UResearcher

Liu, Christopher
Sharma, Ritesh
Tata, Micheal
Zhao, Jack
Frank, Will

Team Cool Guys
October 25th, 2019

Table of Contents

Executive Summary

Researchers currently sift through thousands of online research papers regularly to find relevant and useful information pertaining to a particular project. Consequently, an exorbitant amount of time is spent filtering out excessive data, misleading citations, and irrelevant topics, and, sometimes, these efforts prove fruitless because of misinformation and conflicting research results. Consequently, such tedious tasks lead to frustration and impediments. UResearcher remedies these issues plaguing research by significantly reducing the search time for pertinent topics and by efficiently organizing information at search time. The application accomplishes this task by narrowing the search space while finding useful and relevant information through machine learning models.

UResearcher primarily uses unsupervised clustering through public, machine learning libraries to organize the research data. The application retrieves vector information from available abstracts; then, it produces an appropriate model from the gathered information to classify and describe future inputs. Therefore, this feature can easily organize the search space from a user-defined input, reducing not only the search time but also the processing stress. As a result, the program significantly decreases unnecessary research time.

Also, UResearcher identifies previously unidentified knowledge using methods described by researchers from Lawrence Berkeley National Laboratory. This preprocessing, known as Latent Knowledge Analysis, uses the Word2Vec and Gensim libraries to estimate and weight abstract information from papers. From this data, the application constructs a machine-learning based model, which identifies areas of interest for research or gaps in knowledge. Consequently, relevant research becomes readily available without a significant loss in information, ensuring a smooth process.

Furthermore, UResearcher analyzes and predicts trends for fruitful research opportunities by looking at proposal calls for past, present, and future funding insights. The program fetches public funding information as data points and models a predictor for future trends in research. As a result, UResearcher provides invaluable assistance for researchers by reducing menial searching and providing more time for other endeavors.

Therefore, UResearcher is an invaluable application for researchers since the included features reduce unnecessary searching while providing worthwhile outlets. Organizing topics, finding missing information, and predicting are invaluable assets for any researcher, so UResearcher is an essential tool for any research endeavor.

Background & Technical Requirements

Idea Space

The advent of the internet has made accessing academic research easier than ever. From nearly any location in the world, individuals may find research information without a loss in time or quality. In addition, research is no longer confined to isolated singular locations and institutions, so prospective researchers do not need excessive levels of clearance to access papers. However, because of the sheer volume of information available, the process of going through academic research has become even more time consuming and complex. Furthermore, with new research breakthroughs, it is no longer feasible for experts and academics to keep up with the developments in his or her respective fields. This issue also creates problems with redundant information and overlapping research. So, searching the internet for specific information regarding research becomes an arduous task fraught with pitfalls.

Currently, there are several tools that aim to help with online academic research. A few notable programs are Scholarcy, the Science Research Assistant, Mendeley, and Zotero. Scholarcy focuses on summarization and automatic highlighting of key points in an article and functions on an article to article basis. The application primarily expedites sifting through a single article and does not address the problems with searching and filtering. The Science Research Assistant aids with searching by analyzing multiple publishers at once and allows for search customization based on keywords provided by the user. Nevertheless, this application does not address redundant information and only facilitates search in a more organized manner. Mendeley, an organization and social media tool, makes citations and bibliographies easier, while offering the ability to connect with other researchers. Similarly, Zotero offers a way to save, cite, and organize research. However, both Mendeley and Zotero lack features for recommendations.

All of these programs offer features that help with one specific area of online research, but they fail to address the larger problems that make online research difficult. UResearcher addresses these issues and aims to help with multiple aspects of research. By focusing on presenting relevant information in an organized fashion, the program helps with both the scope of the search and the issues of redundant information. It also offers other analytical tools such as viewing trends in grant allocation, and keyword trends within an area of research. Consequently, UResearcher fills an essential niche in the research field by offering more useful features and providing necessary insight.

Technology

The final project contains aspects from several frameworks, libraries, and technologies. Such technologies are quite modular and well-defined, consequently, the structure of UResearcher has several components pieced together. Despite this pre-defined foundation, UResearcher uses at least one algorithm built from scratch.

UResearcher implements three key features: keyword searching, unsupervised clustering, and latent knowledge analysis. To perform searching, the program will use a keyword search algorithm to gather the initial set of articles. The application then performs unsupervised clustering to organize research articles for the user. Finally, the program performs Word2Vec using the Gensim library which will perform latent knowledge analysis within the application.

Another feature is the analysis of past grant money allocation to predict the future of such allocations. In this particular area, UResearcher uses probability and statistics to extrapolate possible graphs. Furthermore, the program accesses the grants.gov API to gather raw information regarding grants and allocations, displays that information in a friendly manner to the user with charts and graphs, and makes predictions about possible outcomes. In addition, the simplicity of this feature allows for a quick and easy implementation.

In the front end, the application uses HTML, CSS, JS/JQuery, and Bootstrap. Aside from the first few technologies, which are nearly essential, Bootstrap aids in rapid prototyping for a quick and clean implementation. Furthermore, the Bootstrap library helps provide a consistent look and feel without contracting any graphic designers.

On the backend, UResearcher uses algorithmic implementations in Python 3, so Django, a python-based web framework, serves as the standard. The long lasting and public nature of Django also yields plenty of supporting middleware packages for development. Django also supplies an MVC architecture, an important aspect to the application. Furthermore, Flask interacts with the various APIs and provides a very lightweight framework for handling requests. It is also used to talk to the University of Utah's Center for High Powered Computing, where our most intensive operations will be performed. Because SQLite is the default for Django and it is a python library, UResearcher uses SQLite for its minimal storage needs.

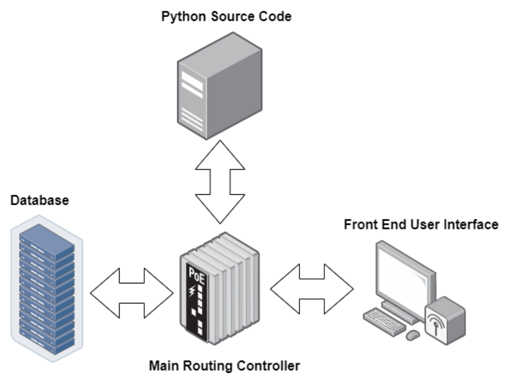
Hardware/Software Requirements

Because our product is delivered as a website there are virtually no hardware or software requirements. Because we will be using Bootstrap 4.3 we will be targeting, at the oldest, Chrome 45, Firefox 38, Edge 12, iOS 9, and Android 4.4, though these requirements should be very easily met. Even though this is a product rife with complex operations they will be performed primarily on the backend, so there is no high end hardware required for the user.

Requirement Analysis

System Architecture

The system architecture is split into four main parts: the front end, the main networking controller, a database, and the data analysis code.



Front End

- This component coordinates the direct interaction between the user and functionality of the application. The user makes various synchronous and asynchronous calls to the networking controller through this layer, and the networking controller sends appropriate responses to the front end.

Networking Controller

- This portion of the application handles all of the incoming and outgoing requests. It also makes calls to the database and to the python analysis code to gather data to serve to the user.
    - Centralized Server - Handles connections of users to the application and communication between the database and the analysis code.
    - Application Center - Hooks into every other portion of the application to gather and serve data.

Data Manipulation

- This layer is the collection of python algorithms for searching and various types of analysis:
    - Searching - Gathers applicable articles from the database based on the user's search.
    - Clustering - Clusters articles from a search into categories based on unsupervised clustering of their abstracts and keyword rank.

- Latent Knowledge Analysis - Analyzes articles from the search space to find relevant articles this will extract the knowledge about the topic that we have not yet harnessed.
- Trend Analysis - Scans the grant repository and search history to determine trends in funding and keywords.

Database

- A simply-architected MySQL database that stores all articles to serve to users. Also stores user account information.

Personnel

- **Jack Zhao - (Article Search, Developer API)**
  - Jack has experience in web software architecture and visualization, and he has an extensive background with full stack development. His knowledge of algorithms also helps with the development of efficient searches and APIs.
- **Ritesh Sharma - (Clustering, Article Search)**
  - Ritesh Sharma has backgrounds in NLP and Data Mining. Knowledge from these areas are essentials for clustering, a keyword-based search of subtopics and similarities in NLP.
- **Will Frank - (Front End)**
  - Will, having taken web software architecture and done some web development for an internship, has the most experience with front end development.
- **Christopher Liu - (Grant Analysis, Network Controller)**
  - Christopher Liu has a sturdy background in machine learning, statistics, mathematics; consequently, he monitors and designs grant analysis features. Furthermore, his knowledge of networking and the model-view-controller design makes him qualified for constructing the network controller.
- **Michael Tata - (Latent Knowledge Analysis, Keyword Analysis)**
  - Michael Tata has a background in machine learning and data analysis and has experience working with multiple machine learning libraries. Therefore, he designs the model for training latent knowledge analysis.

