Drug Repurposing with the Open Targets Dataset

Christo Sasi

Student Number: S1102577hora

MSc. Specialisation: Medical Epigenomics

Daily Supervisor: Siqi Wei

Project Supervisor: Prof. Dr. Peter-Bram 't Hoen

Place: Nijmegen, Netherlands

Date: 3 September 2024

1 Summary

This report presents the design and evaluation of a novel pipeline for drug repurposing, focusing on the use of graph neural networks (GNNs) trained on the OpenTargets dataset. The study explores the construction of biomedical knowledge graphs and the application of various GNN architectures to predict drug-disease interactions. It highlights the challenges encountered with the OpenTargets platform, such as schema inconsistencies and limited drug-specific annotations. The pipeline's performance was evaluated across different negative sampling approaches, revealing that the selection of these samples significantly influences the model's effectiveness. The report concludes with recommendations for improving the pipeline, including adding custom graph creation features and enhancing the interpretability of the results through better tooling and parameter tuning.

2 Introduction

2.1 Introduction to Drug Repurposing

Drug repurposing is a method of conducting drug discovery by identifying new therapeutic uses of approved medicines based on knowledge obtained from previously known drug-target interactions [1]. Historically, successful drug repurposing projects have been conducted by biologists and pharmaceutical domain experts based on experimental evidence published in literature and clinical trial results [2, 3].

These projects require both physicians and human domain experts with a background in pharmacology and theoretical biology for the identification of the potential drug candidates for repurposing. In cases of previously untreated or rare diseases, the relevance of genes and variants linked to the diseases also need to be taken into account for the design and development of a drug repurposing experiment [4]. Such projects require input from multiple domain experts from diverse fields [5]. For rare and previously untreated diseases, even the insights of domain experts can sometimes fall short due to the lack of efficacy of repurposed candidates in clinical trials[6, 7].

2.2 Deep Learning Methods for Drug Repurposing

Machine learning algorithms for drug repurposing can be inspired by classical machine learning techniques such as Random Forest, Support Vector Machines and Logistic Regression [8, 9, 10]. However, modern deep learning techniques have been outperforming these techniques due to the following reasons:

1. Feature Engineering Limitations: While classical machine learning techniques has been useful in the domain of drug repurposing [8, 10, 9], it typically requires extensive feature engineering—manually extracting and selecting data features, that are important for model predictions. This process is not only time-consuming but also limits the scope of data exploration to the expertise and biases of the researchers. Deep learning methods allow for automatic feature learning from massive datasets [11].

2. Handling Complex Biomedical Data: Drug repurposing draws upon diverse data like chemical structures, genomic profiles, and protein-protein interactions. Given the diversity in the innate properties of the entities in biomedical datasets, deep learning frameworks allow machine learning models to learn from seen and unseen relationships despite the variability in data source and data type. Deep learning excels at modeling the intricate relationships within heterogeneous data sources such as genetic, proteomic, drug profiles, electronic health records and gene expression data. [12].

3. Enhanced Predictive Power: The nonlinear modeling capabilities of deep learning architectures often result in superior predictive performance for identifying potential drug repurposing candidates, outperforming simpler models commonly used in classical machine learning [13].

2.3 Motivation for graph representation of biomedical knowledge

The cumulative biological and pharmacological knowledge published by domain experts from clinical and molecular interaction studies is published on biological databases such as ChEMBL [14], DrugBank[15], PubChem [16], PDB [17], etc. Data extracted from these databases can be transformed and processed to generate biomedical knowledge graphs. These biomedical knowledge graphs represent complex pharmacological and biological interactions follow based on the network medicine model of disease [18].

Biomedical knowledge graphs are a type of non-Euclidean data structure that cannot be fully represented in traditional, finite-dimensional spaces. Graphs can be used for the representation of functional, structural, and other complex information inherent in non-standard real world data to train a GNN [19, 20]. A more expressive variant of a graph which can be used for the representation of knowledge is called a knowledge graph. A knowledge graphs consists of 3 basic entities: nodes, features and edges.

Following is an example of a biomedical knowledge graph:

1. Nodes: Entities that can be represented with unique keys. For e.g., ChEMBL IDs for molecules, EFO IDs for diseases, ENSG IDs for genes. These keys can be mapped to unique indices in a Knowledge Graph consisting of n nodes as (1, 2, 3 .....n).

2. Features: Features represent the properties or characteristics associated with each unique node in a graph. These properties need to be encoded as tensors (numerical arrays) based on the type of feature, enabling the graph to capture and represent the diverse attributes of its nodes. By encoding features in this way, the graph can effectively represent complex entities—such as molecules, diseases, or genes—in a format that machine learning models can process. The encoded features, typically as binary (1s and 0s) or numerical vectors, allow models to learn and distinguish the properties of different nodes, facilitating tasks like classification, prediction, and inference within the graph structure [21].

   For application in drug repurposing, some types of features for a drug, disease or gene node might include:

   (a) Boolean features: Features that can be represented as True or False, 'BlackBoxWarning'; [0, 1]

   (b) Numerical features: Features molecular weight ; [0.625]

(c) Categorical features: Categorical features of a node which allows it to be placed in only category among an number of given options. Drug Type : [0, 0, 0, 0, 1]

(d) Textual Features: Descriptions, clinical trial summaries, or adverse effect reports. These can be encoded using text embedding techniques (e.g., Word2Vec [22], Doc2Vec [23]) to create numerical tensor representations. [0.23, -0.11, 0.87, 0.04, ..., -0.34]

3. Edges: Entities that can be used to represent relationships between 2 node entities. For e.g., If a molecule m is approved for treating disease d, it can be represented as (m, d).

Drug repurposing methods based on biomedical knowledge graphs have gained popularity in recent years because they facilitate the simulation of complex system interactions between drugs and their targets [24, 25].

2.4 GNNs for Drug Repurposing

Due to the geometric nature of graph representation, graph machine learning algorithms can be trained on the topology and features of the nodes in a graph to predict new links between previously unconnected nodes. This allows for Graph Neural Networks (GNNs) to be used for performing classification and prediction tasks on graph-structured data. These methods leverage the connectivity information and features of a graph to iteratively propagate information from a node to its neighbors, thus allowing node-specific updates based on both the node's own attributes and its neighbors' states. In the context of drug repurposing using biomedical knowledge graphs as input, these algorithms have the following advantages over other deep learning techniques.

1. Message Passing Function: This enables the model to capture the complex dependencies in the graph structure [21].

2. Directly Modeling Relationships: GNNs process the connections between drugs, diseases, and molecular targets directly, preserving crucial relational information that may be obscured when converting graphs into other representations [24, 25].

3. Leveraging Structural Information: The topology of biomedical knowledge graphs itself contains valuable patterns. GNNs are designed to learn from this structure, potentially identifying drug candidates that other methods might miss [26].

4. Potential for Improved Interpretability : While the complexity of deep learning can hinder understanding, GNNs aligned with biological pathways offer the potential to explain their reasoning. This aids in understanding mechanisms of action and builds greater confidence in predictions [27].

5. Impact of Edge Connectivity: GNNs are expected to make better predictions when the edge connectivity in a graph is denser, as it provides more information for the model to learn from. This is especially true in biomedical knowledge graphs where multiple relationships between nodes (e.g., drugs, genes, diseases) can offer valuable insights. However, it is unknown whether that important to note that the effectiveness of this approach can diminish in scenarios where the graph is too small with too many edges. This pipeline is designed to evaluate the impact of adding edge connectivity to make the input graph fully connected without any isolated nodes on the prediction performance of GNN models.

6. Impact of Negative Sampling: Negative sampling plays a crucial role in training machine learning models to predict drug-disease links. The absence of true negatives and the challenge of dealing with false negatives (i.e., drug-disease links that are unlikely or impossible) can significantly limit the performance of models designed for drug repurposing. Previous research has emphasized that carefully selected negative samples are essential for motivating models to distinguish between likely and unlikely drug-disease associations effectively [28, 29].

GNN models are expected to learn from the representations of nodes that already have established connections with other entities within the graph. In the context of a drug repurposing experiment, the representations of drugs and diseases that are already linked are expected to help the GNN predict additional links for these drugs [30]. The loss function for these GNNs is designed to encourage the model to minimize false positives by correctly identifying non-existent links.

In this context, a positive link prediction occurs when the model accurately predicts a link that exists in the validation dataset. Conversely, a negative link prediction is when the model correctly predicts the absence of a link that does not exist in the validation dataset. This ability to distinguish between true and false drug-disease associations is critical for the practical application of the GNNs for drug repurposing

2.5 Overview of the OpenTargets Platform

The OpenTargets platform provides users a single source ∼20 publically available expert annotated biomedical datasets. Some salient features of the dataset that can be useful in creating a graph representation of the data and a machine learning pipeline for drug repurposing:

1. The Open Targets dataset is updated every quarter with new annotations based on real world discoveries in literature of gene-disease associations and drug-disease approvals due to successful clinical trials. This allows users to train models on historical data while also allowing for the use of validation test sets from later updates. Such a train-validate-test (illustrated in Figures 3, 4, 5) framework is useful for drug repurposing experiments.

2. Annotation of gene-to-disease association datasets based on data sources and theoretical biology metrics allows for positive and negative relationship representation for genes to diseases. All associations datasets have association scores upon which a threshold for association can be applied. This allows for the inclusion and exclusion of gene-disease edges based on the user's preference while also allowing for the evaluation of these scores for real world applications such as drug repurposing.

3. Flexibility of use for machine learning applications and knowledge representation: An analysis [31] of the OpenTargets Dataset showed that 33 out of 50 (66 %) drugs approved by the FDA in 2021, had either one or all the following features:

   (a) The genes encoding their primary assigned targets had been previously associated with the drug's indication.

   (b) Proteins known to physically interact with the drug targets had established associations with the indication.

(c) There existed a closely related phenotype to the drug's indication that was genetically linked to the drug target.

These findings point towards the influence of genetic evidence in determining the probability of a drug-disease relationship existing in the real world. This reasoning can also conversely suggest that drugs showing low or no genetic evidence would be less likely to be approved for diseases. The experiment detailed in this report provides a framework for testing the validity of this claim. The input graph used to train the model includes the gene-disease associations as a yes/no relationship based on the threshold of association described by the user. The pipeline generated for this experiment also tests the usefulness of these associations in generating less likely drug-disease links for negative sampling.

2.6 Updated Research Question/Hypothesis

The application of machine learning frameworks for identifying drug repurposing candidates has been widely explored in scientific literature. However, there is a pressing need for tools that allow researchers to benchmark these algorithms through retrospective analyses, assessing whether the predicted candidates have proven effective in real-world clinical trials. The framework developed in this pipeline addresses this need by enabling the model to be trained on a historical dataset, reflecting drug approval data from a specific period, and subsequently validating and testing the model on data published in future updates. This approach not only strengthens the model's predictive power but also provides a practical means to evaluate its real-world applicability over time.

The datasets provided by the OpenTargets Platform serve as a rich source for constructing graph representations of complex biomedical data. This platform's capabilities allow for the design of a versatile machine learning pipeline, adaptable to data updates across different versions.

Based on this understanding of the data source and the possibility of building a machine learning pipeline, the following Research Question is posed for this report: What is the impact of edge connectivity, negative sampling, and GNN architecture on the performance of predictive models in drug repurposing?

Hypothesis: By fine-tuning edge connectivity, applying strategic negative sampling, and carefully selecting the GNN architecture, it is possible to significantly improve the predictive performance of drug repurposing pipelines.

3 Proposed Methods

3.1 Pipeline Overview

The pipeline for performing the experiment can be (described in Figure 1) detailed as:

- Step 1: Split available datasets into train-validate and test triplets based on the availability of updated validation and test data for drugs that are present in the training set and got repurposed for diseases in the validation and test sets (Figure 3, 4, 5).

- Step 2: Query all versions of the OpenTargets datasets to identify sources to extract graph entities (nodes, features and edges)
- Step 3: Pre-process graph entities and encode each column into suitable Tensors
- Step 4: Generate a knowledge graph representation for the training version
- Step 5: Generate a validation set dataset of drug-target-disease relationships
- Step 6: Train a GNN with this knowledge graph to perform link prediction (new drug-disease links) and test its performance on the validation and test sets described in Step 1 based on the following parameters:

    1. Varying combinations of graph entities (features, nodes and edges)
    2. Negative Sampling Methods
    3. Graph machine learning architecture

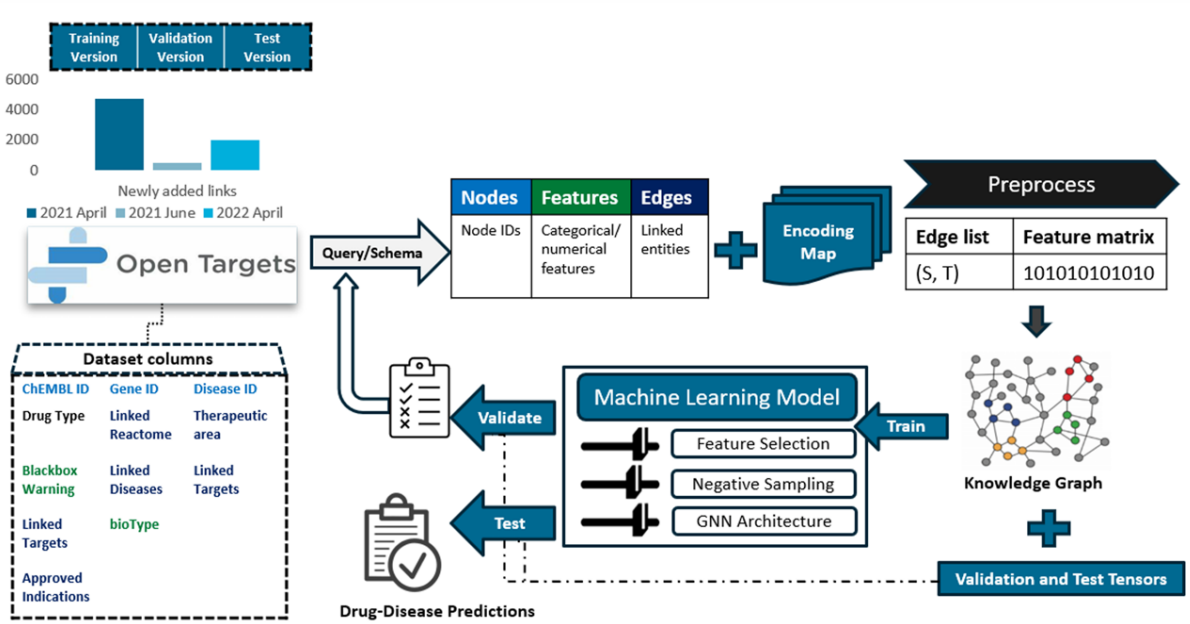

Figure 1: Overview of Drug Repurposing Pipeline

3.2 Parquet Dataset Viewer

The parquet dataset viewer (Figure 2) was developed using the curses library to allow for the row-wise viewing of parquet datasets. This allows to get a view inside each dataset and view data stored inside each column. It also helps take a look at nested data structures that would otherwise only be deducible by parsing the files with a python script and analyzing a detailed schema.

The Python API for Apache Arrow (PyArrow) was used to generate queries for extracting data from the parquet datasets [32]

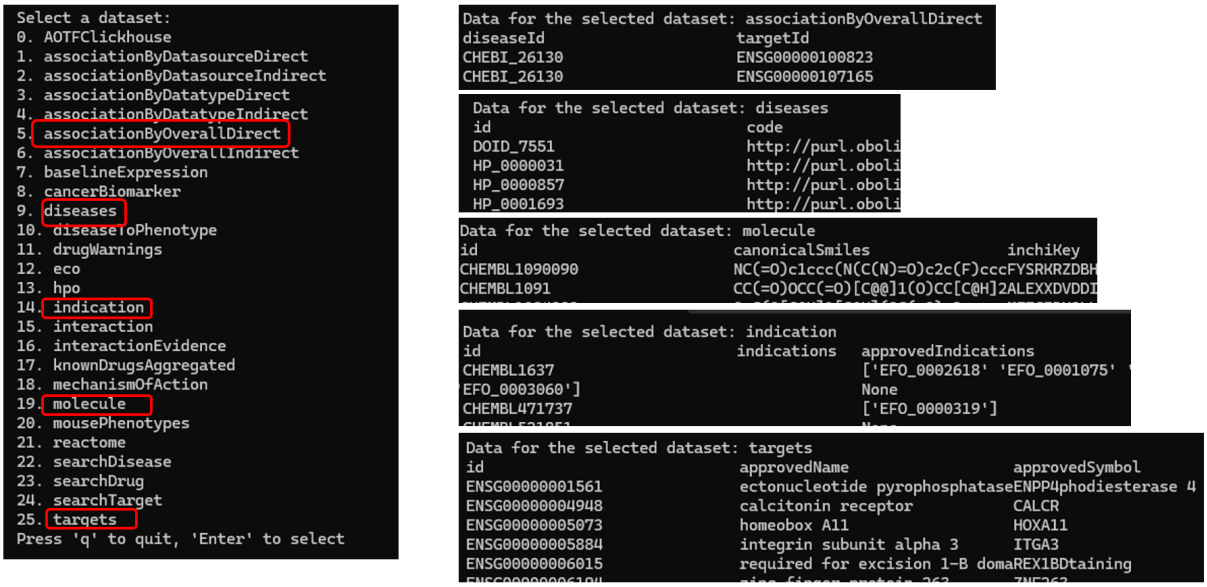

Figure 2: Parquet Dataset Viewer

3.3 Train/Validate/Test split with a Sliding Window

The Open Targets database allows users to access datasets from 2019-07-01 to the present. After viewing the datasets available across versions with the parquet dataset viewer, the 'molecule' , 'targets' and 'disease' datasets were selected for graph entity extraction as described in Table 1. For this experiment, versions 21.04 (Published 2021-04-28) to 23.12 (Published 2023-11-29) were selected due to the consistency in schema amongst versions [33]. While some column names have been changed by the publishers, the graph creation script in the pipeline accounts for such cases.

