A Demo of Interactive Thematic Data Collection on the Live Web

Michael West
tuo95818@temple.edu
Temple University
Philadelphia, Pennsylvania, USA

Eduard Dragut
edragut@temple.edu
Temple University
Philadelphia, Pennsylvania, USA

ABSTRACT

Many emerging AI systems can search and reason over web content, but most are designed for task-bounded objectives such as answering a question or producing a report. In contrast, many real-world workflows require thematic web-data collection: identifying and gathering large numbers of topically relevant documents distributed across the live web. We present a demonstration of an interactive web-data collection system for thematic collection building, implemented on top of a multi-agent focused crawling architecture with human-in-the-loop relevance and steering feedback. The system decomposes data collection into cooperating agents for query generation, page relevance assessment, and frontier exploration. These agents operate within a dynamic re-seeding loop that periodically re-enters the web through newly generated search queries, enabling exploration beyond the currently reachable crawl frontier and into disjoint regions of the web graph. A browser-based interface exposes the discovery process to users, who can inspect intermediate results, correct relevance judgments, and provide steering feedback that directly influences system behavior. The system records query histories, crawl statistics, and discovered pages as exportable artifacts. This demonstration highlights how interactive, agent-driven focused crawling can support long-horizon thematic data collection workflows in which users aim to build reusable collections of web resources rather than obtain a single synthesized answer.

PVLDB Reference Format:
Michael West and Eduard Dragut. A Demo of Interactive Thematic Data Collection on the Live Web. PVLDB, 19(12): 4554 - 4557, 2026.
doi:10.14778/3827998.3828064

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/JustusWest/Web-Data-Discovery.

1 INTRODUCTION

Large Language Models (LLMs) have accelerated the development of systems that search, browse, and reason over web content. Recent web-interacting agents revisit core challenges in information retrieval such as selecting promising entry points, iteratively updating relevance estimates, allocating limited interaction budgets, and deciding when to stop [11, 14, 19]. However, these systems are predominantly designed around *task-bounded* objectives: they seek sufficient evidence to answer a question, generate a report, or satisfy a single user request.

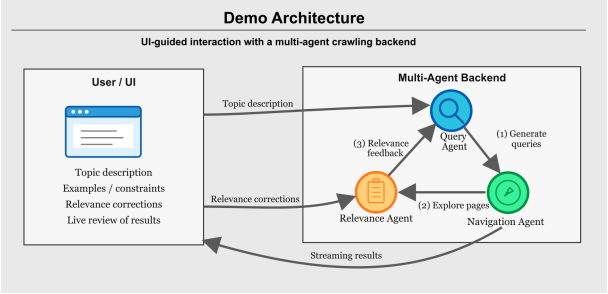

Figure 1: Architecture of the interactive thematic web-data collection system. The user defines a discovery task, while the backend coordinates query generation, relevance evaluation, and frontier exploration through local traversal and search-driven re-seeding. Results stream to the interface for inspection, feedback, and steering.

In contrast, many real-world workflows require *thematic web-data collection*. In such tasks, the objective is not to produce a single answer, but to discover and aggregate large sets of topically relevant documents distributed across the web [3]. For example, scientific data-management workflows have long required building reusable collections from distributed data sources, such as large-scale radar archives for hydrometeorology research [9]. A domain scientist studying public-health communication may similarly need to collect policy pages from hundreds of municipal websites [10].Similarly, identifying university programs such as Autism Support Programs (ASPs)—comprehensive support systems designed to help students transition to college and improve academic outcomes—requires aggregating information scattered across many institutional websites [12, 15, 17, 18]. More broadly, these tasks involve compiling structured collections of web pages (e.g., institutions, job postings) that satisfy specific criteria across decentralized sources [6]. Success therefore depends not only on local relevance judgments, but on sustained exploration efficiency and coverage across disjoint regions of the web graph [13].

Existing LLM-based web agents are not well-suited for this setting. Their interaction strategies are optimized for early stopping once sufficient evidence is found, *rather than for maximizing coverage or systematically exploring the web*. As a result, they tend to under-explore the search space and do not expose intermediate discovery decisions to the user.

Focused crawling provides a classical systems approach to long-horizon thematic exploration. It aims to discover topic-relevant pages while minimizing traversal of irrelevant regions of the web graph [4]. However, traditional focused crawlers rely on fixed seeds, static relevance estimators, and heuristic traversal policies [2, 5, 7].

