DSAI 103 Project
Network Citation Search

Amr Yasser - 202301043

Introduction:

This report presents a detailed explanation of a web application developed for searching and visualizing citation networks of research papers. The application is built using FastAPI, a modern, fast (high-performance), web framework for building APIs with Python 3.6+ based on standard Python type hints. The application utilizes Google Scholar's API through SerpAPI for fetching citation data and NetworkX for graph creation and visualization.

Objective:

The main objective of this application is to allow users to search for research papers and visualize their citation networks. The application extracts the top 20 cited papers related to the search query, constructs a graph representing the citation relationships, and generates visual images of these graphs.

Technology Stack:

1. **FastAPI**: For building the web application and handling HTTP requests.
2. **Jinja2**: For rendering HTML templates.
3. **SerpAPI**: To interact with Google Scholar and fetch citation data.
4. **NetworkX**: For creating and manipulating the citation graph.
5. **Matplotlib**: For visualizing the citation graph.
6. **uvicorn**: ASGI server for serving the FastAPI application.
7. **HTML/CSS**: For the frontend interface.

Application Structure:

1. **main.py**: The main backend script that sets up the FastAPI app, handles search requests, processes citation data, and generates citation graphs.
2. **index.html**: The homepage template where users can input their search query.

3. **results.html**: The results page template that displays the citation graphs and related papers.
4. **style.css**: Stylesheet for the frontend templates.

Code Explanation:

main.py:

- **Imports**: Essential libraries and modules are imported for the application.
- **FastAPI Setup**: The FastAPI app is initialized, and static files and templates are configured.
- **Utility Functions**:
    - `clean_title(raw_title)`: Cleans up the title strings by removing leading brackets.
    - `extract_citation_info(search_results)`: Extracts titles and citation counts from the search results.
    - `create_citation_graph(citations)`: Constructs a NetworkX graph from the citation data.
    - `generate_graph_image(citation_graph, start, stop)`: Generates a visual image of the citation graph for a subset of papers.
- **Endpoints**:
    - `@app.get("/")`: Serves the homepage.
    - `@app.post("/search")`: Handles the search form submission, interacts with the SerpAPI, processes the results, and renders the results page.

index.html:

- Contains a form for users to input the research paper title they want to search for.

results.html:

- Displays the citation graphs and lists the papers with their citation counts.
- Uses Jinja2 templating to dynamically insert the search results and images.

style.css:

- Defines styles for the HTML elements to ensure a clean and user-friendly interface.

Workflow

1. **User Interaction**:
    - The user visits the homepage (`index.html`), inputs a research paper title, and submits the form.

2. **Search Request Handling**:
    - The form submission triggers the `/search` endpoint, which performs the following steps:
        - Uses SerpAPI to fetch the top 20 citation results from Google Scholar.
        - Cleans and processes the search results to extract relevant citation data.
        - Constructs a citation graph using NetworkX.
        - Generates visual images of the citation graph in chunks of 10 papers each.
3. **Results Rendering**:
    - The results, including the citation graph images and paper details, are passed to the `results.html` template and rendered for the user.

Conclusion:

The developed web application successfully integrates various technologies to provide a functional and interactive tool for visualizing citation networks. By leveraging FastAPI for backend development, SerpAPI for data retrieval, and NetworkX and Matplotlib for graph visualization, the application offers a comprehensive solution for researchers and academics to explore citation relationships efficiently.