The 'molecule' dataset was filtered to select molecule nodes which already have at least one approved indication across versions. The increase in the number of these approved molecules across datasets indicate that the new approvals for old and new molecules are present in these datasets. However, to create a train/validate/test framework, a validation and test set where new approvals for drugs already present in the training set is required. To accomplish this, the selected versions (a total of 13) were divided into consecutive pairs of training and validation versions. This allows for the pipeline to be used for training GNN models on a graph generated from the training version. Then this model can be validated on the validation and test set of new drug-disease edges extracted from future versions of the dataset.

To determine the best pair for train/validate/test distribution configuration, Figures 4 and 5 were generated. These figures help visualize possible options for training, validation, and test sets based on the differences in drug-disease edges across versions. These edges are expected to increase across versions as more drugs get approved for more diseases in the real world. While Figure 4 shows a general increase in new approvals for both new and existing molecules, Figure 5 highlights that a significant number of new approvals for already approved drugs are observed primarily in version 21.06 (497 new edges) and version 22.04 (1931 new edges). This indicates that the edges representing successful real-world drug repurposing approved by regulatory agencies, such as the FDA, are most concentrated in these two versions.

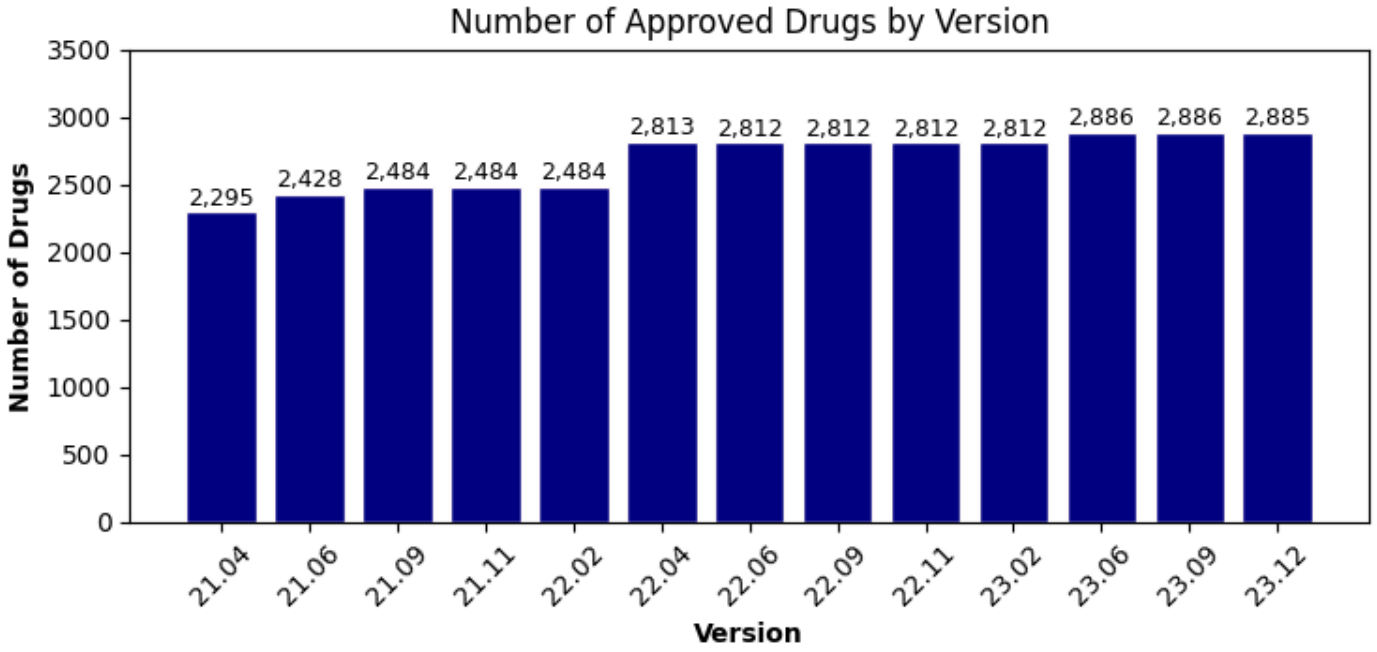

Figure 3: Number of drugs across versions on OpenTargets

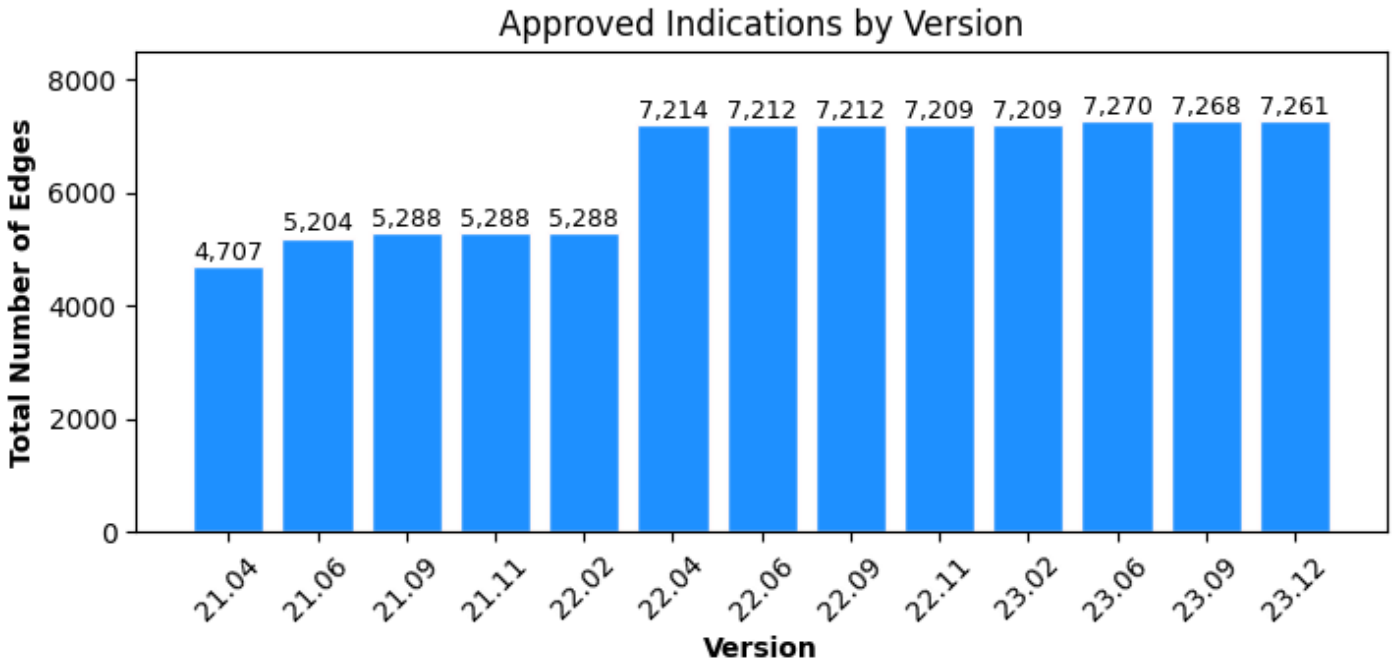

Figure 4: Count of Approved Indications across versions on OpenTargets

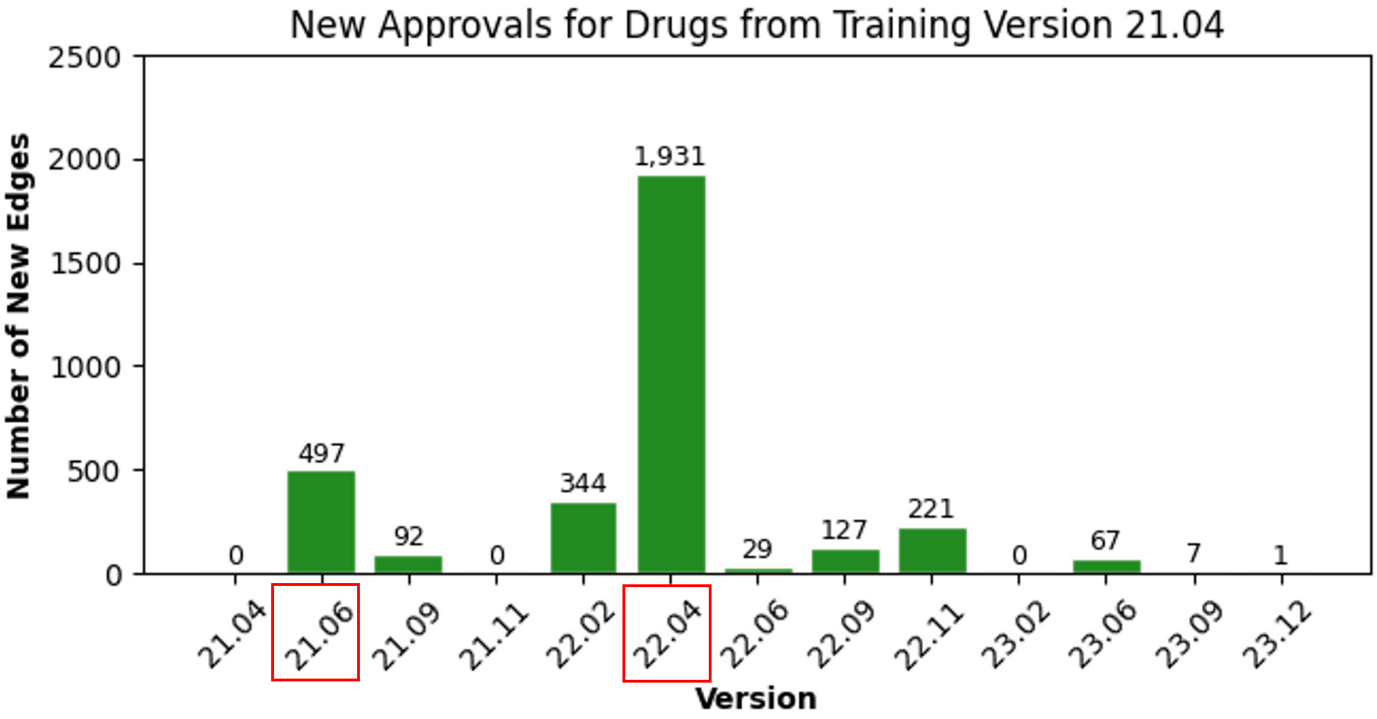

Figure 5: Edge Distribution in Validation and Test Sets

As illustrated in Figures 3, 4 and 5, the sliding window approach of viewing datasets across versions allows for the selection of training, validation and test sets that hold the most potential to be used for evaluating the GNNs and negative sampling used in this experiment.

- Training Set: Version 21.04
- Validation Set: Version 21.06
- Test Set: Version 22.04

3.4 Knowledge Graph Schema and creation

3.4.1 Data selection:

1. Parquet datasets were viewed with the dataset viewer and datasets to be used to extract graph entities were selected.
2. Column names alongside their datatypes were printed into a csv for all selected datasets. Team members collaborated on selecting columns and annotating graph entity type (node, feature and edge).
3. The following maps were created for describing the various relationships between the graph's entities along with the various properties each node:
    - Node-to-Feature Map: Defines the specific properties that each node will represent, which the model will learn from.

- Edge Map: Clearly outlines the connections between nodes (e.g., drug-gene-disease relationships), specifying how different nodes are linked, such as drug-disease edges representing therapeutic interactions.
- Feature Encoding Map: Specifies the encoding strategy for each feature type, ensuring they are interpreted correctly by the GNN.

The datasets and the columns selected for graph creation are listed in the table below along with their graph entity mapping:

Table 1: Graph Entities selected for input graph creation

Dataset	Column Name	Graph Entities
Molecule	id	drug node
	drugType	drug-type nodes; drug to drug-type edges
	blackBoxWarning	drug node feature
	yearOfFirstApproval	drug node feature
	linkedTargets	drug to gene edges
Targets	id	gene node
	bioType	gene node feature
	pathway	pathway nodes; gene to pathway edges
Diseases	therapeuticAreas	therapeutic area nodes; disease to therapeutic area edges
	ancestors	disease to ancestors edges
	descendants	disease to descendants edges
	children	disease to children edges
Indication	approvedIndications	drug and disease nodes; drug and disease edges

3.4.2 Graph creation:

1. Node Selection : drug, drug-type, gene, pathway, disease, therapeutic-area. As this is a drug repurposing experiment, the drug nodes that were selected from the 'id' column of the molecule dataset were filtered for drugs with at least 1 approved indication (filtered with 'approvedIndications>0')

2. Feature Selection: The following node-feature mapping and feature-encoding map were used to include feature representations in the generated graph

Table 2: Node-feature and feature encoding map

node	feature	Encoding
Drug	blackBoxWarning	boolean
	yearOfFirstApproval	normalize
	node-type	one-hot-encoding
Drug-Type	node-type	one-hot-encoding
Gene	bioType	one hot encoding
	node-type	one-hot-encoding
Pathway	node-type	one-hot-encoding
Disease	node-type	one-hot-encoding
Therapeutic-Area	node-type	one-hot-encoding

3. Edge Definition: The following undirected edge relationships were selected for the graph's edge index:

   (a) drug to drug type
   (b) drug to gene
   (c) drug to disease
   (d) gene to pathway
   (e) disease to gene
   (f) disease to therapeutic area
   (g) disease to ancestors edges
   (h) disease to descendants edges
   (i) disease to children edges

Only edges a-f are included in the graph with isolated nodes. For the fully connected graph, edges g,h and i are also included. The graph connectivity is further illustrated in Figure 6 and Figure 7 below:

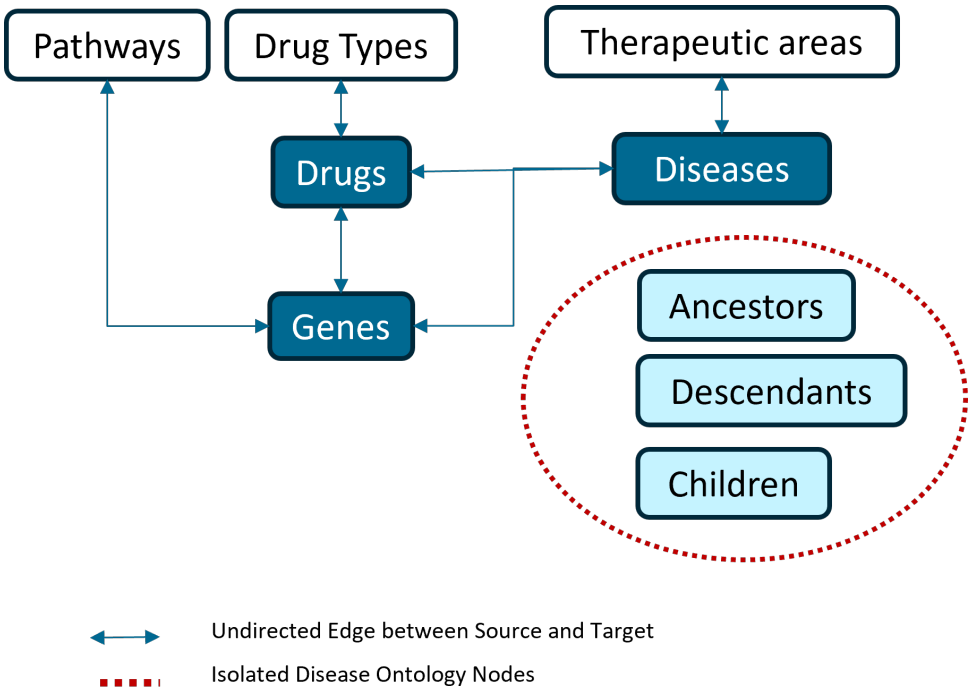

Figure 6: Graph with isolated Disease Ontology nodes

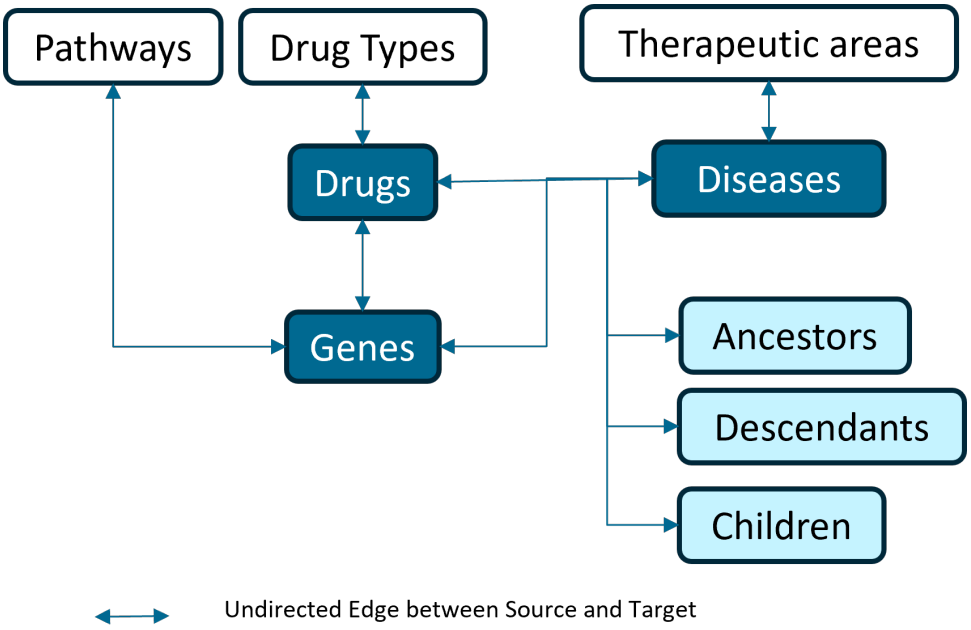

Figure 7: Fully Connected Graph

3.4.3 Graph Statistics:

To further understand the structural properties of these graphs, the following statistics were computed:

1. Degree Distribution Histogram: The degree distribution provides insight into how nodes are connected within the graph. A higher average node degree suggests a denser network, which can be beneficial for GNNs, as it provides more context and relationships for the model to learn from. A balanced degree distribution can indicate a more homogeneous graph, whereas a skewed distribution might suggest the presence of hubs or highly connected nodes[34, 35].

2. Degree Assortativity Coefficient: This coefficient measures the tendency of nodes to connect to other nodes with similar degrees. A high assortativity coefficient might suggest the presence of clusters within the graph, which can affect how the GNN learns patterns within the data [36, 37].

3.5 Machine learning models tested in this pipeline

Consider an undirected graph $G\ (N,E)$ where N is the number of nodes, E is the set of edges between the nodes in the graph. For each node i in the graph (where i belongs to the set $(0,....N)$) x_i is the feature tensor that represents the features of node i.

Each node is represented with a one-hot encoded vector that represent its unique node type. The rest of the features for each node type are in enlisted in Table 2. The feature tensors of each node type are padded to match the size of the feature tensor of the node type which has the most amount of features. In this experiment, the 'drug' node type has the most number of

features. In undirected graphs, all existing edges between nodes are edges without an explicit direction being defined for these relationships. This is implemented by including 2 sets of edges in E which representating interchangeable source-node relationships . This allows the GNN we train to treat edges between nodes symmetrically, which is appropriate for the link prediction task.

The GNN implemented in this project is written in Python (Code available at the MBS Gitlab Repo [38]). The individual components of the GNN are listed below in the order of each operation performed to get the final the predictions.

The current architecture consist of 3 essential components GNN consists of the following operations:

1. GNN node-wise operation: The input graph is passed through the selected model over 10 iterations. This operation transforms the input features of each node based on its neighbors as described by the architecture of the selected model.
2. Layer Normalization: This Operation is applied after each pass through the selected model, layer normalization helps stabilize the learning process by normalizing the outputs.
3. ReLU Activation Function: Used after each layer (except the last one), ReLU (Rectified Linear Unit) introduces non-linearity into the model, enabling it to learn complex representations.

The node-wise formulation of each GNN model applied on the input graph described above are discussed below:

1. GCNConv: A semi-supervised learning algorithm for graph neural networks based on a first order approximation of spectral graph convolutions [39]. This model aggregates and normalizes the features from a node's neighbors to update its representation. The node wise formulation is written as:

$$\mathbf{x}'_i = \mathbf{\Theta}^\top \sum_{j \in \mathcal{N}(i) \cup \{i\}} \frac{e_{j,i}}{\sqrt{\hat{d}_j \hat{d}_i}} \mathbf{x}_j$$

   (a) $\Theta^\top$: This is the learnable weight matrix that is applied to the aggregated features of the node i and its neighbors. The matrix $\Theta^\top$ is crucial for transforming the combined information into a new representation for the node.

   (b) $e_{j,i}$: Represents the edge weight between the source node j and the target node i. Since the graph has no specific edge weights, $e_{j,i}$=1 for all edges.

   (c) $\hat{d}_i$ and $\hat{d}_j$: These are the normalization constants derived from the degrees of nodes i and j, respectively. The degrees include self-loops, ensuring that the contribution of each neighboring node is appropriately scaled relative to the graph structure.

   (d) $\mathbf{x}_j$: feature vector of j^{th} node

   This model ensures that each node's updated representation reflects both its own features and those of its neighbors, scaled by their relative importance in the graph.

2. GraphSAGE: The GraphSAGE operator is inspired from a graph isomorphism testing algorithm. [40]. This model samples a subset of neighbors and aggregates their features

using a specified function (mean was selected in this experiment). The node wise formulation is written as:

$$\mathbf{x}_i' = \mathbf{W}_1\mathbf{x}_i + \mathbf{W}_2 \cdot \text{mean}_{j\in\mathcal{N}(\rangle)}\mathbf{x}_j$$

Here, the node representation $\mathbf{x}_i'$ is updated based on the following components:

(a) $\mathbf{W}_1$ and $\mathbf{W}_2$: These matrices are learnable parameters that control how information is propagated between layers in the model. $\mathbf{W}_1$ transforms the node's own features, while $\mathbf{W}_2$ transforms the aggregated features from neighboring nodes.

(b) $\mathbf{x}_i$: The feature vector of the i^{th} node.

(c) $\text{mean}_{j\in\mathcal{N}(\rangle)}\mathbf{x}_j$: The mean of the feature vectors from the neighbors of node i.

This approach allows the model to generalize to unseen nodes by learning a function that generates embeddings based on sampled neighborhoods.

3. TransformerConv: It integrates Graph Neural Networks with label propagation algorithms and introduces a masked label prediction strategy to prevent overfitting. This model uses attention mechanisms to focus on the most relevant neighboring nodes [41]. The node wise formulation for TransformerConv can be written as :

$$\mathbf{x}_i' = \mathbf{W}_1\mathbf{x}_i + \sum_{j\in\mathcal{N}(i)} \alpha_{i,j}\mathbf{W}_2\mathbf{x}_j$$

Here the node representation $\mathbf{x}_i'$ is updated based on the following components:

(a) $\mathbf{W_1}$: A learnable weight matrix applied to the node's own feature vector $\mathbf{x_i}$, similar to GraphSAGE.

(b) $\mathbf{W_2}$: A weight matrix applied to the features of neighboring nodes after they have been weighted by the attention coefficients.

(c) $\alpha_{i,j}$ The attention coefficient that determines the importance of node j's contribution to node i's new representation. This coefficient is computed using a multi-head dot product attention mechanism, which allows the model to selectively focus on the most relevant neighbors.

The use of attention mechanisms in TransformerConv allows for a more nuanced aggregation of neighbor information, making it particularly powerful in capturing complex relationships within the graph.

4. GNN-BPR: This custom model integrates a GNN with Bayesian Personalized Ranking (BPR) to predict drug-disease interactions. This approach leverages the strengths of GNNs in learning from graph-structured data and BPR's effectiveness in ranking positive and negative drug-disease links[42] . BPR is a ranking-based learning algorithm typically used in collaborative filtering for recommendation systems. BPR is particularly effective for implicit feedback [43] scenarios where the goal is to rank items (e.g., drugs) based on user preferences (e.g., their associations with diseases). In the context of drug repurposing, implicit feedback allows models to infer the effectiveness or relevance of a drug for a previously unlinked disease based on observational data available within the system that the model is being trained on (the input graph). Instead of relying on explicit ratings or direct feedback from users (such as clinical trial outcomes or expert opinions), this technique allows the model to analyze patterns in the input graph to predict whether a drug might be approved for a new disease. This approach leverages existing data to make informed predictions about potential drug-disease relationships, enabling the identification of repurposing opportunities without the need for extensive new experimental data. The

GNN-BPR class takes the graph and a random set of triplets as input and generates the following sub components to produce the final output of $prediction_i$ and $prediction_j$.

In addition to the initial GNN pass based on the models discussed above, this custom GNN was implemented with the following components :

(a) Custom triplet generation: To train the model, triplets consisting of a drug, a linked disease (positive example), and a not-linked disease (negative example) are generated. A triplet would look like the following: *drug, linked disease, non-linked disease*

(b) Node Embeddings: The GNN-generated node embeddings corresponding to the drugs and diseases in each triplet are passed to the BPR model. The goal is to derive a personalized ranking of all diseases for all drugs following posterior probability where theta represents the parameter vector of the GNN

Given the embeddings generated by the selected GNN, the BPR model performs pairwise ranking. The embeddings for drugs u, positive diseases i, and negative diseases j are denoted as z_u,z_i, z_j respectively.

i. z_u: Embeddings of all the drugs in the graph.

ii. z_i: Embeddings of the diseases linked to the drugs.

iii. z_j: Embeddings of the diseases not linked to the drugs.

(c) Forward Pass: In the forward pass, the model computes the preference scores for pairs of items (drug-disease pairs). These predictions are made by calculating the dot product of the aforementioned embeddings, which measures the similarity between the drug and disease embeddings. A higher dot product indicates a stronger predicted association. The forward pass yields the following scores:

i. $prediction_i = {z_u}^\top z_i$, the predicted score for the known (positive) drug-disease interaction.

ii. $prediction_j = {z_u}^\top z_j$ the predicted score for the randomly sampled (negative) drug-disease interaction.

3.6 Difference in Objective and Loss Functions in BPR vs Non-BPR Models

The loss functions used in this experiment are aligned with different objective functions, reflecting how the model is encouraged to learn during training.

3.6.1 Loss Functions

- Binary Cross Entropy (BCE) Loss: The non-BPR model uses Binary Cross Entropy (BCE) Loss to perform binary classification for each drug-disease pair. The goal is to predict whether a specific drug-disease interaction exists [44]. The BCE loss function measures the discrepancy between the predicted probability and the actual label, guiding the model to improve its predictions over time. The objective is to predict whether a given drug-disease edge exists or not. In this experiment, the BCE loss function is specifically applied to edges representing drug-disease interactions in the graph, where a drug-disease edge is only included if a drug is approved for a disease.

  The loss function is defined as:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\sigma(\hat{y}_i)) + (1-y_i)\log(1-\sigma(\hat{y}_i))\right]$$

Where:

- $\hat{y}_i$ is the predicted logit for the i_{th} sample.
- y_i is the actual label (1 for an existing link, 0 for a non-existing link).
- σ is the sigmoid operation.

- BPR Loss: The BPR loss function is used to encourage the model to rank positive interactions higher than negative ones based on the pairwise ranking of drug-disease interactions learned from their embeddings. The BPR-GNN model optimizes this ranking by training the GNN to produce embeddings that are particularly suited for the BPR task. Similar to the BCE loss, in this experiment, BPR Loss is specifically applied to drug-disease edges, focusing on ranking known positive drug-disease interactions higher than unknown or negative ones. This allows the model to be optimized to improve the ranking of diseases for each drug by maximizing the likelihood of correct rankings in the observed preference data (existing drug-disease links).

The loss function is defined as:

$$\mathcal{L}_{\text{BPR}} = - \sum_{(u,i,j) \in \mathcal{D}_S} \log \sigma \left(x_{u,i} - x_{u,j} \right)$$

Where:

- $x_{u,i}$ is the predicted score for the known (positive) drug-disease interaction.
- $x_{u,j}$ is the predicted score for the randomly sampled (negative) drug-disease interaction.
- $\sigma(\cdot)$ is the sigmoid function, defined as $\sigma(x) = \frac{1}{1+e^{-x}}$.
- $\mathcal{D}_S$ is the set of all observed triplets (u, i, j), where u is a drug, i is a linked disease (positive example), and j is an unlinked disease (negative example).

3.7 Negative Sampling Approaches

Negative sampling has been considered a significant factor in motivating machine learning models to predict possible drug-disease link. Previous work has highlighted how the absence of True Negatives and the prevailing notion of false negatives (impossible and or drug-disease links with a low likelihood of existing) is a fundamental limitation of machine learning frameworks designed for drug repurposing [28, 29]. The drug repurposing pipeline developed in this experiment allows users to benchmark various GNN architectures trained on an input graph on the following negative sampling approaches :

1. Random Sampling: Randomly generated negative samples generated from a set of non-existent drug-disease edges in the training version. This set is a subset extracted from a set of all possible drug-disease edges with the positive edges removed.
2. Association score based sampling: In the OpenTargets Dataset, there exist the following 6 gene to disease association scores for linked gene-disease pairs in 6 different associations datasets.

(a) associationByDatasourceDirect

(b) associationByDatasourceInDirect

(c) associationByDatatypeDirect

(d) associationByDatatypeInDirect

(e) associationByOverallDirect

(f) associationByOverallInDirect

The aforementioned associations can be filtered by strength of association on the basis of a user defined threshold on the association score reported in each dataset [45]. Additionally, each molecule dataset contains the 'linkedTargets' column consisting of molecule-gene associations annotated based on literature. A join operation of the molecule-gene tables with gene-disease tables can yield a table with molecule-gene-disease triplets which can be ranked based on the gene-disease association scores. Each molecule-gene link provides us with a yes/no relationship between a gene and a disease. Each gene-disease link in the associations datasets provides a yes/no relationship along with a strength of the association. This relationship and its score can be used to generate scores which can be used to rank molecule-disease associations. The lowest ranking associations in this table can be used to generate negative samples for the model to train on.

A similar approach has been used in previous work where curated FDA-approved drug-biomarker combinations were considered true positives and as gold standard and all non-FDA-approved drug-biomarker combinations were labelled as true negatives to train the model [46]

3. BPR: Custom Embedding-Informed Triplets-Based Sampling: In this approach, the BPR framework is utilized to generate negative samples. This method involves creating triplets that consist of a drug, a positively associated disease, and a negatively associated disease. The negative samples are not selected randomly; instead, they are informed by the current embeddings of the model, focusing on challenging cases where the model might struggle to distinguish between positive and negative interactions. This approach is expected to improve the model's ranking capability by encouraging it to learn from difficult (least likely) examples, ultimately leading to more robust and accurate predictions.

3.8 Model Training, Optimization and Evaluation

Training: In this experiment, the GNN model was trained using different negative sampling approaches. Each approach was expected to help improve the model's ability to distinguish between positive and negative drug-disease interactions by providing varying types of negative samples during training.

Optimization: During training, the network optimized the parameters of the convolutional layers and the message-passing functions within the Graph Neural Network (GNN). For the random and association score-based sampling approaches, the Binary Cross Entropy (BCE) loss was used to train the model by calculating the loss between the predicted and actual interactions, with additional emphasis placed on negative samples through a weighted loss function. This process encouraged the GNN to learn embeddings that could correctly classify individual interactions.

In contrast, when the BPR-based sampling approach was used, the model was optimized using the BPR loss. This loss function encouraged the GNN to generate embeddings that helped

the BPR component rank positive interactions higher than negative ones. The training loop included generating triplets, computing the BPR loss, and updating the model parameters based on this loss, which focused on improving the relative ranking of drug-disease pairs.

Evaluation: To evaluate the performance of each model, the following four metrics from scikit-learn [47] were computed during training, including:

- **Average Precision Recall (APR)**: Measures the precision [48] of the model in ranking positive drug-disease links across various thresholds. It summarizes the precision-recall curve as the weighted mean of precision achieved at each threshold, with the increase in recall from the previous threshold used as the weight.

  $$\text{AP} = \sum_n (R_n - R_{n-1}) P_n$$

  Where P_n and R_n are the precision and recall at the n-th threshold.

