Unveiling Gene-Disease Connections: A Graph Neural Network Approach for Biological Knowledge Graphs

Joao Pedro Volpi
CentraleSupélec
joao-pedro.monteiro-volpi@student-cs.fr

Lucas Vitoriano
CentraleSupélec
lucasvitoriano.queirozlira@student-cs.fr

Lucas José Veloso
CentraleSupélec
lucasjose.velosodesouza@student-cs.fr

Abstract

This study explores the use of Graph Neural Networks (GNNs) for identifying gene-disease associations through a Knowledge Graph derived from the Stanford Biomedical Network Dataset Collection, which includes over 21,000 connections. By representing genes and diseases as nodes, and their relationships as edges, we enable GNNs to uncover complex patterns and predict novel associations. This approach not only reveals previously unknown gene-disease links but also offers fresh insights into the molecular basis of diseases.

1. Introduction

The intersection of genetics and disease pathology represents a critical area of biomedical research, with the potential to significantly advance our understanding of disease mechanisms and inform the development of targeted therapies. The Stanford Biomedical Network Dataset Collection provides a comprehensive resource for exploring this intersection, offering detailed datasets on gene-disease associations, disease classifications, and gene annotations. This work utilizes Graph Neural Networks (GNNs) to delve into the intricate relationships captured within these datasets and the *Knowledge Graph* we constructed from them. Our primary focus is on the disease-gene association network, aiming to perform **Knowledge Graph Completion**. Through this process, we strive to discover new connections between diseases and genes, enriching our understanding of their relationships.

The Graph Neural Networks (GNNs), a class of deep learning models designed for graph data, are well-suited to model the intricate, non-linear relationships present in biological data. By representing genes and diseases as nodes and their associations as edges, GNNs can learn from the topology of the network as well as node and edge attributes, enabling the identification of novel gene-disease links and the prediction of disease phenotypes based on genetic data.

This study aims to harness the disease-gene association network dataset, containing information on 7,813 nodes and 21,357 edges, to uncover patterns and associations that may not be immediately apparent through traditional analysis methods.

2. The Dataset

The Stanford Biomedical Network Dataset Collection(1) encompasses a rich compilation of biomedical datasets, containing information of drugs, proteins, cells, tissues, etc. We will focus on gene-disease connections. This section elaborates on the composition of the disease and gene datasets within the collection.

2.1. Disease Datasets

The disease datasets are comprehensive, incorporating disease names, definitions, synonyms, and mappings to disease categories. These datasets serve as a fundamental resource for understanding the molecular basis and classification of various diseases:

- **Disease Descriptions and Synonyms**: Derived from the Disease Ontology, this dataset (1.7MB, `D-DoMiner_miner-diseaseDOID`) contains detailed descriptions and features of diseases. It is an essential tool for annotating disease-related information with standardized ontology terms.
- **Disease Descriptions**: Focusing on molecular diseases and environmentally influenced medical conditions, this dataset (3.3MB, `D-MeshMiner_miner-disease`) provides synopses, including names, definitions, and synonyms of diseases. It draws references from the Compar-

ative Toxicogenomics Database and the MINER network, highlighting its utility in understanding disease mechanisms and associations.

- **Classification of Diseases into Disease Categories**: This dataset (15KB, `D-DoPathways_diseaseclasses`) offers a mapping of diseases to their respective categories, based on etiology and anatomical location. It includes diverse examples such as Marfan syndrome (monogenic diseases), rheumatoid arthritis (musculoskeletal system diseases), and liver neoplasms (cancers), facilitating a structured approach to disease classification.

2.2. Gene Datasets

- **Gene Names, Descriptions, and Synonyms**: This dataset (12MB, `G-SynMiner_miner-geneHUGO`) comprises information on 35,000 human genes, including their names, descriptions, synonyms, familial relationships, and chromosomal locations. It forms the backbone of our gene-centric analyses, enabling the exploration of gene functions, relationships, and their roles in diseases.

2.3. Disease-gene Association Network

This dataset (`DG-AssocMiner_miner-disease-gene`, 818KB) represents a disease-gene association network that encapsulates information on genes associated with diseases. In the first sketch of our Knowledge Graph (KG), we define that genes and disease correspond to nodes, and edges denote the associations between them. It provides a structured means to analyze the network of disease-gene interactions.

