Summary

Biological researchers collect clinical samples to study living organisms and quantify thousands of proteins, the fundamental building blocks of cells, to understand biological processes. Various statistical approaches, including functional enrichment analysis, are employed to analyze these proteins and identify pathways critical for discovering disease causes, new drugs, and specialized treatments. Functional enrichment analysis assesses a set of proteins to determine if specific biological pathways or processes are overrepresented, providing insights into the biological significance of the data.

However, biological research often faces the challenge of small sample sizes and numerous features due to the high costs and time-consuming nature of sample collection. Traditional statistical methods are often inadequate in this context. To address this issue, leveraging large biological databases is essential. Databases such as Reactome, UniProt, and KEGG are repositories of biological information, including genetic sequences, protein interactions, pathways, and molecular structures, crucial for understanding complex biological systems and processes.

This thesis investigates advanced methods for analyzing biological samples by utilizing Graph Neural Networks (GNNs) combined with graph sampling techniques to manage the complexities of large-scale graph data extracted from these databases. Specifically, a Graph AutoEncoder (GAE) is employed for learning compact, informative node representations. The GAE consists of an encoder and a decoder: the encoder embeds nodes into a latent space, and the decoder reconstructs the graph's adjacency matrix from these embeddings. This approach ensures that the learned embeddings capture both topological and feature-based similarities. Additionally, a pipeline combining GAE with graph sampling techniques, such as Random Walk with Restart, Forest Fire, and Cluster-GAE, is proposed to manage large graph databases while preserving essential structural properties. These methods help manage computational load and enhance the efficiency of subsequent analysis stages.

The thesis further delves into various clustering algorithms, such as K-means and Hierarchical Clustering, to group protein embeddings after training with the GAE. This clustering may reveal new biological insights by identifying patterns and relationships among proteins. Additionally, the k-Nearest Neighbors (k-NN) algorithm is proposed for analyzing these embeddings to predict classifications based on the nearest neighbors in the feature space. However, it is noted that this method requires a larger dataset for reliable evaluation to ensure accurate and meaningful predictions. The analysis also incorporates functional enrichment analysis to identify overrepresented biological pathways or processes within the protein data, leveraging the context provided by biological databases to enhance interpretability.

In conclusion, this thesis introduces a robust framework for analyzing biological data with limited sample sizes. By integrating graph sampling and GNNs, we effectively leverage the vast information available in biological knowledge graphs to enhance the analysis of protein data. The results demonstrate the potential of this approach to uncover meaningful patterns, relationships, and biological insights, paving the way for new discoveries and applications in biological research.

Enriching Clinical Sample Analysis With Pathway Knowledge Graphs and GNNs

Fatemeh Shad Bakhsh
fbakhs22@student.aau.dk
Department of Computer Science, Aalborg University

***Abstract*—Biological research often faces challenges in analyzing protein data due to limited sample sizes, hindering the use of traditional statistical methods. Integrating knowledge from extensive graph databases like Reactome and UniProt can offer valuable insights, but efficient techniques are required for effective analysis. This thesis proposes Cluster-GAE, a novel method that combines graph sampling techniques with Graph Neural Networks (GNNs) to learn informative representations from large-scale biological networks. By adapting the cluster-GCN algorithm for graph representation learning, Cluster-GAE addresses the computational challenges of analyzing large graphs while preserving crucial structural information. Our evaluation, which included a comparison with Random Walk, Forest Fire, and No Sampling methods, demonstrates Cluster-GAE's superior performance in preserving graph structure and generating meaningful protein embeddings. Through t-SNE visualization and functional enrichment analysis, we showcase the ability of Cluster-GAE to identify distinct protein clusters and reveal over-represented biological pathways, potentially leading to the discovery of novel biological mechanisms. This thesis establishes a robust framework for analyzing biological samples with limited data, enhancing the interpretability of protein data analysis and opening new possibilities for biological discovery.**

***Index Terms*—Graph Neural Networks; Graph sampling; Pathways; Graph Autoencoder.**

I. Introduction

Biological researchers collect clinical samples to study living organisms, quantifying the amounts of thousands of proteins of interest. Proteins, serving as the building blocks of cells, are fundamental to understanding biological processes. Researchers utilize various statistical approaches, including functional enrichment analysis, to analyze these proteins and identify pathways essential for discovering disease causes, new drugs, and specialized treatments. Functional enrichment analysis involves assessing a set of proteins to determine if specific biological pathways or processes are overrepresented, thereby providing insights into the biological significance of the data [1].

However, collecting samples can be expensive and time-consuming, and trials often include only samples from a few subjects, usually in the order of tens. The process of obtaining clinical samples involves complex procedures such as patient recruitment, ethical approvals, and meticulous sample handling and processing, all of which contribute to the high costs and lengthy durations. Consequently, researchers frequently work with limited sample sizes. Having few data points with numerous features (i.e., proteins) poses significant challenges for traditional statistical analysis [1].

Our study involves an analysis of a dataset $D = (X, y)$. In this context, X denotes the proteins or variables derived from the initial samples, and y corresponds to the group associated with each variable. D comprises a notably small number of samples. The scarcity of samples in D poses significant challenges for the deployment of conventional machine learning algorithms, which typically require substantial data to learn representative features effectively. Moreover, the collection of biological samples is costly and time-consuming, with trials often involving samples from a limited number of subjects. This combination of few data points and numerous features (proteins) presents substantial challenges for conventional statistical analysis.