Proceedings of the VLDB Endowment, Vol. 19, No. 12 ISSN 2150-8097.
doi:10.14778/3827998.3828064

These design choices limit their ability to escape local regions of the graph and adapt to evolving discovery signals [1, 16]. Moreover, they offer limited support for user interaction and steering during the collection process.

In this demonstration, we present an interactive system for *thematic web-data collection*, built on top of a multi-agent focused crawling architecture with LLM-powered components. Figure 1 illustrates the system. A user defines a data collection task through a natural-language collection theme and optional example pages. The system decomposes the process into cooperating agents responsible for query generation, page relevance evaluation, and frontier exploration. It alternates between local graph traversal and search-driven re-entry through dynamically generated queries, enabling exploration beyond the current crawl frontier and into previously disconnected regions of the web. Discovered pages stream continuously to a browser-based interface, where users can inspect results, correct relevance judgments, and provide steering feedback that directly influences system behavior. This design treats focused crawling as an underlying mechanism while elevating interactive, long-horizon collection building as the primary objective.

The demonstration makes three main contributions:

- An interactive system for thematic web-data collection that integrates query generation, relevance estimation, and frontier exploration through specialized agents.
- An interactive interface enabling human-in-the-loop steering as users build thematic collections of web documents.
- A session-based workflow that generates structured, exportable artifacts for inspection and downstream analysis.

2 SYSTEM OVERVIEW

The system consists of a Python-based asynchronous backend for thematic web-data collection and a React web interface. The backend uses FastAPI to support session management, human-in-the-loop steering, and artifact export. Query, relevance, and navigation agents are implemented using the DeepSeek-V3.1 API.

2.1 Data Collection Loop

Focused crawling offers a natural starting point for thematic web-data collection. It is a selective web traversal paradigm designed to discover pages relevant to a target topic while avoiding exhaustive exploration of the web graph [1, 2, 4, 5, 8]. A traditional focused crawler relies on three key components: seed pages that define entry points into the web graph, a relevance estimator for evaluating visited pages, and a frontier manager that prioritizes which links to explore next. However, this formulation is not well-suited for interactive, long-horizon data collection workflows. Seed pages are typically fixed prior to crawling, limiting the system's ability to adapt entry points as new signals about the topic emerge [16]. Relevance estimation is often based on static, pre-trained classifiers, which restricts the system's ability to refine its notion of relevance as it encounters additional positive and negative examples. Moreover, link-based exploration alone can become inefficient in sparse or weakly connected regions of the web graph, making it difficult to reach disjoint areas that may contain relevant content. Hence, while focused crawling provides a useful conceptual baseline, it lacks the adaptability, coverage, and user interaction required for effective thematic web-data collection in modern settings.

Algorithm 1 Dynamic Re-Seeding Data Collection Loop

Require: theme description t, examples E, crawl budget B
initialize frontier $F \leftarrow \emptyset$, visited $V \leftarrow \emptyset$, relevance $R \leftarrow \emptyset$
while B not exhausted **do**
  $Q \leftarrow \textsc{QueryAgent}(t, E, R)$
  $C \leftarrow \textsc{Search}(Q)$ ▷ candidate entry pages
  $F \leftarrow F \cup \{c \in C \mid c \notin V\}$
  while $F \neq \emptyset$ and local exploration budget remains **do**
    select highest-priority page p from F
    visit p and add p to V
    $(p, s, r) \leftarrow \textsc{RelevanceAgent}(p, t, E, R)$
    record page, score, and rationale in R
    if user feedback available **then**
      update R with corrections and guidance
    $L \leftarrow \textsc{ExtractLinks}(p)$
    $L' \leftarrow \textsc{NavigationAgent}(L, p, t, R)$
    $F \leftarrow F \cup \{\ell \in L' \mid \ell \notin V\}$

Our system addresses these limitations with a *dynamic re-seeding loop*. Instead of relying on a pre-compiled set of seed pages, we adopt a few-shot approach in which the user begins with a natural language description of the collection theme and either provides a few examples of relevant URLs and/or provides direct relevance feedback on the first few URLs the crawler encounters.

Algorithm 1 outlines the iterative collection process, integrating query generation, local exploration, relevance evaluation, and user feedback into a unified adaptive crawl loop. This loop balances *local exploitation* with *global exploration* across the web.

