A Geo-Contextualized Multi-Genre Scientific Search Engine

A Novel Conceptual Design and Prototype Evaluation

Johannes Honeder[1], Roxanne El Baff[2], Tobias Hecking[2]
Alexander Nussbaumer[1], and Christian Guetl[1]

[1] Graz University of Technology, Graz, Austria
{alexander.nussbaumer,c.guetl}@tugraz.at
[2] German Aerospace Center (DLR), Germany
{roxanne.elbaff,tobias.hecking}@dlr.de

Abstract. Scientific research is nowadays challenging due to the overwhelming access to information. Current scientific search tools have two shortcomings: (1) they lack a contextualized search framework where the user can search based on geolocation, and (2) they do not interconnect different text genres (e.g., publications and blogs). To tackle these issues, this paper presents a novel, generic conceptual design for a scientific search tool that applies to any domain. Our approach focuses on two main components within the search engine: (1) steering search results towards geolocation-oriented search where *location* expressions are extracted from the user query and used within the search process, and (2) enabling multi-genre text retrieval to increase synergy discovery. We built a prototype for the earth observation and science domain, where geolocation is salient. Our qualitative evaluations show the benefits of visually representing scientific research on map views, but they also reveal crucial points to improve our design further. The code is available here: https://zenodo.org/records/13788713, and the post-evaluation version is available here: https://zenodo.org/records/13789303.

Keywords: Open Web Search, science search, Specialized information retrieval

1 Introduction

Apart from traditional outlets today, much scientifically relevant information can be found online (e.g., science blogs, news reports). However, research data and websites are still very much disentangled. In this work, we aim to tighten the integration of geo-annotated web information and earth observation data products to facilitate scientific discovery in the environmental sciences, allowing for:1) enabling contextualized search using geolocation for more meaningful results and, 2) interlinking relevant scientific data across different genres (e.g., scientific papers, blogs).

Even though existing work facilitates science search through specialized search applications, they have two shortcomings: 1) The lack of incorporation of geolocation information limits users from conducting a nuanced search and discovering regionally scientific work for scientific knowledge or collaboration. And, 2) the usage of one-genre science-related data (e.g., publications, research datasets, or open-source software), ignoring the considerable potential of relevant information available on the web nowadays [4], consequently restricting users from exploring and interlinking different scientific genres.

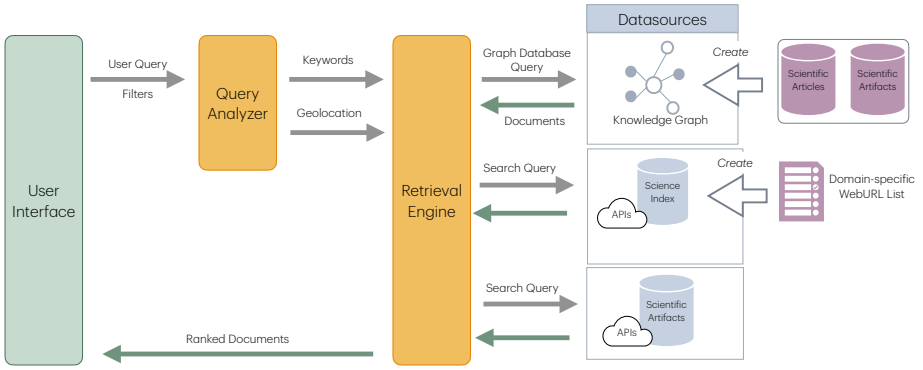


Fig. 1: A modular overview of the conceptual architecture of the search engine. It includes on the frontend side the *User Interface* (Green) where the user sends natural language queries, then, on the Backend side (in Yellow), the *Query Analyzer* extract the keywords and geolocation information from the query and sends it to the *Retrieval Engine* (RE). The RE retrieves ranked documents from three different datasources, curated for the scientific domain (in Blue): a.) a knowledge graph combining scientific articles and artifacts, b) a search index containing scientific domain-specific URLs, and c) domain-specific artifacts.

To overcome these challenges, we develop a conceptual design for a search tool for the science domain, as illustrated in Figure 1. The presented design focuses on augmenting the search results with geolocation data to increase contextual search and interlink different scientific genres. Our architecture incorporates two core components to achieve this: The *query analyzer* (yellow) provides categorized expressions such as location and keywords for rich contextual results, and a *multi-structural search indices* for different text genre interlinkage.

Figure 1 reveals three main components (from left to right): 1) The *User Interface* (UI) (Frontend) allows the user to search via a natural language query and other filters (e.g., Author's name). After the search hits are fetched, the UI visualizes the search hits in multiple views, a map view pinning each work to a location on a map or a list view employing traditional search view results.

2) The *Query Analyzer* (Backend) extracts geolocation information and keywords and other criteria from the user's query, steering the results towards not only keyword- but also criteria-based focus.

3) The keywords and criteria are sent to the *Retrieval Engine* containing different structures of search, each serving different capabilities: a.) *Knowledge Graph Search* enables exploratory science search by interconnecting science articles and artifacts in a graphical database, b.) *Web Search*: enables access to a broad type of web-content (e.g., blogs, journals) with the ease of expansion. And c.) specialized search enables access to curated domain-specific up-to-date data. Figure 1-Blue shows an illustration.

The presented concept is generic and, therefore, can be applied to any scientific domain. For this work, we build a prototype specifically targeting the Earth Observation and Earth Science domains, focusing on studying planet Earth's natural systems through satellite data. Given the nature of using satellite imagery, geospatial context is salient in these domains. The application targets geoscientists, earth observation researchers, and users generally interested in the domains. The development phase included gathering data, building and integrating multi-structural indices, and developing a web application to visualize the data and facilitate user interaction.

