently extract this information. This work proposes to automate this collection using Web Scraping, Natural Language Processing and Language Models. Techniques with BeautifulSoup (for static websites) and Selenium (for dynamic websites) were applied, with support from the GPT-4o-mini model. The tests showed a significant reduction in collection and structuring time. The methodology obtained 90% accuracy on static websites and 73% on dynamic websites. Despite limitations with extensive texts and Document Object Model (DOM) structure, the results show that the approach is viable.*

1. Introduction

The Internet gives us access to a wealth of important information related to the health sector, much of which is spread across various websites and organizations that provide information about health around the world [Abdurakhmonova et al. 2022]. Often, the amount of data is so large that it is easy to get lost in the middle of a search without finding the information you are looking for. Therefore, there is a need to group together as much of this data as possible and make it available in a way that facilitates the extraction of knowledge and decision-making based on it, in order to facilitate research in the health sector [Lunn et al. 2020].

Despite the great need to present data in a structured manner that makes it easy to extract knowledge, there are some implications in this process. The large volume of data already available on public health, a volume that is growing every day, can become an obstacle in the research phase, which is often manual, in the search for a medical solution, delaying the arrival of conclusions about the problem in question [Abdullah et al. 2013]. The existence of static and dynamic websites also slows down this process, since both function and respond to the user in different ways and require different approaches to extract information [Li et al. 2019].

There are some tools and ways to apply certain knowledge, such as web scraping, Natural Language Processing (NLP) and Large Language Models (LLM) capable of facilitating the process of automation and data collection, organizing what was found in a structured and designated way for the information extraction, although they are not widely explored in the area of computing or research in this way [Lunn et al. 2020, Kim et al. 2021].

In this context, this paper proposes an intelligent system that uses LLM to automate the health data collection process[1], performing automated and intelligent searches on the internet through web scraping and web crawling, and presenting the requested information to the user. In this way, the user can define the information they want to extract, and the system performs automatic and intelligent data collection, saving time on the project employed, reserving more time for project decision-making.

To validate the proposed approach, experiments were conducted on healthcare platforms that present both static and dynamic characteristics. The system was tested on specific information extraction tasks, such as mortality rates and disease prevalence, with the help of language models for automated filter selection and organization of the extracted data. The results demonstrated the effectiveness of the methodology in automating the information collection and structuring process, significantly reducing the time required for manual analysis and contributing to improving access to organized and relevant data in the context of public health.

The text is structured as follows: Section 2 reports on the related works to this research, while Section 3 presents the methodology used in the process described, being divided into materials and methods. Section 4 discusses the experiments performed and the results found. Finally, Section 5 concludes the research, presenting the main results and future works planned in this field of knowledge.

2. Related Works

The use of web scraping and the applied knowledge of NLP to automate data collection is an area that has been constantly researched, bringing great advances and new tools that aim to facilitate this process.

[Wu et al. 2018] developed a system that applies the concepts of NLP for the automated extraction of radiotherapy events from clinical texts. The study combines NLP models for the named entity recognition and the extraction of relationships between properties associated with radiotherapy events. This work presents both the need and the possibility of extracting information from medical texts, extracting knowledge from the collected data.

[Liu et al. 2019] proposed a deep neural network-based model for the recognition and extraction of named entities in online medical diagnosis data. The model combines different types of deep neural networks to improve the accuracy in identifying disease names, medical measurements, and therapies. The study highlights the use of a scalable web crawler to collect large volumes of online medical data, demonstrating the effectiveness of the model compared to previous approaches.

[Lunn et al. 2020] explored the use of web scraping and applied the knowledge of NLP to inform pedagogical practitioners, demonstrating how these technologies can be

[1]The developed algorithms are publicly available in our research repository: https://github.com/cart-pucminas/automated-healthcare-data-collection

used to collect and analyze large volumes of textual data, allowing for a deeper understanding of educational trends and student needs in the context of the study conducted. The authors highlight the importance of integrating these tools to automate data acquisition and improve evidence-based decision-making, regardless of the context or area of knowledge in which such tools and methodologies are applied.

[Single et al. 2020] investigated the application of NLP and ontology-based techniques for the automated information extraction from chemical accident databases. The study highlights the use of web scraping to enrich the extracted information with additional details about chemical substances, which are then organized into an ontological structure. This approach facilitates the retrieval and analysis of complex information, allowing the identification of causal relationships between accidents. The relevance of this work for current research lies in demonstrating how to integrate NLP and web scraping techniques to structure unorganized data, making it more accessible and useful for future analysis.