System Features

-Base

- **Searching -** Perform a search and get scholarly article results related to searched keywords.
- **Clustering -** With some search results, perform basic clustering, which would filter results into keyword-based clusters.
- **Latent Knowledge Analysis -** Given the resulting articles found through a search or clustering, encode their contents as word embeddings and provide four ways to view the data: a 2D tSNE projection of the vectors, a list of the most similar words to a target word, a diagram of connections between a base word and some other words, and an 'analogy generator' to do math on the underlying embeddings.

-Planned

- **Citation Analysis -** View what articles have been cited in an article and observe how many times those articles have been cited.
- **Grant Information -** For a specific keyword or search result, view information related to grants on that keyword.
- **Keyword Analysis And Trends -** For keywords found in a search, view the history and frequency of occurrence for found keywords.
- **Loading Icon -** Provide a loading icon to indicate that analysis is being performed because of the time consuming nature of some of these algorithms.
- **Help Page -** Provide a basic help page which describes functionality and instructions for features.
- **Feedback Page -** Allow users to submit feedback on the application.

- **Developer API -** Expose the ability to get search results or the resulting data of various analysis algorithms from API calls.

-Advanced

- **Config File -** A replacement to user accounts, allow users to upload their own configuration files containing their API keys to private/paid databases.
- **Advanced Search -** Similar to other database searches, allow users to filter results by aspects such as the publisher or publication date.
- **Progress bar -** Display the actual progress of certain analysis algorithms (as opposed to just a loading icon or spinner).
- **User Accounts -** Store user accounts so users can upload their API keys to paid databases or change the site theme.
- **In-app back button -** Effectively turn the site into a single-page application with a non-browser back button.
- **Saving** - Saving of search results and the underlying data of the various analysis functions.
- **Fully Interactive Graphics** - Implement graphics that let the user mouse over, click, and generally interact with them to better expose the data.
- **Animations** - Rather than just replacing content on the web page as the user requests it, show an animation to smooth the transition.

Development Timeline

-Alpha (January 6, 2020 - February 10, 2020)

- Frank, Will:
  - Build out the file structure for front end and back end
  - Develop the front end for searching, search results, clustering, and latent knowledge analysis
  - Hook up synchronous and asynchronous calls to serve data in combination with front end code
  - Begin database integration for storing data efficiently

- Liu, Christopher:
  - Develop a program that graphs and displays grant information into a digestible format.
  - Create methods and processes to download or cache grant information for use.
  - Implement a basic client-server model for network communication.

- Sharma, Ritesh:
  - Develop an algorithm that cluster the research papers based on abstract using NLP libraries.
  - Implement a function to train the model to extract more meaningful clusters.
  - Implement and algorithm that summarizes the given label based on all the articles, this will be used if the summary for the word cannot be extracted from wikipedia.

- Tata, Micheal:
  - Implement keyword gathering and basic graphing of keywords for keyword analysis
  - Assist with the implementation of grant analysis and hooking up with the Grant.gov API, and utilizing the HHS TAGGS data for older grant information
  - Rework Latent Knowledge Analysis layout to make it more intuitive for the user, and come up with a tutorial for its use in the help page.

- Zhao, Jack:
  - Build out the file structure for front end and back end
  - Develop the crawler for Elsevier.com
  - Begin implementation for the API document

-Beta (February 10, 2020 - March 16, 2020)

- Frank, Will:
  - Develop and connect front end for grant analysis and keyword trend analysis

  - Integrate user accounts into the user interface
  - Begin migration to ElasticSearch database for articles

- Liu, Christopher:
  - Find the ideal graphing methods and display them to them properly on the application.
  - Expedite the process for gathering grant information to ensure appropriate response times.
  - Implement a structured multi-client server to handle several clients concurrently.

- Sharma, Ritesh:
  - Make the clusters more accurate by using several other Machine learning methods.
  - Summarize the articles to a meaningful summary so that it can be used whenever the summary is not present in Wikipedia.
  - Rank the search result so that articles with the highest weight are displayed first.

- Tata, Micheal:
  - Refine Latent Knowledge Analysis training and implement pre-processing (Removing pronouns and names, finding optimal word count for inclusion, etc…)
  - Expand on keyword gathering and implement a keyword frequency timeline for plotting
  - Implement citation information for resulting search articles

- Zhao, Jack:
  - Improve UI for the web application
  - Develop the script to automate the installation process for both Windows and Unix-like system
  - Improve the configuration functions for the application

-Production (March 16, 2020 - April 21, 2020)

- Frank, Will:
  - Polish and refine existing UI/UX with advanced features such as progress icons
  - Finalize database migration
  - Publish the website

- Liu, Christopher:
  - Refine the design of the provided graphs into simpler and more pleasing displays.
  - Provide a robust server, which can handle multiple calls at an ideal rate while avoiding interference issues.

- Sharma, Ritash:
  - Polish and test the clustering and ranking algorithms

    - Fix bugs if required.

- Tata, Micheal:
    - Further LKA model training optimization
    - Polish & Bug fixes

- Zhao, Jack:
    - Bug fixes
    - Improve Dev Docs

Appendix

Use-Cases

For User

Number	1
Title	Searching for a Paper or Subject
Description	A user wishes to investigate or find a specific area of research papers on the website.
Steps	1. Type a query into the search bear and click “Search.” 2. Optionally use a refined search or cluster to dive deeper into the website. 3. Choose paper from the results .
Related UI	Figure 1, Figure 2, Figure 3, Figure 4

Number	2
Title	Reverting Searches and Starting a New Search
Description	A user desires to backtrack to a previous result page or start a search for a new subject space.
Steps	1. Continue to click the back button that is provided in your web-browser until you get to your desired page. 2. Stop reverting after reaching the home page. 3. Use a new search box contained in each page to search for new search result.
Related UI	Figure 1, Figure 2, Figure 3

Number	3
Title	Doing Latent Knowledge Analysis
Description	After doing a search, the user requests latent knowledge analysis in the results.
Steps	1. Perform a search. 2. Optionally refine the search results. 3. Select to perform latent knowledge analysis on the results. 4. View latent knowledge analysis.
Related UI	Figure 2, Figure 12

Number	4
Title	Analyzing Grant Trends
Description	After performing a search, the user wishes to analyze grant trends.
Steps	1. Perform a search. 2. Optionally refine the search results. 3. Select to perform grant trend analysis on the results. 4. View the grant trend analysis graph.
Related UI	Figure 6

Number	5
Title	Finding No Articles in an Area.
Description	The user finds little to no articles with a search request.
Steps	1. Perform a search. 2. Potentially refine the search further. 3. Arrive at the “No Articles Found” page once the server reaches the minimum number of pages provided in a search. 4. If there are any results, explore the provided pages, or search or revert from that point.

Related UI	Figure 5

Number	6
Title	Saving Search Results
Description	The user has performed a search, potentially refined it, and wants to save a list of all the articles.
Steps	1. Perform a search 2. Click the Save Search button.
Related UI	Figure 2 - Figure 12

Number	7
Title	Error Messaging
Description	The user encounters an unforeseen error in the application.
Steps	1. Encounter the error in the application. 2. Optionally reply to a feedback system. 3. Revert or restart a search.
Related UI	Figure 11

Number	8
Title	Help, Feedback, and Documentation
Description	The user desires to learn more about the application and provide input to the developers.
Steps	1. Click on the “Help” icon on the application. 2. Choose “Help” to view directions for running the application. 3. Choose “Feedback” to send feedback to the developers. 4. Choose “Documentation” to view the documentation documents. 5. Revert or delete to exit these pages.
Related UI	Figure 1 - Figure 13 (Figure 7: Help Page, Figure 8: Feedback Page, Figure 10: Documentation Page)

Number	9
Title	Advance Search
Description	A user can add more specific filters for their search (like publish date, author, etc.)
Steps	1. From any page, click the “Advanced Search” button. 2. Fill in any desired fields and click “Search.”
Related UI	Figure 13

Developer Use Cases:

Number	1
Title	Search Results API
Description	We provide APIs to get search results as JSON rather than HTML, as well as plaintext of specific articles.
Steps	1. Send a GET request to the API using the correct parameters
Related UI	Example API Call: http://uresearcher.com/api/search/your-search-query

Number	2
Title	Configuration file
Description	Configuration file (in .yaml format) 1. Add key for other publishers 2. Add preset for advance search
Steps	1. Download the template configuration file. 2. Make any necessary changes.