Dataset Statistics

- Number of Nodes: 7,813
  - Disease Nodes: 519
  - Gene Nodes: 7,294
- Number of Edges: 21,357

This collection provides a comprehensive framework for our study. The disease datasets, with their rich descriptive and classificatory information, combined with the gene datasets' extensive coverage of gene attributes, offer a opportunity to explore and predict gene-disease relationships

3. Modeling as a Knowledge Graph

The transformation of the Stanford Biomedical Network Dataset Collection into a Knowledge Graph (KG) represents a pivotal step in our work. Utilizing the Web Ontology Language (OWL), we delineate a structured, semantic framework that not only encapsulates the entities and their relationships within the biomedical domain but also facilitates advanced analysis and inference. In this section, we detail the creation of this ontology, emphasizing the classes, properties, and instances that form the backbone of our knowledge graph. The namespace prefix **bio:** is systematically employed to distinguish our domain-specific elements.

3.1. Defining Classes

At the core of our OWL scheme are two primary classes that reflect the fundamental entities of our investigation:

Disease encapsulates individual diseases, serving as a repository for various attributes including name, classification, and symptoms.

Gene represents genes, focusing on aspects such as name, function, and chromosomal location.

```
bio:Gene rdf:type owl:Class.
bio:Disease rdf:type owl:Class.
```

3.2. Specifying Properties

Properties play a crucial role in defining the attributes of our classes and the relationships between them. We introduce a mix of datatype and object properties to richly annotate our entities and their interconnections:

Datatype Properties for Disease include *hasDiseaseClass*, *hasDefinition*, *hasMainSymptom*, and *hasRiskFactor*, among others. These properties allow for the detailed characterization of diseases.

Datatype Properties for Gene such as *locatedAt* and *belongsToFamily* provide essential information on gene attributes.

Object Property *isAssociatedWith* directly links genes to diseases, establishing a foundational basis for analyzing gene-disease associations.

```
bio:isAssociatedWith rdf:type owl:ObjectProperty;
                     rdfs:domain bio:Gene;
                     rdfs:range bio:Disease.
```

3.3. Instantiating Individuals

The instantiation of individuals—specific diseases and genes—along with the assignment of their properties is the final step in populating our knowledge graph. This includes:

- Creating instances of the **Disease** and **Gene** classes, imbued with relevant properties derived from our dataset.
- Establishing the *isAssociatedWith* relationships between genes and diseases, laying the groundwork for our analysis.

The relationship *isAssociatedWith* is precisely the one we aim to complete within our graph. Figure 1 displays several instances within the knowledge graph and illustrates the task we are attempting to perform: **Knowledge Graph Completion**. This involves predicting potential missing links in our graph.