A possible strategy to address these challenges is centered around subset selection, a method aimed at identifying the most informative features within the dataset D [2]. Given the small sample size, the direct application of a learning function f, such as traditional machine learning methods like logistic regression, to find the best features could lead to overfitting or biased results.

Biological databases such as Reactome [3], KEGG [4], UniProt [5], and other relevant resources are invaluable tools for enhancing the analysis of biological data. Reactome is a curated database of pathways and reactions in biology, which provides detailed information about molecular processes [3]. KEGG (Kyoto Encyclopedia of Genes and Genomes) offers a collection of manually curated databases dealing with genomes, biological pathways, diseases, drugs, and chemical substances [4]. UniProt (Universal Protein Resource) is a comprehensive resource for protein sequence and annotation data, providing extensive information on protein functions, structures, and interactions [5]. Considering their wealth of information, leveraging these biological databases offers another approach to handle these challenges. These databases provide valuable biological context that can be used to enhance the analysis of D, even with a limited number of samples. However, the question remains: How can we use such biological databases effectively to analyze D despite having tens of samples in it? And how can we integrate this external biological knowledge to improve the reliability and interpretability of our findings?

The goal of this project is to support biological researchers by enriching their analyses with contextual data about proteins and pathways. Such elements are described in extensive

biological graph databases. We therefore rely on graph representations to capture the relevant structural and semantic information from the graph, using Graph Neural Networks (GNN). GNNs work by iteratively updating the features of a node by incorporating information from its neighbouring nodes, a process known as message passing. GNNs are highly effective in finding embeddings in high-dimensional protein data, providing a compact, lower-dimensional representation that captures the essential features and relationships present in the data [6]. This process is crucial for various bioinformatics tasks, including protein function prediction, interaction prediction, and structure analysis [7].

To cope with the large scale of these databases, we study GNN architectures to handle larger graphs efficiently. This is optionally combined with graph sampling methods, which offer a pathway to manage and analyze large-scale graphs without compromising the performance or scalability of GNNs. This thesis investigates how to effectively utilize GNNs in combination with graph sampling methods to learn and represent relevant information stored in large graph databases such as Reactome and UniProt. This approach helps to analyze the sparse instances within biological samples and identify differences between groups in the samples, despite the challenges posed by the large scale and complexity of graph databases.

To address these challenges, we conduct a comprehensive review of current graph sampling techniques, meticulously examining their strengths and weaknesses to identify the most suitable method for our project. Our analysis focuses on methods such as Cluster-GCN, Random Walk sampling (RW), and Forest Fire sampling (FF). By integrating these sampling methods with GNNs, we aim to efficiently manage large-scale graph data, ultimately enhancing the representation and understanding of complex biological networks.

The contributions of this thesis are manifold and revolve around advancing the state-of-the-art in the analysis of biological data using graph-based methods and GNNs. Specifically, the contributions can be summarized as follows:

- We propose a method named **Cluster-GAE**, which combines graph sampling techniques specifically designed for large graphs with GNNs to effectively learn representations in expansive graph structures.
- We leverage biological knowledge graphs such as **Reactome** and **UniProt** to analyze biological samples with a limited number of instances using GNNs, enhancing the understanding and interpretation of these datasets.
- We employ **functional enrichment analysis** technique to evaluate our proposed method, assessing the biological relevance of the identified groups and offering potential new insights into protein functions and interactions.

In Section II, we review related works within the problem domain, providing context and highlighting previous advancements. Section III delves into the preliminaries and key concepts related to the problem, including Graph Neural Networks, Graph Convolutional Networks, and Cluster-GCN. In Section IV, we detail the proposed method to address the challenges outlined in the introduction. Section V describes the dataset utilized, outlines the various experiments conducted, and presents the results obtained from these experiments. Finally, in Section VI, we summarize our findings and discuss potential directions for future research.

II. Related Works

A. Knowledge Graphs

Knowledge graphs represent a compelling advancement in the management and utilization of information, acting as structured semantic networks that connect entities (such as individuals, places, and things) through edges that describe their interrelations. They encapsulate complex relationships and attributes within data in a way that is both computationally efficient and semantically rich, making them particularly valuable for enhancing search technologies, powering recommendation systems, and enabling advanced artificial intelligence applications. By integrating diverse data sources into a coherent graph structure, knowledge graphs provide a unique framework that supports not only information retrieval but also data inference, allowing for more sophisticated analytics and decision-making processes. Their application ranges from enhancing semantic search capabilities in digital assistant technologies to improving data interoperability across disparate systems, highlighting their role as a pivotal technology in the ongoing evolution of data handling and analysis in academic, commercial, and technological domains [8].

One notable application of knowledge graphs is in the medical domain. They are used to integrate and interpret vast amounts of biomedical data, facilitating breakthroughs in research and clinical decision-making. For instance, UniProt uses knowledge graphs to link proteins with relevant biological information, such as functions, interactions, and pathways. Similarly, Reactome, a database of biological pathways, employs knowledge graphs to map complex biochemical reactions and processes within human biology. By doing so, these resources enhance the understanding of molecular biology and support the discovery of new therapeutic targets [3], [5], [8].