	3. Upload the file to apply your changes.
Related UI	ELS_API_KEY: some_keys_here PRESET_SEARCH: some_keyword_here

Figures

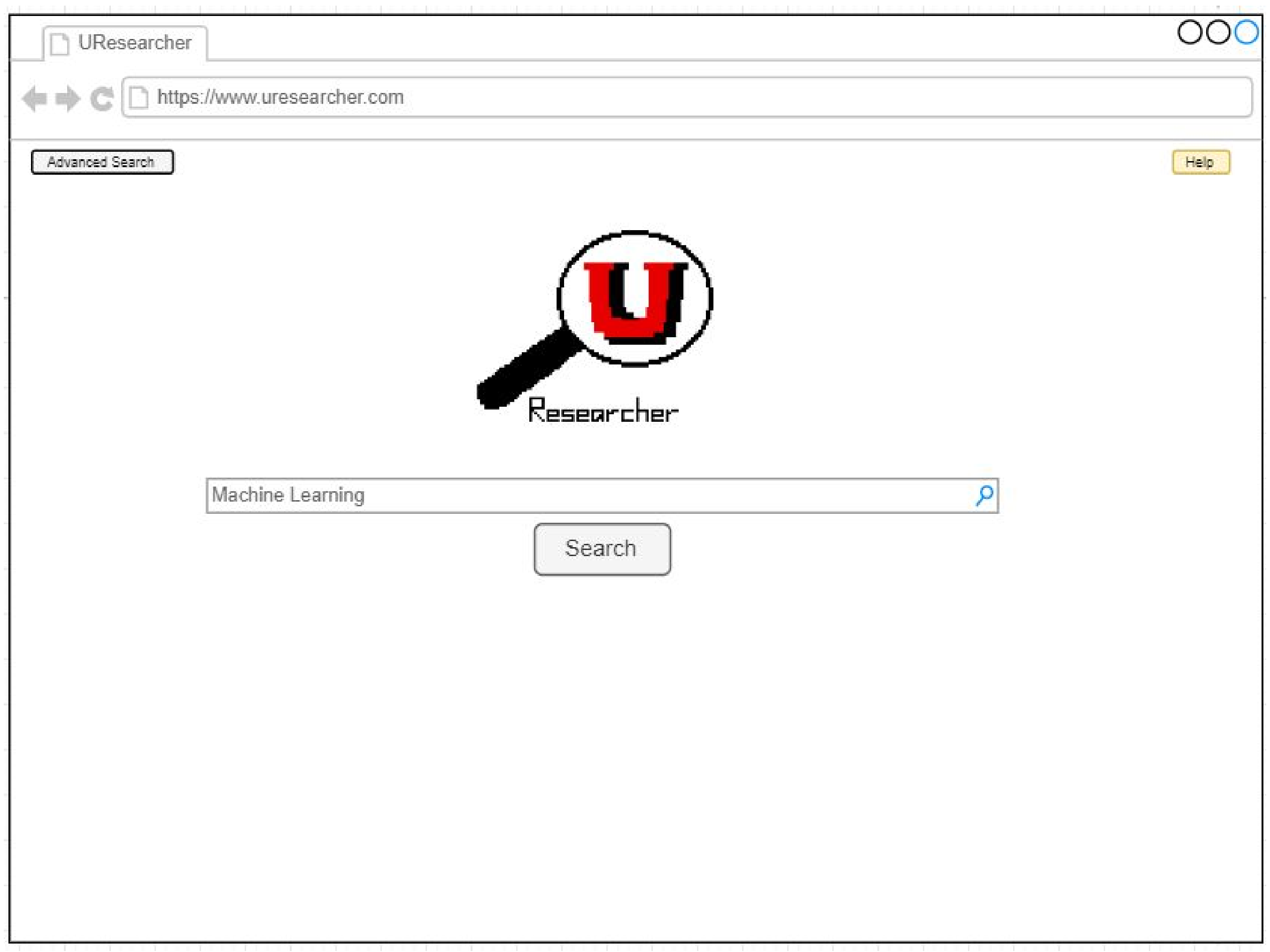
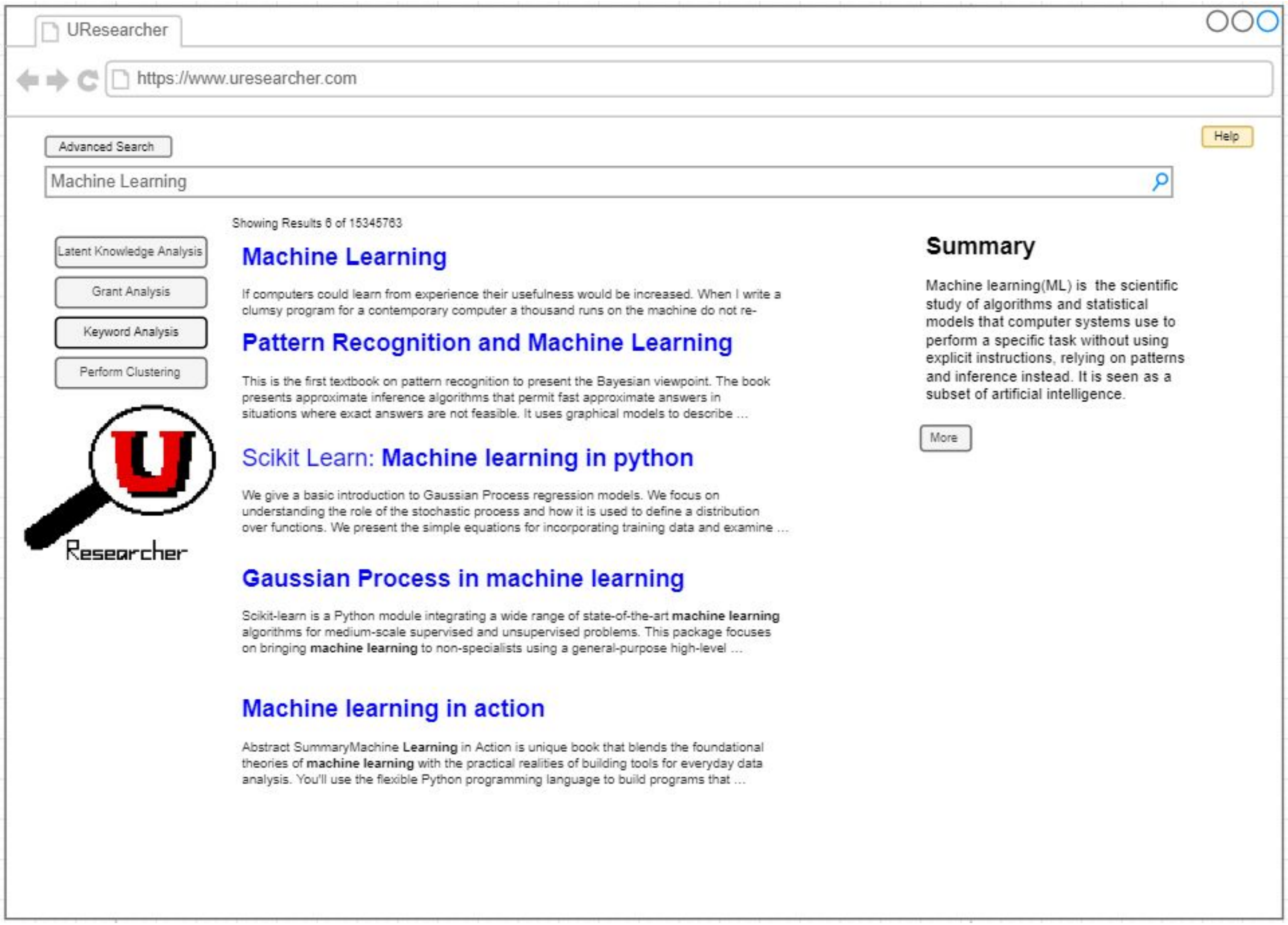


Figure 1 - Home Page



Showing Results 6 of 15345763

Machine Learning

If computers could learn from experience their usefulness would be increased. When I write a clumsy program for a contemporary computer a thousand runs on the machine do not re-

Pattern Recognition and Machine Learning

This is the first textbook on pattern recognition to present the Bayesian viewpoint. The book presents approximate inference algorithms that permit fast approximate answers in situations where exact answers are not feasible. It uses graphical models to describe …

Scikit Learn: **Machine learning in python**

We give a basic introduction to Gaussian Process regression models. We focus on understanding the role of the stochastic process and how it is used to define a distribution over functions. We present the simple equations for incorporating training data and examine …

Gaussian Process in machine learning

Scikit-learn is a Python module integrating a wide range of state-of-the-art **machine learning** algorithms for medium-scale supervised and unsupervised problems. This package focuses on bringing **machine learning** to non-specialists using a general-purpose high-level …

Machine learning in action

Abstract SummaryMachine **Learning** in Action is unique book that blends the foundational theories of **machine learning** with the practical realities of building tools for everyday data analysis. You'll use the flexible Python programming language to build programs that …

Summary

Machine learning(ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence.