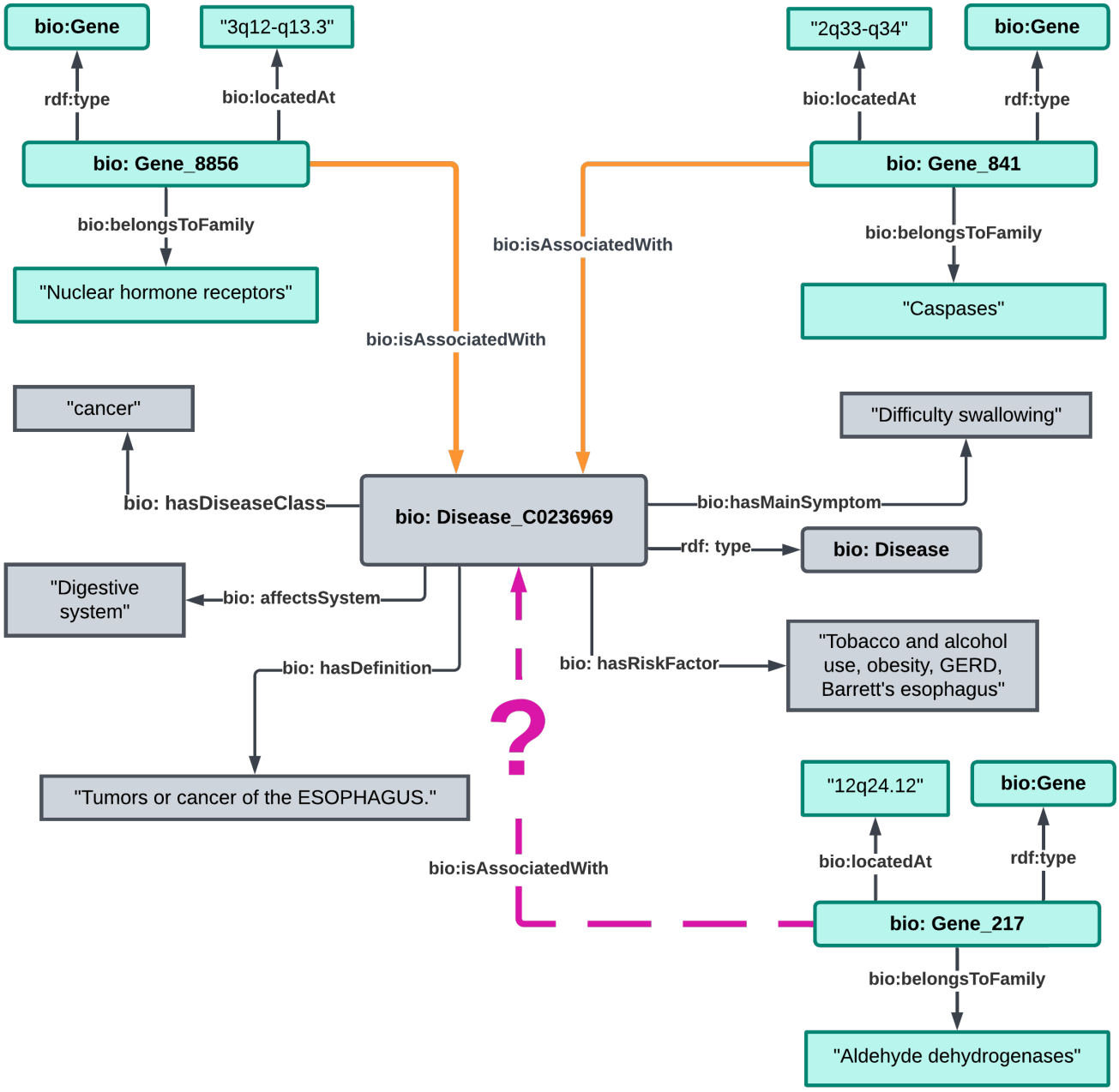

Figure 1. The image depicts several instances from our knowledge graph, illustrating the critical challenge we address in this work: **Knowledge Graph Completion**. Specifically, we focus on utilizing the existing relations (highlighted in orange) and attributes within our incomplete knowledge graph to predict possible missing relationships (indicated in pink) between genes and diseases.

4. Graph Neural Networks (GNN)

Graph Neural Networks (GNNs) represent a cutting-edge class of deep learning models designed to perform inference on data structured as graphs. Unlike traditional neural networks, GNNs can capture the complex relationships and interdependencies between nodes2 (e.g., genes, diseases) within a graph, making them particularly suited for analyzing biological networks. By leveraging node features and edge information, GNNs can learn to predict not only the properties of individual nodes but also the strength and significance of the connections between them. This capability is critical for our work, as it allows us to model the intricate gene-disease associations within the Knowledge Graph, potentially uncovering novel insights and predictive markers for diseases.

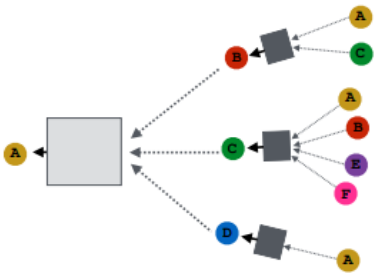

Figure 2. How GNNs generates encodings. Image from https://web.stanford.edu/class/cs224w/

5. Our Approach

5.1. Preprocessing

In the preprocessing phase of our research, we executed a series of essential steps to refine the Knowledge Graph derived from the Stanford Biomedical Network Dataset Collection. These steps were crucial to ensure the graph's consistency and robustness, laying a solid foundation for our subsequent analysis.

Disease Feature Standardization Our initial challenge was to harmonize the disease identifiers across the three disease feature datasets, which used different IDs and naming conventions. This standardization process was crucial to accurately match diseases across datasets and consolidate their features into a unified table. However, this integration revealed that many diseases lacked comprehensive feature information, such as definitions, categories, and synonyms. To maintain the integrity of our analysis, diseases missing a significant portion of these features were excluded from the dataset.

Disease Feature Augmentation To augment the disease feature table, we employed OpenAI's GPT API, using GPT 3.5 Turbo, leveraging its capacity to generate contextualized information based on prompts. For diseases with existing definitions in our dataset, we formulated prompts to extract additional features in a structured JSON format, specifically focusing on the main symptom, risk factors, disease class, and main system affected. This approach was predicated on the assumption that providing a specific disease definition would guide the language model towards generating accurate and relevant information, thereby minimizing the risk of data hallucination. This method enabled us to enrich the dataset with four new features for each applicable disease.