- **Area Under the ROC Curve (AUC)**: Assesses the model's ability to discriminate between positive and negative drug-disease links. [49] A higher AUC value indicates better model performance, as it represents the probability that the model ranks a randomly chosen positive instance higher than a randomly chosen negative one.

  $$\text{AUC} = \int_0^1 \text{TPR}(\text{FPR}^{-1}(t))\, dt$$

  TPR (True Positive Rate) is also known as recall or sensitivity [48], and it is calculated as:

  $$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

  where TP represents the number of true positives (correctly predicted positive instances), and FN represents the number of false negatives (actual positive instances that were incorrectly predicted as negative).

  FPR (False Positive Rate) is calculated as:

  $$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

  where FP represents the number of false positives (actual negative instances that were incorrectly predicted as positive), and TN represents the number of true negatives (correctly predicted negative instances).

  t represents a threshold value ranging from 0 to 1. The expression $\text{TPR}(\text{FPR}^{-1}(t))$ refers to the true positive rate as a function of the false positive rate for a given threshold t.

  The AUC is obtained by integrating the true positive rate over the range of false positive rates from 0 to 1.

- **Recall**: Represents the proportion of actual positive drug-disease links that the model correctly identified. It is the ratio of true positives to the sum of true positives and false negatives.

  $$\text{Recall} = \frac{TP}{TP + FN}$$

  Where TP is the number of true positives and FN is the number of false negatives.

- **Accuracy**: Indicates the overall correctness of the model's predictions by considering both positive and negative drug-disease links. It is the ratio of correctly predicted instances (both true positives and true negatives) to the total number of instances.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

These metrics provided insights into how well the model generalized to unseen data, reflecting its ability to accurately predict drug-disease interactions in new, unobserved cases. These metrics were logged per epoch to track the model's performance and progress.

4 Results

The Results produced by the pipeline described in Figure 1 are explained in the sections below.

4.1 Description of the generated Graphs

The graph creation script was used to generate two distinct graphs using the schema discussed in the graph creation section above. Both graphs share the same node distribution, as illustrated in Figure 8.

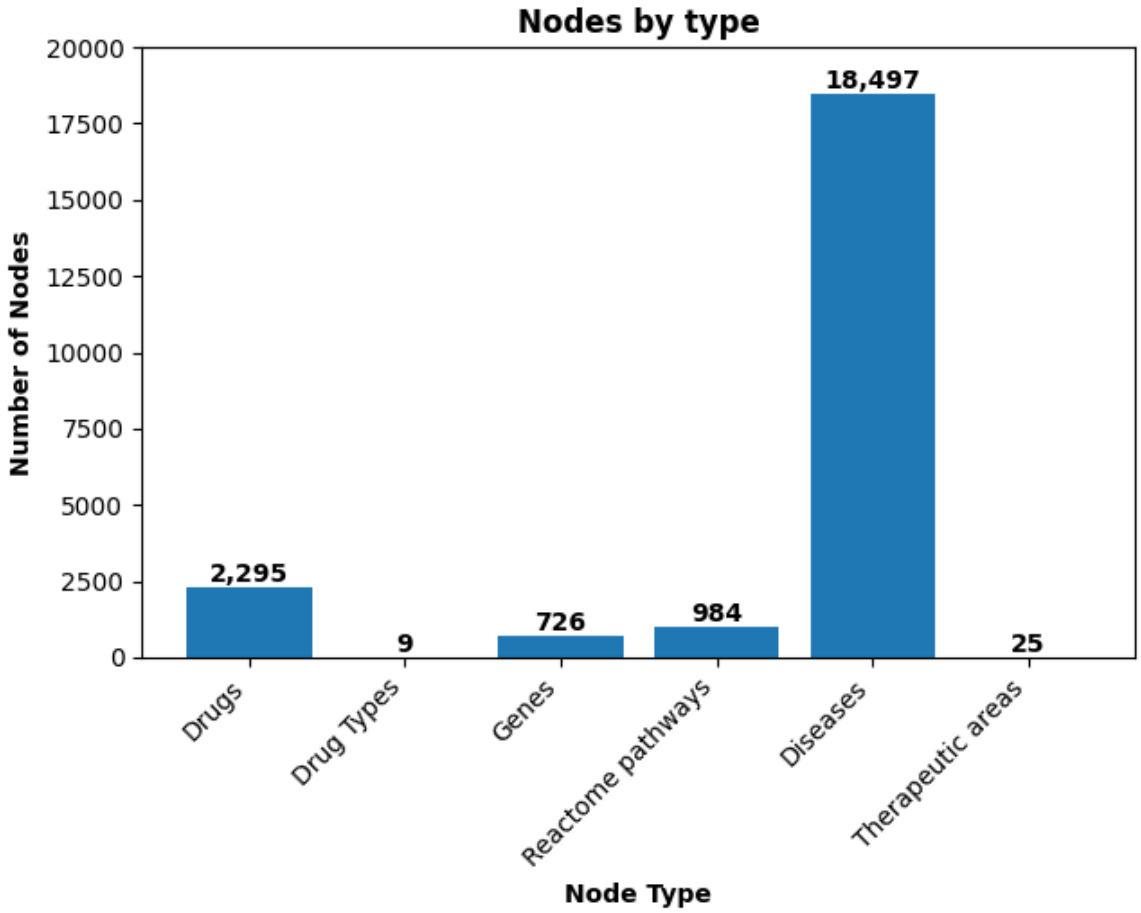

Figure 8: Node Distribution of Graph

4.2 Graph Analysis and Key Statistics

The two graphs generated were analyzed to understand their structural properties, which are crucial for interpreting the behavior and performance of Graph Neural Networks (GNNs) trained on these graphs. Below, the key statistics and their implications are discussed:

1. Graph with isolated nodes disease similarity network) This graph was generated following the schema described in section 3.4. Isolated nodes in this graph were identified, and it was observed that these nodes correspond to tissues and locations in organs that are linked to disease nodes in the 'disease' dataset. These isolated nodes do not belong to any therapeutic area and can only be connected to the rest of the graph by establishing relationships with the disease nodes. Isolated nodes are significant because they do not participate in the message-passing process of GNNs, potentially limiting the model's ability to learn from these nodes and potential edges.

2. Graph with Disease Similarity Network (Fully Connected) : This graph was generated to ensure all disease ontologies described in Table 1 are connected by their respective edges. In this fully connected graph, the absence of isolated nodes indicates that all nodes are part of the network's structure, allowing GNNs to effectively propagate information across the entire graph. The edge distribution of the 2 generated graphs are illustrated in Figure 9 and 12

respectively:

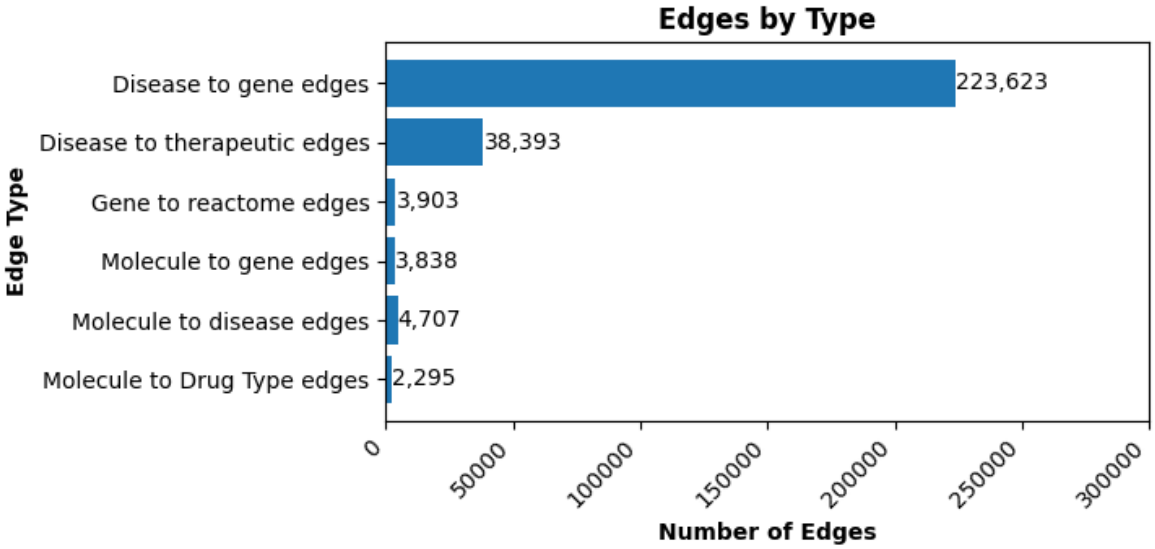

Figure 9: Edge Distribution of Graph with isolated disease ontology nodes

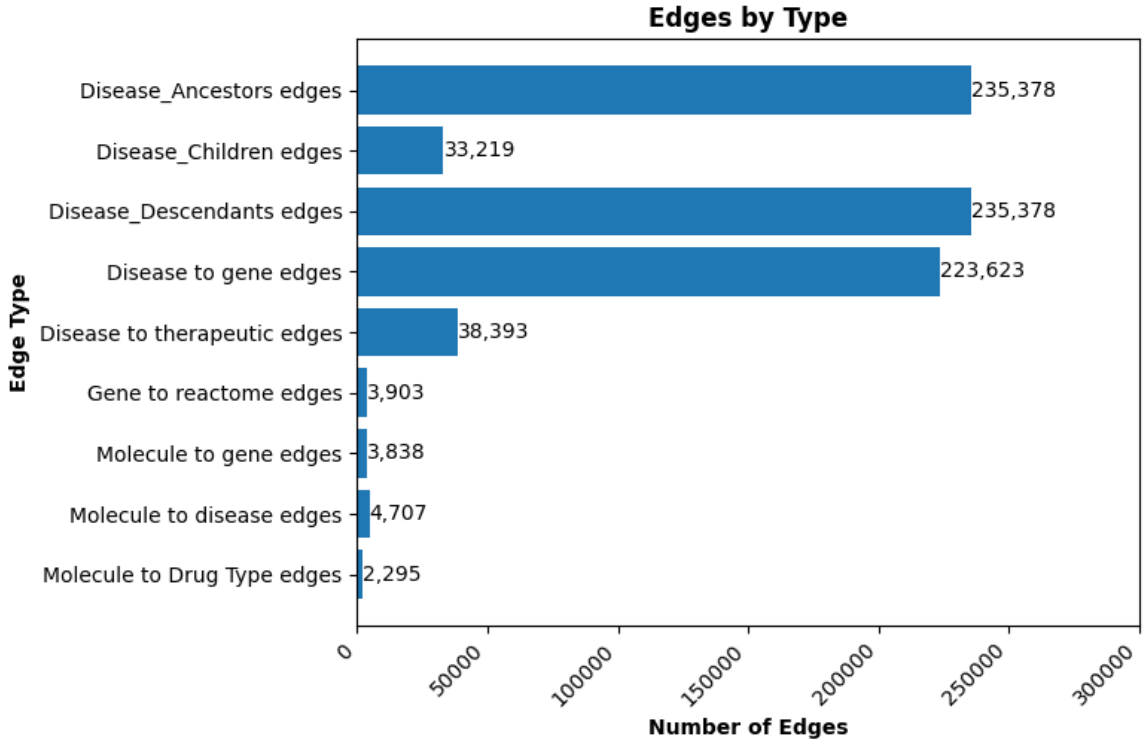

Figure 10: Edge Distribution of Fully connected graph with linked disease ontologies

Additional Graph Statistics:

Some basic statistics about both graphs (as discussed in section 3.4.3) are given in Table 3.

Table 3: Comparison of Graph Statistics

Graph Type	**Statistic**	**Value**
Graph with Isolated Nodes	Average Node Degree	24.56
	Number of Isolated Nodes	1028
	Median of Node Degree	16.0
	Assortativity Coefficient	-0.17
Fully Connected Graph	Average Node Degree	45.45
	Number of Isolated Nodes	0
	Median of Node Degree	3.0
	Assortativity Coefficient	-0.18

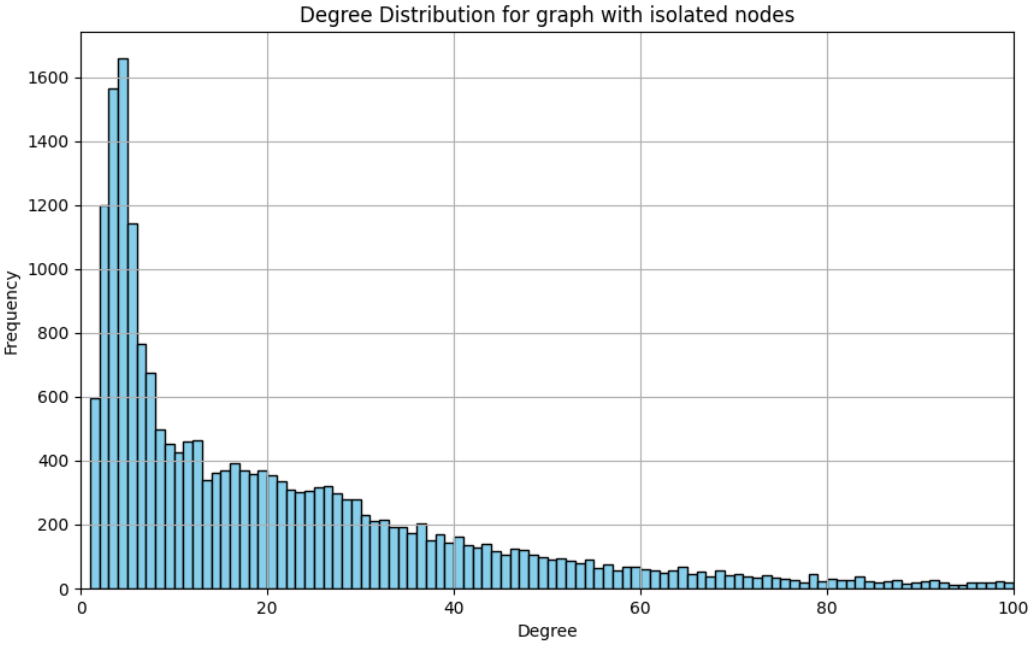

Figure 11: Degree Distribution of graph without linked disease ontologies

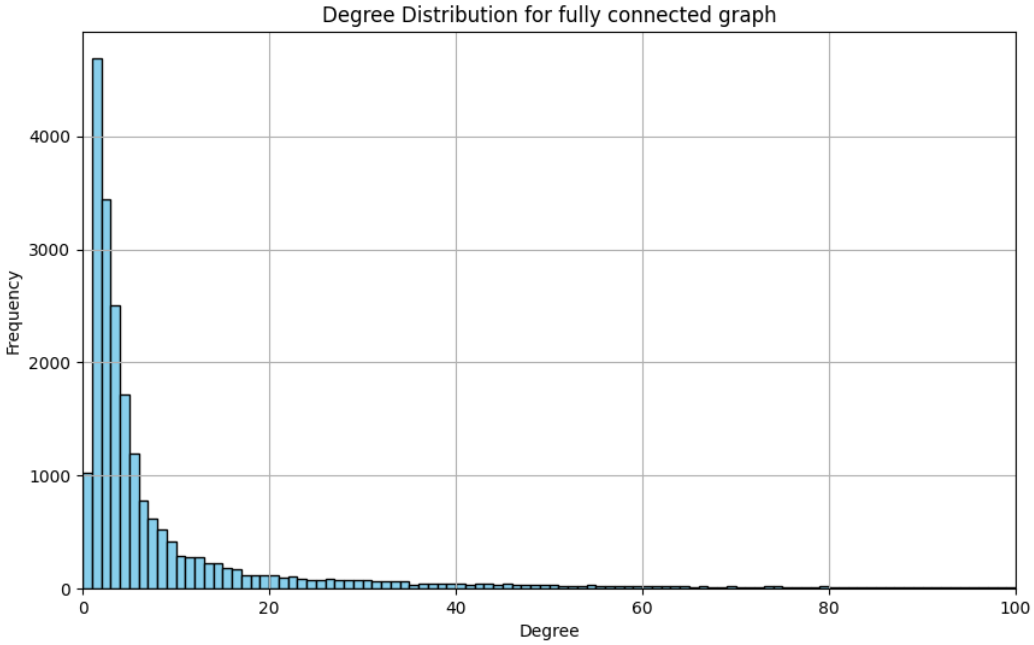

Figure 12: Degree Distribution of Fully connected graph with linked disease ontologies

4.3 Link Prediction Results

Link prediction with Graph Neural Networks (GNNs) involves forecasting the existence of a connection, or "link," between two nodes within a graph. In the context of drug repurposing, this means predicting potential new relationships between drugs and diseases that are not yet established. The GNN models leverage the topological structure of the graph and the features of the nodes to learn patterns that suggest where new links might form, effectively identifying drug-disease pairs that could be explored as new therapeutic opportunities.