[Abdurakhmonova et al. 2022] investigated the use of web crawling technologies for the compilation of parallel corpora as an essential step in NLP. The study details the main techniques for automated data collection on web pages, highlighting the importance of these approaches for information retrieval and the creation of databases based on several languages. The authors emphasize the challenges encountered when dealing with the diversity and dynamicity of web pages, in addition to discussing solutions to improve the efficiency of the extraction process.

[Pichiyan et al. 2023] presented a detailed study on the combination of web scraping with NLP for the extraction and analysis of large volumes of unstructured textual data, transforming them into organized and accessible information. The article addresses web content extraction techniques, in addition to the use of specialized libraries, such as BeautifulSoup from the Python language, which is also used in this study.

[Guo et al. 2023] investigated the use of Large Language Models (LLM) to automate the search and compilation of archives. The study highlights the application of fine-tuned models for the information extraction from vast textual repositories, demonstrating how these technologies can significantly reduce manual effort in complex document analysis tasks. This work makes use of automated techniques for collecting and processing large volumes of textual data, taking advantage of the potential of LLMs to organize and interpret unstructured information efficiently.

[Bitterman et al. 2023] developed an end-to-end NLP system for the automated extraction of radiotherapy events from clinical texts. The study combines NLP models, such as BERT, for the named entity recognition and the extraction of relations between properties associated with radiotherapy events. The relevance of this work lies in demonstrating the feasibility of automated systems to collect and structure specific information in medical texts, addressing similar challenges in the healthcare field.

Therefore, it is possible to perceive the advances in the area of data collection automation and its evolution over the years, creating space for new techniques and tools that seek to solve the problem in various areas of knowledge, including the area of public health, validating the objective of this research.

3. Methodology

This work proposes an automated data collection methodology combining traditional web scraping tools with modern Natural Language Processing models. The approach involves two distinct data extraction pipelines: one for static web pages using BeautifulSoup, and another for dynamic websites using Selenium. In both cases, the extracted content is processed using the GPT-4o-mini language model to identify and structure relevant health-related information. The following subsections describe the materials used and the proposed method.

3.1. Materials

Developing this system required the integration of several technologies and frameworks to address the challanges of both dynamic and static web applications. The decision to adopt BeautifulSoup was driven by its ease of use, lightweight performance, and seamless integration with Python's ecosystem. Unlike heavier frameworks, BeautifulSoup requires minimal setup and provides a natural syntax for navigating tag hierarchies, which facilitates the identification of target elements such as tables, paragraphs, or divs marked with unique identifiers or class attributes [Single et al. 2020].

Once the HTML is extracted from a static site, BeautifulSoup parses it and structures the content in a tree format. This structure is then scanned for the components specified by the user's prompt. Although the library does not support JavaScript-rendered content, it is highly effective for static pages that contain pre-loaded HTML elements. This aligns well with the proposed architecture, where BeautifulSoup acts as the entry point for gathering raw data before passing it to the LLM for semantic filtering and structuring [Guo et al. 2023].

When combined with the OpenAI GPT-4o-mini API, BeautifulSoup enables an automated pipeline that converts unstructured static content into actionable, structured data. This synergy demonstrates how classic scraping tools can remain relevant and powerful when integrated with modern AI techniques [Abdullah et al. 2013].

Selenium WebDriver was chosen as the preferred framework for this solution due to its well-established reputation, extensive learning resources and robust documentation. A WebDriver acts as an intermediary between test scripts and web browsers, translating commands into browser-specific actions. Selenium provides broad compatibility with multiple browsers, enabling cross-browser testing with minimal adjustments.

We selected the GPT-4o-mini model for its optimal performance and efficiency in processing structured web data [OpenAI 2024]. The model's cost-effectiveness makes it ideal for dynamic web interaction tasks compared to larger LLMs, due to its better scalability potential.

3.2. Method

To find a solution for automating the collection of external data for the healthcare sector, two algorithms were developed to address the problem in two different ways: extracting information from static websites and dynamic websites. A static website is one that has fixed content, that is, it will present the same content to all its users. A dynamic website, on the other hand, can present different versions of its page to each user, allowing interactive activities and changing parts of how it is viewed according to the actions of each user. The method was developed as shown on Figure 1.