2.2 Multi-Agent Architecture

The discovery process is implemented through three cooperating agents: a *Query Agent*, a *Relevance Agent*, and a *Navigation Agent*. Each agent operates on a multi-turn conversation which includes structured inputs derived from the crawl state and produces outputs that either modify the crawl state or influence the policy of other agents. To ensure reliable interaction between agents, outputs are produced in JSON containing fields for queries, relevance labels, confidence scores, or ranked link candidates.

Query Agent. The Query Agent is responsible for generating search engine queries that identify promising entry points into the web graph. Its inputs include the user-provided collection theme, previously issued queries, and summaries of pages discovered from previously issued queries. The agent is prompted to produce queries that maximize the likelihood of discovering new relevant domains while avoiding redundant exploration. The output of this agent is a list of queries, which are issued through a search engine API to retrieve candidate pages for exploration.

Relevance Agent. The Relevance Agent evaluates whether visited pages are relevant to the collection theme. Its inputs include the URL, page title, extracted content, collection theme, example page content, and user-provided feedback from previous pages. The agent produces a structured output consisting of a binary relevance classification, a confidence score, and a short textual rationale.

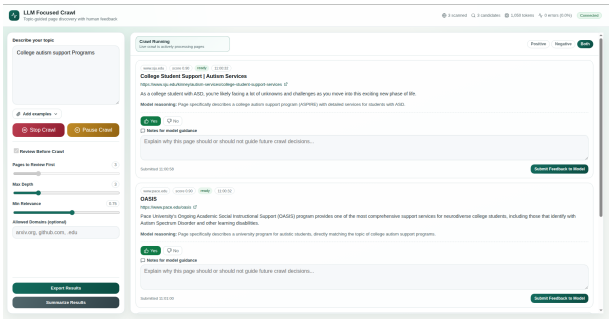

Figure 2: User interface of the thematic web-data collection system. The left panel specifies the task and crawl parameters; the right panel streams discovered pages and supports relevance feedback.

These relevance judgments update the crawl state and provide feedback signals to both the Query and Navigation agents.

Navigation Agent. The Navigation Agent manages the local crawl frontier. Its inputs include outgoing links discovered on recently visited pages, the surrounding page content of the links and relevance assessments of previously selected links. The agent ranks candidate links based on their likelihood of leading to additional relevant pages. The resulting ranked frontier determines which pages are explored next during local traversal.

User feedback provided through the interface is incorporated into this process as an additional signal available to all agents. When a user corrects a relevance judgment or provides textual guidance, this information is appended to the agents' context during subsequent query generation, relevance evaluation, and link prioritization steps. In this way, the collection policy can adapt dynamically as the crawl progresses.

Together these agents form an adaptive system for thematic web-data collection. The Query Agent identifies new entry points, the Navigation Agent explores the local regions of the graph surrounding those entry points, and the Relevance Agent evaluates visited pages to provide feedback that influences both query generation and link prioritization. Each visited webpage is stored as a structured record containing its URL, domain, title, extracted content, relevance score, originating query, and associated user feedback. A session manager maintains ordered results, event logs, and crawl statistics so that sessions can be resumed and artifacts exported after completion.

2.3 User Interface

The web interface, seen in Figure 2, presents the data collection process as a live session. On the left side of the interface the user specifies a collection theme, optional example pages, crawl parameters, and optional domain filters. The crawl parameters allow the user to specify how many pages they want to manually review before the system begins, a confidence threshold for a page to be considered relevant and the maximum traversal depth from an entry point. On the right side the system streams discovered pages in real time. Each result includes the page title, URL, content snippet, relevance score, Relevance Agent reasoning, and feedback controls. The feedback controls allow the user to correct binary relevance classification and optionally provide an open-ended text feedback.

The typical workflow is as follows:

(1) The user defines a collection theme and optionally provides a small number of example URLs.
(2) The *Query Agent* generates search queries and retrieves an initial batch of candidate pages.
(3) If review mode is enabled, the crawl pauses and waits for user feedback on the first batch before continuing.
(4) The crawl resumes, streaming discovered pages incrementally. The user may continue to provide feedback or they may leave the system running independently.
(5) When finished, the user can summarize session results and export crawl artifacts.