To showcase the usability of our web application, we conducted a qualitative and qualitative user study to evaluate the prototype by expert and non-expert users. The evaluation revealed the importance of integrating geolocation data and interlinking different text genres. However, it also shows a need for improving the user interface to avoid overwhelming users with information.

Our contributions are three-fold: (1) We propose a generic concept for a domain-specific scientific search engine that augments geolocation and uses multi-structural index-based search to provide contextual and synergistic results. (2) We implement and evaluate a prototype tailored to Earth Observation/Science domains, revealing our approach's effectiveness. And (3) we conduct a quantitative and qualitative evaluation that highlights the benefits of our approach and identifies weaknesses for future improvement.

2 Related Work

Science search is a form of exploratory search, including many open-ended tasks paired with high timeliness and a large answer volume. Simultaneously, the scientific search process is opportunistic, iterative, and multi-tactical [7]. Therefore, design principles facilitate exploratory search, and possible connections with research datasets must be established. [7] explore the information needs and specific requirements of scientific search tasks to determine essential user interface tools for effective scientific search. They define three sets of tools to improve users' exploratory search tasks focusing on 1) querying, 2) visualizing, and 3) long search: 1) **Tools for querying** are pivotal in aiding users to formulate suitable queries for their search tasks. This category includes features such as rapid query refinement, where the user interface predicts and auto-completes

queries based on previous interactions. Recommending appropriate queries is another tool that aids users in formulating relevant queries. 2) **Tools for analyzing and visualizing search results** are a category of concepts that help the user make sense of the retrieved documents by visualizing certain aspects of the data and providing tools that allow users to filter and restrict the search results (e.g., facets, hierarchical classification, and data visualization tools). 3) **Tools for long search** enable social collaborations for scientists to share their search results among colleagues and save intermediate results to continue search processes over a more extended period [7]. Various providers and institutions have developed web applications within the field of science search, explicitly focusing on earth observation,- environmental, and earth science. This paper identifies and compares analogous applications to draw inspiration for the design. In total, 13 applications were investigated in terms of user interface, visualization techniques, and search capabilities (see Table 2.1), from which we identify and collect valuable features and design ideas for the proposed science search application while taking into consideration the difference in objectives between each of the 13 applications and our work.

For the science search applications in the domain of earth science and EO, two related applications were identified in this domain: Science Discovery Engine [1] developed by NASA and Pangaea [3], a search application and data portal for georeferenced data in the domain of earth system science (see Figure 2.9). Both search applications mainly host scientific data and publications related to the specific domain of the application. Both applications have a simple yet satisfying UI design, with Pangaea also implementing an interactive map component for showing search results and filtering documents based on their geospatial data. Also, both search applications allow for simple keyword searches but also allow more sophisticated searches based on topics, tags, and other criteria.

3 Conceptual Design

This section explains the conceptual design behind the scientific search tool, as illustrated in Figure 1. This architecture has several interconnected modules that work together to enhance the search capabilities of a domain-specific search tool.

As an overview, following the user interaction flow: first, the user submits a query using the user interface (UI) (Figure 1-Green), which triggers a search request to the Backend module (Figure 1-Yellow) instantiating two steps. Within the backend, the first step involves the **Query Analyzer** module extracting relevant information from the keyword query, such as *location* and *topic* expressions, which are sent to the **Retrieval Engine**. The Retrieval Engine processes these parameters to fetch content from multiple indices (Figure 1-Blue) that are based on multiple datasources (Figure 1-Purple). It aggregates the results and ranks them based on relevance and geospatial proximity. After consolidating the responses, the Retrieval Engine forwards the final aggregated and ranked search results to the UI to visualize them in a *map view* or genre-specific *document list*.

Given the importance of geospatial context in many scientific domains, particularly in Earth Sciences, our design integrates geolocation data at the core of the search process. The following sections describe the **Query Analyzer** and the **Retrieval Engine**.

3.1 Query Analyzer

The leading search query is a textual content search, where users can express their information needs with a simple keyword query in natural language. The text query is processed to extract specific information to fine-tune the search with specific criteria (e.g., location expression). Because we focus on the *geospatial use case*, the Query Analyzer mainly takes as input textual data. It extracts the following parameters: (a) **Spatial (Geospatial)**: Identifies geographic locations mentioned in the query, and (b) **Topic / Theme**: Extracts keywords and phrases from the natural language query.

3.2 Multi-Structural Search Index

Fig. 2: Intersecting domains of the search application

To provide comprehensive search results, we employ a multi-structural design for the search indices, targeting three multi-purpose search types, as illustrated in Figure 2: 1) **Exploratory Science search** uses a knowledge graph to interconnect scientific publications with other artifacts, such as geospatial data and code repositories, empowering users to discover relationships between different resources. 2) **Web Search** builds a broad scientific-domain web index that contains scientific articles, blogs, and more. The indexed links are manually curated to ensure domain-specific relevance and genre diversity go hand in hand. 3) **Specialized Search** uses scientific domain-specific datasource(s) curated by experts for fetching the latest research and information that usually are not indexed on the web.

The **Retrieval Engine** concurrently queries all three search indices and aggregates the results.

Figure 1-Blue shows that the presented approach intersects exploratory search, web search, and specialized search, combining information from various sources into a single user interface. Ultimately, the proposed search application will enable researchers to find synergy by interlinking different sources.