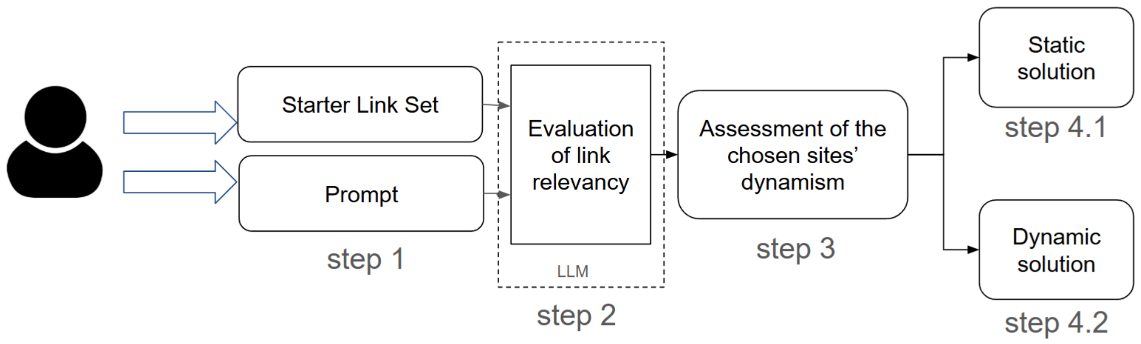

Figure 1. Core decision-making flowchart: LLM-assisted link assessment leading to static or dynamic solution paths.

The program starts by receiving links and prompts of what should be searched for from the user (step 1), while the next step is to analyze the website link and determine its dynamism using LLM (step 2). After that, it will be treated according to the result (step 3). For static sites (step 4.1), BeautifulSoup was used, while for dynamic sites (step 4.2), the Selenium data library was used.

3.2.1. Step 4.1 - Static Components Retrieval:

For static websites, the process of the algorithm is shown on Figure 2. The program accesses the relevant link (step 1 in Figure 2) and extracts the HTML using the Beautiful Soup library (step 2 in Figure 2). The extracted content is then sent to the GPT-4o-mini API (step 3 in Figure 2), which parses the data and extracts the relevant information according to the specified user input (step 4 in Figure 2). The extracted topics are processed and stored in a .json file for future use (step 5 in Figure 2). The following prompt was utilized in step 4.1:

You will receive the HTML code of a website. Your task is to identify and list the main topics directly related to the following theme: "{user-input}". Each topic should begin with the topic name (without introductions such as "relevant topic"), followed by a colon ':' and a list of directly related information separated by commas, ending each line with a semicolon ';'. For example: Infectious diseases: fever, chills, malaise; Prevention: use of repellent, vaccination, avoid stagnant water;. Focus only on content that truly addresses the requested theme. Ignore generic information, advertisements, or unrelated items. Be concise, clear, and direct in the information provided.

However, the GPT-4o-mini API has a limit of 128,000 tokens per call. This means that when processing HTML from websites, it is necessary to limit the number of characters sent for analysis to avoid exceeding this limit. As a result, pages that contain more text than the allowed amount may only be analyzed partially. To mitigate this limitation, more efficient strategies will be explored in the future, such as content segmentation and distributed processing of large pages.

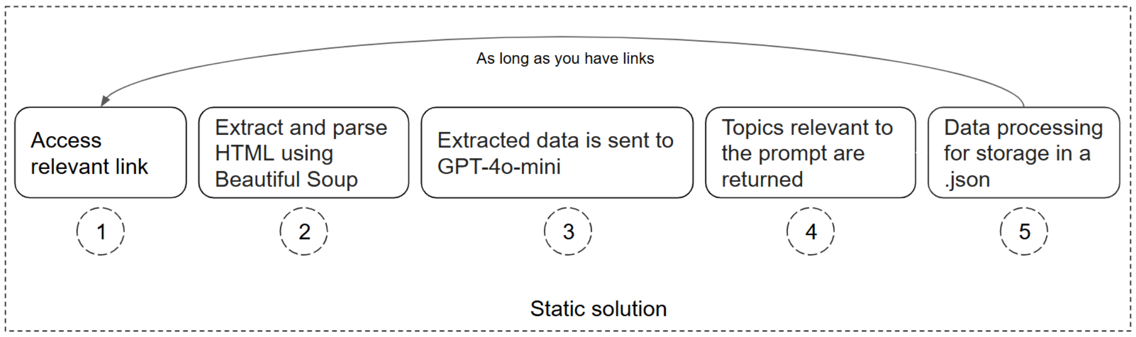
As long as you have links
Access relevant link
Extract and parse HTML using Beautiful Soup
Extracted data is sent to GPT-4o-mini
Topics relevant to the prompt are returned
Data processing for storage in a .json
1
2
3
4
5
Static solution

Figure 2. Static content processing workflow involving HTML extraction, LLM-driven topic refinement, and JSON serialization.