Prompt used:

> *The disease has the following definition: definition Based on your available knowledge and in the definition provided, give me information about disease only in JSON format on:*
> {
> *"main symptom": "",*
> *"risk factors": "",*
> *"disease class": "",*
> *"main system affected": ""*
> }
> *Go straight to the point: only list the important terms and don't talk too much.*

Gene Feature Enhancement The gene dataset presented a different set of challengess for enhancement. Initially, the gene table included features such as Gene ID, hgnc_id, name, locus group, locus type, and location. We identified that the "location" feature, which specifies the chromosomal position of genes, contained valuable information that was not explicitly parsed in the dataset.

To extract this latent information, we applied multiple regular expressions to dissect the "location" feature into more granular components: Start Chromosome, Start Chromosome Arm, Start Chromosome Loc, Start Chromosome SubLoc, End Chromosome Arm, End Chromosome Loc, and End Chromosome SubLoc. This breakdown allowed for a more nuanced understanding of gene locations and facilitated more detailed analyses regarding gene distribution and potential implications for disease associations.

Example: For the gene

Location
5q14.2-q14.3

Table 1. Response preferences between fine-tuned Gemma and fine-tuned Phi model

Became after preprocessing:

Feature	Value
Start Chromosome	5
Start Arm	q
Start Loc	14
Start SubLoc	2
End Arm	q
End Loc	14
End SubLoc	3

Table 2. Detailed breakdown of gene location information

The preprocessing stage was fundamental in refining the Knowledge Graph for our analysis. By standardizing disease names, augmenting disease feature and enhancing gene feature granularity, we laid a solid foundation for applying Graph Neural Networks to explore the relationships between genes and diseases.

5.2. Implementation Details

In our quest to accurately predict gene-disease relationships, we explored various Graph Neural Network architectures to determine the most effective model for our heterogeneous graph data. This iterative process involved evaluating different models based on their ability to encode complex node and edge relationships inherent in gene-disease interaction data. Below, we outline the revised approach to our model's encoder design and architecture selection process.

Model Architecture Selection Contrary to employing a single architecture, our approach was to test multiple models, including GATv2Conv, GCNConv, and SAGEConv, among others. This strategy was driven by the understanding that the optimal architecture might vary depending on the specific characteristics of the gene-disease graph, such as its sparsity, connectivity patterns, and node feature distributions. Each GNN model has unique strengths; for instance, GATv2Conv excels in capturing node interdependencies through attention mechanisms, while GCNConv offers a more generalized method of aggregating neighbor information, and SAGEConv focuses on sampling neighbor nodes for efficient computation.

Training Process Adaptation Adapting to multiple architectures necessitated a flexible training process. We employed a standardized training loop that could accommodate any GNN model by adjusting parameters such as the number of epochs, batch size, and learning rate based on the architecture in use. This approach ensured that each model was trained under optimal conditions, allowing for a fair comparison of their performances.

Technical Stack Our experiments were conducted using the PyTorch framework alongside PyTorch Geometric (PyG)(2), which offers extensive support for various GNN layers and architectures. This choice provided us with the versatility needed to implement and test different GNN models efficiently.

6. Results

Our experimental setup evaluated three different Graph Neural Network (GNN) architectures: SAGEConv(3), GraphConv(4), and GATv2(5). The performance of each model was measured in terms of accuracy and F1 score across epochs during training and testing phases. This section presents the results and discusses the implications of our findings, supported by visual analyses in Figures 3 and 4.

6.1. Accuracy Analysis

Figure 3 illustrates the accuracy progression for each GNN model across the training epochs. The SAGEConv architecture consistently demonstrated superior performance in the testing phase, achieving an accuracy that plateaued around 70% after initial fluctuations. The GraphConv model exhibited a comparable performance, with its accuracy converging towards that of the SAGEConv model towards the later epochs. In contrast, the GATv2 model showed slightly less stability, with a test performance that gradually improved but remained below the 70

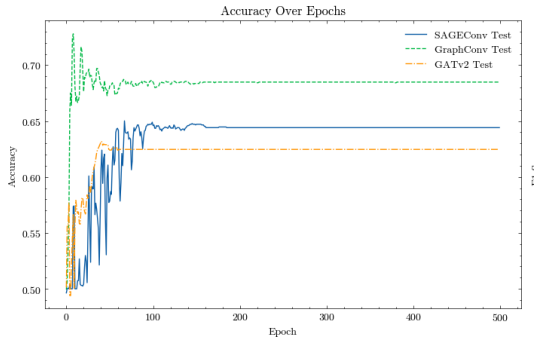

Figure 3. Accuracy of SAGEConv, GraphConv, and GATv2 models over training epochs.