Integrating query-geolocation focus with these three search types would lead to a more holistic and general approach to discovering and accessing scientific information and data sources while enabling contextualized search.

3.3 Results view

A search query triggers each of the three search types. Each returns the top hits based on location and keywords. The results are displayed in a map view or document list view.

In the *map view*, search results are visualized as markers on an interactive map, allowing users to explore results based on location. Users can also click on a marker to explore further and read more details. Whereas for the *document list* view, the search hits are shown in the traditional view of standard search engines, where the user can toggle between text genres (e.g., publications, web).

This conceptual design integrates geolocation data and diverse data genres, increasing contextualized search and interdisciplinary synergy.

4 Data

This section specifies the data types and sources used for the Earth Observation Search Engine. Then, we define the design of each index structure using the data specified. The following subsections overview the different datasources used within the three search types and describe the knowledge graph within the exploratory search.

4.1 The Different Data Sources

Our prototype integrates data from multiple sources to support multi-genre search capabilities. These sources include Web Documents, scientific publications, and SpatioTemporal Asset Catalog (STAC) Collections, which is a standardized format for representing properties of geospatial data, and other: **(1) Web Documents:** A curated list of 440 websites related to Earth Observation and natural disasters was used to create a web index partition for web searches provided by the OWS Service, which includes news outlets, scientific journals, and popular scientific websites from agencies like NASA and ESA. Scientific Publications: A dataset of approximately 12,000 scientific publications in Earth Observation research focusing on natural disasters, provided by the German Aerospace Center (DLR). The dataset was preprocessed to extract keywords and author names, **(2) SpecioTemporal Asset Catalog (STAC) Collections:** A collection of 166 STAC collections from open Earth Observation catalogs, including providers like Planetary Computer, Terrabyte STAC API, and DLR's EOC Geoservice. These collections were selected based on relevance and open-access availability. And **(3) Additional Earth Observation Data:** Data on 344 Earth Observation missions and 431 instruments from the Committee on Earth Observation Satellites (CEOS) Database to enrich search results and enable further interactions.

4.2 Knowledge Graph

As the core of the system that enables the exploratory search of scientific publications interconnected with other aspects specific to the Earth Observation

domain, a knowledge graph was built from different data sources that interlink different information items; earth observation data was systematically harvested from several providers offering SpatioTemporal Asset Catalogs (STACs)[3]. Since STAC constitutes a common language to describe geospatial information and make it accessible through APIs, several data sources could be integrated.

We describe below the different datasources used, the graph ontology revealing the interconnection between the different datasources, and technical tools.

Datasources

- **Publication**: is a scientific publication.
- **STAC Source**: is a STAC provider that provides EO data via an API
- **STAC Collection**: is a STAC Collection
- **Author**: is a publication author.
- **EO Mission**: is a satellite mission that provides EO data.
- **EO Instruments**: is a scientific measurement device on board of an EO satellite mission.
- **Keyword**: is a keyword in the context of scientific literature, concepts, or use-case-specific terms.

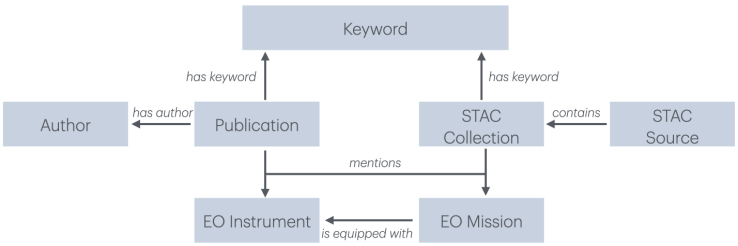


Fig. 3: The Ontology of the Knowledge Graph.

Ontology The ontology of a knowledge graph (KG) defines the existing entity classes and types of relationships between these entities. Figure 3 shows a sketch of the ontology, which includes all the data described in Section §4.2

The KG holds the publications and STAC collections, that are connected with each other through the **Keyword** collection with the relationship *HasKeyword*. Publications and STAC Collections are connected to EO Missions and EO Instruments with the relationship *Mentions*. Additionally, an EO Mission can contain multiple EO Instruments, which is represented in the edge relation *Is Equipped With*. Relevant to the topic of EO are *EO Missions* and *EO Instruments*. Similar to the relationship **HasKeyword**, both publications and STAC

[3] https://stacspec.org/

Collections are connected to EO Instruments and Missions with the relationship **Mentions**. Also, each EO Mission can contain multiple EO Instruments and is connected via **Is Equipped With**.

The **HasAuthor** edge connects Publications to at least one author. Each STAC Collection has a STAC provider; therefore, the edge **Contains** connects a **STAC Source** with a **STAC Collection**. The shared connections from publications and STAC collections to keywords, EO missions, and EO instruments allow advanced graph queries connecting the domains of science and EO.

5 Implementation

This section illustrates the development of a search engine prototype for the Earth Observation and Science domain. Following the presented conceptual design (§3), below we elaborate on the technical details of the *User Interface* (frontend) where we describe the input and output, and the two backend components: *Query Analyzer* and *Retrieval Engine*. The prototype focuses on natural disaster events within Earth Observation and Earth Science. We first frame the prototype with a focus, then give an overview of the system architecture and the technological stack we use. After that, we delve into the technical implementation of each part.

5.1 Focus

For our prototype, we focus on natural disasters such as floods, earthquakes, wildfires, and landslides, which are highly monitored events within the Earth Observation and Science domain. Narrowing the scope makes our prototype feasible within resource constraints.