Figure 2 - Results Page

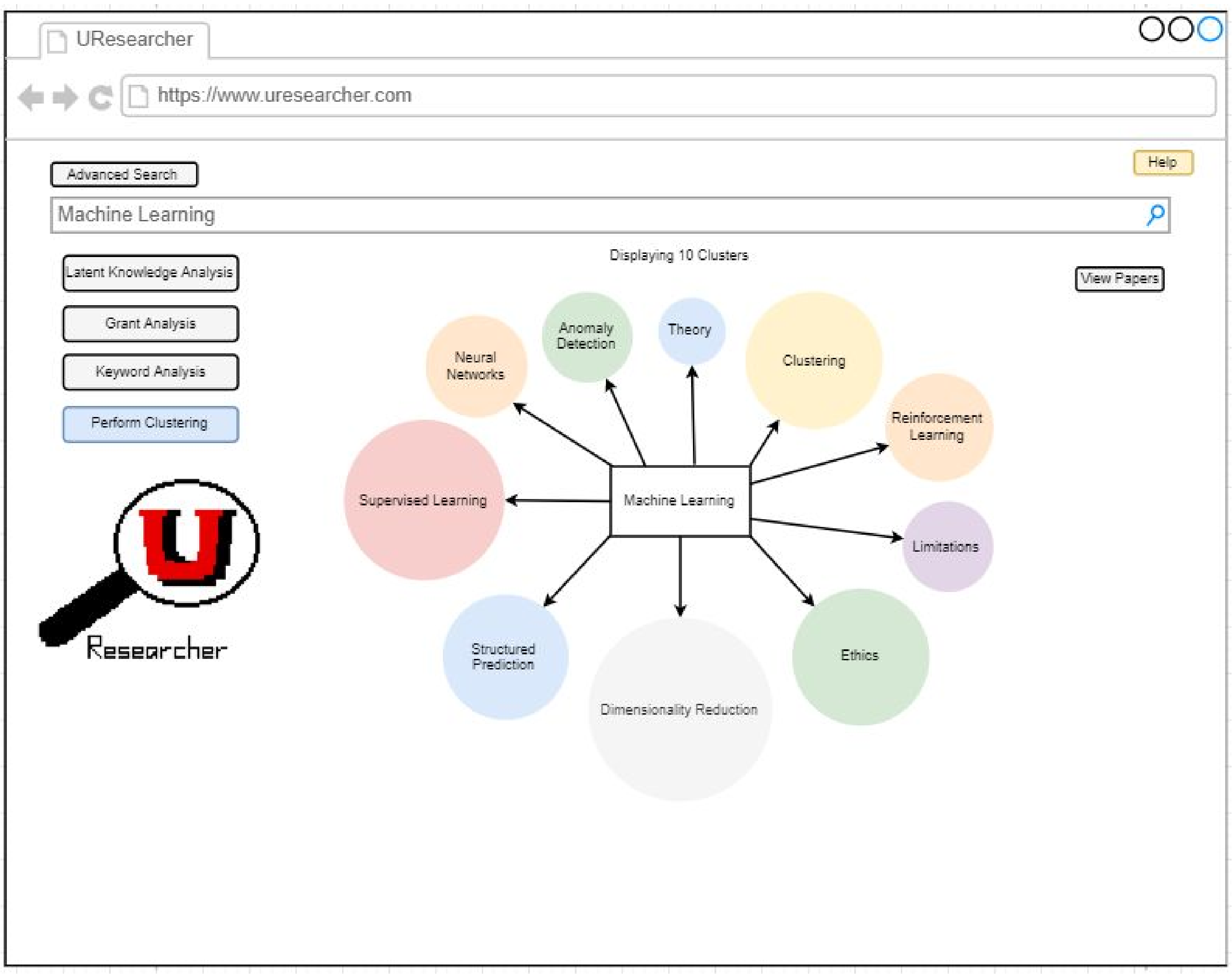


Figure 3 - Clustering Page

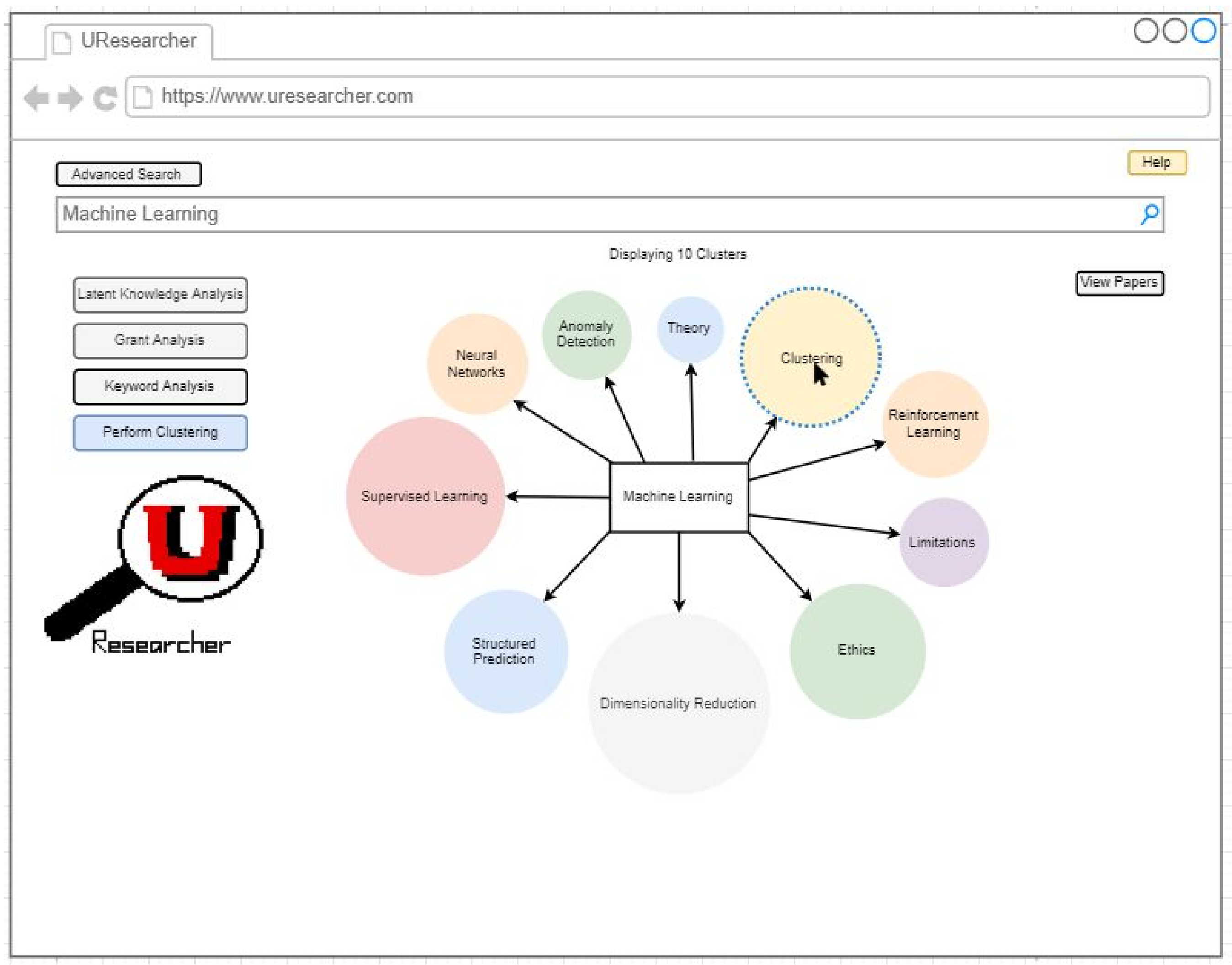
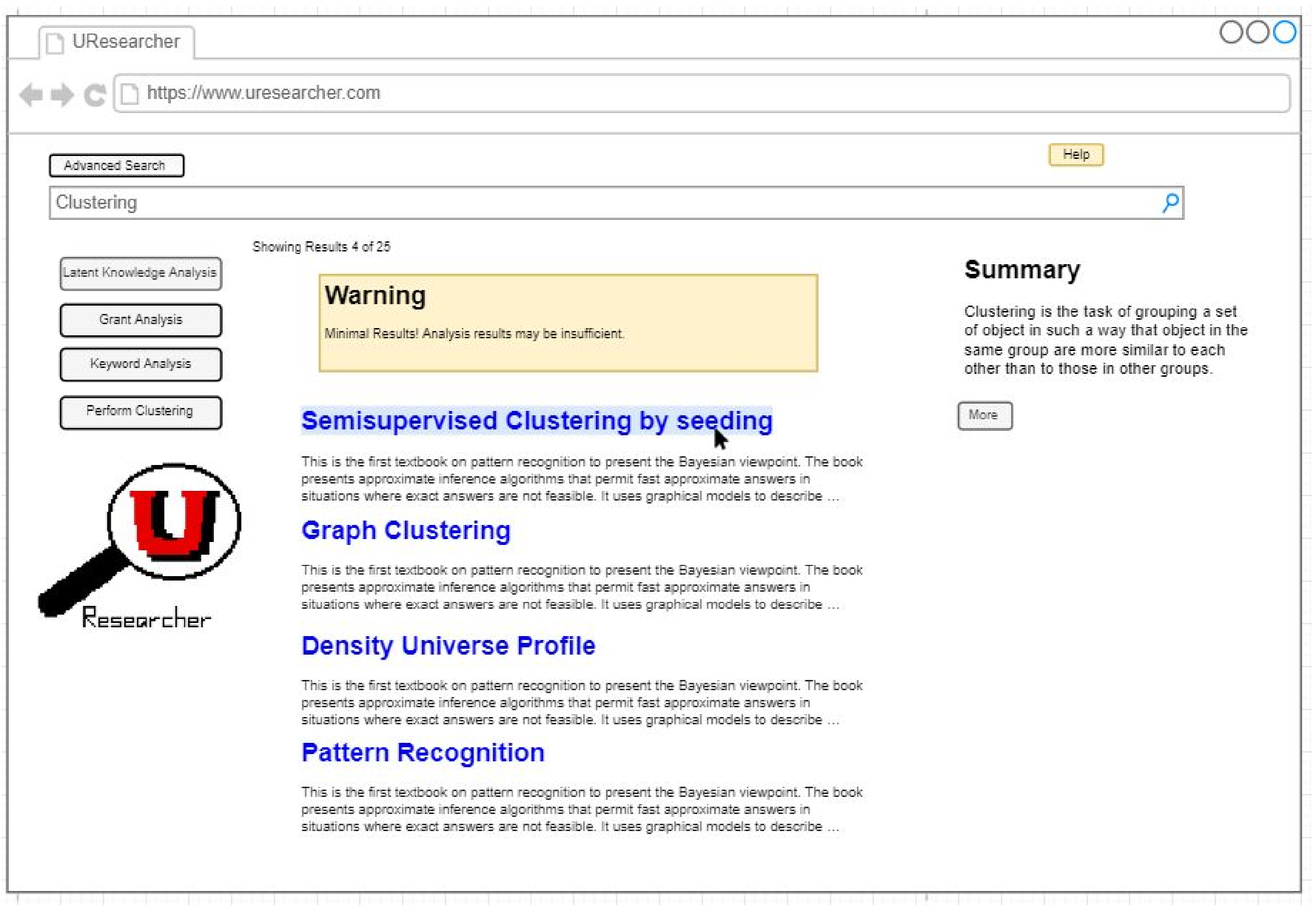