B. Graph Sampling Methods

Graph sampling methods are crucial for analyzing large-scale networks efficiently by selecting representative subsets of nodes and edges. These techniques extract subgraphs that maintain the structural properties of the original graph, enabling accurate estimations of characteristics such as degree distribution, clustering coefficients, and path lengths. Common approaches include random node sampling, where nodes are chosen at random; edge sampling, which randomly selects edges; and traversal-based sampling, like Breadth-First Search (BFS) and Depth-First Search (DFS), which systematically explore nodes starting from a random node. More sophisticated methods like snowball sampling involve selecting a node and then progressively adding its neighbors [9], [10].

Random walk sampling, a sampling technique, starts from a random node and moves to a randomly chosen neighbor,

repeating this process to form a path. This method is particularly useful for capturing the local structure and connectivity patterns within the graph. Forest Fire sampling, on the other hand, begins with a set of seed nodes and "burns" through the graph by selecting neighbors with a certain probability, mimicking the spread of a fire. This method can efficiently sample large graphs while preserving community structures [9].

These sampling techniques are essential in applications where direct analysis of the entire graph is computationally prohibitive, providing a scalable solution for data scientists working with massive network datasets.

C. Graph Embeddings

Graph embedding is a technique in graph analysis, enabling the transformation of complex graph-structured data into low-dimensional vector spaces while preserving the essential structural properties of the original graph. This technique has gained significant attention in recent years due to its capability to facilitate various downstream tasks such as node classification, link prediction, clustering, and visualization [11].

Traditional methods for graph embedding, known as matrix factorization techniques, include Laplacian Eigenmaps and graph factorization. These methods focus on capturing the spectral properties of graphs, aiming to preserve their global structure. However, they often face challenges related to scalability and computational efficiency, especially when applied to large graphs [12].

Advancements in neural network architectures have significantly revolutionized graph embedding techniques. Graph Autoencoders (GAEs) represent one such advancement. GAEs learn embeddings by reconstructing the input graph from encoded representations, effectively capturing the underlying structure in an unsupervised manner. Additionally, Variational Graph Autoencoders (VGAEs) introduce a probabilistic layer, modeling the latent space with a Gaussian distribution. This enhancement improves the model's ability to capture complex graph structures [13].

In bioinformatics, graph embeddings have been crucial in analyzing protein-protein interaction networks. They facilitate the discovery of new biological insights and functional annotations, highlighting their importance in advancing our understanding of complex biological processes [14].

III. Background

A. Graph Neural Networks

Graph Neural Networks [6], [15] are a transformative approach in machine learning that leverage the natural graph structure within data to perform deep learning tasks directly on graphs. These networks are designed to capture the dependencies between nodes in a graph through message passing or neighborhood aggregation techniques, where each node updates its state by recursively aggregating and transforming feature information from its neighbors. This process, iterated over multiple layers, allows GNNs to learn complex node representations that reflect both their local graph topology and node-specific features.

Formally, consider a graph $G = (V, E)$ where V is the set of nodes and E is the set of edges. Let $\mathbf{h}_v^{(k)}$ represent the feature vector of node v at the k-th layer. The general operation of a GNN layer can be described as:

$$\mathbf{h}_v^{(k+1)} = \sigma\Bigg(\mathbf{W}^{(k)} \cdot \text{AGGREGATE}^{(k)}\Big(\Big\{\mathbf{h}_u^{(k)} : u \in \mathcal{N}(v) \cup \{v\}\Big\}\Big)\Bigg), \tag{1}$$

where $\mathbf{W}^{(k)}$ is a learnable weight matrix, $\mathcal{N}(v)$ denotes the set of neighbors of node v, $\text{AGGREGATE}^{(k)}$ is an aggregation function (e.g., sum, mean, or max), and σ is an activation function (e.g., ReLU).

The flexibility of GNNs has led to their application across a wide range of fields, from predicting protein interactions in computational biology to optimizing network traffic in telecommunications. In social network analysis, for instance, GNNs can identify influential community members or predict the evolution of social ties. Their ability to directly incorporate relational data makes them particularly effective for tasks that traditional neural network architectures struggle with, such as graph classification, node classification, and link prediction [6], [15].

1) Graph Autoencoders: Graph Autoencoders [13] are a class of neural networks specifically designed for unsupervised learning on graph-structured data. They function by compressing the graph into a low-dimensional embedding that captures the essence of its topology and then reconstructing the original graph from this compressed representation. This process involves two primary components: an encoder and a decoder.

The encoder, typically implemented using a GNN, maps each node $v_i \in V$ (or sometimes entire subgraphs) to a vector $\mathbf{z}_i \in \mathbb{R}^d$ in a latent space, where V denotes the set of nodes and d is the dimensionality of the latent space. The embedding $\mathbf{z}_i$ effectively captures the node's role and connectivity within the overall graph structure. Mathematically, the encoder function can be expressed as:

$$\mathbf{Z} = f_{\text{enc}}(\mathbf{X}, \mathbf{A}), \tag{2}$$

where $\mathbf{X} \in \mathbb{R}^{N \times F}$ is the feature matrix with N nodes and F features per node, $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix, and $\mathbf{Z} \in \mathbb{R}^{N \times d}$ is the matrix of latent representations.

