Project Report: Gene-Disease Graph

Repository: https://github.com/ArtigasChristopher/Gene-Disease-Graph
Language: Python
Topic: Biomedical Informatics / Graph-based Data Modeling

1. Introduction

The "Gene-Disease Graph" project presents a structured and programmatic approach to exploring gene-disease associations using graph theory. The project leverages the DisGeNET API to retrieve biomedical relationship data and models these relationships as a graph, allowing for downstream analytical applications. The primary goal of the project is to represent the intricate relationships between genes and diseases as a graph structure to facilitate understanding, discovery, and exploration within biomedical research.

In recent years, the biomedical field has seen a growing need for systems that can integrate large-scale data from multiple sources and offer intelligible representations of complex biological networks. This project serves as a foundational step toward that direction by providing a clean and accessible pipeline for transforming API data into graph models.

2. Project Objectives

The core objectives of the project are:

- To fetch and structure gene-disease association data using the DisGeNET public API.
- To transform this structured data into a graph-based representation, where nodes represent genes and diseases, and edges represent known or inferred associations.
- To create a framework that could be extended to perform network-based analysis, visualization, and predictive modeling.

Through these objectives, the project aims to support biomedical researchers and developers in extracting meaningful insights from vast biomedical data repositories.

3. Background: The DisGeNET Database

DisGeNET is one of the most comprehensive public resources of gene-disease associations, integrating data from curated repositories, GWAS catalogs, animal models, and scientific literature. The API allows researchers to query the associations in a RESTful manner and supports a variety of endpoints for different levels of granularity (gene-centric, disease-centric, score-based filtering, etc.).

By utilizing DisGeNET, this project ensures its foundation lies in a well-maintained and peer-reviewed biomedical knowledge base.

4. Technical Implementation

Programming Language: Python
External APIs: DisGeNET API
Dependencies: requests, JSON handling libraries (potential for NetworkX or Neo4j in extended versions)

Main Workflow:

1. Authentication and API Query

   a. Uses a registered DisGeNET API key to authenticate requests.

   b. Retrieves gene-disease association data in JSON format based on specified parameters.

2. Data Processing and Parsing

   a. Extracts relevant fields such as gene identifiers, disease names, association scores, and publication evidence.

   b. Filters or formats the data into tuples representing graph relationships.

3. Graph Modeling

   a. Constructs a graph where:

      - Nodes = Genes or Diseases

      - Edges = Associations between genes and diseases

   b. Each edge may be enriched with metadata such as confidence scores, source databases, and publication counts.

4. Output and Storage

   a. The resulting graph can be exported into standard formats like CSV, JSON, or GML.

   b. Enables integration with visualization tools (e.g., Cytoscape, Gephi) or analytic frameworks (e.g., NetworkX, PyG).

5. Validation and Statistics Module

The project also includes a powerful graph validation and statistics module that analyzes the structural, semantic, and statistical properties of the generated gene–disease network. This functionality is designed to evaluate the quality of the dataset and its fitness for downstream applications like machine learning or hypothesis generation.

Upon running the validator, a comprehensive summary like the following is generated:

a. Graph Overview: Number of nodes (e.g., 135) and edges (e.g., 184), raw associations loaded (e.g., 241).

b. Association Quality Metrics:

- Score range, mean score, evidence index
- Mean PMID count (e.g., 173.9)
- High-confidence associations (score ≥ 0.75)

c. Network Topology:

- Density, clustering coefficient, degree distribution
- Connected components, average path length, diameter
- Scale-free network test (R^2 value)

d. Gene–Disease Specificity:

- Avg. diseases per gene and genes per disease
- Top multi-gene diseases (e.g., Alzheimer's)

e. Precision–Recall Classification:

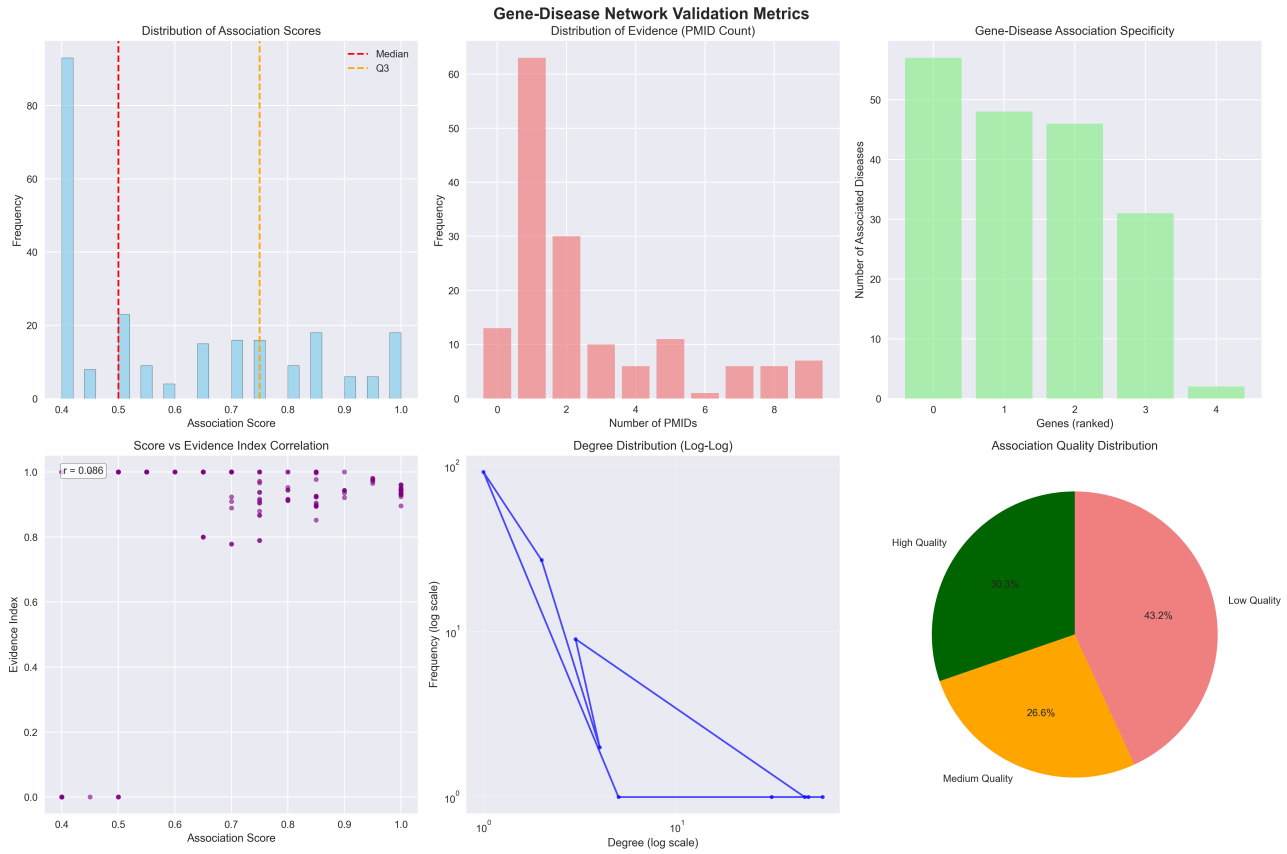
- Distribution of high- and medium-quality associations

f. Community Structure Analysis:

- Modularity score (e.g., 0.4649)
- Number and purity of community clusters
- Intra-community edge density

These metrics are printed in the console and also saved in:

- JSON format: output/validation_results.json
- Image summary: output/validation_metrics_summary.png



6. Use Cases and Applications

This project has broad applicability in various biomedical and data science contexts:

- Biomarker Discovery: Identify key genes that are central to multiple diseases.
- Disease Classification: Cluster diseases based on shared genetic associations.
- Graph-Based Learning: Feed graph data into Graph Neural Networks (GNNs) for prediction tasks (e.g., predicting novel gene-disease links).
- Visualization Dashboards: Integrate with web-based frontends to provide exploratory data interfaces for clinicians and researchers.

By abstracting raw biological data into structured networks, the project facilitates a more accessible and powerful interface for knowledge discovery.

(Refer to the project README or module documentation for specific setup and execution instructions.)

7. Limitations and Future Work

While the current version of the project provides a clear pipeline from data retrieval to graph construction and validation, it leaves room for important improvements:

- Scalability: Implementing asynchronous API requests and data chunking for large-scale data ingestion.

- Data Integration: Combining DisGeNET data with other ontologies such as MeSH, OMIM, or UniProt.
- User Interface: Adding a visualization dashboard or REST API service for real-time queries.
- Machine Learning Integration: Direct support for exporting graph embeddings or interfacing with ML models.

As such, this repository should be viewed as a base module that can evolve into a full-fledged biomedical knowledge graph system.

8. Conclusion

"Gene-Disease Graph" is a compact but impactful project demonstrating the power of open data and graph-based modeling in biomedical research. By enabling researchers to fetch and visualize complex gene-disease relationships, the project provides an essential tool for hypothesis generation, data exploration, and knowledge graph development.

With further development, this project has the potential to scale into a powerful framework for AI-driven biomedical analysis, precision medicine, and therapeutic research.

Keywords: DisGeNET, Gene-Disease Associations, Biomedical Graphs, Network Analysis, Knowledge Graph, Python, Bioinformatics