Figure 4 - Select Cluster Page



Showing Results 4 of 25

Warning

Minimal Results! Analysis results may be insufficient.

Semisupervised Clustering by seeding

This is the first textbook on pattern recognition to present the Bayesian viewpoint. The book presents approximate inference algorithms that permit fast approximate answers in situations where exact answers are not feasible. It uses graphical models to describe ...

Graph Clustering

This is the first textbook on pattern recognition to present the Bayesian viewpoint. The book presents approximate inference algorithms that permit fast approximate answers in situations where exact answers are not feasible. It uses graphical models to describe ...

Density Universe Profile

This is the first textbook on pattern recognition to present the Bayesian viewpoint. The book presents approximate inference algorithms that permit fast approximate answers in situations where exact answers are not feasible. It uses graphical models to describe ...

Pattern Recognition

This is the first textbook on pattern recognition to present the Bayesian viewpoint. The book presents approximate inference algorithms that permit fast approximate answers in situations where exact answers are not feasible. It uses graphical models to describe ...

Summary

Clustering is the task of grouping a set of object in such a way that object in the same group are more similar to each other than to those in other groups.

Figure 5 - End of Search Page

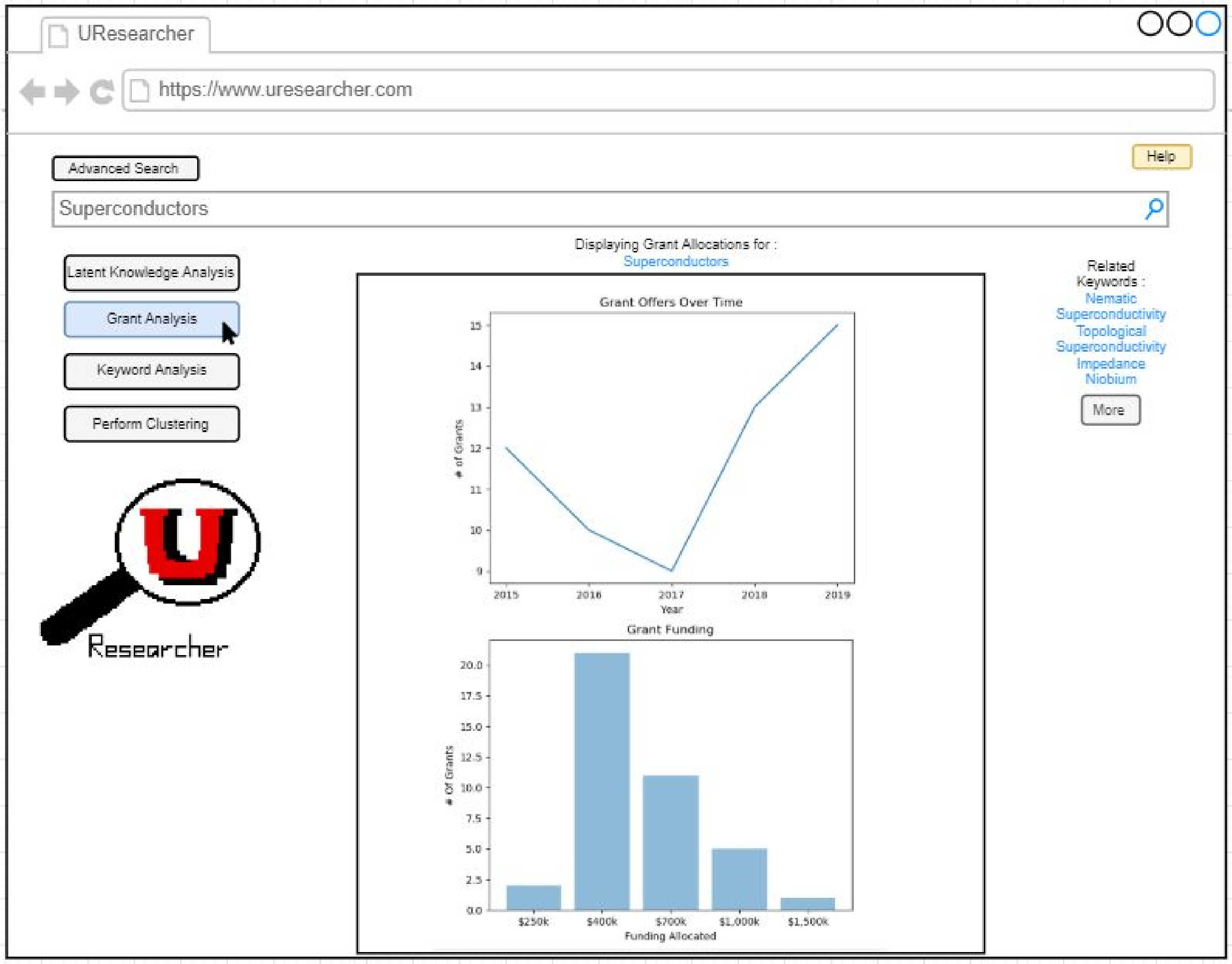


Figure 6 - Grant Analysis Page

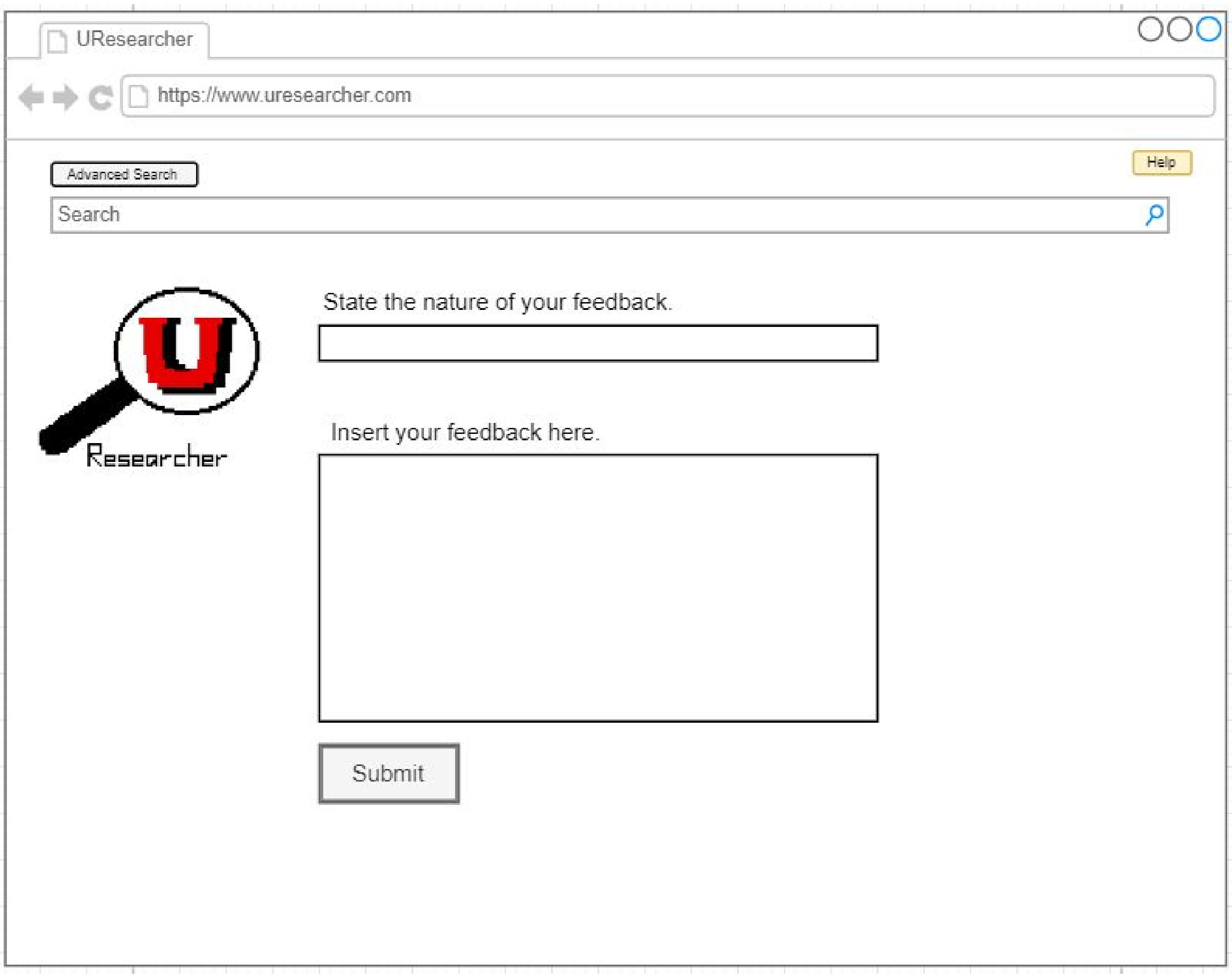
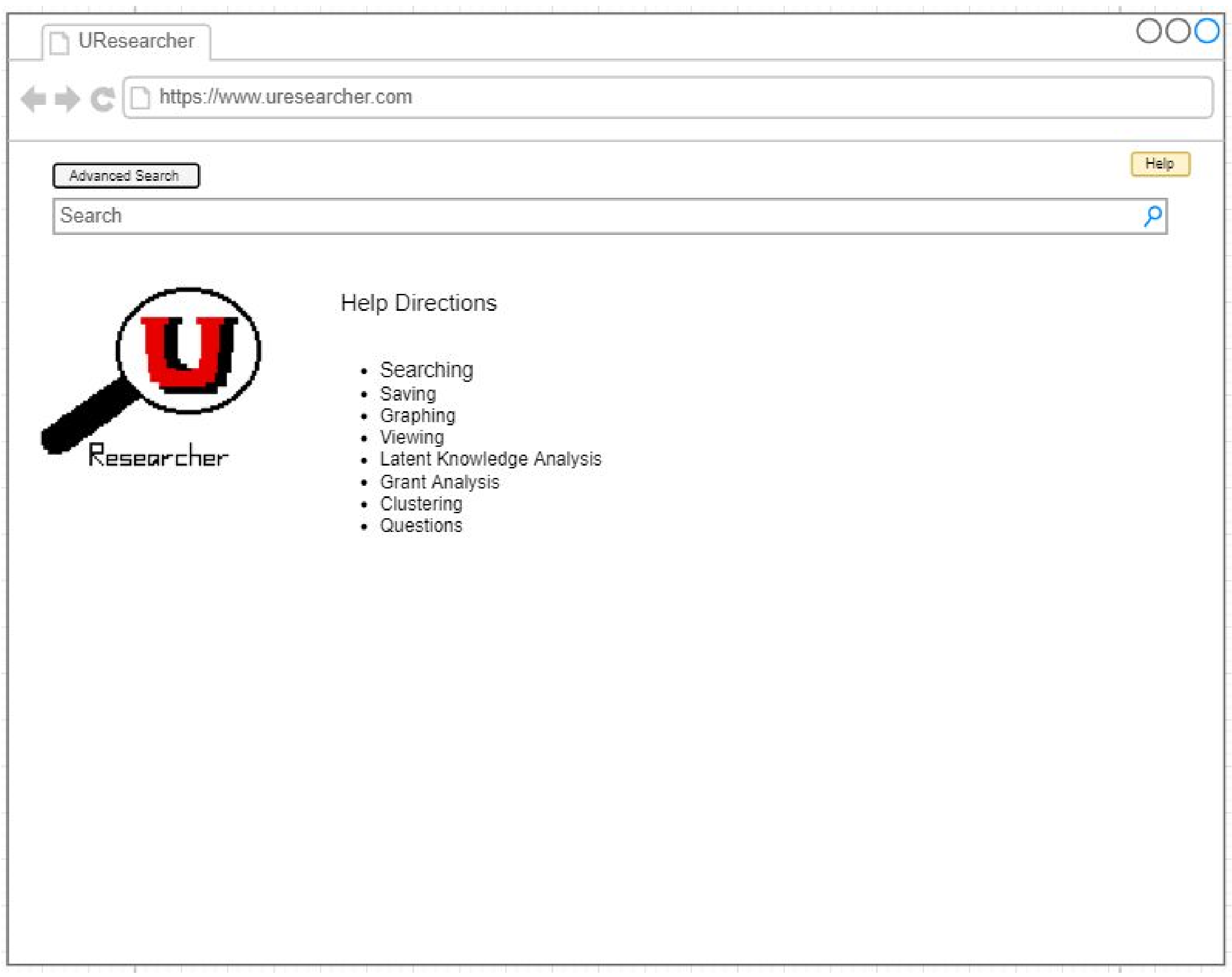

UResearcher
https://www.uresearcher.com
Advanced Search
Help
Search
Researcher
State the nature of your feedback.
Insert your feedback here.
Submit