6.2. F1 Score Analysis

The F1 scores for the models are shown in Figure 4. Similar to accuracy, the SAGEConv and GraphConv models performed comparably, with the SAGEConv slightly outperforming GraphConv in the test F1 score, reaching close to 0.7. The GATv2 architecture, while still delivering an acceptable level of performance, had a test F1 score consistently under 0.7, indicating some limitations in the balance between precision and recall for this model.

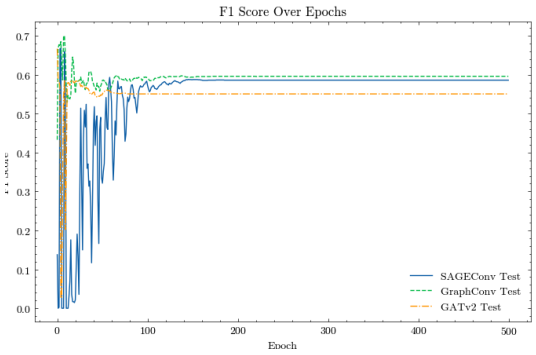

Figure 4. F1 Score of SAGEConv, GraphConv, and GATv2 models over training epochs.

6.3. Comparative Performance

Throughout the training and testing phases, the SAGEConv model maintained a slight edge over the others in both accuracy and F1 score. The performance stability of SAGEConv suggests it may be better suited to this particular gene-disease prediction task, as further evidenced by the trends depicted in Figures 3 and 4.

6.4. Model Selection Considerations

The choice of the most suitable model for gene-disease prediction should take into account not only accuracy and F1 score but also model complexity, interpretability, and computational efficiency. Our results indicate that the SAGEConv model stands out as a robust choice for our dataset. However, the GraphConv model could be preferred for its simpler architecture and potentially faster inference times.

7. Limitations

Despite the promising results, our study encompasses several limitations that must be acknowledged. Firstly, the quality of the predictions is highly contingent on the data's accuracy and comprehensiveness. While the Stanford Biomedical Network Dataset Collection is extensive, it may not capture the entire spectrum of gene-disease associations, especially for less studied or newly discovered diseases.

Another limitation stems from the inherent complexity of biological networks. While GNNs are adept at modeling non-linear relationships, they may oversimplify complex biological phenomena that are influenced by a multitude of factors beyond the scope of the present dataset, such as environment and personal lifestyle, given that diseases are not solely determined by genetic predispositions.

Furthermore, the interpretability of GNNs remains a challenge. Although our models can predict associations, understanding the underlying biological mechanisms that lead to these predictions is not straightforward. In practice, these models can suggest directions with a higher likelihood of success (better than a mere guess), but they still require the technical verification by professionals to confirm the existence of the predicted relationships. They can be utilized by these professionals to determine which relational studies to pursue.

8. Conclusion

In conclusion, our exploration into the application of Graph Neural Networks for the prediction of gene-disease relationships in the created Knowledge Graph has demonstrated the viability and potential of GNNs in this domain. Our results show that the SAGEConv model marginally outperforms the GraphConv and GATv2 models in both accuracy and F1 score, as evidenced by 3 4

While the SAGEConv model is preferred for further work in this area, the limitations outlined point to areas for improvement and future research. These include expanding the dataset and then the Knowledge Graph, enhancing model interpretability, and exploring a broader range of GNN architectures and configurations.

References

[1] Sagar Maheshwari Marinka Zitnik, Rok Sosič and Jure Leskovec. BioSNAP Datasets: Stanford biomedical network dataset collection. http://snap.stanford.edu/biodata, August 2018. 1

[2] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. 5

[3] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs, September 2018. arXiv:1706.02216 [cs, stat]. 5

[4] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks, February 2017. arXiv:1609.02907 [cs, stat]. 5

[5] Shaked Brody, Uri Alon, and Eran Yahav. How Attentive are Graph Attention Networks?, January 2022. arXiv:2105.14491 [cs]. 5