3.2.2. Step 4.2 - Dynamic Components Retrieval:

To handle dynamic web applications, a distinct approach is required due to the necessity of automated interaction with dynamic content, as shown on Figure 3. The initial phase of scraping dynamic websites involves configuring a browser automation tool (step 1 in Figure 3), in this case, Selenium. The setup process includes performance optimization and initializing the user's browser profile settings (e.g., cookies, cache, or saved credentials) to bypass potential authentication barriers.

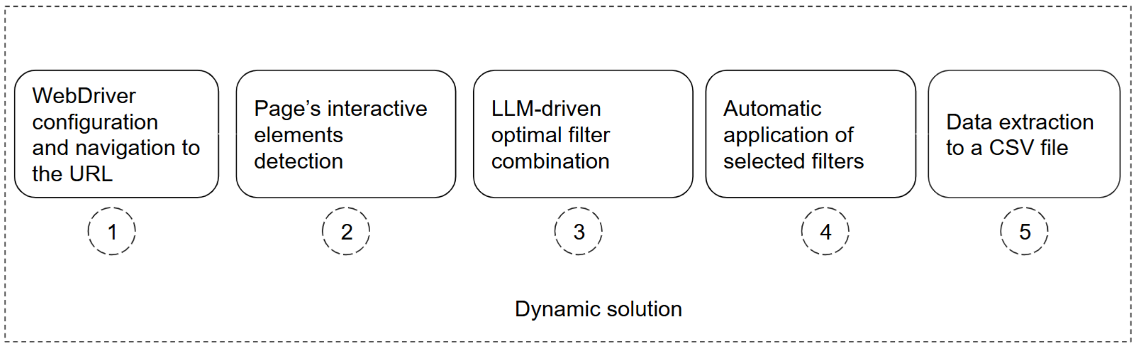
WebDriver configuration and navigation to the URL
Page's interactive elements detection
LLM-driven optimal filter combination
Automatic application of selected filters
Data extraction to a CSV file
1
2
3
4
5
Dynamic solution

Figure 3. Dynamic content processing pipeline with URL navigation via Web-Driver, followed by LLM-driven filter selection and structured CSV extraction.

The system carries out an extensive scan of the web interface to identify every user-interactive component. An algorithm analyses both standard HTML elements and UI frameworks, categorizing them by type, functionality and page structure(step 2 in Figure 3). Each element is then mapped using multiple identifiers to ensure reliable localization in dynamic environments.

The system formats the extracted element data into a structured input, and uses the OpenAI API to dynamically determine the optimal filter combination based on the user's requirements (step 3 in Figure 3). The model processes contextual data, such as element types, functionalities, and page structure, and returns a structured recommendation of filters tailored to the user's specific needs. This AI-driven approach ensures the selected filters align with the user's query and the dynamic web interface.

The filters refer to interactive user interface elements, primarily dropdown menus and selection components, that enable users to specify search criteria and refine data queries on dynamic web applications. These elements contain multiple selectable options organized by categories such as data types, geographic parameters, temporal ranges, demographic characteristics, and other domain-specific attributes.

In designing our prompt for filter combination analysis, we prioritized a structured, rule-based approach given that logical reasoning frameworks are better suited for tasks requiring deterministic precision [Son et al. 2025]. Our prompt establishes interconnected evaluation rules, explicit output formatting requirements, and unambiguous criteria for filter prioritization. This design ensures consistent label matching. The following is a representative prompt used in our experiments:

> *You are a specialized analyzer for web-based dropdown filters, tasked with evaluating interconnected dropdown menus to determine the optimal combination of selections based on their labels. Your analysis should treat all dropdowns as a unified system, prioritize the most relevant results based on user needs, and allow multiple selections per dropdown. The output must be a precise list in the exact format below, preserving all labels, options, punctuation, accents, and formatting:*
> *["Dropdown 1 Label", "Available Option A", "Available Option B"]*
> *["Dropdown 2 Label", "Available Option C", "Available Option D"]*
> *["Dropdown 3 Label", "Available Option E", "Available Option F"].*

During the filter application stage, the system executes a coordinated sequence of interactions with the web interface (step 4 in Figure 3), precisely following the AI model's recommendations. The selected filters are applied using the WebDriver approach, ensuring reliable manipulation of dynamic web elements. Following complete filter implementation, the system performs structural analysis of the resultant page, systematically extracting the resulting data (step 5 in Figure 3).

