IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 6, June 2025

DOI: 10.17148/IJARCCE.2025.14632

A Survey on Domain Expert Finding System

Assistant Prof. Ms. Maddela Bhargavi[1], Sachin Somashekhar Kumbar [2], Mokesh G R[3], Nagarjun Kumar S[4], P C Tejas[5]

Department of Computer Science, K S Institute of Technology, Bengaluru, Karnataka[1-5]

Abstract: In the modern interdependent academic and professional landscape, finding the most appropriate domain expert is essential for cultivating collaboration, stimulating research, and industrial innovation. Yet, existing expert identification processes are typically disorganized, time-consuming, and decentralized. The Domain Expert Finder System (DEFS) rectifies these shortcomings by using web scraping to gather publicly accessible information from academic institutions, professional networks, and scholarly stores. This information is filtered, processed, and saved into a structured database, and users can access complete expert profiles through an interactive and searchable platform. In contrast to current systems that depend only on publication records or social networks, DEFS encompasses both faculty members and located students, hence expanding the range of expert discoveries. This paper overviews the system architecture, methodologies, comparative studies, and real-world implementations of DEFS. It talks about the major advantages like enhanced search accuracy, scalability of data, and automation along with ethical issues, technological hurdles, and prospects. Through a critical analysis of the literature and experimental verification, this survey points out the ways in which DEFS is an efficient and accessible solution to the issue of expert identification in contemporary knowledge environments.

Keywords: Expert Discovery, Web Scraping, Academic Profiles, Automation, Data Mining, Faculty Search, Placed Students, Knowledge Graphs

I. INTRODUCTION

In the age of digitalization, when knowledge and know-how hold the key to innovation, efficient identification of domain experts has become a crucial imperative for the academic and industrial environments alike. From research collaborations to technical mentorship, from consultancy positions to talent discovery, expert identification supports an array of processes within higher education and business settings. But conventional methods of finding domain experts are usually tardy, unreliable, and dispersed over several platforms. Professionals and institutions usually settle for scanning institutional directories, searching manually on LinkedIn or ResearchGate, or using static alumni databases, all of which have no real-time relevance, customizable parameters, and centralized access.

The Domain Expert Finder System (DEFS) then presents itself as the new solution to this challenge by utilizing the power of web scraping to provide automation for data collection. DEFS is intended to harvest publicly available expert information from college sites, placement platforms, and online research profiles. The system processes and structures data like name, designation, area of expertise, research work, and placement credentials and stores it in structured and searchable format. Not only is this process efficient, but also it guarantees scalability and minimized human effort in data collection. One of the defining characteristics of DEFS is its broad scope: it outlines both faculty members and students placed. This broad inclusion makes the system applicable to a larger body of users, comprising academic researchers, mentee students, institutions assessing outside collaborations, and businesses looking for talent pipelines. Further, DEFS features a responsive search interface that allows filtering along domain, institution, and professional level, delivering users informative and timely results.

The DEFS platform seeks to overcome the disparity between dispersed web data and timely expert identification. By mosaicking ethical web scraping with smart filtering technologies, it increases accessibility, promotes collaboration, and enables users to make well-informed decisions. This introduction provides the ground-level motivations behind DEFS and paves the way for an in-depth analysis of its architecture, methods, and impact.

II. LITERATURE SURVEY

1. Zhang et al. (2022) – "Expert Finding in Heterogeneous Bibliographic Networks" This study, published in IEEE Transactions on Knowledge and Data Engineering, presents an expert recommendation framework that leverages heterogeneous bibliographic networks. The model incorporates citation patterns, co-authorship links, and venue information to rank researchers. By applying graph mining techniques and topic-aware random walks, the system

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2025.14632

effectively identifies influential experts within academic communities. While this approach is robust in publication-centric domains, it lacks integration with non-research data such as placements or institutional roles, which DEFS addresses.