The decoder then attempts to reconstruct the graph's adjacency matrix $\mathbf{A}$ or some function of the graph from these embeddings. Typically, the decoder computes the probability of an edge between pairs of nodes (v_i, v_j) by applying a function g_{dec} on their latent representations:

$$\hat{\mathbf{A}}_{ij} = g_{\text{dec}}(\mathbf{z}_i, \mathbf{z}_j). \tag{3}$$

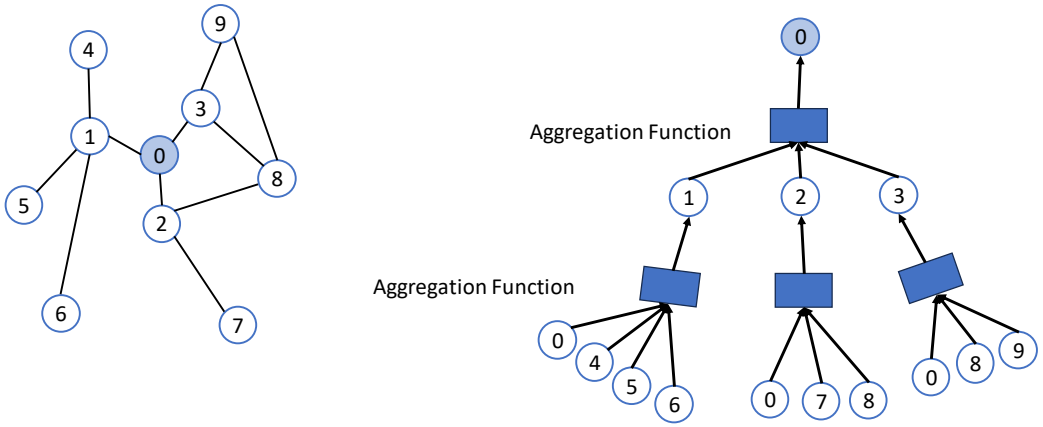

Fig. 1: GNN Network Overview. This figure illustrates the message passing process in a two-layer GNN. On the left, the original graph is shown with node 0 and its neighbors. On the right, the hierarchical aggregation process is depicted where each node aggregates messages from its neighboring nodes. The aggregation function combines the feature information from neighboring nodes recursively across multiple layers, allowing the central node (node 0) to update its state based on the aggregated information. This process captures both the local graph topology and node-specific features, enabling the learning of rich node representations.

In many cases, g_{dec} is modeled as a simple inner product:

$$\hat{\mathbf{A}} = \sigma(\mathbf{Z}\mathbf{Z}^\top), \tag{4}$$

where σ denotes the sigmoid function applied element-wise.

The training objective for GAEs often involves minimizing the reconstruction loss between the original adjacency matrix $\mathbf{A}$ and the reconstructed adjacency matrix $\hat{\mathbf{A}}$. This can be formulated as a binary cross-entropy loss:

$$\mathcal{L}_{\text{rec}} = -\frac{1}{N}\sum_{i,j}\left[\mathbf{A}_{ij}\log\hat{\mathbf{A}}_{ij} + (1-\mathbf{A}_{ij})\log(1-\hat{\mathbf{A}}_{ij})\right]. \tag{5}$$

Overall, GAEs provide a powerful framework for learning compact and informative representations of graph-structured data, facilitating various downstream tasks such as node classification, link prediction, and graph clustering. Furthermore, by learning efficient embeddings, GAEs can also facilitate dimensionality reduction, anomaly detection, and even graph generation tasks [13].

The effectiveness of GAEs hinges on the ability of the encoder to produce meaningful latent variables and the capacity of the decoder to reconstruct the graph's structural properties accurately. Enhancements in GAE architectures often focus on improving these components through more sophisticated GNN models, incorporating attention mechanisms, or using variational approaches that add a regularization term to the encoder's output, leading to more robust and generalizable graph representations [13].

B. Graph Convolutional Networks (GCN)

GCNs were first introduced by Kipf and Welling in their 2016 paper, "Semi-Supervised Classification with Graph Convolutional Networks" [16]. The core idea is to extend the concept of convolutional neural networks (CNNs) to graph-structured data. In a traditional CNN, the convolutional layer processes data by scanning a small window over the input data, computing a dot product with learnable weights, and summing the results. Similarly, in a GCN, the graph convolutional layer processes data by scanning over the neighboring nodes of a given node, computing a dot product with learnable weights, and summing the results. This process is repeated for multiple layers to capture complex relationships between nodes. At a high level, the key idea behind GCNs is to iteratively update the node representations by aggregating information from neighboring nodes. This is achieved through a graph convolution operation, where each node aggregates feature information from its neighbors weighted by the edge connections. By stacking multiple graph convolutional layers, GCNs can capture increasingly complex relationships and higher-order structures in the graph.

C. Cluster-GCN

Cluster-GCN is a cutting-edge algorithm designed to efficiently train GCNs on large-scale graphs by leveraging graph clustering structures. Traditional GCN training methods face high computational costs and memory demands due to the extensive neighborhood expansion required to compute node embeddings. Cluster-GCN addresses these challenges by partitioning the graph into densely connected subgraphs, or clusters, using algorithms like METIS. During training, the algorithm samples these clusters and restricts computations to the subgraphs, significantly reducing memory usage and computational overhead. This approach maintains high efficiency and enables the training of much deeper GCNs, leading to improved prediction accuracy. Furthermore, Cluster-GCN incorporates a stochastic multi-clustering framework

that combines multiple clusters in each batch, reintroducing between-cluster links and enhancing convergence. By focusing on dense subgraphs and stochastically reintroducing between-cluster connections, Cluster-GCN effectively trains deep GCNs on large-scale graphs with improved efficiency and scalability [17].

