NOVARTIS-NEST

User Manual for Problem Statement-1

Overview

This user manual provides step-by-step instructions to understand and execute the codebase for the clinical trials project. The workflow consists of three main Jupyter Notebooks: Relationship_extraction.ipynb, Merge_similar_nodes.ipynb, and Final_recommended_nodes.ipynb. Each notebook is designed to accomplish specific tasks in the pipeline: relationship extraction, node normalization, and recommendation generation.

1. Relationship Extraction (Relationship_extraction.ipynb):

Objective: Extract structured relationships from raw clinical trial data.

Steps:

1. **Input Data:** Ensure the raw clinical trial data (e.g., clinical_trials.csv) is in the correct directory and contains columns such as Study Title, Primary Outcome Measures, Secondary Outcome Measures, and Criteria.
2. **Merge Columns:** The notebook combines text from these columns into a unified field for each trial.
3. **Run Relationship Extraction:**
    - The notebook uses the Groq LLM API to process each trial and extract relationships in a tab-separated format.
    - Relationships include generic connections such as RELATIONSHIP and their corresponding objects.
4. **Output Files:**
    - relationships.csv: Contains the structured relationships with fields like Subject, Relationship, and Object.
    - Object_Value_Counts2.csv: Includes frequency counts for unique objects.

How to Use:

- Update the input file path in the notebook.
- Run each cell sequentially to process and save the results.

2. Node Normalization (Merge_similar_nodes.ipynb):

Objective: Normalize and deduplicate extracted nodes to ensure efficient representation.

Steps:

1. **Input Data:** Use the output file relationships.csv generated from the previous notebook.
2. **Generate Embeddings:**
    - The notebook uses the Stella 1.5B transformer model to create embeddings for each unique object.
    - Embeddings encode semantic similarity between objects.
3. **Similarity Search:**
    - Uses FAISS to perform a similarity search based on cosine similarity.
    - Identifies and merges nodes with similarity scores above a predefined threshold (e.g., 0.8).
4. **Output Files:**
    - filtered_results_with_similars.csv: Contains the normalized nodes and their similar objects.

How to Use:

- Ensure the embeddings library (FAISS) and transformer model (Stella 1.5B) are installed and configured.
- Update file paths as required and run the notebook sequentially.

3. Recommendation Generation (Final_recommended_nodes.ipynb):

Objective: Generate clinical trial recommendations using a graph-based approach.

Steps:

1. **Input Data:** Use the output file filtered_results_with_similars.csv and load it into Neo4j.
2. **Knowledge Graph Construction:**
    - Run Cypher queries to create SubjectNode, ObjectNode, and RELATIONSHIP edges.
    - Transactions are optimized for concurrent execution.
3. **Graph Analysis:**
    - Project the graph using Neo4j's Graph Data Science (GDS) library.
    - Calculate Jaccard similarity between nodes using gds.nodeSimilarity.stream.
4. **User Input:**
    - Prompt the user to input trial IDs for recommendation generation.
    - Validate trial existence in the graph and display the top 10 similar trials ranked by similarity scores.

How to Use:

- Ensure Neo4j is installed and configured correctly.
- Update database connection details in the notebook.
- Follow the prompts to input trial IDs and view recommendations.

General Notes

- **Dependencies:** Ensure all required Python libraries and tools are installed, including Neo4j, FAISS, and transformer models.
- **Error Handling:** The notebooks include error messages for common issues (e.g., missing files, API connection errors). Would help in for troubleshooting.

Execution Order

1. Run Relationship_extraction.ipynb to preprocess raw data and extract relationships.
2. Use the output to run Merge_similar_nodes.ipynb for node normalization.
3. Finally, execute Final_recommended_nodes.ipynb to generate and view recommendations.