In addition to optimizing prompt engineering, ensuring reliability in dynamically interfaced systems requires addressing key technical challenges. These include maintaining linguistic coherence, managing asynchronous processes. Code-switching, which is the practice of alternating between languages within a single utterance, can negatively affect the performance of LLM models [Yuan et al. 2024]. A challenge in dynamic interfaces arises from the asynchronous loading of content, sometimes necessitating precise timing control mechanisms. This constraint may limit the generalization of potential solutions

4. Experiments and Results

A set of URLs composed of predominantly static and dynamic pages was used. For each of them, the HTML content was extracted using the BeautifulSoup library. Then, the first 16,384 characters of the extracted HTML were sent to the GPT-4o mini model, accompanied by a prompt formulated based on the criteria established in this work for classifying pages as static or dynamic. The model analyzed the structure of the content and returned, in general, whether the page presented more typical characteristics of a static or dynamic website. The prompt provided to the model considered that static pages tend to contain a greater volume of continuous and organized text, while dynamic pages have less text density and a predominance of visual or interactive elements.

Based on this automated classification, the system adopted two distinct processing flows: one aimed at handling pages classified as static and another with specific techniques aimed at dynamic pages.

4.1. Static Solution

In the static stage of the experiment, a method was created to extract titles from a vector containing 15 links, and all of them can found in the GitHub repository. These links were used to test the ability of the GPT-4o mini model to identify which titles are relevant according to a given thematic prompt provided by the user. The titles extracted from the links, through the BeautifulSoup library, are organized in an enumerated list (indexed from zero) and then the list is passed to the model, which returns the indexes of the titles considered relevant. The accuracy of the method was 90%, with 4 correct answers out of 5 tests. The only error occurred in the classification of *Tropical diseases*, in which the model possibly considered HIV/AIDS as belonging to the category. Considering this occuring as a partially correct answer, we were able to find the accuracy of the method. Table 1 shows the results of each test.

Table 1. Results of experiments performed on the static website data collection system

Test	Prompt	Corresponding Links	Results
1	Malaria	`https://bvsms.saude.gov.br/malaria-5/`	Correct
2	AIDS	`https://bvsms.saude.gov.br/hiv-e-aids/`	Correct
3	Artificial Inteligence	`https://pt.wikipedia.org/wiki/Intelig%C3%AAncia_artificial`	Correct
4	Tropical Diseases	`https://bvsms.saude.gov.br/malaria-5` `https://www.saude.pr.gov.br/Pagina/Febre-Maculosa` `https://bvsms.saude.gov.br/febre-amarela/` `https://bvsms.saude.gov.br/hiv-e-aids/` `https://bvsms.saude.gov.br/infeccao-pelo-virus-zika/` `https://bvsms.saude.gov.br/febre-de-chikungunya/`	HIV/AIDS may be debatable
5	Electric Vehicles	—	Correct

Finally, with the links selected from the text, the method that obtains the relevant topics for each page accessed is called. This method makes an HTTP request to the relevant link and, with the help of the BeautifulSoup library, transforms the website's HTML into a textual representation using the prettify() method, which organizes the content in a readable way. This textual representation (limited to approximately 550,000 characters to respect the model's capacity) is then sent to the GPT-4o mini model with a specific prompt, requesting the extraction of the most relevant topics according to the initial theme provided by the user. The model returns an organized list of topics, followed by their respective most significant information. This data is stored in a .json file, allowing for later analysis or structured use of the extracted information.

4.2. Dynamic Solution

For the system evaluation, we utilized Global Burden of Disease (GBD) Results Tool[1], a health data visualization platform presenting three characteristic challenges for dynamic web scraping: (1) React-based UI components requiring specialized interaction handling, (2) context-dependent Document Object Model (DOM) elements that only render after specific user-triggered events, and (3) multi-level hierarchical dropdown menus controlling dynamically generated tabular displays.

For this analysis, the standard system user input was *"Search for information on the death rate and prevalence of HIV in women located in Central Asia, between 2010 and 2018"*.