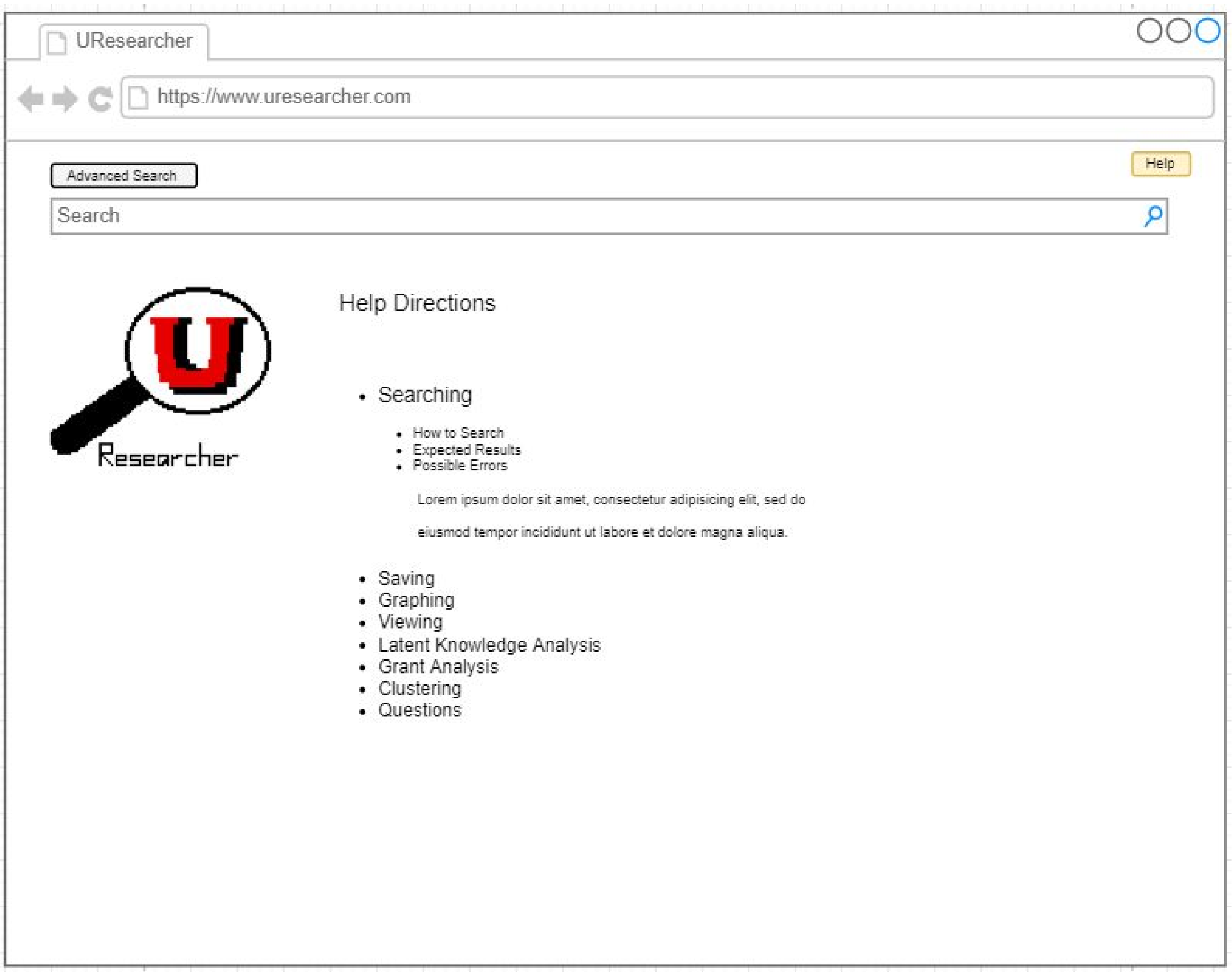
Figure 7 - Feedback Page



Help Directions

- Searching
- Saving
- Graphing
- Viewing
- Latent Knowledge Analysis
- Grant Analysis
- Clustering
- Questions

Figure 8 - Help Page



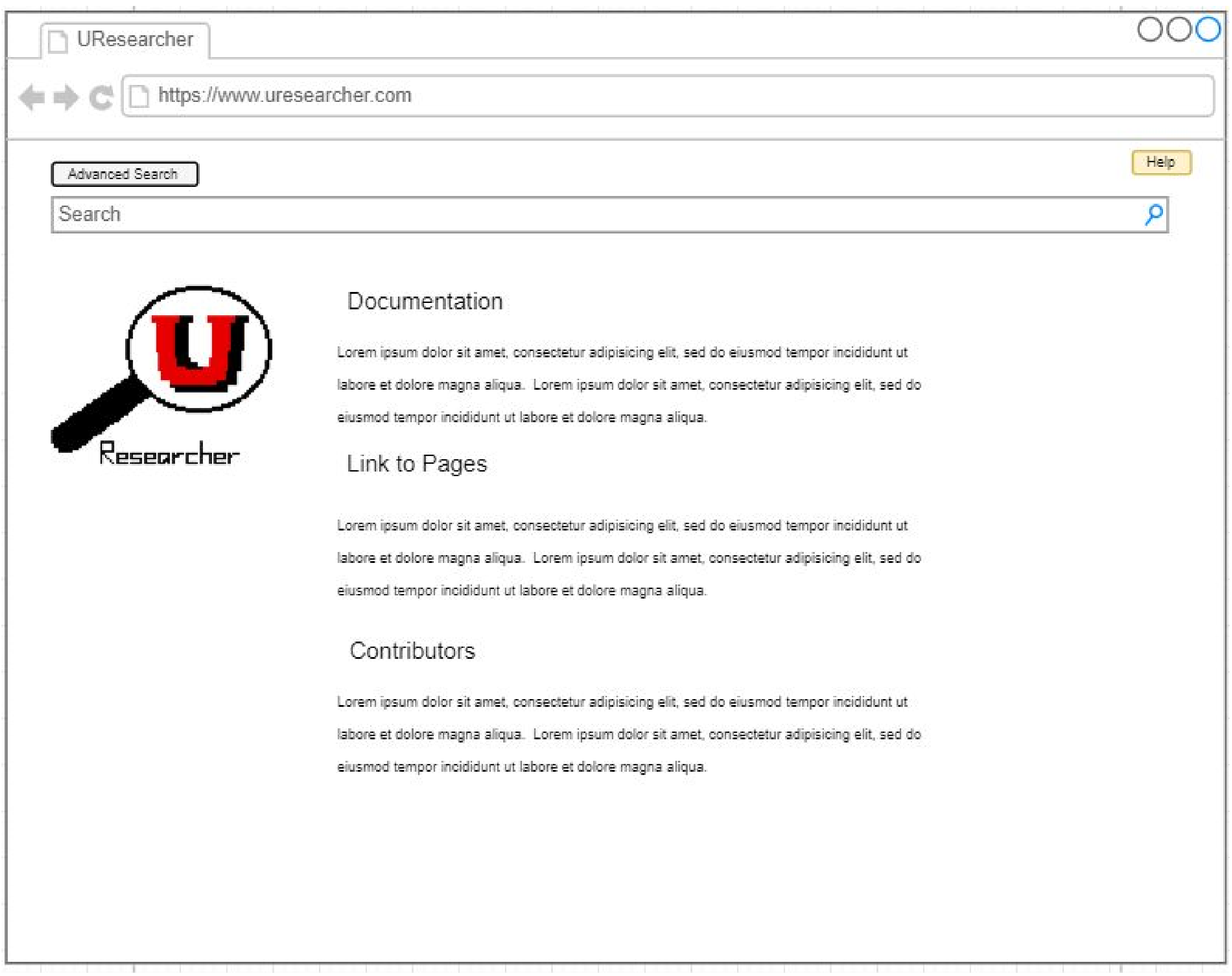
Help Directions

- Searching
  - How to Search
  - Expected Results
  - Possible Errors

  Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do

  eiusmod tempor incididunt ut labore et dolore magna aliqua.
- Saving
- Graphing
- Viewing
- Latent Knowledge Analysis
- Grant Analysis
- Clustering
- Questions

Figure 9 - Help Page (Extended)



Documentation

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

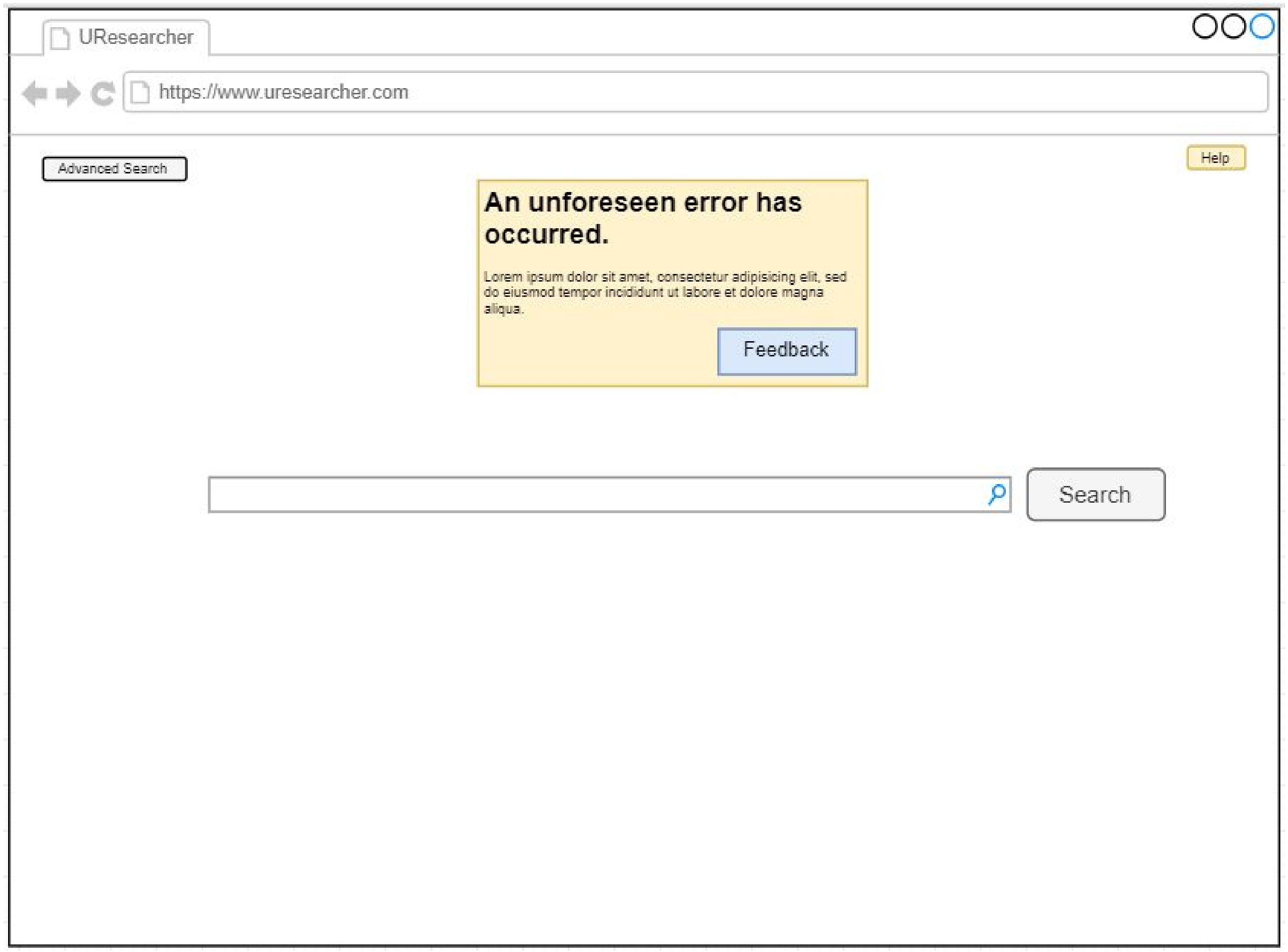
Link to Pages

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Contributors

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Figure 10 - Documentation Page



