Building Lightweight Semantic Search Engines

Michael Rovatsos
School of Informatics
The University of Edinburgh
Edinburgh, United Kingdom
Michael.Rovatsos@ed.ac.uk

Rosa Filgueira
School of Computer Science
University of St Andrews
St Andrews, United Kingdom
rf208@st-andrews.ac.uk

***Abstract*—Despite significant advances in methods for processing large volumes of structured and unstructured data, surprisingly little attention has been devoted to developing practical methodologies that leverage state-of-the-art technologies to build domain-specific semantic search engines tailored to use cases where they could provide substantial benefits.**

This paper presents a methodology for developing these kinds of systems in a lightweight, modular, and flexible way with a particular focus on providing powerful search tools in domains where non-expert users encounter challenges in exploring the data repository at hand.

Using an academic expertise finder tool as a case study, we demonstrate how this methodology allows us to leverage powerful off-the-shelf technology to enable the rapid, low-cost development of semantic search engines, while also affording developers with the necessary flexibility to embed user-centric design in their development in order to maximise uptake and application value.

***Index Terms*—Semantic search, natural language technologies, knowledge graphs, neural information retrieval**

I. Introduction

Advances in semantic technologies [1], [2], natural language processing (NLP) [3], and machine learning (ML) [4] have given rise to widely available tools that can be used to create software applications that can help individuals and organisations navigate large volumes of both structured and unstructured information. However, surprisingly little prior work focuses on developing practical methodologies to guide the lightweight, flexible integration of off-the-shelf technologies when developing custom search engines for specific use cases from scratch; and which afford developers with the freedom to choose and adapt the algorithms used to train search models, their preferred approach to data and privacy management, and the user interaction and collaboration modalities they want to embed in the resulting application.

In this paper, we present a methodology for building search engines in domains where users need to find information in large volumes of typically *unstructured* datasets (text, images, audio, or video). In these domains, the raw data itself cannot be navigated purposefully (users do not "speak the language" of the raw data), but *structured* metadata is available that lends itself to human interpretation and navigation, at least at a level that allows for implementing key functionality that matters to the user. To link unstructured to structured data, such search engines need to provide a *semantic* connection between raw data, metadata, and user queries, which can nowadays be achieved by extracting semantic information from unstructured data using state-of-the-art NLP and ML techniques [5].

Crucially, we cannot expect that source datasets have always been enriched with expressive metadata, so we have to carefully think about how users can interact with the data. Even in cases where a search may return little more than a link to the original data item (e.g. a document, image, audio or video file) that has to be explored manually, search engines of this kind must be able to narrow down large search spaces using only relatively *uninformed* queries (in terms of how closely they describe the kind of content the user is looking for) to provide tangible value to end users.

Use cases where such technology can provide significant application value abound, and include: (a) Tools that enable non-expert users to identify experts in a given profession, discipline, or area of business, e.g. for purposes of recruitment, to source expertise for collaborations or suppliers of commercial services; (b) enterprise information systems that pull together documents from a range of corporate databases in ways that can be searched by all employees to retrieve relevant business information, or find the right colleague to speak to about a certain matter; and (c) systems where the target data cannot be queried directly using textual queries, e.g. when a user wants to describe what kind of musical piece they are looking for in a database of audio files [6], a clinician tries to retrieve medical images of scans that have similar characteristics to the case they are trying to evaluate [7], or when a video content editor is sourcing sequences of archive footage to put together a feature on a specific topic [8].

In many of these scenarios, information is effectively only available in one or more "language(s)" the user is not conversant in, or may not have been made available for the purpose the user ultimately wants to use it for.

Our work aims to support the development of this class of applications by providing a meta-model of their general structure that underpins a simple, yet flexible design methodology that allows developers to build them rapidly using off-the-shelf components. We demonstrate the benefits of this methodology with the case study of a system designed to help lay users find academic experts that has been implemented and deployed for use at a large research university.

The remainder of this paper is structured as follows: Section II outlines key desiderata for the class of systems we are interested in, both regarding their functionality and in terms of their development and deployment. We present out methodology, which has been designed to satisfy these requirements and is driven by a high-level meta-model of their structure in Section III. Section IV provides a detailed account of a prototypical system developed using this methodology. Related work is covered in Section V, and Section VI concludes with a summary of our main contributions and a discussion of future avenues for further research.

II. Desiderata

The common foundation of most, if not all, search-based applications is that they aim to assist users in navigating large amounts of data, including in situations where users might not know exactly what information they are looking for, or whether it even exists. Unlike general-purpose web search engines (e.g. Google Search, Bing, Baidu), many more specific use cases focus on searching specific datasets, for example document, image, audio or video repositories, product or service catalogues, or people profiles.

Several commercial products such as Apache Solr[1], ElasticSearch[2], or Algolia[3] provide semantic search technology that can be readily used by developers, while other vendors such as Starmind[4] offer complete entreprise knowledgement solutions that use similar kinds of technology to enable smart search and knowledge sharing within client organisations.

Also, with recent immense advances in large language models (LLMs) [9] and generative AI, many new ways of using systems like ChatGPT [10] and search technology built on top of it (e.g. Moveworks[5] or Microsoft Prometheus[6]) provide conversational interfaces that enable users to explore information in more intuitive ways and offer AI-assisted functionality that offers huge benefits in terms of productivity.

As these technologies have advanced, we observe that many of their underpinning components are now available in free and often open-source stacks (e.g. LangChain[7]), which suggests that similar applications tailored to specific use cases could be developed from scratch with little effort in more cost-effective ways, avoiding vendor lock-in, and giving developers and users full control over how data and algorithms are used in the resulting software application.

This would not only support the democratisation of semantic search technology, but also allow for a more *user-centric* development of practical solutions through rapid prototyping and testing with target users, even in cases where commercial solutions might be ultimately adopted.

Realising this vision requires focusing on a number of requirements that underpin our methodology:

[1]solr.apache.org
[2]elastic.co
[3]algolia.com
[4]www.starmind.ai
[5]www.moveworks.com
[6]gpt3demo.com/apps/microsoft-prometheus
[7]python.langchain.com/

a) Focus on a minimal, lightweight architecture: We seek to identify only the components that provide the functionality that is strictly needed to deliver what users need, and to integrate these in in a "bare-bones" architecture that provides a simple, generic, and reusable design pattern. While feature-rich applications and platforms may well be built on top of our core framework, they are beyond the scope of this work.

b) Modular, flexible design: We aim to use readily available, off-the-shelf software components and to integrate them in a modular way so they can be individually configured, modified or swapped out for alternatives, and easily migrated to different computing platforms for deployment and scale-up. The objective of our framework is not to advance the performance of any single component or its underpinning algorithms, but to be able to experiment with existing technologies for the purpose of rapid prototyping and testing.

c) "Out of the box" search engine functionality: Our target developer and user audiences will often work in domains or organisations with relatively small numbers of users. Therefore, our systems need to avoid the "cold start" problems of recommender systems that rely on extensive user engagement.