D. Clustering

Clustering is an unsupervised learning technique in data analysis and machine learning that involves grouping a set of data points into clusters, such that points within the same cluster exhibit higher similarity to each other than to those in different clusters. This technique is invaluable for uncovering underlying patterns and structures in datasets without predefined labels. Common clustering algorithms include k-means, hierarchical clustering, DBSCAN, and Gaussian Mixture Models, each offering unique advantages depending on the nature of the data and the specific application [10].

The applications of clustering are extensive and diverse, spanning numerous fields such as marketing, image analysis, bioinformatics, and anomaly detection. For instance, in customer segmentation, clustering can help businesses identify distinct customer groups based on purchasing behavior and demographics, thereby enabling targeted marketing strategies. In bioinformatics, clustering facilitates the classification of genes and proteins, aiding in the discovery of gene expression patterns and evolutionary relationships. Evaluating the effectiveness of clustering algorithms involves metrics such as the Silhouette Score, Davies-Bouldin Index, Adjusted Rand Index, and Normalized Mutual Information, which assess the coherence and validity of the formed clusters. As data continues to grow in complexity and volume, clustering remains an essential tool for extracting meaningful insights and supporting data-driven decisions [18], [19].

IV. Method

This section presents a comprehensive pipeline designed for biological samples analysis, including some steps to facilitate effective graph representation learning using GAE. As shown in Fig 2 and Fig 3, the pipeline incorporates various components, including preprocessing, graph sampling, GAE to address challenges associated with large-scale graph data. The specific components and processes within this pipeline will be explained below. Moreover, the proposed methodology is summarized in Algorithm 1.

The proposed method uses graph data from the Reactome database, processed through distinct pipelines for RW, FF, and Cluster-GAE approaches. For RW and FF, the steps include querying Reactome, sampling the graph, preprocessing by normalizing and converting node features to tensor, and learning the embedding using a GAE, followed by sample analysis. The Cluster-GAE pipeline differs in preprocessing order and sampling method. Detailed descriptions of these pipelines will be discussed in the following subsections.

A. Graph Querying

In this step, we query the Reactome graph database using the Cypher query language via Neo4j [20] to extract relevant information. Specifically, we focus on identifying nodes connected to the proteins listed in dataset D. By executing these queries, we retrieve a subgraph including all relevant nodes and their interactions, thereby ensuring that the analysis consists of all related biological pathways and processes.

The extracted data is then integrated into a single comprehensive graph, resulting in a structure consisting of 50,164 nodes and 1,667,138 edges. This comprehensive graph captures the intricate network of interactions between the proteins and their associated entities, providing a detailed representation of the biological pathways and processes.

This graph serves as a foundational dataset for subsequent analysis and modeling. By enabling the exploration of connectivity and relationships within the biological network, it facilitates the comprehensive analysis of the samples discussed in Section I. Leveraging Neo4j for querying ensures efficient and precise data retrieval, taking full advantage of the robust capabilities of the Reactome graph database. This approach not only enhances the accuracy of the data but also optimizes the process of integrating complex biological interactions into our analysis framework.

The query 1 is written using Cypher, the query language for Neo4j, and makes use of the apoc.path.subgraphAll procedure to explore the graph. To be more specific, the query takes a list of protein names, matches them against relevant nodes in the Reactome database, extracts a subgraph of connected nodes and relationships up to two levels deep, filters the nodes to include only those relevant to the species Mus musculus, and returns the filtered subgraph. This process ensures that the analysis is focused on biologically pertinent information for the specified proteins and species.

Listing 1: Neo4j query

```
UNWIND $proteinNames AS proteinName
MATCH (p)
WHERE ("EntityWithAccessionedSequence" IN
    labels(p) OR "GenomeEncodedEntity" IN
    labels(p)) AND ANY(name IN p.name WHERE
    name = proteinName)
CALL apoc.path.subgraphAll(p, {
  maxLevel: 2,
  minLevel:1
})
YIELD nodes, relationships
WITH p,
     [node in nodes WHERE "Mus musculus" IN
         labels(node) OR node.speciesName = "
         Mus musculus"] AS filteredNodes,
     relationships
RETURN p AS protein, filteredNodes AS nodes,
    relationships
```

B. Model Training

The model training process differs for the RW and FF sampling methods compared to the Cluster-GAE method, as illustrated in Figures 2 and 3 respectively.