Employing the element mapping methodology, we derived a set of filter configurations from the predefined categorical dimensions: *Measure, Metric, Cause, Location, Age, Sex, and Year*. A comparative analysis between manually curated filters and those selected by the LLM is presented in Table 2, highlighting discrepancies and alignment in selection criteria.

Table 2. Filter Mapping Evaluation for HIV Query in Central Asian Women

Filter Category	Expected Selection	LLM Selection	Match Status
Measure	Deaths, Prevalence	Deaths, Prevalence	Correct
Metric	Number, Rate	Number, Percent	Partial
Cause	HIV/AIDS	HIV/AIDS	Correct
Location	Central Asia	Central Asia	Correct
Age	All ages	15-49 years, 20-24 years	Incorrect
Sex	Female	Female	Correct
Year	2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018	2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018	Correct

The Expected Selection column represents the standardized baseline filter options established for the control group, derived from the task specifications. The LLM Selection column records the options selected by the LLM during inference. To evaluate performance, the Match Status column categorizes the agreement between the LLM's output and the expected selections into three classes: Correct (exact match, 100% alignment), Partial (semantic or functional similarity but non-identical), and Incorrect (complete mismatch, 0% alignment).

[1]https://vizhub.healthdata.org/gbd-results/. Accessed April 16, 2025

The filter application and data extraction components of the system employ a multi-stage process to ensure accurate query configuration and reliable retrieval of epidemiological data. Following the mapping of all interactive elements, the system methodically applies user-specified filters by manipulating dropdown selections and resolving complex UI interactions, automatically clearing existing selections when necessary to prevent filter conflicts. Once filters are configured, the search execution module triggers the data retrieval process while implementing appropriate wait mechanisms to accommodate the asynchronous loading behavior characteristic of the GBD Results platform.

The data extraction module systematically processes the resulting tabular information by first detecting the underlying table structure, then extracting both headers and row data through strategic CSS selector targeting, before finally exporting the structured data to CSV format with appropriate timestamping.

To validate the system's accuracy, a similarity assessment was performed between autonomously extracted results and predetermined test cases, yielding a final similarity score of 72.62%. This validation incorporated weighted metrics across three dimensions: structural similarity (95.83%), schema alignment (100%), and distribution similarity (61.78%). While perfect alignment was achieved at the schema level, the distribution analysis revealed varying degrees of correspondence across individual columns, with cause and location attributes showing strong similarity (86.49%), but metric and demographic fields demonstrating more significant variation, highlighting areas for potential refinement in the filtering mechanism.

The similarity assessment results presented here are specific to this research, derived from a direct comparison between autonomously extracted data and predefined test cases using a custom analysis tool. The percentages were calculated through a multi-dimensional approach: structural similarity (95.83%) was determined by comparing row and column counts, normalized to their maximum values; schema alignment (100%) reflected perfect matching of column names; and distribution similarity (61.78%) was computed using Jaccard similarity to evaluate value frequency overlaps across shared columns. Weighting was applied to prioritize data fidelity, with distribution similarity assigned the highest weight (0.7) due to its direct impact on analytical outcomes, while structure and schema received lower weights (0.15 each) as they primarily reflect organizational consistency. This weighting scheme emphasizes the system's ability to reproduce meaningful data patterns, with discrepancies in metric and demographic fields suggesting targeted improvements for future iterations.

The current solution exhibits three key limitations. First, its implementation specificity restricts optimization to dropdown-type elements, offering limited support for alternative interaction patterns such as sliders, interactive maps, or date pickers. Second, language model constraints tie the system's filter selection performance to the GPT-4o-mini model's contextual understanding of user requirements, meaning ambiguous or overly complex queries may yield suboptimal filter recommendations. Third, the approach's heavy reliance on the web application's DOM hierarchy renders it vulnerable to interface redesigns or structural changes that disrupt element locators.

5. Conclusion

This study proposed an automated data collection approach for the healthcare sector, addressing the challenges posed by both static and dynamic websites. By integrating web scraping techniques with NLP and LLMs, the proposed method allows for structured and efficient extraction of relevant health data from online sources.

The experiments demonstrated the feasibility of the approach, particularly in handling complex web interfaces and dynamically generated content. The evaluation highlighted the system's effectiveness in extracting key health indicators while also revealing limitations related to prompt sensitivity, Document Object Model (DOM) dependencies, and specific UI interactions. Despite these challenges, the methodology proved capable of significantly reducing manual effort in data collection, improving accessibility to structured healthcare data.