This implies that we are limited to using algorithms that do not rely on user feedback or profiling to inform the search models used by the system. This does not mean we discourage developers from adding features that take user input into account; rather, we want to be able to be able to demonstrate the core functionality of the system as soon as it is deployed in order to be able to iterate its design with users.

d) Full control over data and algorithms: Many commercially available tools supply algorithmic components that have been (understandably) packaged up in ways that do not allow developers to modify and adapt those freely. Some of them may also have been pre-trained on datasets in ways that cannot be controlled (or verified) *ex post*, as is the case, for example, in some LLM-based technologies (users may also not know whether the data they supply to these tools is used for further training).

By contrast, our approach is aimed at domains where the data supplied to the search engine may be sensitive in terms of privacy or commercial confidentiality and needs to remain safely under the control of the locally deployed system. Additionally, we need to ensure search results are confined to controlled data sets and can be linked to specific source data items in terms of provenance.

e) User-centric design: The design of the application should be driven by user requirements. This means that, when selecting individual components and designing specific data and algorithmic processing pipelines, we must take into account what capabilities users have in terms of navigating information, how they might articulate their queries, and what the most common user workflows look like.

To design for a diverse range of potential types of users, it is also important to separate the user interaction layer clearly from data and algorithmic processing layers, so that different interfaces could be provided while reusing algorithmic compo-

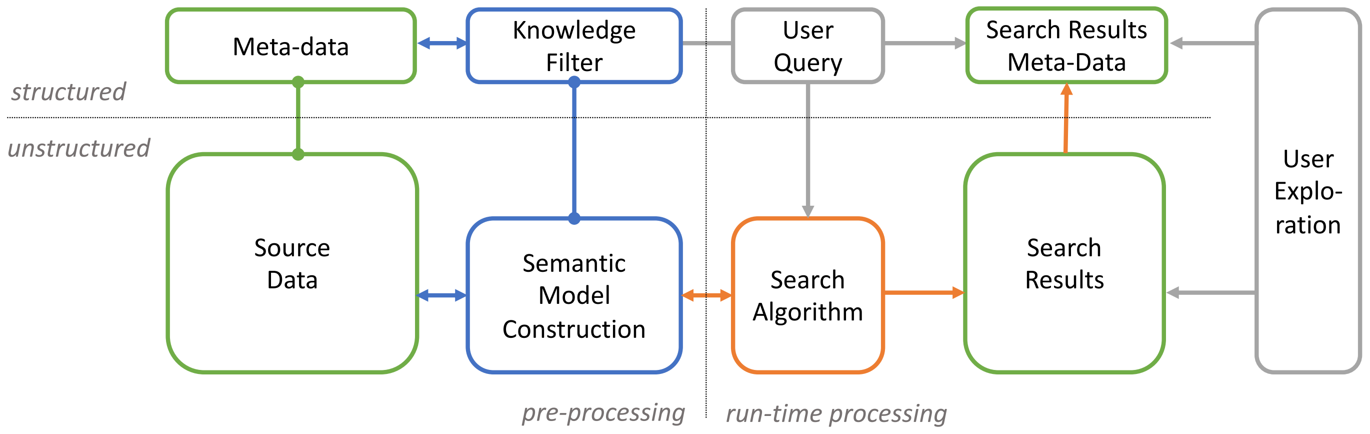


Fig. 1. Semantic search engine structure and information flows between data components (green), user activities (grey), algorithmic components (blue/orange)

nents across different use cases, or, e.g., making them available as SaaS components to other developers.

III. Methodology

Guided by these desiderata, we develop our methodology by decomposing semantic search engine design into a number of key steps as per the high-level meta-model shown in Fig. 1:

1) The *source data* that is to be searched and linked to *metadata* that contains descriptions of source data items. As described above, we are mainly interested in domains where the source data is *unstructured* and can either not be searched effectively using descriptions provided by users in their queries (as, e.g., in the case of non-textual data); or, it may be accessible in principle but not easy to describe for the target user audience (e.g. because it uses specialised technical language). We do not make any assumptions regarding the nature of the available metadate. In many cases, it will merely contain a resolvable identifier to access the original data object in a database or a URL. Ideally, however, it would contain key information relevant to the use case (e.g. the names of document authors if users typically look for relevant authors).

2) *Knowledge filters* that structure metadata in ways that allow users to constrain their queries at run-time. Such filters may operate directly on the original metadata (e.g. specific data facets that the search can be narrowed down to) or on entities derived using information extraction algorithms (e.g. attributes extracted from documents or images using information retrieval or image recognition algorithms). Depending on the richness of the metadata structure, knowledge filters might offer anything from simple attribute-value constraints to a fully-fledged knowledge graph query language [1], [2].

3) The core *semantic model construction* algorithm that that enables relevance-based search by providing a facility to compare the user's search query against source data items. The model construction step will typically involve using an intermediate representation (bag of words, vector embedding etc) that both source data and user query can be translated to, and which the unstructured source data has been augmented with in a pre-processing step. Model construction will often require extensive training of a model on the source data, with inference against the search query conducted at run-time. It is essential that the representations used allow for returning search results within an acceptable amount of time, as this will directly affect search execution times experienced by users.

4) The *search algorithm* itself maps the *user query* to the intermediate representation used by the semantic model and matches it against the source data items, taking any knowledge filtering choices into account the user has made when running their search. Typically, this will involve an algorithmic process of exploring the (semantic model part of the) source database to identify most relevant "matches" to the query, and ranking them for presentation to the user in the *search results*.

The key challenge we are faced with in the design of this component is ensuring that the search algorithm can scale to the dimensions of the dataset at hand. This may require, for example, generating search results dynamically in order of decreasing relevance to reduce query execution times while allowing users to explore further results incrementally.

5) *User exploration*, finally, refers to the modalities users are afforded to search any results returned by the search engine. These will at least include providing interfaces that allow for navigating and processing the metadata corresponding to search results (including, frequently, the application of additional filters on the results) and/or the actual unstructured data items (e.g. reading a document, playing a media file).