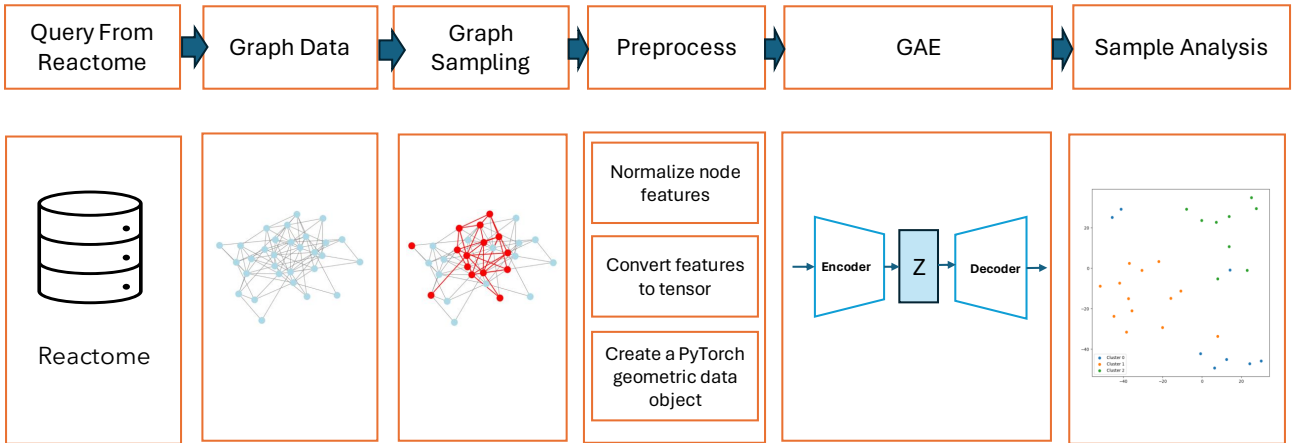

Fig. 2: Pipeline Architecture for RW and FF

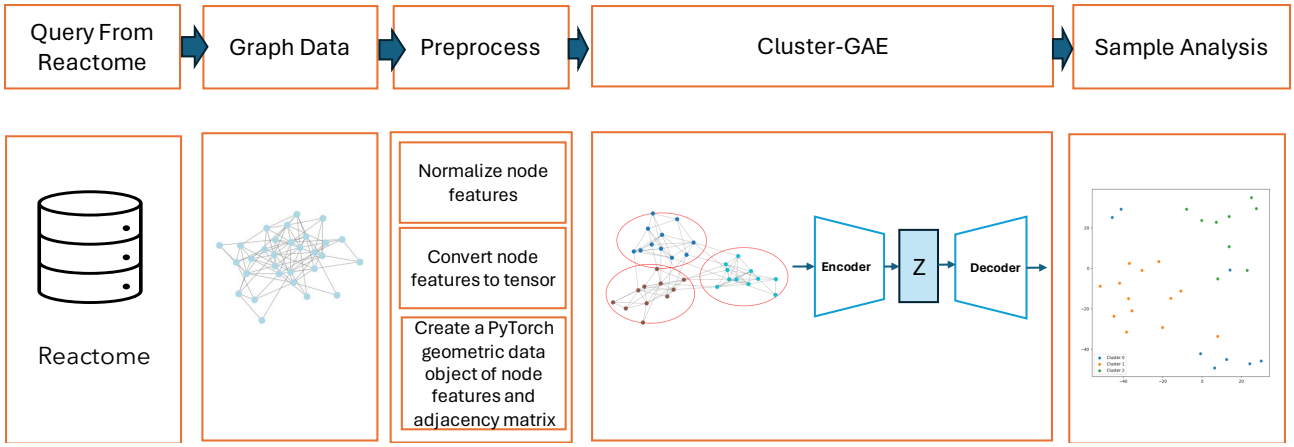

Fig. 3: Pipeline Architecture for Cluster-GAE

1) Forrest Fire and Random Walk Sampling:

a) Graph Sampling: The pipeline in Fig 2 after querying the data from Reactome and transforming it to a graph follows with the selection of a graph sampling method. Graph sampling is an important step that aims to reduce the size of the graph, ideally reducing computation load while retaining the essential structural properties of the original graph. Two sampling techniques are considered: Random Walk with Restart [9], Forest Fire [9]. The RW with Restart Sampler simulates random walks on the graph, providing a localized view of the graph's structure. The FF Sampler, on the other hand, uses a probabilistic approach to create a smaller graph that mimics the connectivity of the larger graph.

b) Preprocessing: The pipeline includes a feature preprocessing stage. This step involves converting node labels to integers, handling non-numeric features, and normalizing feature vectors. The conversion of node labels to integers standardizes node identification across different graph processing steps. For non-numeric features, a simple placeholder value is assigned, ensuring that all features are represented in a uniform numeric format. Feature normalization is then performed to scale the feature vectors, enhancing the stability and performance of learning algorithms.

c) Graph Autoencoder: The core of the pipeline is the GAE model, which is utilized for learning low-dimensional, yet informative representations of the graph's nodes. The GAE model comprises an encoder and a decoder, with the encoder being a GCN that embeds nodes into a latent space. The decoder then attempts to reconstruct the graph's adjacency matrix from these embeddings. This process is facilitated by a reconstruction loss function, guiding the model to capture the essential topological and feature-based similarities between nodes [13].

2) Cluster-GAE: In contrast, the Cluster-GAE method, depicted in Figure 3, follows a different training approach.

a) Preprocessing: Similar to the RW and FF pipelines, this stage involves normalizing node features and converting them to tensors. These tensors, along with the adjacency matrix, are used to create a PyTorch geometric data object, facilitating further graph processing.

b) Cluster-GAE Training: The Cluster-GCN algorithm enhances computational efficiency by dividing the graph into multiple clusters, enabling parallel processing of these components. In our scenario, it is crucial to learn a compact representation of the graph data (input). Therefore, instead of using a traditional GCN within the Cluster-GCN method, we employ a GAE model. While GCNs are typically used for semi-supervised learning, GAEs are specifically designed for unsupervised learning, making them more suitable for capturing and learning latent representations of graph data. This modification allows us to generate meaningful embeddings and understand the underlying structure of the graph. As mentioned in Subsection III-C, this method divides the data into partitions and then samples from these partitions during the training phase. Furthermore, the GAE in the Cluster-GAE model follows the same structure as the standard GAE with an encoder and decoder.