5.2 System Architecture

Our prototype is developed as a dynamic web application with several interconnected modules. The key components, illustrated in Figure 4, include:

- **Search User Interface (Web Client):** A dynamic, single-page web application that allows users to submit queries and visualize search results.
- **BackEnd (ASGI[4] Web Server):** The backend infrastructure that handles search requests sends them to the **Query Analyzer**, then to the **Retrieval Engine** (RE), where it initiates retrieval modules and communicates with external services. Within the RE, there are three components:
  - **Knowledge Graph (Graph Database):** A central database that hosts the knowledge graph, enabling the search of scientific publications and SpatioTemporal Asset Catalog (STAC) collections.

[4] Asynchronous Server Gateway Interface (ASGI) specification, handles asynchronous Python web applications. It is a standardized interface between asynchronous Python web servers, frameworks, and applications, mainly supporting asynchronous communication and concurrency.

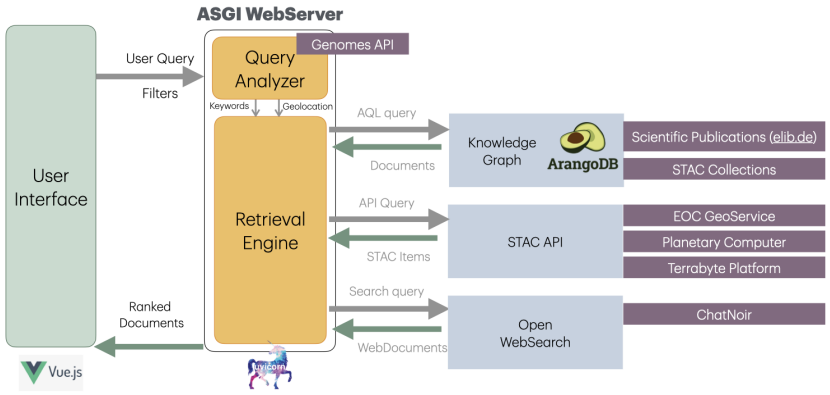


Fig. 4: Simplified sketch of the final software architecture (derived from the conceptual architecture in Figure 1), including the Search User Interface developed with VUE.JS, ASGI Web Server for the backend, and the data pathways

- **External Services**:
- **Open Web Search (OWS) Service**: Provides web search capabilities for web documents.
- **STAC API:** Allows retrieval of Earth Observation data from different providers.

Figure 4 presents a simplified software architecture sketch highlighting the interactions between these components.

5.3 Technology Stack

This subsection gives an overview of all the technology and frameworks we used for our prototype, divided into Frontend and Backend Technology Stack:

Frontend

- **Vue.js[13]:** A JavaScript framework for building user interfaces [13], with **PrimeVue[8]** to build a responsive interface, along with its development and deployment tool: VueCLI and **Webpack**.
- **Leaflet[12]:** An open-source JavaScript library for interactive maps.
- **Axios**[5]**:** For making HTTP requests from the browser.

Backend

- **ASGI Webserver:** The backend infrastructure is configured as an Asynchronous Server Gateway Interface (ASGI) web server. ASGI[6] is a Python

[5] https://axios-http.com/
[6] https://github.com/django/asgiref/tree/main

standard facilitating asynchronous communication between web applications and servers. The web server is instantiated using Uvicorn[11]. The API which facilitates the communication between the backend server and the web client is constructed using FastAPI[9].

- **spaCy[5]:** For natural language processing tasks, including named entity recognition [5]. It is used to enable the semantic labeling in the query analyzer.
- **SentenceTransformers[10] :**These are used to create text embeddings used in retrieval.
- **Geocoder**[7]**:** For geoparsing location expressions.
- **Graph Database:** We use ArangoDB, a NoSQL database system used to implement the knowledge graph. On top of ArangoDB, we use Corpus Annotation Graph Builder (CAG)[2] for building and populating the knowledge graph.
- Containerization with Docker:For deploying application modules in containers [6].

5.4 Knowledge Graph Implementation

The knowledge graph is central to our prototype, connecting different data types and enabling exploratory search capabilities. Following Section 4.2, where we defined the ontology, we detail below the creation of the knowledge graph.

The knowledge graph is hosted in an ArangoDB [8] graph database, and its primary purpose is to store, retrieve, and search data. For the search, ArangoDB provides the ArangoDB Query Language (AQL)[9].

After the data acquisition, the knowledge graph is created with CAG [2], a framework for creating graphs with ArangoDB. After defining the graph structure (ontology) within the CAG framework, the graph is initialized and populated with the provided data corpus. The pipeline for populating the graph consists of four data sources that are used to create the graph sequentially: EO Instruments ($n = 432$), EO Missions ($n = 345$), Publications ($n = 11379$), and STAC Collections ($n = 166$).

The EO Instruments are initialized in the first step using the available CSV file from the CEOS database[10]. After the EO Instruments are included in the graph, the EO Missions, also provided as CSV files from the CEOS database, are the next data collection added to the graph. Since a single EO Mission can potentially contain one or more EO Instruments, an outgoing edge *Is Equipped With* is created for each connection to an EO Instrument. After initializing the graph with the EO missions and instruments, the next step is to populate the graph with the scientific publications data corpus. Each file contains several authors and keywords, creating new nodes for these data types. The author

[7] https://geocoder.readthedocs.io/
[8] https://arangodb.com/download/
[9] https://docs.arangodb.com/stable/aql/
[10] https://database.eohandbook.com/

names are parsed from the publication file, and for each found author, a data node and an outgoing edge connection *HasAuthor* from the publication to the author node is created. Similarly, *HasKeyword* connects a publication node to a keyword node.