In more fully developed search engine applications, many other features can be added that enable, e.g., archival of previous searches, the elicitation of user feedback on search results to improve the semantic models, or further processing steps related to using and managing source data items (e.g. data download and export functionalities).

This meta-model highlights key design decisions that need to be taken in the development of semantic search engines of this kind such as: What data sources are to be used, what metadata is available for (or could be reliably extracted from) them? What semantic representation can be used for searching unstructured data, and what off-the-shelf tools are available to develop the algorithmic components that will (a) use it to augment the unstructured data and (b) provide efficient search functionality that exploits this representation?

While answers to these questions will allow developers to assess whether implementation of the system is feasible (with the resources available), the starting point of the design process should always be a clear understanding of the functionality that the search engine can provide for target users, and which depends on their specific needs and capabilities.

This is because such systems aim to close a "gap" between very large volumes of data that is typically hard to navigate for users due to the unmanageable human effort that would required, or because their skills and knowledge require them to access information in ways that require "translating" the source data to user-appropriate content.

In the following, we describe how we have approached this in the development of a prototypical system that was designed using this methodology.

IV. Case Study: OpportunityMatch

OpportunityMatch (OM) is an academic expertise finder tool which we have developed for the University of Edinburgh, a large research university with over 11,000 research-active staff and PhD students. Despite the fact that the University provides a publicly available research database[8] that contains (near-)up-to-date information on all of its research outputs (publications, projects, research datasets, etc), it is hard to identify the right experts for specific opportunities for several reasons:

Firstly, such opportunities are often expressed using very different language from that used in scholarly repositories. Examples for this include calls issued by funding agencies, enquiries from non-academic organisations interested in collaborating with researchers, or from media outlets trying to identify experts who can provide commentary on a news topic. Non-expert users may struggle to express what they are looking for in the language used in highly specialised academic outputs, and it is also hard for them to describe exactly what experise they are looking for (as an example, there are probably over 1,000 people at the University who use AI, but only a handful are experts in self-driving cars).

Secondly, official research databases only contain information on past or ongoing research that has already attracted funding or produced publications. They do not capture any new initiatives, directions, or areas of interest that scientists may currently be working on or thinking about. Even though such databases offer facilities for researchers to update their profiles, there is little incentive to update information manually, especially if researchers do not know who might be interested in it. To our knowledge, no widely used corporate tools enablthose *searching* for academic expertise to access this "hidden" information, or for those *providing* expertise to monitor what others might be looking for systematically.

To address these issues, the development of OM sought to capitalise on advances in recent NLP technologies, which no longer restrict document search to simple keyword matching, but instead allow for entire *documents* to be compared in terms of semantic similarity. This can be achieved by using *document embeddings* [11], i.e. high-dimensional numerical vector representations of longer pieces of text that can be derived using deep neural network models trained on a specific corpus of documents to encode its latent semantic space.

[8] www.research.ed.ac.uk

Similarity search on documents enables users to specify their query using longer pieces of text, for example key paragraphs of a funding call, e-mail enquiry, or newspaper article. They can also write their own query text, circumscribing the opportunity in different ways, or using words and phrases they have encountered in previous search results.

While traditional "hard" keyword search can be used to specify terms that *must* appear in results, supplying longer pieces for "soft" open-ended similarity matching is much more likely to yield an approximate description of the content the user is looking for.

The other issue OM aims to address is that traditional scholarly repositories do not support user-side content curation and collaboration between users. Based on input from target user communities (researchers, students, support officers, tech transfer and commercialisation staff), we identified the following functionalities as key to filling this gap:

1) The ability for researchers to modify and adapt their own profiles, i.e. the items associated with them on the research database;
2) the ability to maintain an up-to-date portfolio of experts in specific thematic areas without having to repeat searches on an ongoing basis;
3) facilities to visually explore researcher outputs and, in particular, researcher networks that are not obvious from tabular results presented as a list of items;
4) receiving updates on who is searching for what kind of information (while being able to keep search activity private where appropriate); and
5) being able to limit searches to specific parts of the organisation (e.g. to understand how active a department is in a certain area) or outputs (e.g. considering only funded projects to track investment in certain fields).

Several of these features cut across concerns normally associated with front-end and back-end functionality, and therefore necessitate careful consideration of how data, semantic models, and user input are managed in the system.

A. Features

The design of OM is guided by the considerations above, which can be summarised as two core requirements: (a) Supporting long-form text queries that return results based on semantic similarity matching and (b) treating the database of expertise as a "live" resource that can be augmented by users and drives collaboration between them.

To satisfy (a), we train a model that uses high-dimensional document embeddings when running search queries in a pre-processing step performed in the background periodically (typically overnight). At query time, vectors representing all items in the document database (including additional items created by researchers, which we call *research interests*) are ranked by similarity against the vector the search query is mapped to using the same model, with the top n results returned depending on a configuration parameter setting.

Additionally, the system provides simple keyword search, which is performed using the standard TF-IDF method [12]. Users can choose to perform similarity search, keyword search, or both. When used in conjunction, keyword matching will act as a hard filter on results returned by similarity matching.

In terms of source data, we build on an internal project that already pulls together various corporate and external data sources in a Neo4j[9] graph database called ROAG. Data ingestion involves querying ROAG to create a local copy of all metadata (document authors, research ouput types, organisational units, narrative summaries), which contains links to online resources such as publication URLs, official records of project grants, or research dataset repositories.

The summaries used for training our semantic model typically consist of relatively short abstracts, though (subject to licensing clearance) we could process full-text documents to create a more expressive and detailed semantic model. Note that, while that might substantially increase ingestion and training times, it would make no difference to the responsiveness of the search engine at run-time as the time it takes to compute a full relevance ranking only depends on the number of (constant-size) vector comparisons.