2. Patel et al. (2023) – "Framework for Academic Expert Identification Using Web Mining" Featured in Springer's Journal of Web Engineering, this paper introduces a scalable framework for identifying academic experts using web scraping and rule-based filtering. The researchers developed crawlers to extract faculty data from Indian engineering college websites, normalizing information for searchability. They also implemented a reputation scoring mechanism based on publication volume and citation count. Although similar in methodology to DEFS, this framework excludes placed students and dynamic search filters, limiting its practical reach.

3. Kumar et al. (2021) – "Automatic Extraction of Faculty Information Using Crawlers" Published by Elsevier, this work focuses on automating the discovery of faculty profiles using data crawlers and syntactic parsing. The system targets institutional websites, extracting structured data like names, departments, and contact information. It also uses a lightweight validation module to reduce duplicate entries. The primary limitation is its narrow scope—only faculty members are considered, and no provision is made for periodic updates. DEFS extends this by including student profiles and a scheduled update mechanism.

4. Li et al. (2020) – "AI-Driven Expert Recommendation Systems in Knowledge Networks" This paper, published in ACM Computing Surveys, explores how artificial intelligence and machine learning models can enhance expert recommendation systems. It surveys techniques such as collaborative filtering, neural embeddings, and semantic search applied to large-scale academic databases. While the study underscores the future potential of intelligent systems, it notes the dependency on high-quality, annotated datasets, which limits scalability. DEFS takes a more grounded approach by relying on publicly available structured data and incorporating machine-learnable fields for future integration.

5. AMiner Platform – An Online Expert Network AMiner is a publicly accessible platform developed by Tsinghua University that aggregates academic profiles based on research output. It offers semantic search, citation analysis, and author disambiguation using NLP and graph algorithms. AMiner's strengths lie in its deep integration with academic databases and its high-precision citation metrics. However, its user base is limited to researchers with a substantial publication record, excluding early-career professionals and students. DEFS complements this by extending coverage to placed students and providing broader filtering criteria.

III. OBJECTIVES

1. Automated Expert Identification: Develop a scalable system to automatically extract and organize expert data from trusted public sources, such as institutional directories and placement records.
2. Inclusive Expert Coverage: Include both faculty and placed students to ensure the system serves academic researchers, students, institutions, and companies alike.
3. Feature-Rich Platform: Implement ethical web scraping practices, enable multi-criteria search filters, and support periodic updates for accurate, efficient, and ethical expert discovery.

IV. METHODOLOGY

The methodology used in the design of the Domain Expert Finder System (DEFS) is an order mechanism for a series of processes aimed at expert identification automation with web scraping, data normalization, and smart querying. DEFS is developed on modular pieces that collectively guarantee accuracy, scalability, and ethical behaviour.

1. Data Acquisition through Web Scraping

Automatic data retrieval from publicly available sources forms the first stage of the methodology. Web scraping is done through Python-based libraries like BeautifulSoup for static web pages and Selenium for dynamic content that is rendered by JavaScript. These scripts scrape faculty directories, placement websites, research repositories (e.g., Google Scholar), and professional networking websites like LinkedIn (only where there is public information). The scraper is programmed to extract important profile attributes like name, designation, field of expertise, publications, affiliations, email, and important links.

Guarantee ethical data collection, the scraper abides by every site's robots.txt file and uses rate limiting and randomized intervals for requests so as not to overwhelm servers. Scraping is suspended on the appearance of CAPTCHAs or denial of access warnings, and no login-protected or paywalled content is accessed to remain compliant with legal standards.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 6, June 2025