Each keyword in the publication is checked to determine whether it refers to an EO Mission or Instrument. Given that a publication file lacks a specific attribute for referencing EO missions or instruments, this connection is established by comparing each keyword. Upon identifying a link, indicating that a keyword relates to an EO node, a connection labeled *Mentions* is established with the relevant existing EO node.

Lastly, the graph is populated with the STAC Collection nodes. Each STAC Collection file has dedicated attributes "platforms" and "instruments," referring to EO missions and EO instruments. For each occurrence, an edge connection *Mentions* to the corresponding node is created. Additionally, the keywords from each STAC Collection are parsed, and *HasKeyword* edge connections are created.

Sentence Embeddings We generated text embeddings for document attributes (titles, abstracts, descriptions) using SentenceTransformers. The embeddings support semantic search between queries and documents.

Search Indices In order to ensure a fast document search within the graph, ArangoSearch was used to create full-text search indices on top of the document collections: publications and STAC collections. ArangoDB Analyzers do the text processing to create the search index. The first analyzer is a text tokenizer that applies stemming and stopword removal on the text attributes. The second analyzer is an N-Gram analyzer, which computes trigrams on top of the text to enable fuzzy search, which was applied to enable efficient and fuzzy search capabilities for fast document retrieval.

The relevant document attributes on which the search views are created are "title" and "abstract" for publication documents and "title" and "description" attributes for the STAC Collections.

5.5 Query Analyzer and Retrieval Engine

The Query Analyzer and the Retrieval Engine handle search requests and coordinate communication between the user interface and retrieval modules. It consists of: (1) Query Analyzer – This module extracts location expressions and concept keywords from user queries using spaCy's NER model and the Geonames API. (2) Graph Retrieval Module (Part of RE) – This module queries the knowledge graph to retrieve relevant publications and STAC collections. It supports simple search using full-text indices and advanced search using embedding-based similarity calculations. (3) Web Search Module (Part of RE): This module handles web search requests via the OWS Service and communicates with the OWS API to retrieve relevant web documents based on the user query and spatial filters. And (4) the STAC Module (Part of RE), which enables retrieval of STAC items from external providers based on user-specified parameters such as location and

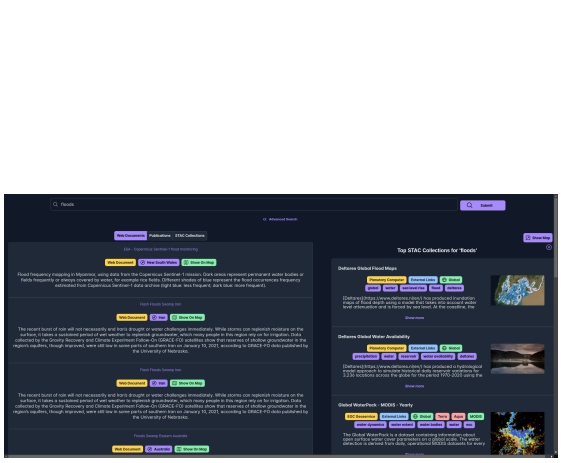
time interval. Figure 4 shows a detailed architecture of the Retrieval Engine and its data pathways.

5.6 Search User Interface

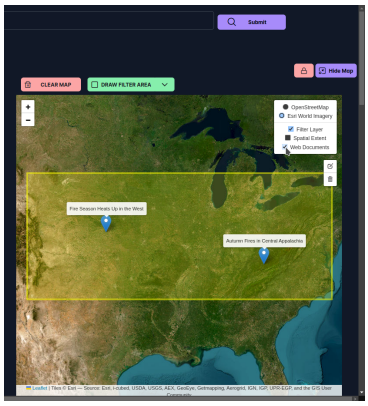
The user interface has a multi-view dashboard comprising:

- **Search Header:** Allows users to submit queries and filters (e.g., keywords, authors, and EO missions/instruments).
- **Document List:** Displays search results in a list view, with custom templates for different document types (web documents, publications, STAC collections). Each template highlights relevant metadata.
- **Interactive Map:** Visualizes search results geospatially using Leaflet. Users can define spatial filters, switch between base layers (e.g., OpenStreetMap, World Imagery), and interact with map layers representing web documents, STAC collections, and STAC items.

Figure 5a shows a screenshot of the multi-view search user interface.



(a) Screenshot of the multi-view search user interface.



(b) Screenshot of the map component, including a filter layer (yellow) and the open *Edit Menu* in the right corner.

Fig. 5: Screenshots from the Scientific search tool prototype.

The map component in the search user interface consists of an interactive map built with Leaflet and some additional buttons for user interaction (see also Figure 5b).

There are two base layers available in the map component that can be displayed: 1) OpenStreetMap[11] visualizes natural and human-made features on the

[11] https://www.openstreetmap.org/

map in a classical street view, highlighting the names of places. And 2) base layer, which is the Esri World Imagery[12] layer that is based on EO Imagery.

The base layers can be switched in the Edit Menu in the right corner (Figure 5b). The edit menu also allows the user to show or hide specific layers in the map, including the Filter Layer, the Spatial Extent Layer, and the Web Document Layer. These layers are dynamically layered on top of the base layer.