In this experiment, all GNN models are trained, validated, and tested using datasets that only include drug-disease interactions where the drugs are already approved for at least one disease. Each model is trained on the representations of approved drugs from the training dataset (published in April 2021) to predict which drugs might receive repurposing approval in the validation set (published in June 2021). Finally, the models are evaluated on a test set published in April 2022. Unlike previous studies that validate and test model performance based on the prediction of drugs entering clinical trials, this pipeline assesses the models' ability to predict drugs that are likely to be approved by regulatory agencies.

The results of the GNN models (GCNConv, TransformerConv, and SAGEConv) trained with the graph with isolated nodes are summarized in Tables 4-6 below. These tables also capture the models' performance across each of the negative sampling approaches outlined in Section 3.7. Figures 11 shows the confusion matrix of the pipeline configuration which generated the highest accuracy.

Table 4: Training Results- Isolated Nodes

Model	Negative Sampling	APR	AUC	Accuracy	Recall
GCNConv	Random Sampling	97.13%	97.00%	77.49%	60.99%
	AS: ByOverallDirect	89.25%	88.89%	62.61%	32.02%
	BPR	99.76%	99.84%	96.09%	99.98%
GraphSAGE	Random Sampling	99.39%	99.56%	67.45%	64.12%
	AS: ByOverallDirect	97.10%	96.87%	62.65%	34.08%
	BPR	99.89%	99.89%	98.57%	99.87%
TransformerConv	Random Sampling	99.76%	99.80%	72.90%	63.95%
	AS: ByOverallDirect	99.58%	99.53%	90.44%	84.64%
	BPR	99.98%	99.98%	99.15%	100.00%

Table 5: Validation Results - Isolated Nodes

Model	Negative Sampling	APR	AUC	Accuracy	Recall
GCNConv	Random Sampling	87.31%	90.62%	64.39%	32.60%
	AS: ByOverallDirect	65.34%	65.99%	59.46%	33.40%
	BPR	99.27%	99.33%	93.86%	99.80%
GraphSAGE	Random Sampling	97.02%	98.18%	74.55%	49.90%
	AS: ByOverallDirect	54.14%	58.82%	47.18%	27.57%
	BPR	99.34%	99.33%	95.67%	95.17%
TransformerConv	Random Sampling	97.04%	98.18%	71.83%	44.47%
	AS: ByOverallDirect	69.58%	76.79%	63.48%	48.49%
	BPR	99.56%	99.51%	96.48%	96.38%

Table 6: Test Results - Isolated Nodes

Model	Negative Sampling	APR	AUC	Accuracy	Recall
GCNConv	Random Sampling	83.00%	84.42%	28.02%	63.51%
	AS: ByOverallDirect	45.94%	52.63%	20.79%	54.02%
	BPR	86.56%	89.21%	79.51%	82.14%
GraphSAGE	Random Sampling	91.25%	91.69%	37.73%	68.03%
	AS: ByOverallDirect	49.11%	45.41%	20.44%	41.58%
	BPR	88.69%	89.77%	68.34%	81.54%
TransformerConv	Random Sampling	88.29%	89.86%	37.38%	67.91%
	AS: ByOverallDirect	69.24%	63.95%	43.55%	58.22%
	BPR	89.09%	90.07%	64.14%	80.17%

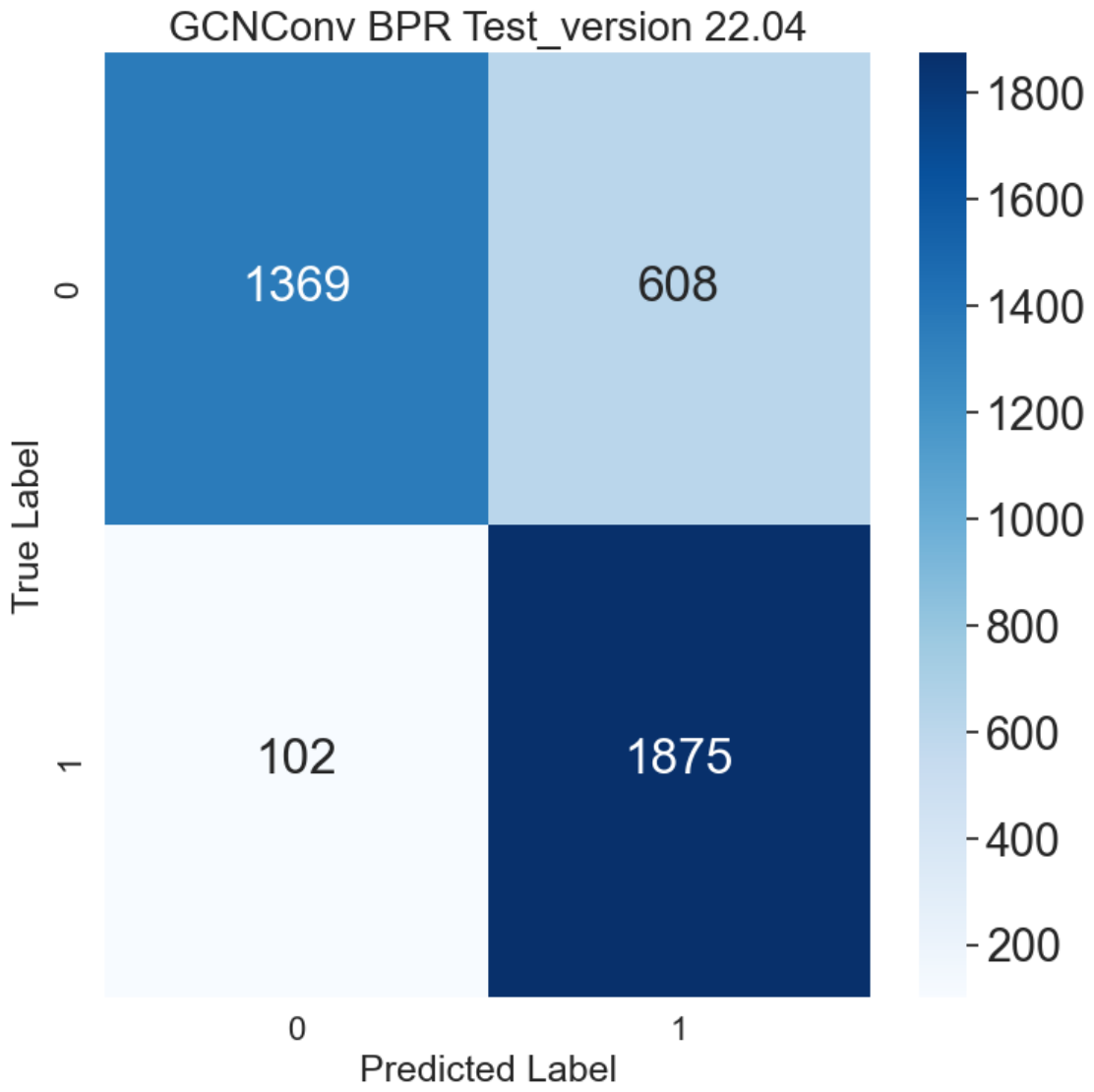

Figure 13: Confusion matrix of GCNConv trained on a graph without disease similarity network edges; BPR negative sampling (AUC:89%, APR:86 %, Recall:79 %, Accuracy: 82%)

Similarly, the results for the fully connected graph are presented in Tables 7-9 and Figures 12 shows the confusion matrix of the pipeline configuration which generated the highest accuracy.

Table 7: Training Results - Fully Connected Graph

Model	Negative Sampling	APR	AUC	Recall	Accuracy
GCNConv	Random Sampling	95.06%	94.87%	82.49%	88.50%
	AS: ByOverallDirect	85.02%	83.38%	70.66%	76.26%
	BPR	96.21%	97.38%	100.00%	72.34%
GraphSAGE	Random Sampling	75.00%	74.01%	3.48%	50.78%
	AS: ByOverallDirect	58.70%	58.59%	100.00%	50.00%
	BPR	98.07%	98.82%	100.00%	63.73%
TransformerConv	Random Sampling	96.58%	97.04%	38.62%	67.80%
	AS: ByOverallDirect	89.11%	87.29%	21.86%	57.33%
	BPR	98.94%	99.32%	100.00%	88.12%

Table 8: Validation Results - Fully Connected Graph

Model	Negative Sampling	APR	AUC	Recall	Accuracy
GCNConv	Random Sampling	91.65%	90.48%	72.43%	83.50%
	AS: ByOverallDirect	74.88%	64.75%	61.57%	62.47%
	BPR	93.35%	94.75%	100.00%	66.40%
GraphSAGE	Random Sampling	65.10%	76.80%	13.48%	51.41%
	AS: ByOverallDirect	73.26%	76.52%	99.80%	56.54%
	BPR	93.78%	95.69%	100.00%	66.40%
TransformerConv	Random Sampling	89.70%	89.08%	33.40%	66.20%
	AS: ByOverallDirect	87.57%	86.30%	31.59%	65.39%
	BPR	98.07%	98.72%	100.00%	91.25%

Table 9: Test Results - Fully Connected Graph

Model	Negative Sampling	APR	AUC	Accuracy	Recall
GCNConv	Random Sampling	75.66%	79.79%	49.77%	72.10%
	AS: ByOverallDirect	51.55%	62.80%	41.68%	55.11%
	BPR	90.82%	88.52%	98.89%	69.27%
GraphSAGE	Random Sampling	77.67%	76.24%	25.64%	60.70%
	AS: ByOverallDirect	71.49%	65.06%	96.56%	58.60%
	BPR	92.89%	91.61%	97.98%	68.11%
TransformerConv	Random Sampling	79.58%	81.36%	20.89%	59.71%
	AS: ByOverallDirect	78.38%	79.74%	19.17%	58.98%
	BPR	91.12%	90.97%	88.67%	84.37%

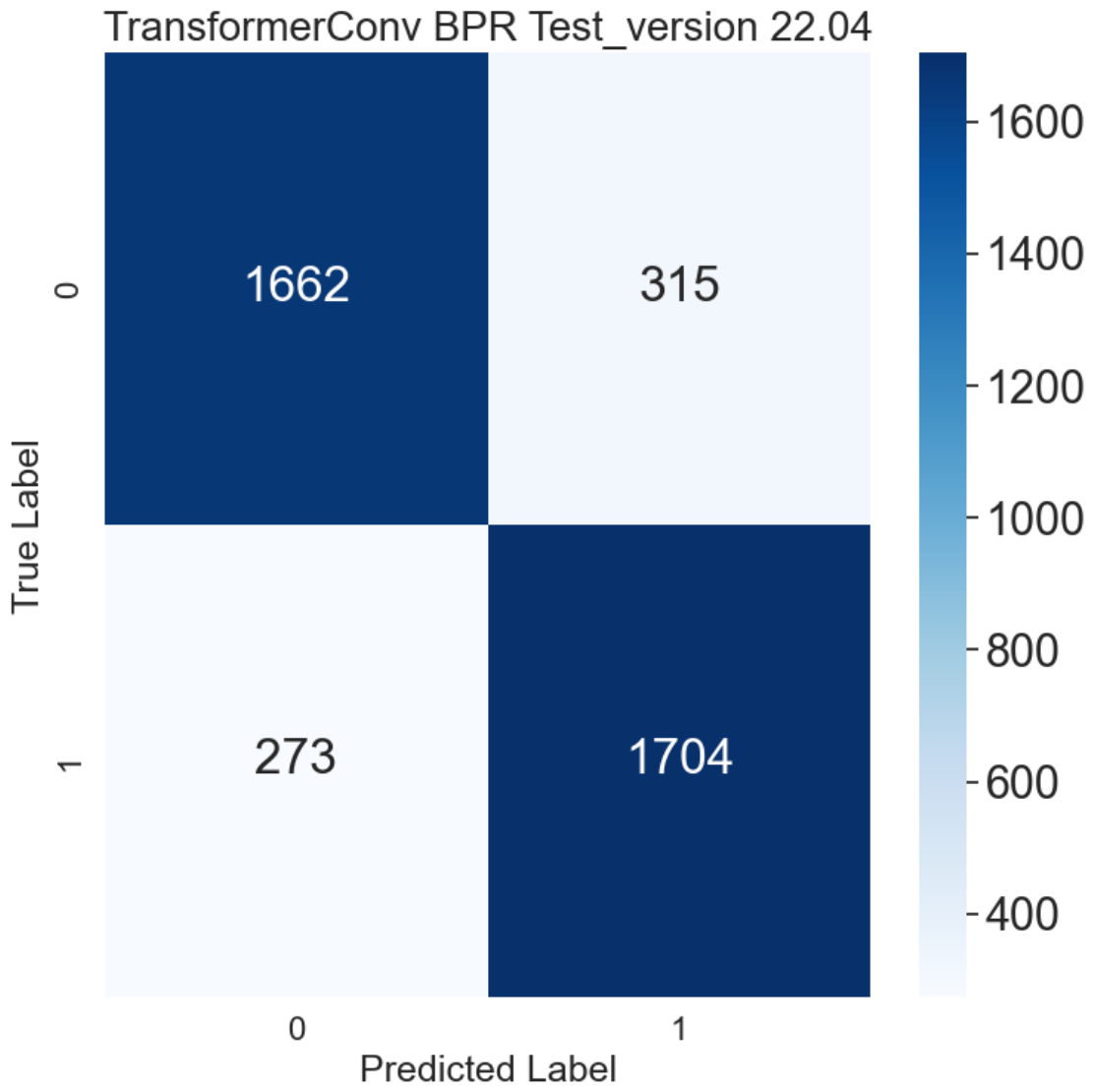

Figure 14: Confusion matrix of TransformerConv trained on a fully connected graph with disease similarity network edges; BPR negative sampling (AUC: 91%, APR: 90 %, Recall: 88 %, Accuracy: 84%).

Note: A detailed comparison of each model's performance with bar plots is available in figures S1- S8 in the Appendix

4.4 Observations on Overfitting

For the initial versions of the pipeline, all models were trained using different configurations for 1,000 epochs. However, it quickly became evident that overfitting occurred after just 100 epochs, as both training and validation performance began to diverge. To address this, the decision was made to focus on the results from models trained for 100 epochs, which offered a better balance between computational efficiency and performance. The models demonstrated strong

performance within this reduced training period, without the risks associated with overfitting. Additionally, testing the models at 200 epochs (Figure 16) further confirmed signs of overfitting, reinforcing the choice to limit training to 100 epochs for optimal results.