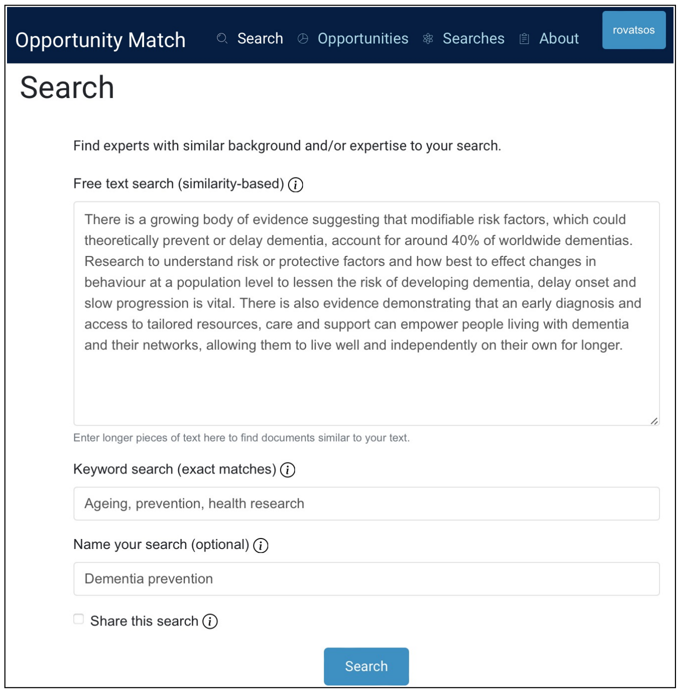
Fig. 2 shows the OM search page, which provides options to use similarity- and/or keyword-based search, to save the current search under a name specified by the user, and to opt into sharing the search with other users of the system. These features address user demands for retaining and adapting their searches over time, being notified about changes to the results of previous searches, and to receive updates on others' searches whose results included their own research outputs (if those searches were shared). Users can set daily, weekly, or monthly automated e-mail alerts for this purpose.

After a search is performed, OM displays results in the format shown in Fig. 3. To enable visual result exploration, a clickable graph is shown at the top of the page, which provides a specific view of the ROAG sub-graph corresponding to the results returned for the search. This graph and the filters users can apply to results together make up the *knowledge filters* of the application, and provide the *user exploration* functionality of OM that combines different modalities for exploring results.

Apart from enabling access to all source documents and web resources hyperlinks for all metadata items (researchers, outputs, interests), OM encourages engagement with results by providing the option to upvote/downvote results, so that crowdsourced feedback about the quality of specific matches can be used to fine-tune the semantic model.

The *researcher profile* view (see Fig. 4), which is available to all system users whose credentials are linked to a unique researcher ID in the database, is similar to a results page in

[9]neo4j.com



Search

Find experts with similar background and/or expertise to your search.

Free text search (similarity-based)

There is a growing body of evidence suggesting that modifiable risk factors, which could theoretically prevent or delay dementia, account for around 40% of worldwide dementias. Research to understand risk or protective factors and how best to effect changes in behaviour at a population level to lessen the risk of developing dementia, delay onset and slow progression is vital. There is also evidence demonstrating that an early diagnosis and access to tailored resources, care and support can empower people living with dementia and their networks, allowing them to live well and independently on their own for longer.

Enter longer pieces of text here to find documents similar to your text.

Keyword search (exact matches)

Ageing, prevention, health research

Name your search (optional)

Dementia prevention

Share this search

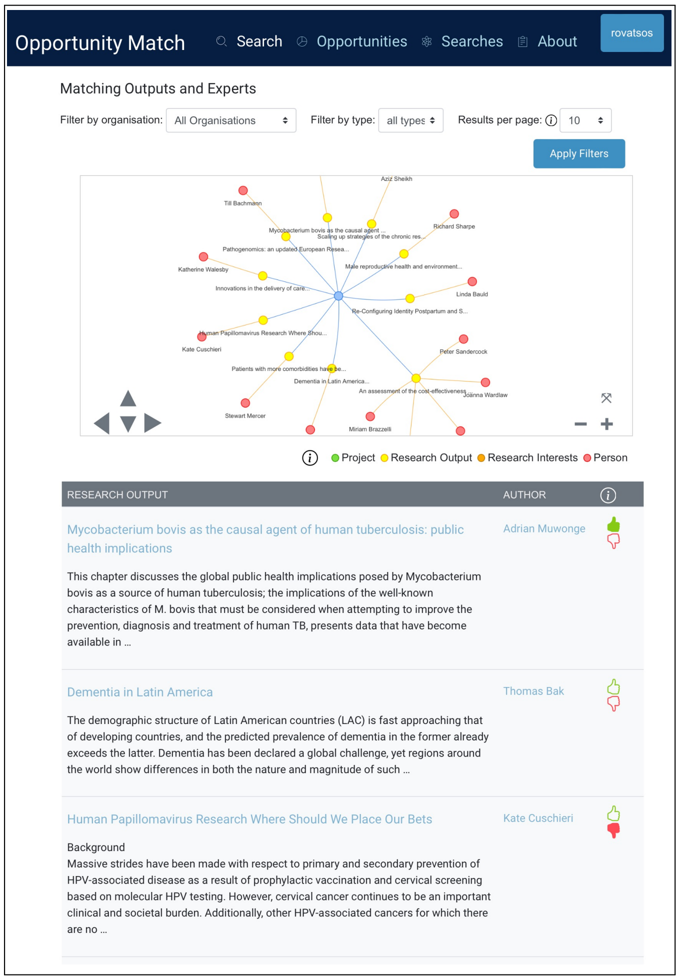
Fig. 2. Search page: The free-text field can be used for longer descriptions of the expertise the user is looking for (here, text from a funding call) in alongside (optional) keyword search. Searches can be stored under a user-defined name and become available under the *Searches* tab so they can be re-run, modified, or deleted. The *Opportunities* tab lists others' searches that returned the present user in their results (if the user who ran the search ticked the *"Share this search"* box when performing it).

structure. It displays a list of all the researcher's outputs and knowledge graph that corresponds to these, but provides a number of additional features: A (clickable) list of their official institutional affiliations, a wordcloud of keywords extracted from their outputs, and a facility for them to extend their profile with additional "research interests" items. Affiliations and wordcloud effectively act as additional semi-structured representations of key information that allows users to explore the researcher's academic expertise and orgnisational role.

Research interests, on the other hand, allow researchers to add arbitrary new items of text to describe current or future interests and ongoing activities, especially where these are not reflected by outputs recorded in the official research database.

These new items are treated just like any other document in the system and added to the training set of the semantic model before it is (periodically) re-trained. Note that, while only the researcher has access to their profile page from the navigation bar and can edit information on it, all users can view a static researcher profile page if they click on the person's name anywhere in the system (this page only contains content tagged as "shared" by the researcher).