The **Spatial Extent Layer** represents the spatial extent of a chosen STAC Collection. This layer is created for specific coordinates and displayed when "Show Spatial Event" is clicked. The spatial extent coordinates are represented as GeoJSON objects encoding one or multiple bounding boxes.

The **Web Document Layer** contains a list of spatial attributes from a set of web documents. When the web documents are retrieved from a search query, the geographic layer on the interactive map is automatically created. Each web document is represented as a single pin on the map (Figure 5b).

The **Filter Layer** represents a user-defined spatial filter for the different requests that allow spatial parameters. When clicking "Draw Filter Area," users can draw either a Polygon or Rectangle directly on the map. The selected filter area can then be used subsequently for search queries as spatial filters or for STAC Item requests that require a spatial parameter.

The "Clear Map" button deletes all created layers on top of the map.

Another layer that can be rendered on top of the map is the **STAC Item Layer**. Whenever STAC Item Requests are successful, the resulting STAC Items are rendered in the map as colored polygons (see also Figure 6). Each STAC Item contains a spatial extent attribute, which decodes the spatial position of the STAC Item as a GeoJSON feature, which is used directly to create the map layer. The STAC Item Layer is interactively connected to the Document List. Clicking on a colored polygon on the map highlights it in red, and the corresponding STAC item in the document list is highlighted and focused on the interface.

5.7 Deployment

The containerization architecture ensures consistent environments across development and deployment, facilitating scalability. We mainly deploy the search tool with Docker, containerizing the frontend, backend, and graph database. ArangoDB container hosts the graph database, listening on internal port 8529. The Backend container runs the ASGI web server and initializes the knowledge graph. It listens on port 5000 for requests from the front end. Lastly, the frontend container deploys the Vue.js web application on port 8080.

[12] https://www.arcgis.com/home/item.html?id=10df2279f9684e4a9f6a7f08febac2a9

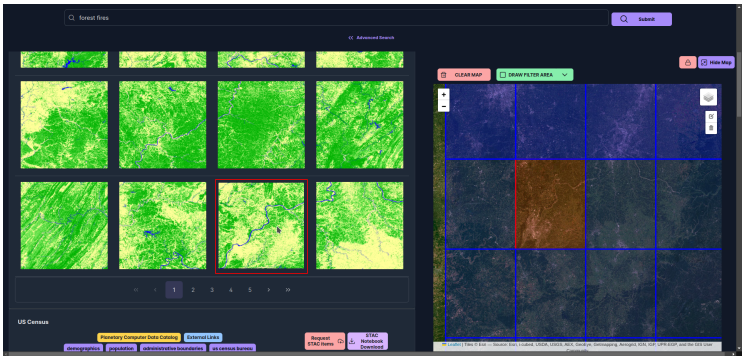


Fig. 6: Screenshot of the document list and the map component in which multiple STAC Items are rendered, with one being selected in red color.

6 Evaluation

This section describes the quantitative and qualitative evaluation of the Earth Observation search tool, highlighting the importance of multi-structural indices and geospatial visualization in the map view.

6.1 Scope and Goal

We prepared a web-based evaluation user guide with three parts: (1) an overview of the search tool, (2) two defined tasks, and (3) a questionnaire capturing both quantitative and qualitative aspects.

The participants completed the evaluation independently, as the evaluation form and the search tool were hosted online. After reading an overview describing the search tool scope (Earth Observation), data sources, and geolocation integration, each participant conducted the following two tasks:

- **Task 1.** Search for *"wildfire events"* and explore all search-hit genres in the Document list view, such as publications, webpages, and satellite images. This task assesses the usability of different data sources for a specific query (*wildfire events*).
- **Task 2.** Search for *"Land cover,"* focusing on the Map View, where the user formulates data requests for satellite imagery for a specific area and STAC Collection.

Research Questions and Questionnaire After completing the tasks, each participant filled out a questionnaire designed to answer three research questions (**RQs**):

- **RQ1** How does an integrated web search within a science search application affect the scientific search process in the context of exploratory science search?

- **RQ2** How usable is the integration of geospatial data in a search tool for science search?
- **RQ3** How usable is the search tool in exploratory science search?
  - **(a)** How usable are the different features and functionality in the search user interface?
  - **(b)** How can the prototype be further improved?

The questionnaire contained 17 questions (Q1 to Q17). (1) For our quantitative evaluation, 10 of these questions were closed questions answered on a Likert scale from 1 (*'Do not Agree'*) to 5 (*'Fully Agree'*). (2) For our qualitative evaluation, the remaining seven questions were open-ended, allowing participants to provide written answers. Overall, the questions addressed the conceptual design of the web-based science search application and the prototype's usability and features, as shown in §6.3.

6.2 Participants

In total, 18 participants (5 female, 12 male) completed the tasks and questionnaires. They were affiliated with either a European research institute or a university. Fifteen participants held a university degree, while three had a High School degree. The majority (n=11) work or study in some field of Computer Science, with the remaining participants spread across Earth Observation, psychology, migration and urban studies, biochemistry, chemical engineering, biomedical engineering, and medicine.

6.3 Results

After performing the two tasks described in §6.1, participants answered the 17 questions summarized in Tables 1 and 2.

Table 1 shows the average scores and standard deviations for Q1–Q10, each linked to its respective research question. Higher scores for Q1, Q3–Q8, and Q10 reflect more positive impressions, whereas for Q2 and Q9, lower scores indicate a more favorable outcome. Participants generally gave high scores (mean ≥ 4), except for Q8 (3.33). The relatively low score for Q8 suggests that the interface might still be too complex. Results also show that participants appreciate the geographical dimension of the search results (Q4: 4.94) and the multi-genre results (Q1: 4.67, Q3: 4.61, and $\downarrow$Q2: 1.78).