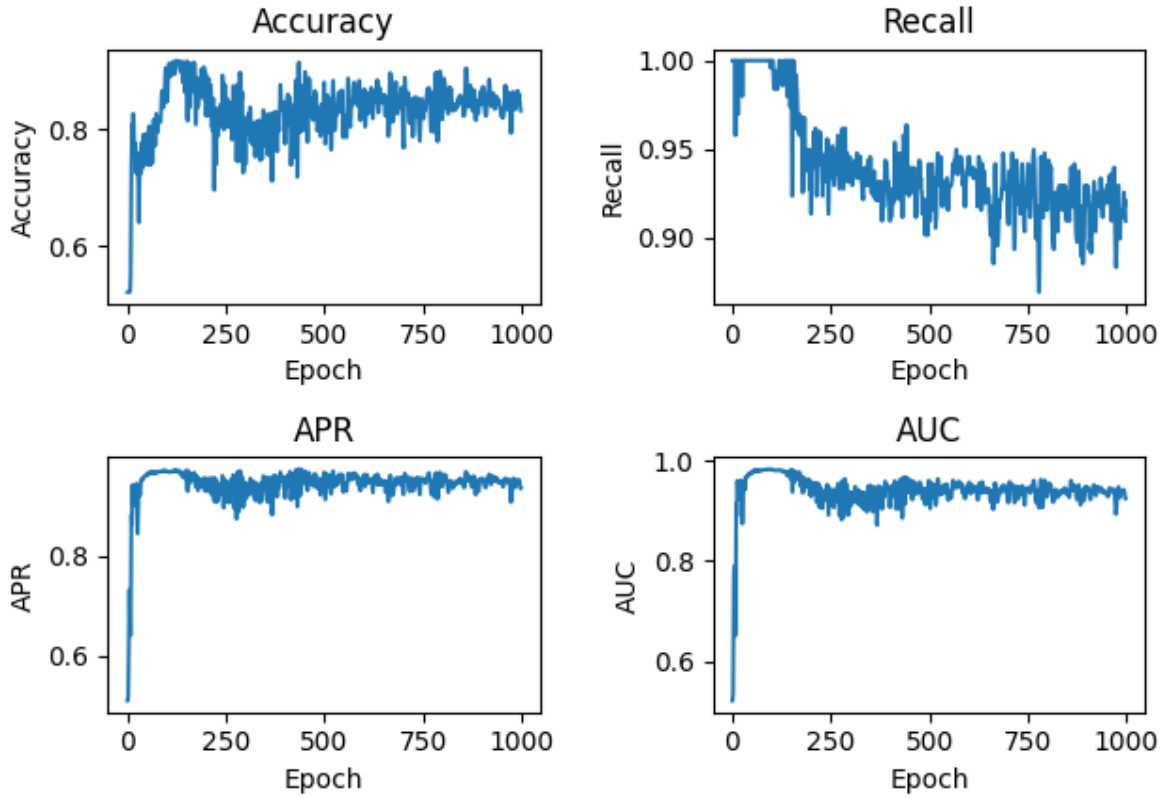

Figure 15: Model Performance on 1000 epochs

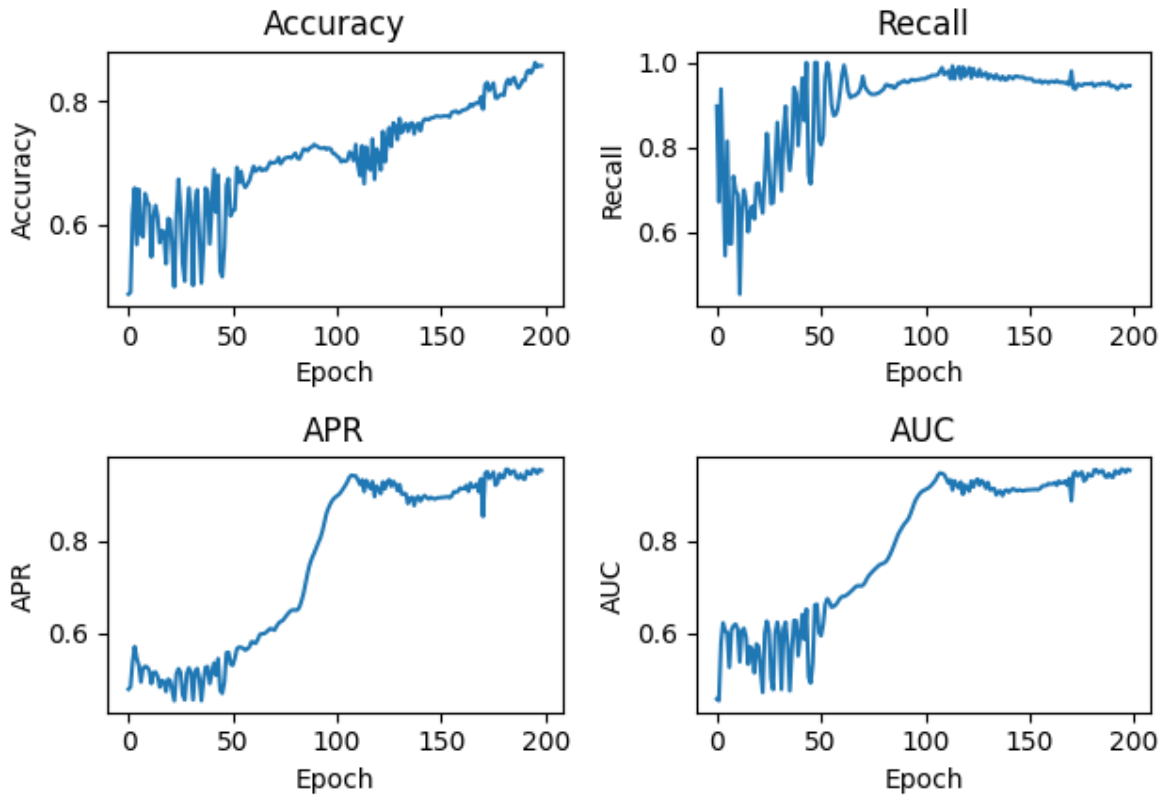

Figure 16: Model Performance on 200 epochs

5 Discussion

The pipeline developed in this project was designed to help users determine optimal parameters for a GNN to predict the best candidates for repurposing from the given training dataset. Two variations of a knowledge graph were generated using nodes, features and edges from the OpenTargets Dataset and they were used to train multiple GNN architectures to determine the best performing models. The pipeline was also designed to allow the use of different negative sampling approaches. Previous work has illustrated how providing true negatives (i.e., drug-disease links that do not not lead to favorable treatment outcomes) helps models predict better candidates for repurposing trials [28, 29]. But many of these studies have trained models to product whether candidate for clinical trials.

5.1 Impact of Negative sampling

5.1.1 Random sampling vs Association scores based sampling

Building on previous research [28, 50], this project employs various negative sampling strategies, all of which focus on non-existent drug-disease links in the training dataset. The GNNs are trained to differentiate between existing and non-existing edges by learning from the dot product of drug and disease embeddings. The findings from this experiment clearly indicate that the

choice of negative sampling method significantly influences the predictive performance of the pipeline—potentially even more than other factors.

In an attempt to improve model accuracy, association scores discussed in Section 3.7 were used to generate negative samples by ranking molecule-disease associations. Surprisingly, the results revealed that using these association scores did not yield better performance than random sampling. In fact, models trained with association-based negative sampling performed worse in some cases.

One possible explanation for this outcome is that the models might be implicitly learning from the patterns in how the FDA has historically approved drugs for diseases [31]. While genetic evidence is often cited as a key factor in FDA approvals, the correct predictions made by the model do not always align with this rationale. This suggests that the models may be capturing other factors beyond genetic associations, which influence FDA decisions.

The performance of the models, particularly when provided with negative samples lacking strong genetic evidence, suggests that genetic associations may not be the most critical factor in the representations of drug-disease edges within the graphs used in this experiment. The GNNs in this pipeline might have underperformed with poorly supported genetic evidence because they learned that FDA approvals are not strictly dependent on molecule-gene-disease associations as captured in the gene-disease association datasets and the 'linkedTargets' columns from the OpenTargets dataset.

Given the possibility of missing annotations in the dataset, rather than incorrect ones, it is plausible that the FDA does not always rely solely on genetic evidence when approving drugs. This might explain why the models did not perform well when the negative sampling strategy heavily depended on genetic associations.

5.1.2 BCE Loss vs BPR Loss Functions

Binary Cross Entropy (BCE) loss is widely used in binary classification tasks, such as predicting whether an existing or non-existent drug-disease link is present. BCE works by minimizing the discrepancy between the model's predicted probabilities and the actual labels for each drug-disease pair independently. The objective is to correctly classify each drug-disease pair as either linked (existing) or not linked (non-existent in the test set). However, BCE can struggle to generalize relationships within a graph when negative samples are not sufficiently challenging or informative. This limitation arises because BCE treats each prediction in isolation, which can lead to sub-optimal learning, especially in cases where many negative samples (non-existent drug-disease relationships) are easy for the model to distinguish from positive samples.

In contrast, the Bayesian Personalized Ranking (BPR) loss function is designed for ranking tasks, where the focus is on the relative order of existing and non-existent interactions rather than their independent classification. BPR operates by comparing pairs of interactions, encouraging the model to assign higher scores to actual (existing) drug-disease links than to non-existent ones. This pairwise comparison better aligns with the relational structure of graphs, making it particularly effective for tasks like drug repurposing. In these tasks, the goal is often to identify the most promising new links, and BPR's ranking-based approach helps the model to prioritize these potential discoveries more effectively.

While the BCE and BPR loss functions specifically evaluate the accuracy of predictions regard-

ing existing and non-existent drug-disease links, the overall node representations in the graph are influenced by all types of edges present in the graph (e.g., drug-target, disease-gene). During training, the model learns to aggregate information from various connected nodes, meaning that different types of relationships contribute to the final node embeddings. These embeddings are then used to make predictions about drug-disease interactions.

Even though the loss functions are used to optimize the model based on the correctness of drug-disease link predictions, the quality of these predictions is inherently tied to the quality of the node representations. These representations, in turn, are shaped by the entire graph structure, including all edge types. Therefore, the influence of other edge types on the weight matrices and, consequently, on the model's predictions while small does exist. This is indicated by the difference in model performance for TransformerConv on both graphs. Added graph connectivity for disease ontology nodes seems to have increased the APR, Recall and Accuracy of the model as shown Table 6 and Table 9.

5.2 Challenges with the Dataset/Pipeline

5.2.1 Challenges with the OpenTargets Platform and Dataset

The OpenTargets platform and datasets present several challenges:

1. **Target-Centric Focus**: The OpenTargets platform is designed primarily for target identification, not for drugs or molecules like DrugBank or ChEMBL. As a result, the platform provides limited explainability and annotation for drugs compared to the extensive bioinformatics data available on OpenTargets and OpenTargets Genetics.
2. **Schema Changes**: The platform has undergone schema changes in the 'molecule' and 'targets' datasets across versions, such as changes in column names and node types, without clear notifications in the release notes.
3. **Lack of Clear Annotations**: There is a lack of clear annotations for the column names used in the datasets. Schema information for interpreting column names is only available through the GraphQL schema documentation.
4. **Community Support**: The author experienced a lack of community support when seeking answers to queries about the dataset. Despite posting on the OpenTargets Community Blog, no response or resources were provided to address basic questions about annotations.

5.2.2 Challenges with the Pipeline Tooling

The current pipeline faces several challenges and limitations:

1. **Edge Imbalance and Sliding Window Approach**: The imbalance between positive and negative edges, combined with the sliding window approach, limits the availability of useful triplets for train/validation/test splits.
2. **Association Score Interpretation**: The association score datasets are difficult to interpret. Although they offer a method for weighing evidence from various sources, the

significance of the six different gene-disease association variants is unclear, making it hard to understand their relevance for model learning. Additionally, the documentation cautions against using these scores as confidence indicators for target-disease associations, further undermining their reliability.

3. **Text Embeddings**: Incorporating natural language text fields for molecules, diseases, and genes as word embeddings could have enhanced this pipeline by providing richer node features. However, choosing the fastest encoder between BioBERT [51], SciKitBio [52] and Word2Vec [22] requires additional testing with various datasets. When word embeddings were used, the pipeline struggled with memory constraints during graph creation and increased computational cost during training. Since the pipeline's best feature is graph creation from a query/schema this feature was excluded to maintain the pipeline's response time and cost-effectiveness.

4. **Feature Matrix Creation**: Creating the feature matrix from the feature encoding map was particularly challenging given the authors' current programming skills. Developing a word embeddings function, creating an algorithm for feature matrix generation and alignment, and ensuring pipeline flexibility across datasets were the most difficult tasks. Of these, only the pipeline's flexibility across different versions was successfully addressed.

5. **Edge Weights**: The pipeline does not currently support assigning gene associations as edge weights, which could potentially improve performance. Adding this feature would also allow for edge weights to be extracted from the OpenTargets Genetics platform.

5.2.3 Implications for Future Research

Enhancing the current version of the pipeline could lead to several practical applications, both in terms of improving performance and interpreting results:

1. **Custom Graph Creation**: The current pipeline lacks the ability to create custom graphs with specific schemas of nodes, features, and edges. However, it does support graph creation with varied edge nodes of the same types from different versions of the same datasets, offering some flexibility in including and excluding real-world data as it is published on the OpenTargets platform.

2. **Parameter Selection for Graph Entities**: Adding functionality for parameter selection with a schema would enable users to incorporate more graph entities, such as new node types, edge types, and edge weights, into their graph representations. While the current version allows for some addition and removal of edges, not all possible combinations of graph entities were used to train models. It's possible that the model's current performance benefits from avoiding the curse of dimensionality [53], which could have occurred if all possible features and graph entities were included in the training graph. However, it is not possible to confirm this without training the models with a graph that has more entities.

3. **Hyperparameter Tuning**: The current pipeline design does not include functionality for hyperparameter tuning. Implementing this feature would allow users to fine-tune the models and configurations tested within the pipeline, leading to potentially better results.

4. **Interpretability of Results**: To assess the real-world usefulness of these models and the pipeline, it would be beneficial to add a querying functionality. This would enable users

to interpret the model's results by generating a list of the best predictions made by the model, thereby increasing the pipeline's overall usefulness and interpretability. A possible solution to tooling limitations could involve creating a well-designed Python class module for easy tensor creation from Parquet datasets. Since PyArrow is platform-agnostic and widely available, extending the PyArrow module with such a class could address the tooling limitations faced during the graph creation process. If the design principles for this class are kept simple, the objects created could also be used to generate sub-graphs for querying both trained models and for testing already trained models.

5. **Custom Negative Sampling**: With minor adjustments, the current pipeline could be configured to accept custom negative samples for evaluating different models. Failed trial results, for instance, could be used for negative sampling, similar to approaches taken in previous research.

References

[1] Ted T Ashburn and Karl B Thor. Drug repositioning: identifying and developing new uses for existing drugs. *Nat. Rev. Drug Discov.*, 3(8):673–683, August 2004.

[2] European medicines agency (ema). `https://www.ema.europa.eu/en/documents/assessment-report/hemangiol-epar-public-assessment-report_en.pdf`. [Accessed 11-04-2024].

[3] Guillaume Canaud, Juan Carlos Lopez Gutierrez, Alan D. Irvine, Pierre Vabres, Jordan R. Hansford, Nii Ankrah, Fabrice Branle, Athanasia Papadimitriou, Antonia Ridolfi, Paul O'Connell, Stuart Turner, and Denise M. Adams. Alpelisib for treatment of patients with pik3ca-related overgrowth spectrum (pros). *Genetics in Medicine*, 25(12):100969, 2023.

[4] Lily Hoffman-Andrews. The known unknown: the challenges of genetic variants of uncertain significance in clinical practice. *J. Law Biosci.*, 4(3):648–657, December 2017.

[5] Maria Cristina De Rosa, Rituraj Purohit, and Alfonso T García-Sosa. Drug repurposing: a nexus of innovation, science, and potential. *Sci. Rep.*, 13(1):17887, October 2023.

[6] Nicola Nosengo. Can you teach old drugs new tricks? *Nature*, 534(7607):314–316, June 2016.

[7] Alexander S Hauser, Misty M Attwood, Mathias Rask-Andersen, Helgi B Schiöth, and David E Gloriam. Trends in GPCR drug discovery: new agents, targets and indications. *Nat. Rev. Drug Discov.*, 16(12):829–842, December 2017.

[8] Iwona E Weidlich, Igor V Filippov, Jodian Brown, Neerja Kaushik-Basu, Ramalingam Krishnan, Marc C Nicklaus, and Ian F Thorpe. Inhibitors for the hepatitis C virus RNA polymerase explored by SAR with advanced machine learning methods. *Bioorg. Med. Chem.*, 21(11):3127–3137, June 2013.

[9] Kai Zhao and Hon-Cheong So. Drug repositioning for schizophrenia and depression/anxiety disorders: A machine learning approach leveraging expression data. *IEEE J. Biomed. Health Inform.*, 23(3):1304–1315, May 2019.

[10] Dong-Sheng Cao, Liu-Xia Zhang, Gui-Shan Tan, Zheng Xiang, Wen-Bin Zeng, Qing-Song Xu, and Alex F Chen. Computational prediction of DrugTarget interactions using chemical, biological, and network features. *Mol. Inform.*, 33(10):669–681, October 2014.

[11] Alexander Aliper, Sergey Plis, Artem Artemov, Alvaro Ulloa, Polina Mamoshina, and Alex Zhavoronkov. Deep learning applications for predicting pharmacological properties of drugs and drug repurposing using transcriptomic data. *Mol. Pharm.*, 13(7):2524–2530, July 2016.

[12] Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do, Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, Wei Xie, Gail L Rosen, Benjamin J Lengerich, Johnny Israeli, Jack Lanchantin, Stephen Woloszynek, Anne E Carpenter, Avanti Shrikumar, Jinbo Xu, Evan M Cofer, Christopher A Lavender, Srinivas C Turaga, Amr M Alexandari, Zhiyong Lu, David J Harris, Dave DeCaprio, Yanjun Qi, Anshul Kundaje, Yifan Peng, Laura K Wiley, Marwin H S Segler, Simina M Boca, S Joshua Swamidass, Austin Huang, Anthony Gitter, and Casey S Greene. Opportunities and obstacles for deep learning in biology and medicine. *J. R. Soc. Interface*, 15(141):20170387, April 2018.

[13] Xiangxiang Zeng, Siyi Zhu, Xiangrong Liu, Yadi Zhou, Ruth Nussinov, and Feixiong Cheng. deepDR: a network-based deep learning approach to in silico drug repositioning. *Bioinformatics*, 35(24):5191–5198, December 2019.

[14] ChEMBL Database — ebi.ac.uk. `https://www.ebi.ac.uk/chembl/`. [Accessed 22-May-2023].

[15] DrugBank Online — Database for Drug and Drug Target Info — go.drugbank.com. `https://go.drugbank.com/`. [Accessed 22-May-2023].