Finally, OM provides researchers with an *Opportunities* tab, which shows users (shared) content items associated with them (searches, profile items) that have appeared in others' searches. Based on privacy concerns expressed by our test users, we opted not to include references to the actual search on this


Opportunity Match
Search
Opportunities
Searches
About
rovatsos
Aziz Sheikh
Till Bachmann
Richard Sharpe
Mycobacterium bovis as the causal agent ...
Scaling up strategies of the chronic res...
Pathogenomics: an updated European Resea...
Katherine Walesby
Male reproductive health and environment...
Innovations in the delivery of care...
Linda Bauld
Re-Configuring Identity Postpartum and S...
Human Papillomavirus Research Where Shou...
Kate Cuschieri
Peter Sandercock
Patients with more comorbidities have be...
Dementia in Latin America...
An assessment of the cost-effectiveness
Joanna Wardlaw
Stewart Mercer
Miriam Brazzelli

Matching Outputs and Experts

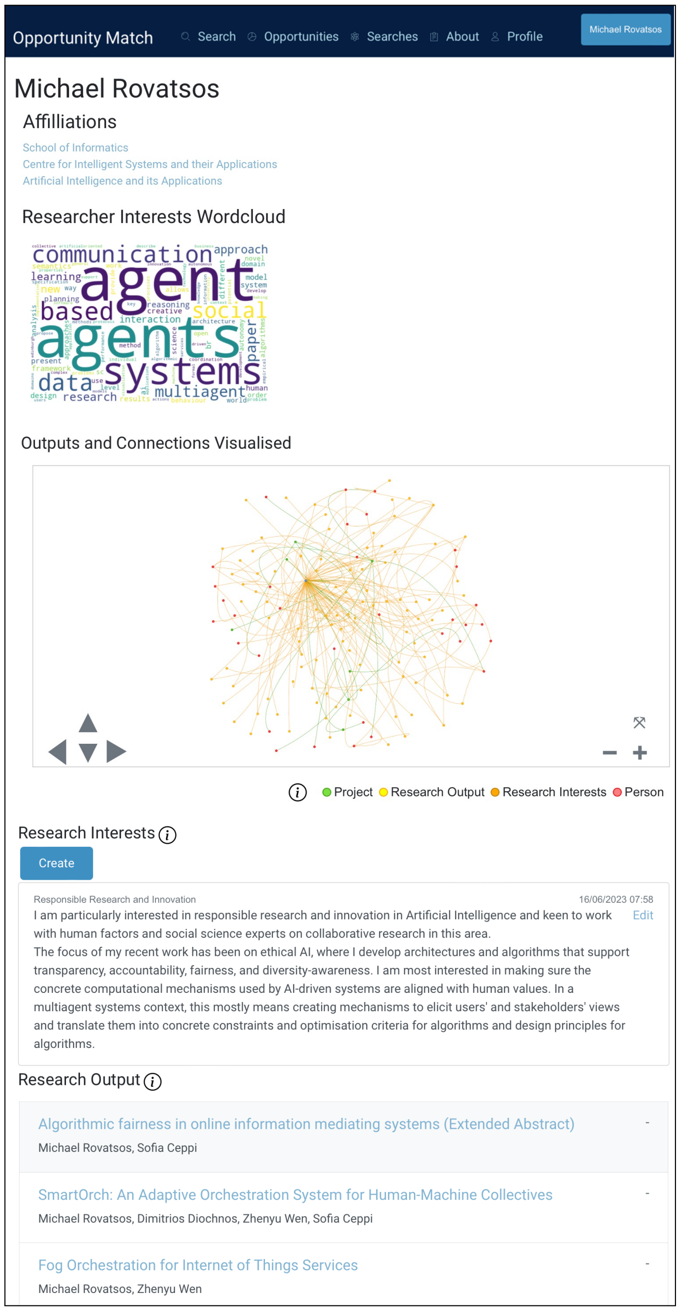
Filter by organisation: All Organisations Filter by type: all types Results per page: 10

Apply Filters

● Project ● Research Output ● Research Interests ● Person

RESEARCH OUTPUT	AUTHOR
Mycobacterium bovis as the causal agent of human tuberculosis: public health implications This chapter discusses the global public health implications posed by Mycobacterium bovis as a source of human tuberculosis; the implications of the well-known characteristics of M. bovis that must be considered when attempting to improve the prevention, diagnosis and treatment of human TB, presents data that have become available in ...	Adrian Muwonge
Dementia in Latin America The demographic structure of Latin American countries (LAC) is fast approaching that of developing countries, and the predicted prevalence of dementia in the former already exceeds the latter. Dementia has been declared a global challenge, yet regions around the world show differences in both the nature and magnitude of such ...	Thomas Bak
Human Papillomavirus Research Where Should We Place Our Bets Background Massive strides have been made with respect to primary and secondary prevention of HPV-associated disease as a result of prophylactic vaccination and cervical screening based on molecular HPV testing. However, cervical cancer continues to be an important clinical and societal burden. Additionally, other HPV-associated cancers for which there are no ...	Kate Cuschieri

Fig. 3. Search results: The knowledge graph corresponding to the search (=blue node) results is shown at the top (with people, projects, outputs, and research interests shown as nodes in different colours that are clickable and link directly to OM and other web resources. The same items are listed below the graph with summaries. Upvote/downvote buttons are provided against each result, which can be used to indicate whether a search result is a good match to the query or not. Results can be filtered by organisation (department) and type (project/output) using dropdown menus.



Michael Rovatsos

Affiliations

School of Informatics
Centre for Intelligent Systems and their Applications
Artificial Intelligence and its Applications

Researcher Interests Wordcloud

Outputs and Connections Visualised

● Project ● Research Output ● Research Interests ● Person

Research Interests

Responsible Research and Innovation 16/06/2023 07:58

I am particularly interested in responsible research and innovation in Artificial Intelligence and keen to work with human factors and social science experts on collaborative research in this area.
The focus of my recent work has been on ethical AI, where I develop architectures and algorithms that support transparency, accountability, fairness, and diversity-awareness. I am most interested in making sure the concrete computational mechanisms used by AI-driven systems are aligned with human values. In a multiagent systems context, this mostly means creating mechanisms to elicit users' and stakeholders' views and translate them into concrete constraints and optimisation criteria for algorithms and design principles for algorithms.