The open questions Q11–Q17 are listed in Table 2. In the following subsection (§6.4), we discuss the results from all 17 questions (Q1–Q17) by grouping them under the three research questions and highlighting key insights.

6.4 Findings

This subsection addresses the three research questions based on the 18 participants' responses (Q1–Q17) and outlines potential improvements and avenues for future work.

#	RQ	Statement	Average (std)
Q1	RQ1	I like searching both the web and scientific databases when exploring new research areas	↑ 4.67 (.58)
Q2	RQ1	I do not like the idea of simultaneously searching multiple data sources (e.g., websites, scientific publications) in a single search interface.	↓ 1.78 (.97)
Q3	RQ1	It makes sense to combine multiple data types and data sources in a single search application and enable a more holistic topic search.	↑ 4.61 (.68)
Q4	RQ2	For some use cases, it makes sense to visualize the geographical dimension of search results.	↑ **4.94 (0.23)**
Q5	RQ1	I like having a simple interface and not having too many distracting visualizations regarding science search.	↑ 4.00 (1.15)
Q6	RQ3(b)	I could imagine other visualizations (e.g., network graphs, timeline) for more aspects and metadata of the results.	↑ 4.50 (0.69)
Q7	RQ2, RQ3(a)	I found the interactive map component useful for searching/visualizing geographical information of the results.	↑ 4.56 (0.59)
Q8	RQ3(a)	the interface does not overload the user with information.	↑ **3.33 (0.94)**
Q9	RQ3(a)	The (advanced) search was difficult to use.	↓ 2.56 (1.01)
Q10	RQ3(a)	I like having a "Dashboard" view to see search results from different sources in a single screen.	↑ 4.50 (0.60)

Table 1: Average scores (and standard deviations) for the 10 Likert-scale questions (#: Q1–Q10). Higher values (↑) are generally more favorable, while lower values (↓) are more favorable for negatively worded statements. Bold indicates the highest or lowest value observed for each type (↑ / ↓).

RQ1. Participants appreciated the multi-structural index results (Q1–Q3) but highlighted the need for a simple interface. Excerpts from open questions include:

- *"I like the integrated search of different research items in addition to web resources."* **(Q15)**
- *"Filtering between Web Documents, Publications, and EO Catalogues separately is a useful feature."* **(Q15)**

RQ2. Participants found the geospatial visualization valuable (Q4, Q7) and showed interest in additional criteria-based views, such as temporal filtering (Q11, Q13, Q15). Some comments:

- *"I really like the feature 'show on map' for the retrieved web documents and the possibility to click on further locations marked on the map."* **(Q15)**
- *"Filter events based on a time range."* **(Q11)**

#	RQ	Open Question
Q11	RQ2, RQ3(b)	Which functionality would help you to explore scientific topics and concepts?
Q12	RQ3(b)	Do you suggest different use cases or research areas that would benefit from such an integrated science search engine?
Q13	RQ2, RQ3(b)	Do you have any suggestions of other (meta) data that can be visualized in the interface?
Q14	RQ3(b)	Do you suggest integrating other data sources in a science search application?
Q15	RQ1, RQ2, RQ3(a)	Which functionalities and features did you like?
Q16	RQ2, RQ3(b)	Which functionality and features did you miss in the prototype?
Q17		Do you have any final remarks about the prototype?

Table 2: Open questions Q11–Q17, along with their corresponding research questions (RQ).

RQ3(a). Participants liked the interactive maps and consolidated view for each data genre (Q7, Q10). While most liked the dashboard view (Q10; e.g., "I like the dashboard view in general ..." – Q15), a few found it overwhelming (Q8; e.g., *"It could be overwhelming for the user if everything is listed at the same time..."* – Q17). For the advanced search feature, several respondents expected a graph-based visualization:

- *"Maybe the concept of a graph search could be made more layman-friendly."* **(Q17)**
- *"When I clicked on Graph search, I somehow expected to see a graph of connected items."* **(Q17)**

RQ3(b). Suggestions for enhancing the prototype included adding connected graphs to visualize topic relationships (Q11) and integrating additional data sources (e.g., the European Open Science initiative). Participants also noted that the core concept can support many use cases (Q12), from urban studies to georisk/natural hazards. They suggested extending metadata (Q13), such as publication dates and open-access information. Finally, UI feedback (Q16) touched on unclear search results (e.g., distinguishing "no results" from user errors) and improving readability (Q16).

Overall, the user study highlights the prototype's strengths and areas for refinement. As a first step, we released an updated version[13] featuring improved font styles and colors, a simplified dashboard to address information overload, added query suggestions, and a snippet summary for each data genre. Figures 7

[13] https://zenodo.org/records/13789303

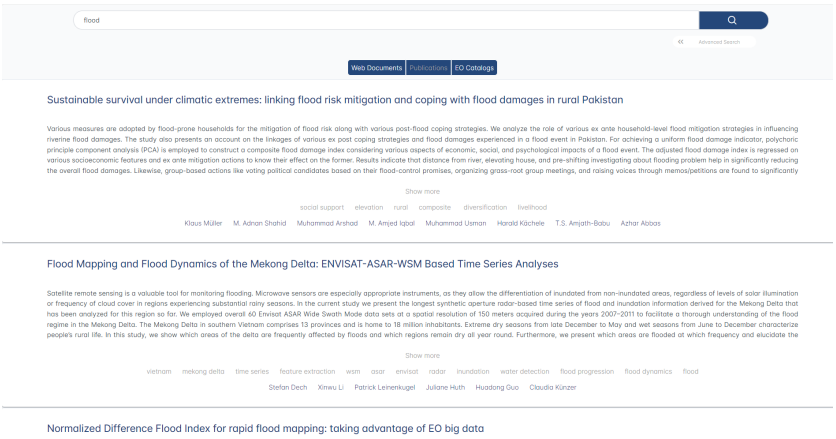


Fig. 7: Post-evaluation screenshot of the improved science search tool dashboard—*Publication* view.