Reviewing the initial batch of pages allows users to verify that the crawler is exploring appropriate regions of the web before large-scale expansion begins. This is also where the distinction from answer-oriented LLM systems becomes concrete. Rather than asking whether the model can already answer the user's question, the system asks whether the direction of exploration is appropriate for building the intended collection and whether the early results are surfacing the right regions of the web.

To support post-run analysis, session artifacts are stored as structured CSV files recording page URL, title, content, relevance, and originating query, along with query history and user feedback. The frontend provides a summary view with crawl statistics and a narrative overview generated from sampled results and query history. Together, these features make the system more than a crawler interface: it becomes a reproducible collection-building environment that preserves both *what* was found and *how* it was found.

3 DEMONSTRATION DESCRIPTION

In the demonstration, we show how the system supports thematic web-data collection through interactive exploration scenarios.

3.1 Example 1: Domain-Science Expert

Consider a researcher attempting to identify universities in the United States that host Post-Secondary Autism Support Programs (ASPs) [12, 17, 18]. This is a sparse collection task where relevant pages are distributed across many independent university domains.

(1) The user enters a collection theme: "College-based support programs for students with autism in the United States."
(2) The *Query Agent* generates initial search queries and retrieves a batch of candidate pages.
(3) With the "review before crawl" setting enabled, the user inspects the initial batch and notices the system has retrieved several pages focused on general autism medical research and programs for special education teachers.
(4) The user intervenes, marking the erroneous pages as "non-relevant" and adding a textual note emphasizing disability services and academic support programs.
(5) The crawl resumes. Updated relevance signals and refined search queries steer exploration toward university disability service websites across new domains.
(6) After running the system for some time, the user pauses the crawl, exports the results, and inspects the collected

pages. They may provide additional feedback and decide whether further exploration is needed.

(7) Upon reaching satisfactory quality and coverage, the user stops the system and exports the results, accompanied by detailed statistics on queries issued, pages retrieved, relevance proportions, and human-verified accuracy.

3.2 Example 2: Job Seeking

A second demonstration scenario focuses on academic job search. Consider a PhD Candidate in Computer Science who is seeking a faculty position at a university in the US. Many websites exist to serve as "job-boards" for these positions, but there is no guarantee that a particular job will be posted on any given site. Thus, our candidate must regularly monitor a number of sites.

(1) The user enters a detailed natural language description: "Search for faculty positions in Computer Science at Universities in the US. Prioritize tenure track positions."
(2) If the user wants to monitor particular job-board or universities, they may limit the search to only those domains using the "Allowed Domains" filter.
(3) The *Query Agent* generates initial search queries and retrieves a batch of candidate pages.
(4) With the "review before crawl" setting enabled, the user inspects the initial batch and notices the system has only returned tenure-track postings.
(5) The user leaves relevance assessment unchanged and provides guidance: "Broaden your search to include all full-time faculty positions, including non-tenure-track and visiting".
(6) The *Query Agent* and *Navigation Agent* incorporate this feedback to broaden the collection.
(7) After running the system for some time, the user pauses the session, exports results and leaves the page to apply to a number of jobs.
(8) This session can be resumed later or repeated periodically to maintain an up-to-date collection of relevant postings.

These examples highlight the distinction between thematic web-data collection and answer-oriented web agents. Question answering systems aim to gather enough evidence for a final response, whereas data collection seeks to build a comprehensive set of relevant documents. Premature stopping risks missing entire regions of the web graph, so our system emphasizes long-horizon exploration and coverage over disjoint regions of the web.

This distinction also changes the role of human oversight. Deep-research assistants typically involve a user query followed by a final report. Our system keeps the user involved throughout the collection process: users inspect intermediate results, correct relevance judgments, provide steering feedback, and export the evolving collection. This is especially important in domain-science settings, where provenance and coverage matter.

4 CONCLUSION

We will present a demonstration of an interactive system for thematic data collection on the live-web. We implement this system through a multi-agent architecture which integrates query generation, relevance estimation, and frontier exploration to support dynamic re-seeding and adaptive web-graph exploration. The demonstration includes an interactive data collection workflow in which users can inspect intermediate results, guide the crawl through feedback, and export the resulting collection in structured form.