Research Output

Algorithmic fairness in online information mediating systems (Extended Abstract)
Michael Rovatsos, Sofia Ceppi

SmartOrch: An Adaptive Orchestration System for Human-Machine Collectives
Michael Rovatsos, Dimitrios Diochnos, Zhenyu Wen, Sofia Ceppi

Fog Orchestration for Internet of Things Services
Michael Rovatsos, Zhenyu Wen

Fig. 4. Researcher profile page: The *Profile* tab shows users registered as researchers their affiliations, a wordcloud of interests extracted from their research outputs, and a graph and list of items that represent all outputs in their public profiles. An additional *Research Interests* section allows them to add further items on additional specific interests they may have.

page, so it currently only provides information of how often one's items have matched searches. However, it would be easy to provide this functionality through links to resources previously created by the software (all pages created by the system are stored automatically using unique REST URIs).

Overall, by following the methodology outlined in Section III, the design of *OpportunityMatch* satisfies the requirements outlined Section II: By leveraging publicly available data and a combination of semantic similarity matching and knowledge graph exploration, it avoids the cold start problem and can provide high-quality search results "out of the box" before any users start using the system.

While all University researchers are "pre-registered" and would be able to access the additional features available to them as soon as they register, the system is open to non-researchers and people outside the University, and it is deliberately designed to rely on users communicating directly outside the application, as we do not wish to add yet another tool users have to use to be able to connect with each other.

This avoids the problems caused by other similar knowledge portals (e.g. Academia.edu, ResearchGate) that create a semblance of researchers having engaged with the tool and "hassle" them to contribute. Apart from never "cold contacting" researchers, OM takes a conservative approach to respecting users' privacy throughout its structure, e.g. by providing "opt-in" functionality to sharing any search activity or user-created content.

It is worth highlighting that the ranking of results is purely based on similarity and keyword matching, and does not take any citation metrics, journal rankings, or cumulative researcher visibility into account. This was a key requirement highlighted in focus group workshops we conducted, as it was seen to disadvantage less established researchers on common scholarly repositories available on the Web.

Finally, to strike a balance between using publicly available data to pre-seed the system that is necessary for its functionality while also allowing researchers to opt out of the system, we provide users with an option to have all their data removed from the system. The internal architecture of the system, further outlined below, utilises only freely available software (gensim[10], MariaDB[11], ElasticSearch[12]), nginx[13], Django[14], Gunicorn[15]) and does not require the implementation of novel algorithms or backend tools.

This enables developers and organisations deploying the system to control, modify, and swap any internal components in a lightweight, modular way. For example, they can choose what parts of a research database should be used for training and during search, or how often models are updated in accordance with the compute resources available to them.

Importantly, this also makes it easy to augment the system with additional functionality. As an example, [13] explored how fairness and diversity considerations can be addressed when deploying such AI-based tools given that the deep neural networks they use are intrinsically opaque. Using information extraction techniques, this project predicted author gender, ethnicity, and seniority to test whether search results would be unduly biased against underrepresented groups, and found this to be the case in some situations.

Based on such insights, it would be straightforward to adjust the relevance ranking function to give those groups more exposure or to provide users with configuration options that control such features. It is worth noting that we are not aware of any open or commercial people search tools that provide this level of flexibility, or, in fact, provide any fairness or bias metrics for their models.

B. Implementation

The overall architecture of the implemented and deployed OpportunityMatch system is shown in Figure 5.

In terms of data storage, we use ElasticSearch for the *Expertise Database*, which stores research project/output information imported via Neo4j queries from the aforementioned institutional ROAG graph database provided by the university, and also any user-generated content added to researcher profiles (where only content marked as shared will be used for semantic model construction).

[10]pypi.org/project/gensim
[11]mariadb.org
[12]www.elastic.co
[13]nginx.org
[14]www.djangoproject.com
[15]gunicorn.org

The choice of ElasticSearch is motivated by its built-in "More Like This" query feature that implements a variant of TF-IDF [12], and which is used for keyword-based search in the system. Apart from the user requirement to provide keyword-based search in addition to long-form queries, extensive testing revealed that TF-IDF performed better for keyword-based searches than the document embedding models used for similarity search.

Since this functionality is not needed for managing data that relates to run-time user activity (storing searches, result ratings produced by upvoting/downvoting individual results, and general REST API resources produced by the web application), we use MariaDB for a separate *Opportunity Database*. In terms of search and semantic model training, this database is treated exactly like the Expertise Database, with previous searches being stored as text documents used to train a different vector embedding model (where these were shared by the user when they ran their query).

In terms of algorithms, semantic model construction relies on using the *doc2vec* algorithm [11], and OM uses the its implementation from the gensim Python library, employing the distributed memory algorithm (PV-DM) for training.

PV-DM was chosen for its ability to capture semantic meaning and contextual relationships in variable-length text, which makes it suitable for our task.

To construct models that capture the data contained in either of our Expertise or Opportunity Databases, we retrieve the core text components from them and perform additional pre-processing steps, which include text conversion, stop word removal, and character cleaning before constructing vocabularies for the two models and training them for 100 epochs using the pre-processed documents.

Training requires making several hyper-parameter choices, such as selecting the right dimensionality for the vector-space embeddings that will be constructed (set to 300, in our case) and the window size, which determines the amount of contextual information captured during training (set to 5 in our case). A learning rate of 0.025, gradually decreased after each epoch, is applied to enable controlled gradient updates and convergence.

These hyper-parameters were chosen based on empirical evidence from our own testing and common best practice, and they proved appropriate to achieve effective representation learning and information retrieval capabilities of the models.

When a OM user performs a search query, the system automatically infers a new vector to represent the query text, utilising the trained document embedding models described above. An important detail to highlight is that, even if the long-form text query field is used, the system will use TF-IDF rather than the document embedding models if the query is less than 30 words. This ensures the better of the two techniques is used without requiring the user to be aware of their differences. For longer queries, the system employ the doc2vec models