[16] PubChem. PubChem — pubchem.ncbi.nlm.nih.gov. `https://pubchem.ncbi.nlm.nih.gov/`. [Accessed 22-May-2023].

[17] RCSB Protein Data Bank. RCSB PDB — rcsb.org. `https://www.rcsb.org/search`. [Accessed 22-May-2023].

[18] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nat. Rev. Genet.*, 12(1):56–68, January 2011.

[19] Barbara Hammer and Brijnesh J Jain. Neural methods for non-standard data. In *ESANN*, pages 281–292. Citeseer, 2004.

[20] Kamilia Zaripova, Luca Cosmo, Anees Kazi, Seyed-Ahmad Ahmadi, Michael M Bronstein, and Nassir Navab. Graph-in-Graph (GiG): Learning interpretable latent graphs in non-euclidean domain for biological and healthcare applications. *Med. Image Anal.*, 88(102839):102839, August 2023.

[21] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Trans. Neural Netw.*, 20(1):61–80, January 2009.

[22] word2vec — Text — TensorFlow — tensorflow.org. `https://www.tensorflow.org/text/tutorials/word2vec`. [Accessed 28-08-2024].

[23] Gensim: topic modelling for humans — radimrehurek.com. `https://radimrehurek.com/gensim/models/doc2vec.html`. [Accessed 26-08-2024].

[24] Daniel Scott Himmelstein, Antoine Lizee, Christine Hessler, Leo Brueggeman, Sabrina L Chen, Dexter Hadley, Ari Green, Pouya Khankhanian, and Sergio E Baranzini. Systematic integration of biomedical knowledge prioritizes drugs for repurposing. *Elife*, 6, September 2017.

[25] Wenhui Wang, Sen Yang, Xiang Zhang, and Jing Li. Drug repositioning by integrating target information through a heterogeneous network model. *Bioinformatics*, 30(20):2923–2930, October 2014.

[26] Mengying Sun, Sendong Zhao, Coryandar Gilvary, Olivier Elemento, Jiayu Zhou, and Fei Wang. Graph convolutional networks for computational drug development and discovery. *Brief. Bioinform.*, 21(3):919–935, May 2020.

[27] Jonathan M Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M Donghia, Craig R MacNair, Shawn French, Lindsey A Carfrae, Zohar Bloom-Ackermann, Victoria M Tran, Anush Chiappino-Pepe, Ahmed H Badran, Ian W Andrews, Emma J Chory, George M Church, Eric D Brown, Tommi S Jaakkola, Regina Barzilay, and James J Collins. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702.e13, February 2020.

[28] Evaluating the performance of drug-repurposing technologies — sciencedirect.com. `https://www.sciencedirect.com/science/article/pii/S1359644621003603`. [Accessed 26-08-2024].

[29] Manh Hung Le, Nam Anh Dao, and Xuan Tho Dang. High potential negative sampling for drug disease association prediction. In *Studies in Systems, Decision and Control*, Studies in systems, decision and control, pages 55–70. Springer Nature Switzerland, Cham, 2024.

[30] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. 2017.

[31] David Ochoa, Mohd Karim, Maya Ghoussaini, David G Hulcoop, Ellen M McDonagh, and Ian Dunham. Human genetics evidence supports two-thirds of the 2021 FDA-approved drugs. *Nat. Rev. Drug Discov.*, 21(8):551, August 2022.

[32] Apache arrow v17.0.0 documentation. `https://arrow.apache.org/docs/python/index.html`. [Accessed 13-08-2024].

[33] Index of /pub/databases/opentargets/platform — ftp.ebi.ac.uk. `https://ftp.ebi.ac.uk/pub/databases/opentargets/platform/`. [Accessed 27-08-2024].

[34] Degree Analysis &x2014; NetworkX 3.3 documentation — networkx.org. `https://networkx.org/documentation/stable/auto_examples/drawing/plot_degree.html`. [Accessed 28-08-2024].

[35] Mustafa Yasir, John Palowitch, Anton Tsitsulin, Long Tran-Thanh, and Bryan Perozzi. Examining the effects of degree distribution and homophily in graph learning models. 2023.

[36] degree_assortativity_coefficient &x2014; NetworkX 3.3 documentation — networkx.org. `https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.assortativity.degree_assortativity_coefficient.html`. [Accessed 28-08-2024].

[37] Susheel Suresh, Vinith Budde, Jennifer Neville, Pan Li, and Jianzhu Ma. Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns. 2021.

[38] Christo Sasi. Gnn benchmarking for drug repurposing. `https://gitlab.cmbi.umcn.nl/cmbi/drugrepo2023/-/blob/master/Final_pipeline_script.py`, 2024. [Created 28-Aug-2024].

[39] Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. 09 2016.

[40] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc.

[41] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification, 2021.

[42] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009*, 05 2012.

[43] Douglas W Oard and Jinmook Kim. Implicit feedback for recommender systems. `https://terpconnect.umd.edu/~oard/pdf/aaai98.pdf`. Accessed: 2024-8-26.

[44] BCEWithLogitsLoss &x2014; PyTorch 2.4 documentation — pytorch.org. `https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html#torch.nn.BCEWithLogitsLoss`. [Accessed 27-08-2024].

[45] Target - disease associations — Open Targets Platform Documentation — platform-docs.opentargets.org. `https://platform-docs.opentargets.org/associations`. [Accessed 26-08-2024].

[46] Imogen R Walpole, Farzana Y Zaman, Peinan Zhao, Vikki M Marshall, Frank P Lin, David M Thomas, Mark Shackleton, Albert A Antolin, and Malaka Ameratunga. Computational repurposing of oncology drugs through off-target drug binding interactions from pharmacological databases. *Clin. Transl. Med.*, 14(4):e1657, April 2024.

[47] Scikit-learn metrics readme. `https://github.com/scikit-learn/scikit-learn/blob/main/sklearn/metrics/_ranking.py`. [Accessed 16-08-2024].

[48] average_precision_score — scikit-learn.org. `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.average_precision_score.html`. [Accessed 29-08-2024].

[49] roc_auc_score — scikit-learn.org. `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html#roc-auc-score`. [Accessed 29-08-2024].

[50] Manh Hung Le, Nam Anh Dao, and Xuan Tho Dang. *High Potential Negative Sampling for Drug Disease Association Prediction*, pages 55–70. Springer Nature Switzerland, Cham, 2024.

[51] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, February 2020.

[52] Biological Embeddings (skbio.embedding) &x2014; scikit-bio 0.6.3-dev documentation — scikit.bio. `https://scikit.bio/docs/dev/embedding.html`. [Accessed 28-08-2024].

[53] T Poggio and Q Liao. Theory i: Deep networks and the curse of dimensionality. *Bull. Pol. Acad. Sci. Tech. Sci.*, November 2023.