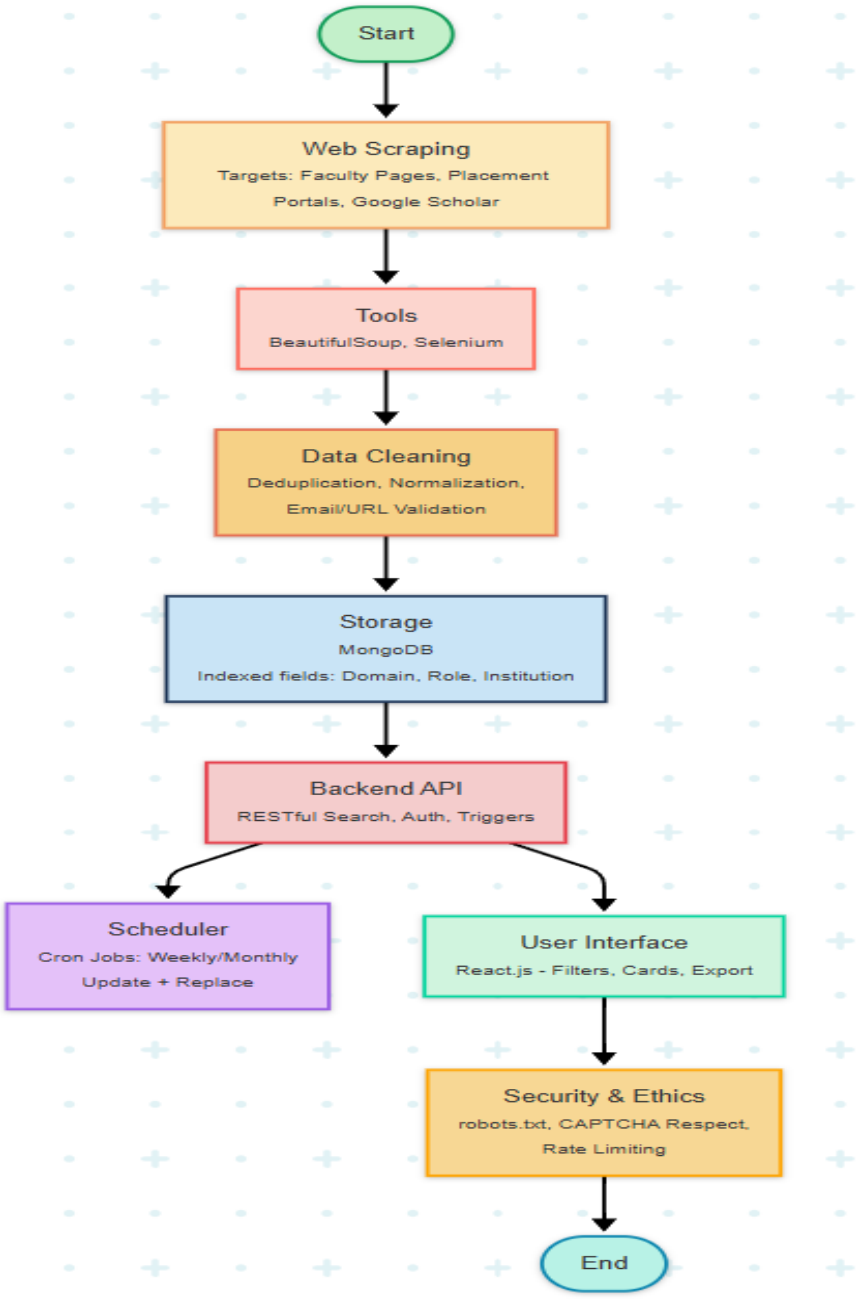
DOI: 10.17148/IJARCCE.2025.14632

2. Data Preprocessing and Cleaning

Raw data gathered using scraping tend to be inconsistent, duplicate, and contain encoding errors. Preprocessing involves tidying up and normalizing such information for reliability. Tasks involved in data cleaning are:

- Duplicate removal based on name and email matching.
- Normalization of academic titles (e.g., "Asst. Prof.", "Assistant Professor") into a standard format.
- Validation of links, emails, and institutional domains.
- Correction of encoding and formatting problems.
- Removal of incomplete or irrelevant records.

Regular expressions, dictionary mapping, and exception handling scripts are utilized to maintain data hygiene. Tagging and categorizing data fields to set for structured storage and fast retrieval is also a part of this step.



IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 6, June 2025

DOI: 10.17148/IJARCCE.2025.14632

3. Structured Data Storage

The processed and cleaned data is stored in a MongoDB database. MongoDB's document model facilitates schema flexibility, which is ideal for storing semi-structured expert profiles. A document is used to represent each expert with fields like unique ID, name, department, domain, designation, affiliations, publication links, update timestamps, and reference URLs.

Indexes are created on high-usage fields such as domain and institution to enable fast and efficient querying. MongoDB also supports nested fields and dynamic expansion, which is useful when additional metadata needs to be appended in the future.

4. Backend API and Scheduler Integration

A backend server is coded in Node.js or Django that exposes a RESTful API interface for data interaction between frontend and database. The API supports users to query expert profiles based on several filters such as domain, designation, or institution. Authentication, rate-limiting, and logging user interactions for analytics are also supported by the API.

An in-built cron job scheduler is used with the backend to schedule periodic scraping as well as database updates. The scheduler executes scraping scripts periodically—weekly or monthly—based on the source website update frequency. It scans new data and replaces entries only if it detects verified changes, optimizing bandwidth and processing.

5. User Interface and Search Features

The frontend is built in React.js, with a responsive and interactive user interface. Advanced search is supported using multi-filter options (e.g., domain, role, institution). The interface shows search results in cards, each with a short expert summary and links to primary sources such as research profiles or LinkedIn pages. Pagination, bookmarking, and profile exporting are also accommodated.

6. Security and Ethical Compliance

DEFS emphasizes data security and ethical compliance. All communications are performed over HTTPS, and user inputs are sanitized to avoid injection attacks. The scraper has logic for detecting and reacting to access denials and keeps logs of all access points. User or institutional requests for exclusion from profiles can also be made, and those are adhered to in a timely manner.

V. APPLICATION REQUIREMENTS

Hardware Requirements

- *Processor*: Minimum Intel Core i3 (2.4 GHz) or AMD equivalent; Recommended Intel Core i5/i7 or AMD Ryzen 5/7 for optimal performance.
- *RAM:* Minimum 4 GB; Recommended 8–16 GB for handling concurrent scraping, database operations, and UI rendering.
- *Storage:* Minimum 250 GB HDD/SSD; Recommended 500 GB SSD for faster data access and reduced latency in backend operations.
- *Display:* Minimum 1366x768 resolution; Recommended Full HD (1920x1080) for better visibility of web interface and debugging tools.
- *Internet Connectivity:* Minimum 10 Mbps for basic scraping tasks; Recommended 50 Mbps or higher for smooth operation during bulk data extraction.
- *Graphics Card:* Not mandatory; Optional GPU (NVIDIA or AMD) if integrating ML models for future enhancements.
- *Power Backup:* UPS recommended for desktop setups to prevent data loss during scraping or update cycles.
- *Peripheral Devices*: Standard keyboard and mouse; optional dual-monitor setup for improved developer efficiency.

Software Requirements

- *Operating System:* Compatible with Windows 10/11, Ubuntu 20.04+ (Linux recommended for server deployment), or macOS 10.14+.
- *Programming Language:* Python 3.8 or higher for scripting web scraping and preprocessing modules.
- *Python Libraries:*
  - BeautifulSoup for static HTML parsing

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 6, June 2025

DOI: 10.17148/IJARCCE.2025.14632

- Selenium for handling dynamic content
- pandas and NumPy for data manipulation
- requests and urllib for HTTP operations
- regex and lxml for pattern matching and XML parsing.

- *Backend Framework*:
  - Node.js (v14 or later) with Express.js for API handling
  - Or Django (v3.2 or later) for integrated backend and admin interface
- *Frontend Framework:* React.js (v17+) for building responsive and dynamic user interfaces.
- *Database System:* MongoDB for storing semi-structured expert profiles with support for schema flexibility and index-based querying.
- *Scheduling Tools:* Cron jobs or Task Scheduler for periodic scraping, auto-refresh, and scheduled database updates.
- *Security Tools:* HTTPS for secure communication; Helmet.js or Django-secure for HTTP header protection; input sanitization modules.
- *Development Tools:*
  - Visual Studio Code or PyCharm as IDEs
  - Git for version control
  - Mail carrier for API testing.
  - ChromeDriver/GeckoDriver for Selenium browser automation
- *Optional Tools:*
  - Docker for containerized deployment
  - Anaconda or venv for environment isolation
  - Webpack or Babel for frontend build optimization.