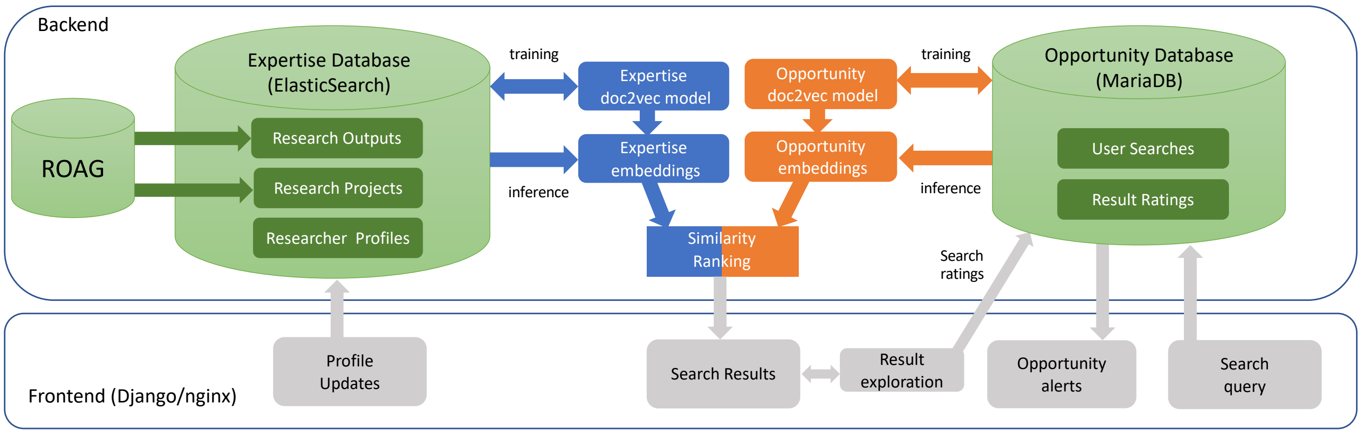


Fig. 5. Architecture overview of our OpportunityMatch implementation, shown using the colour codings of our meta-model for data, algorithm, and user interaction components (see also Fig. 1): Via the Web-based frontend, researchers can directly update their profiles in the Expertise Database; a separate Opportunity Database records previous searches, users' ratings (upvotes/downvotes of search results), and generates opportunity alerts when new matches are found between research outputs and previous searches. The core similarity ranking algorithm is applied to both expertise and opportunity items using models trained on each of the two databases. For readability, we have not included simple service and profile management functions such as user settings management, registration and authentication or admin backend management service functions.

to generate a vector representation that captures the semantic meaning of the input.

Once we have inferred the vector representation of the user input, the system compares it to the vectors of *all* existing documents in the database using cosine similarity. Based on these similarity scores, the system ranks the results in descending order, where documents with higher similarity scores are considered more similar to the user's input. This ranking allows us to retrieve the most similar information from the database, and we limit results to at most 100 items to avoid returning irrelevant items. We then use this list to create the knowledge graph and result lists shown in Figure 3 (with any additional filters applied by the user viewing the results after results are computed).

While this exhaustive similarity-based relevance ranking is performed at query time on the Expertise Database and its corresponding doc2vec embedding model, it is triggered periodically by the backend on the Opportunity Database on the basis of users' alert settings, and will become available under the *Opportunities* tab in the user interface (see Fig. 2).

For this purpose, previous searches are periodically (daily, weekly, or monthly) re-run based on the user's notifications setting to update either (1) researchers of search results in which one of their outputs or research interests appeared or (2) any users who previously ran a search that it matches another users' search. As the system currently generates at most 100 search results for any query, summary notifications are generated that list all matches, but this could be easily be replaced by a more focused approach that only creates notifications for matches over a certain threshold value.

In terms of the design of the frontend and overall OM Web application, we used a standard Django/Gunicorn/nginx stack that allows for secure, fast, and agile RESTful implementation, and also comes with extensive browser-based administration functionality. This was important

C. Development and Deployment

Development of the first prototype required only two person-months of effort, after which the system was released for internal university use. The second release, which allows the system to be opened up to external users and makes it easy to deploy installations for other organisations, required an additional three person-months of work. Installations for other organisations can be deployed by simply providing a JSON file that contains a list (links to) documents tagged with unique IDs for their author(s) alongside a CSV with the names and e-mail addresses of authors associated with these IDs. Using the CSV file, the OM administrator can pre-register these "experts" in bulk on the system via its web interface, so that they can be associated with and curate their (automatically generated) profile once they register using their official e-mail address.

OpportunityMatch ships as a fully dockerized application that can be installed within half an hour by following a set of simple steps within a few hours including model training time. If pre-trained models are used (e.g. after reboot or updates that do not involve changes to the expertise and opportunity databases), deployment time is reduced to around 30 minutes.

In terms of performance, on a 96GB RAM 8-core CentOS Linux VM responses to typical search queries over 22,000 documents return results within around 1s, but the system can also be installed on standard personal computing equipment for local use (training the vector embedding model from scratch takes on the Edinburgh database around four hours on a standard Apple 4-core 16GB laptop). This could also open up interesting additional use cases in terms of using OM as a personal semantic search and document management tool.

Following the system's internal release, we have received resoundingly positive feedback from users, who find the system to be much more powerful in terms of its capability to return relevant results when querying it with longer pieces of text (30 words or more). OM has also already attracted commercial interest from private and governmental organisations.

V. Related work

The history of interactive search is as old as the Web itself [14], and is closely intertwined with research into information retrieval and NLP research for extracting structured information from unstructured data [15]. Within this wider field, semantic search technology research [1] has developed many methods that exploit structured data and knowledge [2]. Meanwhile, neural information retrieval methods [5] have received a great deal of attention over the last ten years with the emergence of neural language models since the development of BERT [16], which led to massive-scale modern-day models like ChatGPT [10].

As highlighted in [17], we are now seeing a confluence between these two approaches, methods from which are combined in so-called "neuro-symbolic approaches". Our approach fits squarely into this category, as it covers keyword-based and natural language-based semantic services in the typology of [14]. Surprisingly, as far as we can see from recent surveys, no approaches seem to exist that exploit longer free text queries for information retrieval purposes. At the level of our application domain, the same seems to be true of scholarly recommendation systems [18] (none of the over 200 papers analysed by this survey seems to suggest that similar approaches exist).

In terms of overall approach, the most similar project to ours is Open Semantic Search[16], which provides a wide range of open source tools for building semantic search engines including data crawling, indexing, enrichment, mining, analytics and visualisation. However, while the project does include a number of NLP tools for disambiguation, entity recognition, and classification, it focuses on keyword (rather than long-form) search, includes no neural information retrieval components, and does not come with a specific design methodology or development guidance. The latter gap is also visible in many other services and platforms that can be used, *inter alia*, for the development of semantic search engines[17].

A similar approach is taken by a paper that presents a semantic search engine called GAIA [19]. The authors focus more on integrating common NLP platforms with information retrieval data sets to support the training of new NLP methods using web corpora. The GAIA search engine is presented as an example for how the combination of both areas allows for building new search engines with ease, but does not constitute a worked example of how to systematically design such systems with a specific use case in mind.