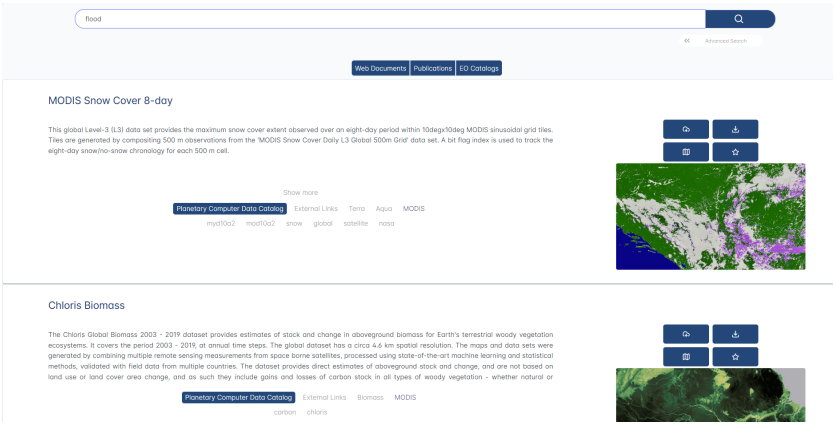


Fig. 8: Post-evaluation screenshot of the improved science search tool dashboard—*EO Catalogs* view.

and 8 show snapshots of the improved UI. Future work will explore incorporating additional data sources, graph database visualization, and a follow-up evaluation round.

7 Conclusion

This paper introduced a novel generic concept for science search applications that empower geolocation-aware search, boosted by integrating multi-genre document search for knowledge discovery. Our prototype showed that combining different information sources enables discoveries in the environmental sciences

and increases search efficiency for locating crises. The user evaluation showed the saliency of geo-based, multi-source search and drew a roadmap for future enhancements. For example, further leveraging the information in documents' geospatial and temporal metadata can increase the explorative phenomena of scientific datasets.

References

1. Bugbee, K., Acharya, A., Davis, C., Foshee, E., Ramachandran, R., Li, X., Ramasubramanian, M.: Nasa's science discovery engine: An interdisciplinary, open science data and information discovery service. Tech. rep., Copernicus Meetings (2023)
2. El Baff, R., Hecking, T., Hamm, A., Korte, J.W., Bartsch, S.: Corpus annotation graph builder (CAG): An architectural framework to create and annotate a multi-source graph. In: Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations. pp. 248–255. Association for Computational Linguistics, Dubrovnik, Croatia (May 2023), https://aclanthology.org/2023.eacl-demo.28
3. Felden, J., Möller, L., Schindler, U., Huber, R., Schumacher, S., Koppe, R., Diepenbroek, M., Glöckner, F.O.: Pangaea - data publisher for earth & environmental science. Scientific Data **10**(1), 347 (Jun 2023). https://doi.org/10.1038/s41597-023-02269-x, https://doi.org/10.1038/s41597-023-02269-x
4. Granitzer, M., Voigt, S., Fathima, N.A., Golasowski, M., Guetl, C., Hecking, T., Hendriksen, G., Hiemstra, D., Martinovič, J., Mitrović, J., Mlakar, I., Moiras, S., Nussbaumer, A., Öster, P., Potthast, M., Srdič, M.S., Megi, S., Slaninová, K., Stein, B., de Vries, A.P., Vondrák, V., Wagner, A., Zerhoudi, S.: Impact and development of an open web index for open web search. Journal of the Association for Information Science and Technology **n/a**(n/a) (2023). https://doi.org/https://doi.org/10.1002/asi.24818, https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.24818
5. Honnibal, M., Montani, I., Van Landeghem, S., Boyd, A.: spaCy: Industrial-strength Natural Language Processing in Python (2020). https://doi.org/10.5281/zenodo.1212303
6. Merkel, D.: Docker: lightweight linux containers for consistent development and deployment. Linux journal **2014**(239), 2 (2014)
7. Nedumov, Y., Kuznetsov, S.: Exploratory search for scientific articles. Programming and Computer Software **45**, 405–416 (12 2019). https://doi.org/10.1134/S0361768819070089
8. PrimeVue: The Next-Gen UI Suite for Vue.js (2023), https://primevue.org/
9. Ramirez, S.: Fastapi (2023), https://fastapi.tiangolo.com
10. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. CoRR **abs/1908.10084** (2019), http://arxiv.org/abs/1908.10084
11. Tom Christie: uvicorn: A Lightning-Fast ASGI Server for Python (2023), https://www.uvicorn.org/
12. Vladimir Agafonkin, Leaflet Contributors: Leaflet.js: An Open-Source JavaScript Library for Mobile-Friendly Interactive Maps (2023), https://leafletjs.com/
13. You, E.: Vue.js: The Progressive JavaScript Framework (2023), https://vuejs.org/