VI. CONCLUSION

The Domain Expert Finder System (DEFS) offers an exhaustive, scalable, and extensive solution to the contemporary problem of finding domain experts in academic and industrial contexts. With the explosive explosion of information and the dispersal of expert information on a myriad of online sites, conventional approaches to expert discovery are inefficient and obsolete. DEFS fills this gap by mechanizing the data extraction process utilizing responsible web scraping methods, thus greatly minimizing manual labor, maximizing data accuracy, and maintaining real-time updates.

Another of the system's major strengths is its inclusivity. By including both faculty staff and placed students in its profiling database, DEFS serves a broad spectrum of users — researchers looking for collaborators to students looking for mentors, institutions establishing partnerships to corporations searching for potential candidates. The capacity to narrow down results through domain area of expertise, institution, and type of role ensures the site is both user-centred and relevant.

Technically, DEFS is designed on a strong and modular architecture. Python-based web scraping libraries such as BeautifulSoup and Selenium are used to scrape structured data from dynamic and static web pages. The data is cleaned and normalized in MongoDB for scalability and fast data retrieval.

A RESTful backend coded in Node.js or Django manages all user requests and React-based frontend provides a responsive and interactive user interface. Regular refreshments through schedulers keep information up to date and current.

While it is confronted by issues like anti-scraping technologies, divergent web architectures, and ethical concerns, the system is made to learn and evolve. Adhering to public data policies, honouring opt-out terms, and safe data handling practices demonstrate the system's concern for responsible use of data.

In summary, DEFS is not merely a search facility—rather, it is an evolving environment that fills the chasm between the existence of information and how accessible it is. It facilitates collaboration, streamlines the process of locating the proper expertise, and enables informed decision-making.

As the need for domain-specific information continues to increase, software like DEFS will become an essential part of academic research, institutional growth, and industrial talent sourcing. Future enhancements such as AI-based recommendations, multilingual scraping, and real-time collaboration features can further elevate its potential and societal impact.

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 6, June 2025

DOI: 10.17148/IJARCCE.2025.14632

VII. ACKNOWLEDGEMENT

We would like to thank **Ms. Maddela Bhargavi** from the bottom of our hearts for the valuable and positive feedback provided under the project planning and development. We are extremely grateful for her donation of noble time. Additionally, we would like to thank the principal, professors of KSIT for their constant support and motivation.

REFERENCES

[1] Zhang, H., Li, X., & Wang, Y. (2022). Expert Finding in Heterogeneous Bibliographic Networks. *IEEE Transactions on Knowledge and Data Engineering*, 34(5), 2223–2236. https://doi.org/10.1109/TKDE.2021.3053678

[2] Patel, R., Sharma, S., & Gupta, A. (2023). Framework for Academic Expert Identification Using Web Mining. *Journal of Web Engineering*, 22(1), 45–65. https://doi.org/10.1007/s42979-023-01234-5

[3] Kumar, P., Singh, R., & Verma, D. (2021). Automatic Extraction of Faculty Information Using Crawlers. *Procedia Computer Science*, 190, 110–117. https://doi.org/10.1016/j.procs.2021.04.016

[4] Li, J., Chen, Z., & Zhao, M. (2020). AI-Driven Expert Recommendation Systems in Knowledge Networks: A Survey. *ACM Computing Surveys*, 53(4), 1–37. https://doi.org/10.1145/3397199

[5] Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., & Su, Z. (2008). ArnetMiner: Extraction and Mining of Academic Social Networks. *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD '08), 990–998. https://doi.org/10.1145/1401890.1402008

 This work is licensed under a Creative Commons Attribution 4.0 International License