WebGPT [20] is a search engine that enables conversational (question answering with full paragraph answers) Web search by exploiting the full capabilities of ChatGPT. The system has been fine-tuned using human feedback on responses, and (unlike ChatGPT) provides links to source web documents that can be inspected by users to verify responses. The fundamental difference to our approach is that the methodology applied for the development of WebGPT is likely only viable for general-purpose Web search rather than a specific corpus of documents as in our case. Even if search was restricted to a specific dataset, the chatbot itself is trained on additional data.

Undoubtedly, existing research has developed much more advanced methods than those used in our work, and many novel tools are emerging that provide exciting opportunities for re-thinking the design of interactive search technologies (e.g. ResearchGPT [18], which enables users to "have a conversation with" research papers). Our methodology complements these efforts by supporting the process of building such systems.

VI. Conclusion

We have presented a methodology for the development of lightweight semantic search engines that use modern-day techniques to support users in navigating large volumes of data. We exemplified the benefits of applying this methodology through an extensive case study that highlights how a simple yet flexible processing pipeline that uses off-the-shelf technologies allows developers to build semantic search engines with ease.

With new AI and NLP research appearing every week, there is much scope for experimenting with more advanced methods that may yield more accurate or focused search results. We have, in a sense, deliberately avoided a focus on this to demonstrate how viable and usable systems can be developed even with fairly generic, tried and tested tools.

Another dimension we did not focus on is scalability and using advanced techniques to speed up search at scale, such as those provided by state-of-the-art vector databases (e.g. Pinecone[19] or Weaviate[20] that provide efficient indexing and query optimisation techniques. Replacing our current exhaustive vector comparison with these would bring massive improvements in terms of scalability, but we have seen that basic approach already produces satisfactory results on non-trivial problem sizes. A more scalable version of OM could open up exciting opportunities to expand its use to global research databases such as OpenAlex[21].

There are also interesting areas for future research that focus more on the architectural and algorithmic level. One of these is to extend our methodology to *federated* semantic search engines, which could allow us to integrate the functionalities of individual implementations seamlessly while safeguarding the integrity of each contributing system and ensuring data is only presented in ways that comply with the constraints of each individual sub-system.

Another avenue worth exploring is combining several items when responding to a query. Vector-space embeddings enable more advanced semantic operations, e.g. $x + y$ computed for two vectors x an y can capture the combined meaning of x and y. This could enable *compositional* search, e.g. in order

[16]opensemanticsearch.org
[17]E.g. kdb.ai and www.deepset.ai
[18]github.com/mukulpatnaik/researchgpt
[19]www.pinecone.io
[20]weaviate.io
[21]openalex.org

to identify the right *team* for a project or to put together a *collection* of documents which, taken together, satisfy the user's information needs.

We hope that the initial work we have presented here will encourage further research into these and other areas by supporting researchers with lightweight techniques they can apply in developing their approaches.

Acknowledgment

The authors wish to thank Amrey Krause and Simon Chapple at the University of Edinburgh and Robbie McGgugan and Tess Vaughan at the Glasgow University Software Service for their contributions to the development of the OpportunityMatch tool. We are also grateful to the over 30 participants in our stakeholder and focus groups for their input, and to the members of University of Edinburgh's AI and Data Ethics Board for their feedback and guidance provided during the development of the tool.

References

[1] H. Bast, B. Buchhold, and E. Haussmann, "Semantic search on text and knowledge bases," *Foundations and Trends in Information Retrieval*, vol. 10, no. 2-3, pp. 119–271, 2016.

[2] D. Fensel, U. Simsek, K. Angele, E. Huaman, K. Elias, O. Panasiuk *et al.*, *Knowledge Graphs - Methodology, Tools and Selected Use Cases*, D. Fensel, Ed. Springer, 2020.

[3] D. Jurafsky and J. H. Martin, *Speech and Language Processing*. 3rd Edition, Prentice Hall, 2023, forthcoming.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[5] B. Mitra and N. Craswell, "An introduction to neural information retrieval," in *Fnds Trends Inf Retrieval*, 2018, vol. 13, no. 1, pp. 1–126.

[6] A. S. Koepke, A.-M. Oncescu, J. Henriques, Z. Akata, and S. Albanie, "Audio retrieval with natural language queries: A benchmark study," *IEEE Transactions on Multimedia*, 2022.

[7] S. Agrawal, A. Chowdhary, S. Agarwala, V. Mayya, and S. Kamath, "Content-based medical image retrieval system for lung diseases using deep cnns," *Int J of Inf Technology*, vol. 14, no. 7, pp. 3619–3627, 2022.

[8] X. Yu, M. Malmir, X. He, J. Chen, T. Wang, Y. Wu, Y. Liu, and Y. Liu, "Cross interaction network for natural language guided video moment retrieval," in *Proceedings of SIGIR 2021*, 2021.

[9] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal *et al.*, "Language models are few-shot learners," *arXiv preprint 2005.14165*, 2020.

[10] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud *et al.*, "Emergent abilities of large language models," *Transactions on Machine Learning Research*, 2022.

[11] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," *CoRR abs/1405.4053*, 2014.

[12] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[13] S. Chowdam, "Identification of social biases in the OpportunityMatch tool," MSc Dissertation, The University of Edinburgh, 2021.

[14] S. Aghaei, K. Angele, E. Huaman, G. Bushati, M. Schiestl, and A. Fensel, "Interactive search on the web: The story so far," *Information*, vol. 13, no. 7, pp. 324–357, 2022.

[15] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL*, Minneapolis, MN, 2019, pp. 4171–4186.

[17] L. Dietz, H. Bast, S. Chatterjee, J. Dalton, E. Meij, and A. de Vries, "Neuro-symbolic approaches for information retrieval," in *Proceedings of ECIR 2023*, ser. LNCS, vol. 13982. Springer, 2023, pp. 324–330.

[18] Z. Zhang, B. G. Patra, A. Yaseen, J. Zhu, R. Sabharwal, K. Roberts *et al.*, "Scholarly recommendation systems: a literature survey," *Knowledge and Information Systems*, May 2023.

[19] A. Piktus, O. Ogundepo, C. Akiki, A. Oladipo, X. Zhang, H. Schoelkopf *et al.*, "GAIA search: Hugging face and pyserini interoperability for nlp training data exploration," *arXiv preprint 2306.01481*, 2023.

[20] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim *et al.*, "WebGPT: Browser-assisted question-answering with human feedback," *arXiv preprint 2112.09332*, 2022.