An unforeseen error has occurred.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Figure 11 - Error Page

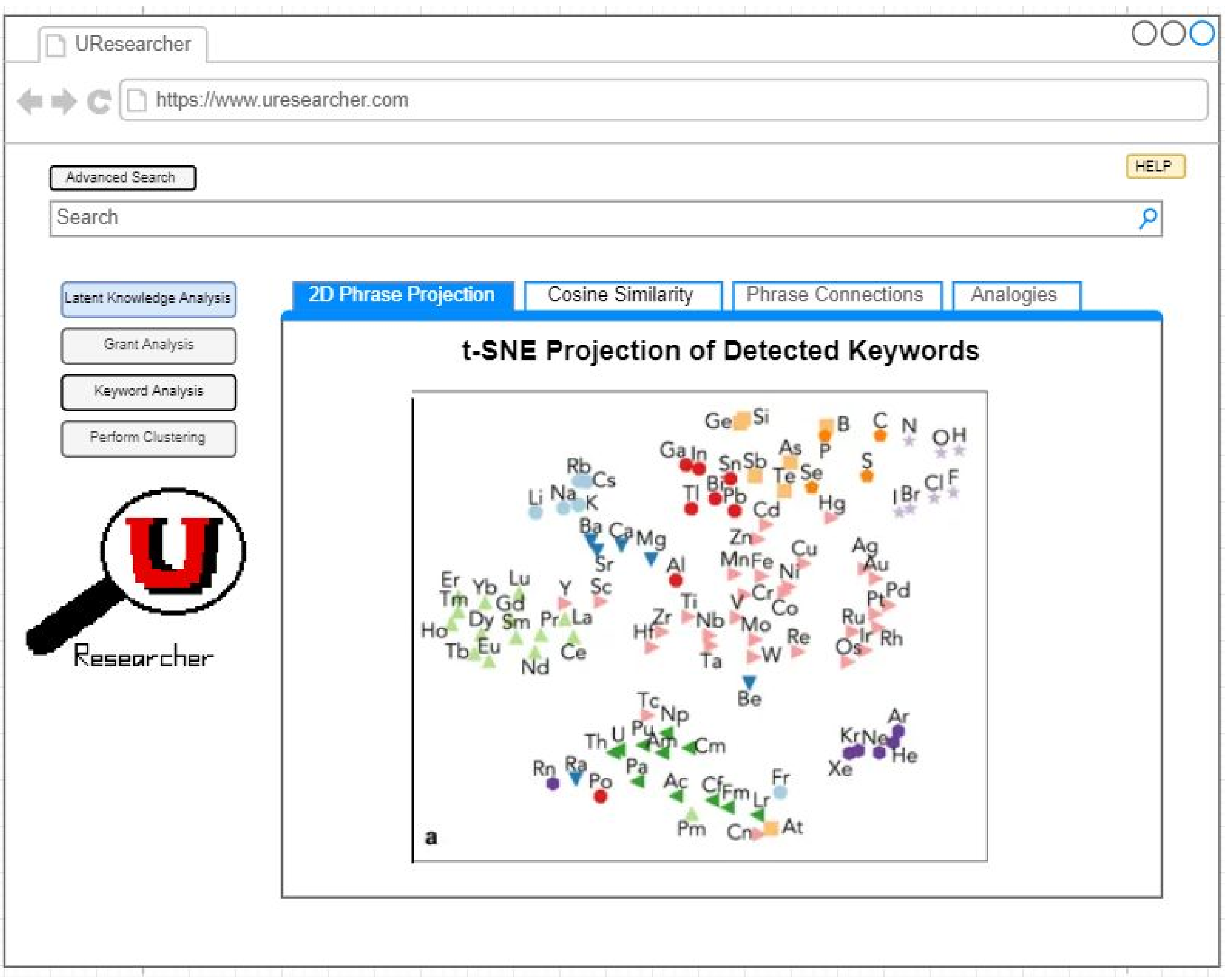


Figure 12 - Knowledge Analysis Page

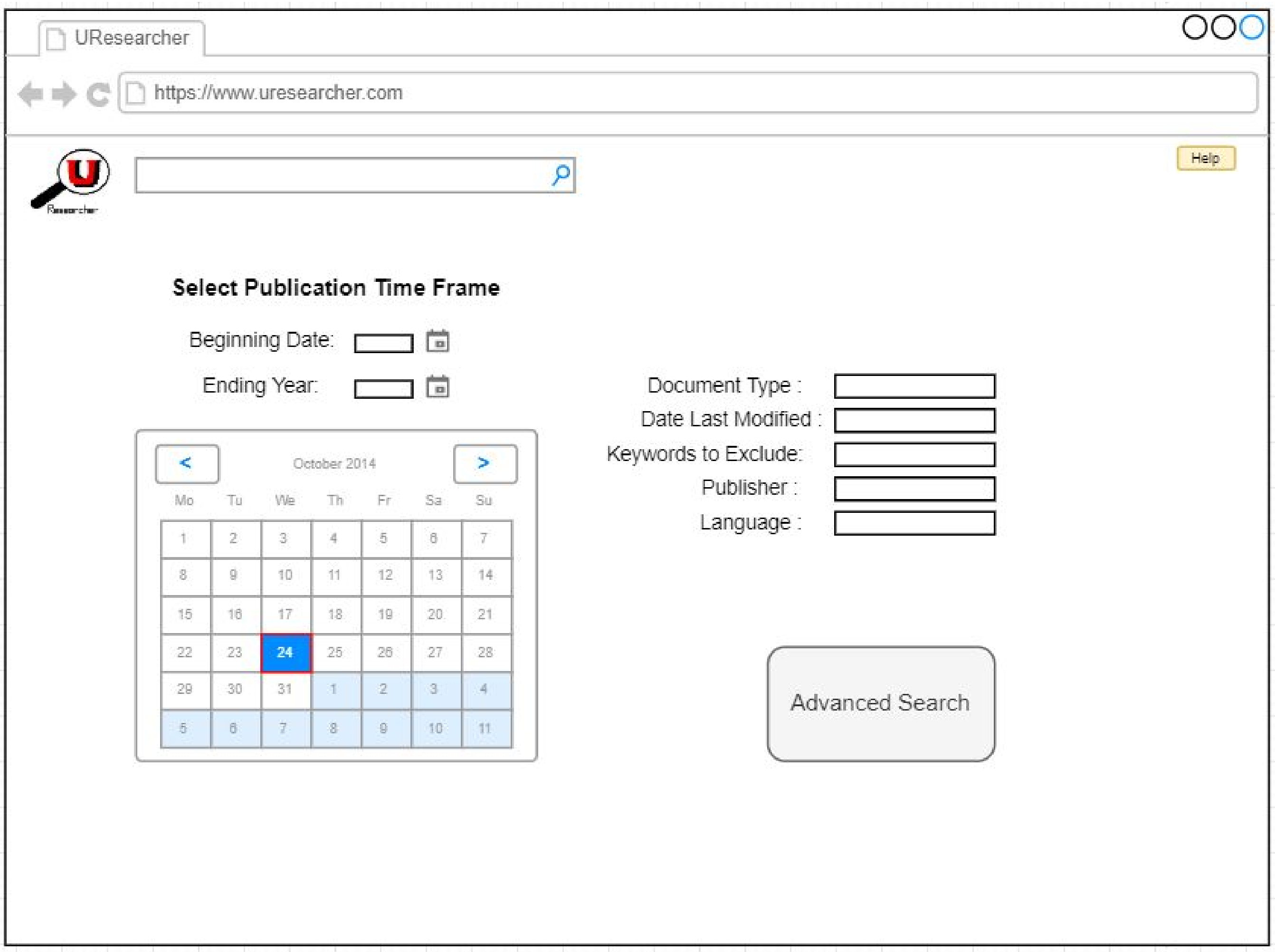


Figure 13 - Advanced Search