In addition to reducing the time required to collect and structure data, the structured outputs generated by the proposed system can be used as input data sets for Machine Learning models in various applications in the healthcare sector. Data extracted on disease prevalence, mortality rates or demographic information, for example, can feed predictive algorithms aimed at epidemiological surveillance, identifying risk patterns or recommending public policies. The standardization of data enables its use in supervised and unsupervised models, expanding the potential of the solution as a basis for more advanced analyses and automated knowledge generation.

Future work will focus on refining filter selection accuracy, enhancing adaptability to diverse web structures, and exploring alternative interaction mechanisms for broader applicability. By addressing these aspects, the proposed solution can contribute to more reliable, scalable and generic data collection processes, supporting research and decision-making in the healthcare domain.

Acknowledgments

The present work was carried out with the support of the Research Incentive Fund of PUC MINAS (FIP/PUC MINAS), Coordination for the Improvement of Higher Education Personnel - Brazil (CAPES, code 001) and Pontifical Catholic University of Minas Gerais.

References

Abdullah, S. S., Rahaman, M. S., and Rahman, M. S. (2013). Analysis of stock market using text mining and natural language processing. In *2013 International Conference on Informatics, Electronics and Vision (ICIEV)*, pages 1–6.

Abdurakhmonova, N., Alisher, I., and Toirova, G. (2022). Applying web crawler technologies for compiling parallel corpora as one stage of natural language processing. In *2022 7th International Conference on Computer Science and Engineering (UBMK)*, pages 73–75.

Bitterman, D. S., Goldner, E., Finan, S., Harris, D., Durbin, E. B., Hochheiser, H., Warner, J. L., Mak, R. H., Miller, T., and Savova, G. K. (2023). An end-to-end natural language processing system for automatically extracting radiation therapy events from clinical texts. *International Journal of Radiation Oncology*Biology*Physics*, 117(1):262–273.

Guo, D., Yue, A., Ning, F., Huang, D., Chang, B., Duan, Q., Zhang, L., Chen, Z., Zhang, Z., Zhan, E., Zhang, Q., Jiang, K., Li, R., Zhao, S., and Wei, Z. (2023). A study case of automatic archival research and compilation using large language models. In *2023 IEEE International Conference on Knowledge Graph (ICKG)*, pages 52–59.

Kim, S., Choi, S., and Seok, J. (2021). Keyword extraction in economics literatures using natural language processing. In *2021 Twelfth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 75–77.

Li, H., Li, Z., and Rao, Z. (2019). Text mining strategy of power customer service work order based on natural language processing technology. In *2019 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*, pages 335–338.

Liu, X., Zhou, Y., and Wang, Z. (2019). Recognition and extraction of named entities in online medical diagnosis data based on a deep neural network. *Journal of Visual Communication and Image Representation*, 60:1–15.

Lunn, S., Zhu, J., and Ross, M. (2020). Utilizing web scraping and natural language processing to better inform pedagogical practice. In *2020 IEEE Frontiers in Education Conference (FIE)*, pages 1–9.

OpenAI (2024). Hello gpt-4o. Available from: https://openai.com/index/hello-gpt-4o/. Accessed: 2025-06-01.

Pichiyan, V., Muthulingam, S., G, S., Nalajala, S., Ch, A., and Das, M. N. (2023). Web scraping using natural language processing: Exploiting unstructured text for data extraction and analysis. *Procedia Computer Science*, 230:193–202. 3rd International Conference on Evolutionary Computing and Mobile Sustainable Networks (ICECMSN 2023).

Single, J. I., Schmidt, J., and Denecke, J. (2020). Knowledge acquisition from chemical accident databases using an ontology-based method and natural language processing. *Safety Science*, 129:104747.

Son, M., Won, Y.-J., and Lee, S. (2025). Optimizing large language models: A deep dive into effective prompt engineering techniques. *Applied Sciences by MDPI*.

Wu, J. T., Dernoncourt, F., Gehrmann, S., Tyler, P. D., Moseley, E. T., Carlson, E. T., Grant, D. W., Li, Y., Welt, J., and Celi, L. A. (2018). Behind the scenes: A medical natural language processing project. *International Journal of Medical Informatics*, 112:68–73.

Yuan, F., Yuan, S., Wu, Z., and Li, L. (2024). How vocabulary sharing facilitates multilingualism in llama?