6 Appendix

6.1 Supplementary Bar plots generated with the pipeline

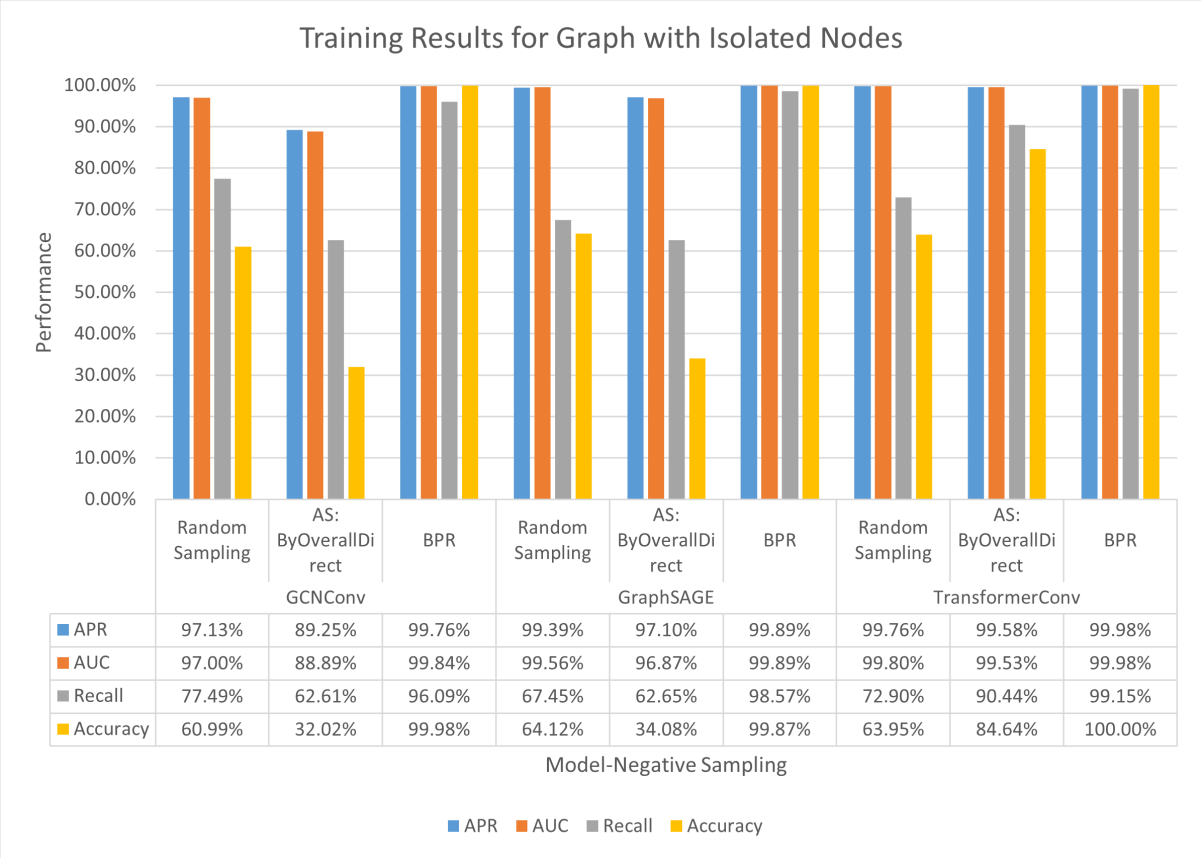

Figure S1: Training Results for graph with isolated nodes

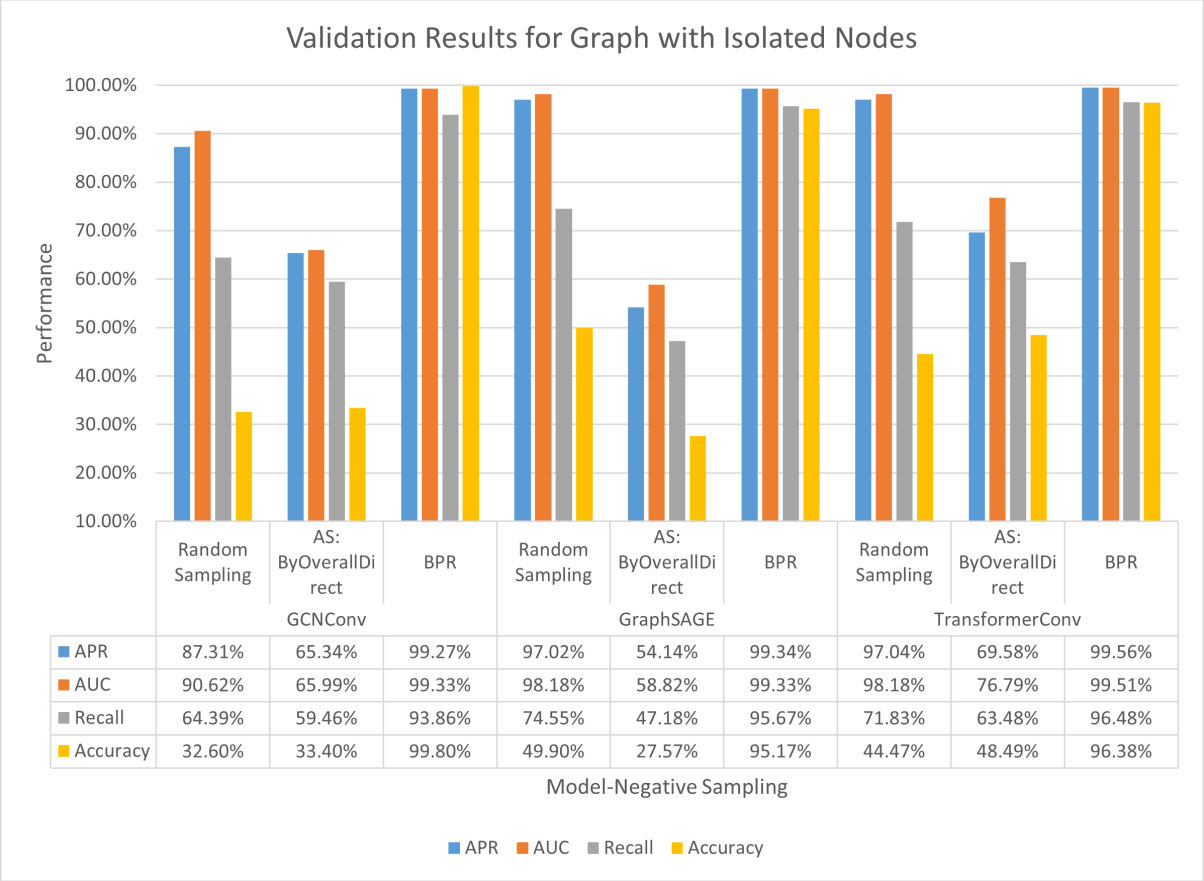

Figure S2: Validation Results for Graph with isolated nodes

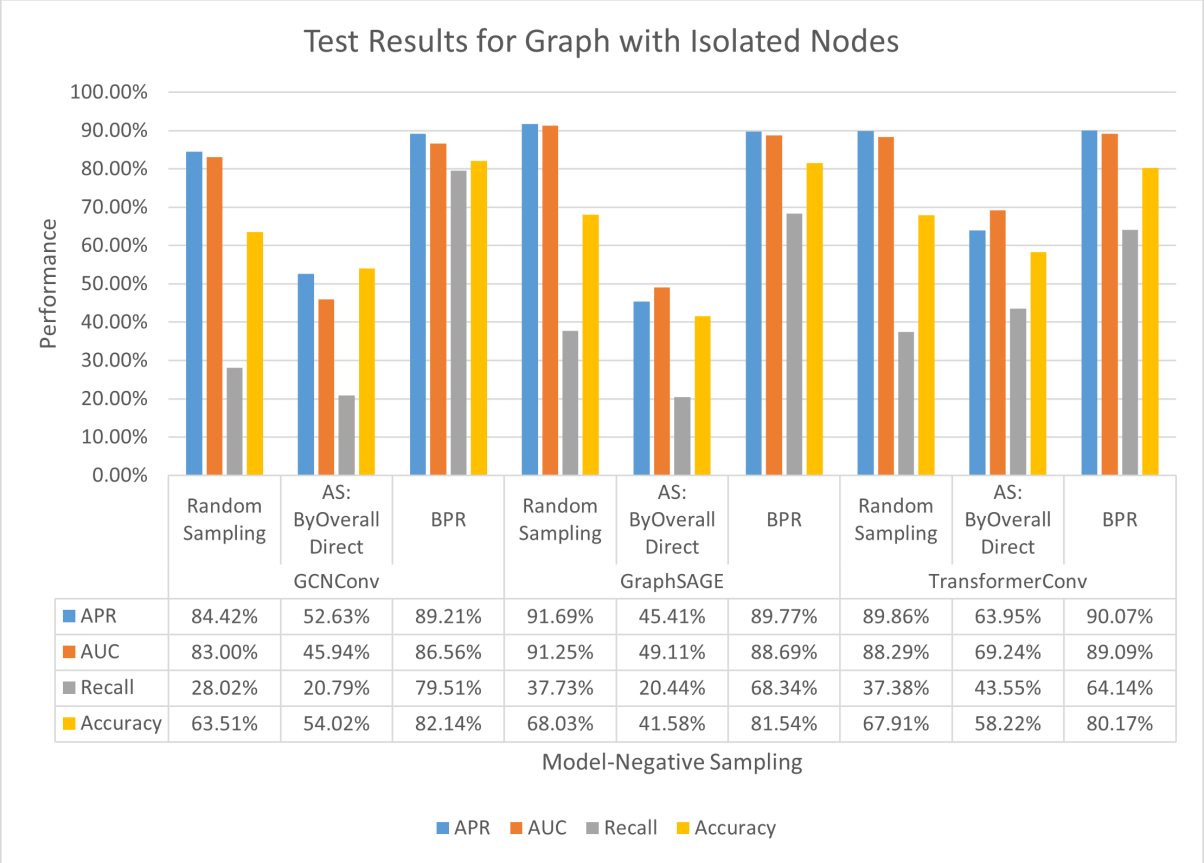

Figure S3: Test Results for Graph with Isolated Nodes

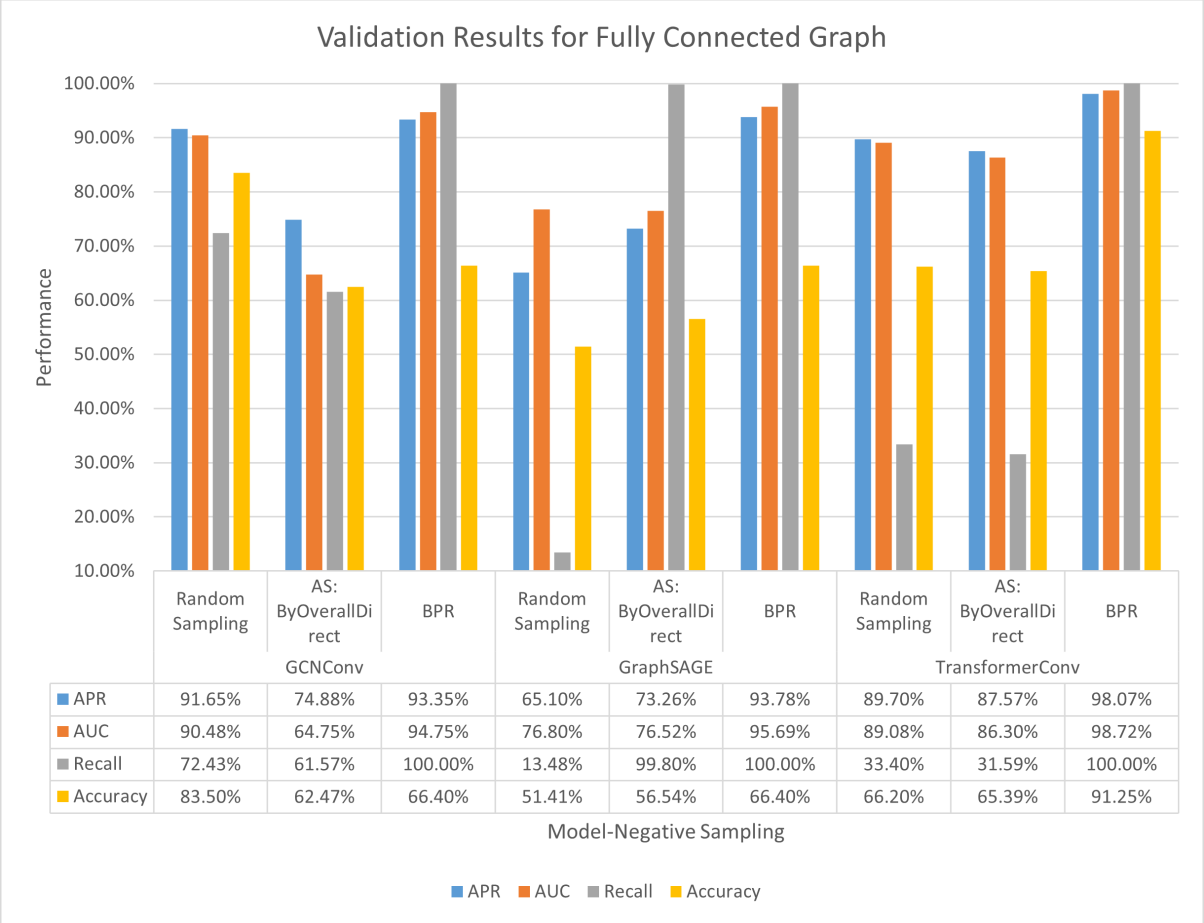

Figure S4: Validation Results for Fully Connected Graph

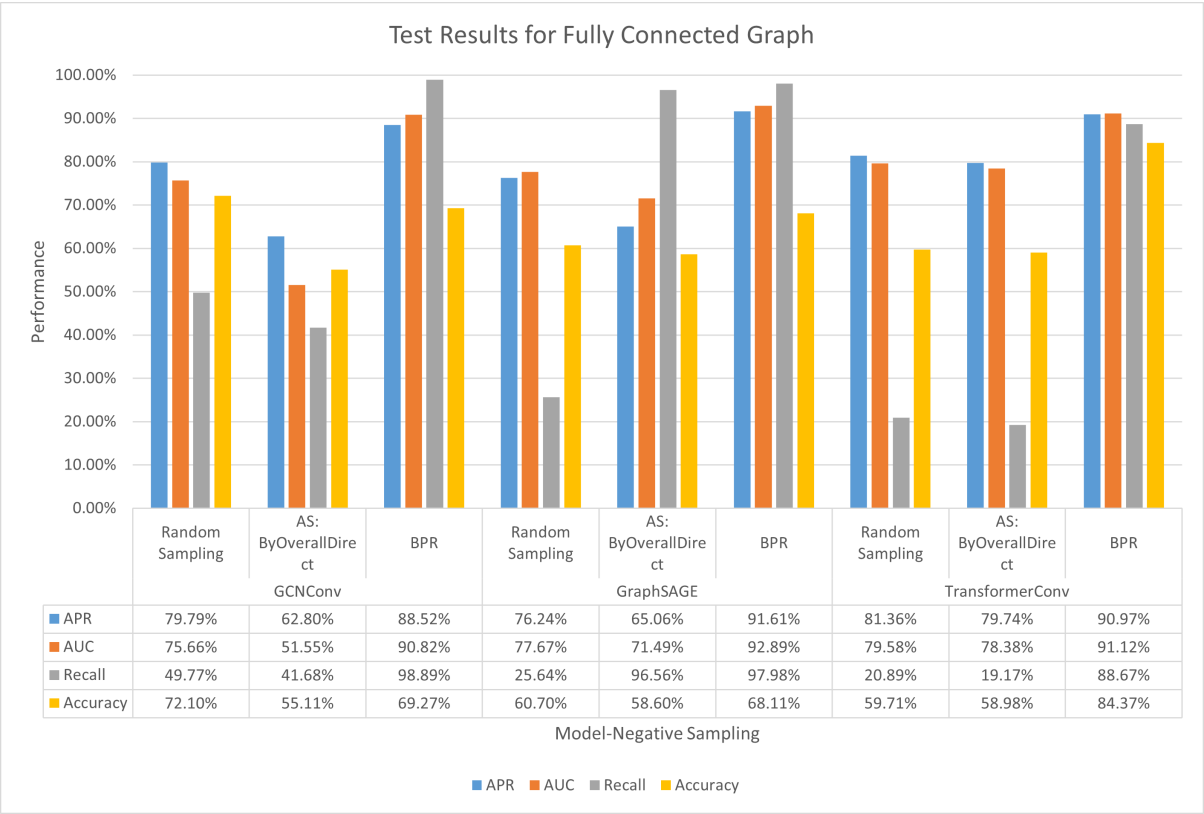

Figure S5: Test Results for Fully Connected Graph

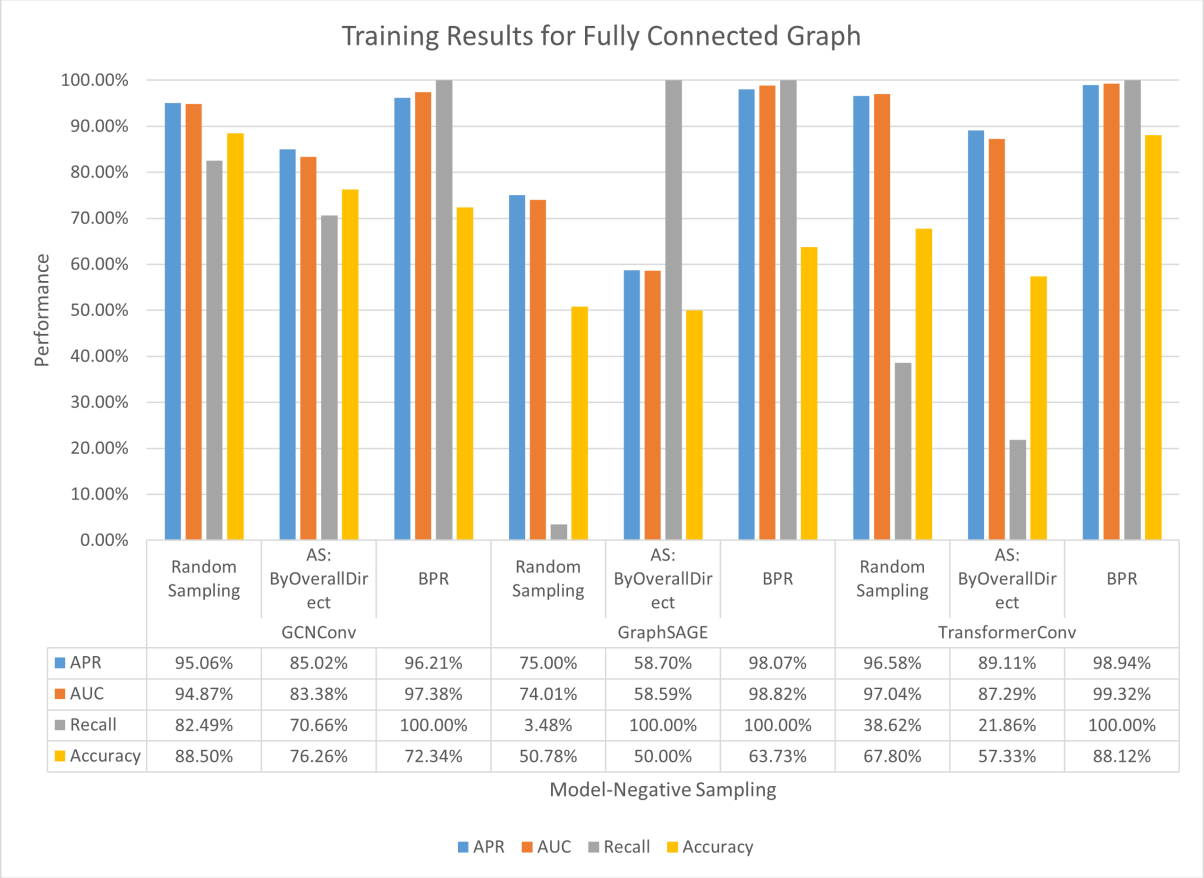

Figure S6: Training Results for Fully Connected Graph

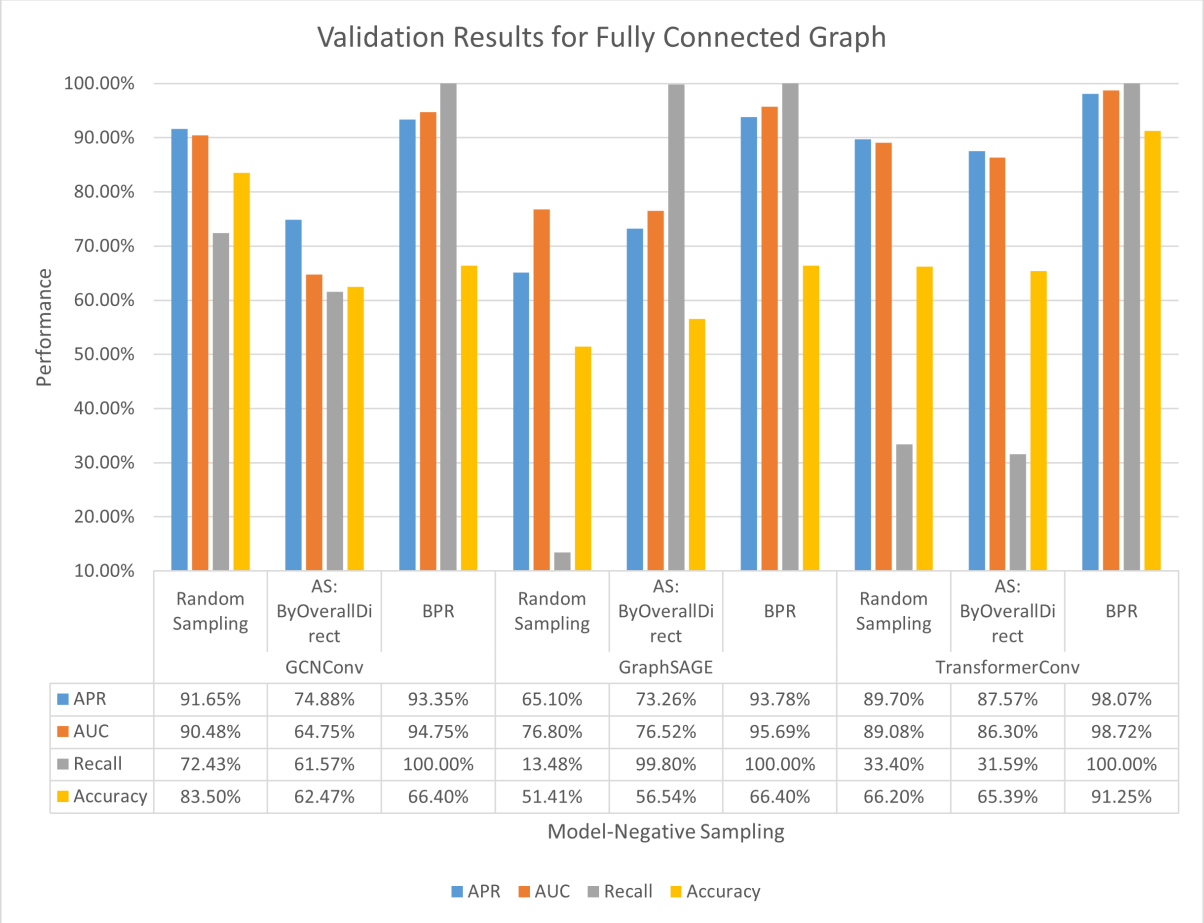

Figure S7: Validation Results for Fully Connected Graph

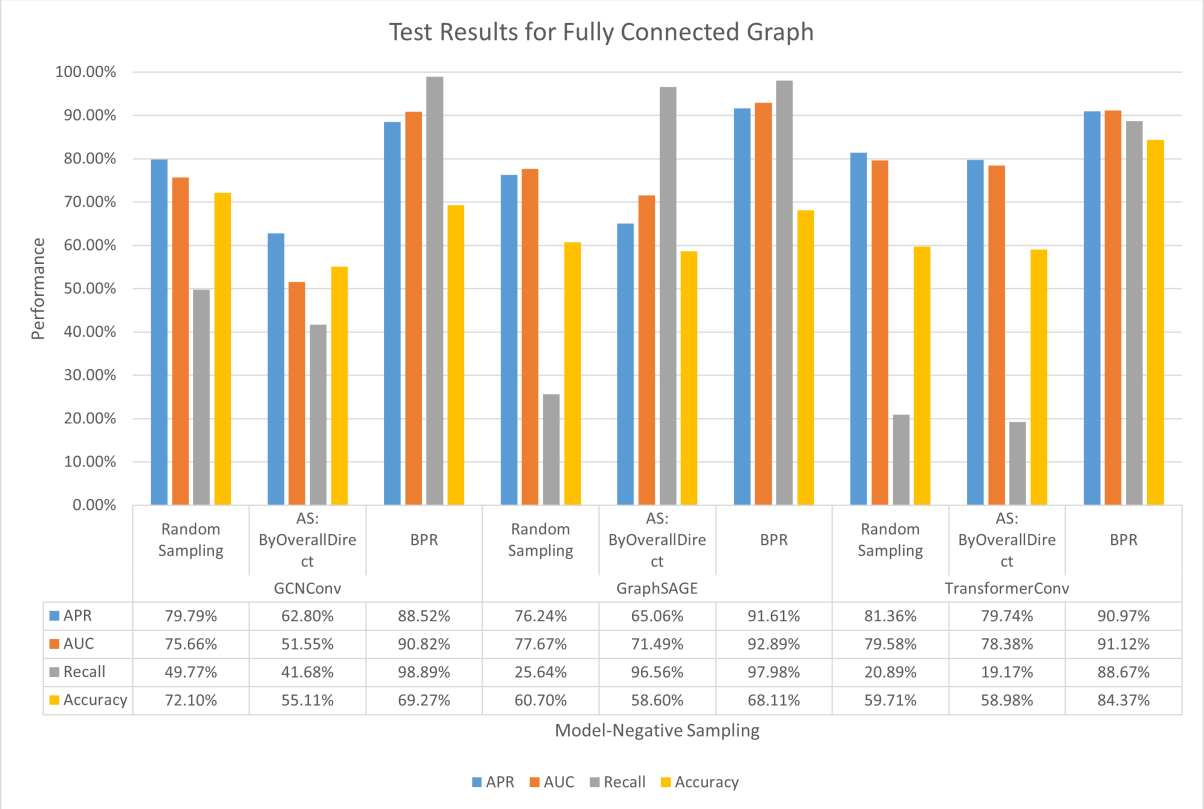

Figure S8: Test Results for Fully Connected Graph