More broadly, we distinguish between answer-oriented deep-research systems and thematic data collection systems. While both rely on search and relevance estimation, they optimize for different outcomes. Our system targets users who need to discover, inspect, and preserve collections of web data rather than stop at a single synthesized answer. As LLM-based agents operate in increasingly open-ended environments, supporting transparent and controllable data collection workflows becomes an important systems challenge. This demonstration illustrates how interactive, agent-driven crawling can support such workflows.

Acknowledgements This work was supported in part by the U.S. National Science Foundation awards III-2107213, 2026513, and ITE-2333789.

REFERENCES

[1] Charu C Aggarwal, Fatima Al-Garawi, and Philip S Yu. 2001. Intelligent crawling on the World Wide Web with arbitrary predicates. In *Proceedings of the 10th international conference on World Wide Web*. 96–105.
[2] Luciano Barbosa and Juliana Freire. 2007. An adaptive crawler for locating hidden-web entry points. In *WWW*. 441–450.
[3] Michael J Cafarella, Alon Y Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. WebTables: Exploring the power of tables on the web. *Proc. VLDB Endow.* 1, 1 (2008), 538–549.
[4] Soumen Chakrabarti, Martin Van den Berg, and Byron Dom. 1999. Focused crawling: a new approach to topic-specific Web resource discovery. *Computer networks* 31, 11-16 (1999), 1623–1640.
[5] Michelangelo Diligenti, Frans Coetzee, Steve Lawrence, C Lee Giles, Marco Gori, et al. 2000. Focused Crawling Using Context Graphs.. In *VLDB*. 527–534.
[6] E. Dragut, Wensheng Wu, P. Sistla, C. Yu, and Weiyi Meng. 2006. Merging Source Query Interfaces onWeb Databases. In *ICDE*. 46–46.
[7] Eduard C. Dragut, Weiyi Meng, and Clement T. Yu. 2012. *Deep Web Query Interface Understanding and Integration.* Springer Nature.
[8] Judy Johnson, Kostas Tsioutsiouliklis, and C Lee Giles. 2003. Evolving strategies for focused web crawling. In *ICML-03*. 298–305.
[9] Anton Kruger, Ramon Lawrence, and Eduard C. Dragut. 2006. Building a Terabyte NEXRAD Radar Database for Hydrometeorology Research. *Computers & Geosciences* 32, 2 (2006), 247–258.
[10] Isabelle S Kusters, Amanda M Gutierrez, Julianna M Dean, Mark Sommer, and Anna Klyueva. 2023. Spanish-language communication of COVID-19 information across US local health department websites. *Journal of Racial and Ethnic Health Disparities* 10, 5 (2023), 2482–2489.
[11] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. In *EMNLP*. 5420–5438.
[12] Brett Ranon Nachman, Catherine Tobin McDermott, and Bradley E Cox. 2022. Brief report: Autism-specific college support programs: Differences across geography and institutional type. *Journal of Autism and Developmental Disorders* 52, 2 (2022), 863–870.
[13] Christopher Olston and Marc Najork. 2010. Web crawling. *Foundations and Trends in Information Retrieval* 4, 3 (2010), 175–246.
[14] Shuang Sun et al. 2025. SimpleDeepSearcher: Deep Information Seeking via Web-Powered Reasoning Trajectory Synthesis. In *Findings of EMNLP*.
[15] Matt Tincani, Hyangeun Ji, Maddie Upthegrove, Elizabeth Garrison, Michael West, Donald Hantula, Slobodan Vucetic, and Eduard Dragut. 2024. Vocational interventions for individuals with ASD: Umbrella review. *Review Journal of Autism and Developmental Disorders* 11, 4 (2024), 806–842.
[16] Karane Vieira, Luciano Barbosa, Altigran Soares Da Silva, Juliana Freire, and Edleno Moura. 2016. Finding seeds to bootstrap focused crawlers. *WWW* 19 (2016), 449–474.
[17] Kathleen D Viezel, Elizabeth Williams, and Wesley H Dotson. 2020. College-based support programs for students with autism. *Focus on Autism and Other Developmental Disabilities* 35, 4 (2020), 234–245.
[18] Michael West, Matthew Tincani, Donald Hantula, Sungsoo Ray Hong, Slobodan Vucetic, and Eduard Dragut. 2025. Applying Data Science Practices to Identify Characteristics of Postsecondary Autism Support Programs From Their Websites. *Journal of Autism and Developmental Disorders* (2025), 1–47.
[19] Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. In *EMNLP*. 414–431.