C. Sample Analysis

Following the learning embedding of graph nodes using the GAE as described in Section IV-B1c, the next phase in our pipeline involves a thorough sample analysis through various techniques. One such technique is clustering, which helps identify inherent groupings within the data that are not immediately apparent. This is particularly crucial for complex biological datasets, where the discovery of clusters can lead to new hypotheses about biological functions or states, as they may represent underlying structures or functional groupings not previously considered.

The embeddings produced by the GAE represent nodes in a reduced-dimensional space. To exploit these embeddings for sample analysis, we extract the desired protein embeddings from the whole embeddings and apply clustering algorithms such as K-means and Hierarchical Clustering to partition the nodes into distinct groups. Each cluster represents a group of nodes that share significant similarities, potentially corresponding to specific biological states or functionalities. For instance, nodes that cluster together and correspond to similar or functionally related biological samples may reveal new patterns and groupings within the embeddings.

In addition to clustering, another possible approach to analyze the embeddings is to employ the k-Nearest Neighbors (k-NN) algorithm as a recommender system to analyze the embeddings [21]. The k-NN algorithm will establish a distance function $f(x_k, x_n)$ to predict the classification of data points by identifying their nearest neighbors in the feature space. This approach includes selecting a subset of features determined to be the most informative through initial exploratory model fitting using logistic regression known as subset selection. Once these optimal features are identified, the k-NN algorithm helps ascertain the closest neighbors to these features, ensuring that the model finds similar proteins in the embedding space to a given protein. However, dataset D contains a very small number of instances, which leads to overfitting. To mitigate this issue, we need a larger dataset of biological samples. Unfortunately, we do not have access to such data at this time. Consequently, it was not possible to evaluate this analysis.

The final analysis uses a statistical approach on the clustered embeddings known as functional enrichment analysis. Statistical methods for enrichment analysis are crucial tools for extracting biological information from biological experiments. Although traditionally used for analyzing gene and protein lists, the advent of high-throughput technologies for regulatory elements necessitates dedicated statistical and bioinformatics tools. Functional enrichment analysis, also referred to as gene set analysis (GSA), is widely utilized to interpret high-throughput experimental results. GSA aims to identify biological annotations that are over-represented in a list of genes relative to a reference background. These annotations help interpret the molecular mechanisms and biological processes associated with the experimental condition under study. g:Profiler is a powerful tool used to analyze overrepresented pathways and biological processes within clusters [22], [23].

Overall, the sample analysis stage is crucial for translating the computational embeddings from the GAE into actionable biological insights, thereby allowing researchers to draw meaningful conclusions about the underlying biological processes represented in the graph data.

V. Evaluation

This section presents the evaluation of the proposed method through a series of experiments. The evaluation process involves detailed descriptions of the experimental setup, the dataset used, the metrics for assessment, the results obtained, and a comprehensive discussion of these results.

A. Data

For the evaluation, two types of datasets are utilized. The primary dataset comprises biological samples characterized by y features or proteins, with a small number of instances. Additionally, the second dataset consists of queried data from the Reactome Graph Database. This data is transformed into a graph structure, including entities connected to the proteins and the relationships between these entities. Table I presents relevant information about the graph data.

TABLE I: Graph Data Summary

Graph Property	**Value**
Number of Nodes	50,164
Number of Edges	1,667,138
Average Degree	66.468
Graph Density	0.001
Is the Graph Directed?	No
Number of Desired Proteins in the Samples that Exist in the Graph	1,549

Algorithm 1 Proposed Method for Analyzing Biological Samples

Require: Dataset $D(x, y)$ containing protein names

procedure GRAPH ANALYSIS PIPELINE($D(x, y)$)
  $N_p \leftarrow$ EXTRACTPROTEINNAMES(D) ▷ Extract protein names from the dataset D
  $G_R \leftarrow$ QUERYREACTOME(N_p) ▷ Query Reactome to get relevant nodes and edges
  $sampling\_method \leftarrow$ SELECTSAMPLINGMETHOD ▷ Select graph sampling method
  if $sampling\_method ==$ "Random Walk" **or** "Forest Fire" **then**
    $G_s \leftarrow$ GRAPHSAMPLING($G_R, sampling\_method$) ▷ Sample a subgraph using the selected method
    $G_p \leftarrow$ PREPROCESS(G_s) ▷ Preprocess the subgraph
    $E \leftarrow$ GRAPHAUTOENCODER(G_p) ▷ Learn node embeddings using GAE
  else if $sampling\_method ==$ "Cluster-GCN" **then**
    $G_p \leftarrow$ PREPROCESS(G_R) ▷ Preprocess the original graph
    $E \leftarrow$ CLUSTERGAE(G_p) ▷ Learn node embeddings using Cluster-GAE
  end if
  $E_p \leftarrow$ EXTRACTPROTEINEMBEDDINGS(E, N_p) ▷ Extract the protein embeddings
  $C \leftarrow$ CLUSTER(E_p) ▷ Cluster the protein embeddings
  $F \leftarrow$ FUNCTIONALENRICHMENTANALYSIS(C) ▷ Perform functional enrichment analysis
  return F ▷ Return the results of functional enrichment analysis
end procedure

B. Evaluation Metrics

The evaluation of the sample analysis consists of three key metrics:

- **Mutual Information Score**: This metric measures the dependency between two variables, providing insight into the comparison of two clustering algorithms. It is used to compare the performance of k-means and agglomerative clustering in clustering the protein embeddings across different sampling methods. A higher mutual information score indicates that both clustering methods agree on the data structure, validating the reliability of the clusters identified. This agreement can enhance confidence in the biological insights derived from the clusters. Furthermore, it can help to evaluate which clustering method and sampling approach best capture the data structure [24], [25].
- **Clustering metrics**: These metrics evaluate the quality of clustering results by assessing how well the identified clusters adhere to desirable properties like compactness, separation, and connectedness. Some commonly used clustering metrics include:
  - **Silhouette Score**: Measures how similar an object is to its own cluster compared to other clusters. A higher silhouette score indicates better-defined clusters [26].
  - **Davies-Bouldin Index**: Measures the average similarity between each cluster and its most similar cluster. A lower Davies-Bouldin index indicates better cluster separation [27].
  - **Calinski-Harabasz Index**: Measures the ratio of the between-cluster dispersion to the within-cluster dispersion. A higher Calinski-Harabasz index indicates denser and more well-separated clusters [28].
- **Earth Mover's Distance (EMD)**: EMD evaluates the dissimilarity between two probability distributions, offering a quantitative assessment of the differences in protein embedding matrices across different sampling methods [29], [30].
- **Frobenius Norm**: The Frobenius norm can be used to measure the difference between two matrices. In this context, it evaluates the differences between the similarity matrices of protein embeddings obtained from various sampling methods. The similarity matrix represents how similar each pair of protein embeddings is to each other. By calculating the Frobenius norm, we can quantify the overall deviation between the structures captured by different sampling techniques

C. Evaluation Setup

The experiments aim to execute the proposed model, introduced in Section IV, utilizing two distinct clustering algorithms: K-Means and Hierarchical Clustering. These algorithms have been selected to determine which one performs better in identifying distinct protein groups. Both algorithms require a parameter K, representing the number of clusters to be identified. In our experiments, K will be set to 2 and 3 to explore the performance under different cluster counts. Choosing a higher number of clusters could complicate the interpretation of cluster representations; therefore, we are initially interested in detecting patterns that can identify specific subgroups. Each clustering algorithm will be applied to the latent space (embeddings) generated by the proposed models.

For these experiments, it is unnecessary to split the dataset into training, validation, and test sets since the focus is on representation learning. Consequently, the entire dataset will be used without splitting. The summary of the graph data is provided in Table I.

We compare four sampling methods: RW, FF, Cluster-GAE, and No Sampling (baseline). Initially, we train models using the algorithm described in Section 1 with a dimensionality of 64 and 128. For the evaluation, we compute the cosine

similarity matrix for each embedding and calculate the Earth Mover's Distance (EMD) and the Frobenius norm of the difference between each pair of sampling methods. After clustering using K-Means and Agglomerative Clustering algorithms, we also compute the Mutual Information Score to compare the consistency of clustering results across different sampling methods. Additionally, we assess the clustering quality using metrics such as the Silhouette Score, Davies-Bouldin Index, and Calinski-Harabasz Score. These metrics provide a comprehensive evaluation of the effectiveness of each sampling method in preserving the graph structure and ensuring high-quality clustering results.

For qualitative evaluations, we visualize the latent space created by these models using t-Distributed Stochastic Neighbor Embedding (t-SNE) [**?**], a statistical method for dimensionality reduction. t-SNE will be employed when the latent space consists of more than two dimensions. By visualizing the latent space, we can better understand how the data is compressed and why the clustering results are either satisfactory or lacking.

The final evaluation involves functional enrichment analysis using the g:Profiler tool [23]. Once we have the cluster labels from K-Means and Hierarchical Clustering, we use g:Profiler to identify over-represented pathways in each cluster. We will then visualize plots and heatmaps to display the differences in pathways across clusters. functional enrichment analysis using g:Profiler involves several key steps. First, an input list of proteins of interest is provided, typically derived from the sample data. g:Profiler then maps these genes or proteins to standardized gene identifiers using data from various biological databases. Next, statistical enrichment analysis is performed, utilizing this test to identify biological terms, such as pathways, that are significantly over-represented in the input list compared to a background set. The over-represented pathways are then visualized using bar plots and heatmaps.

This comprehensive evaluation setup will help us determine the effectiveness of the proposed model and clustering algorithms in identifying meaningful patterns and subgroups within the protein data.

TABLE II: Comparison of Earth Mover's Distance (EMD) of Similarity Matrices between Different Sampling Methods for Dimension $d = 64$

Sampling Method	**RW**	**FF**	**Cluster-GAE**	**No Sampling**
RW	—	0.229	0.431	0.481
FF	0.229	—	0.221	0.277
Cluster-GAE	0.431	0.221	—	0.081
No Sampling	0.481	0.277	0.081	—

Table II presents the Earth Mover's Distance (EMD) values comparing the cosine similarity matrices generated by different graph sampling methods: RW, FF, Cluster-GAE, and No Sampling. Lower values of EMD indicating more similar distributions. The EMD values in the table provide insights into how closely the similarity matrices from each sampling method resemble each other.

RW Sampling: The EMD values indicate that the similarity matrices generated by RW differ significantly from those produced by other methods. These relatively high EMD values suggest that RW sampling, which focuses on local structures, captures a different aspect of the graph compared to the more global perspectives provided by the other methods and may lose some of the information in the graph.

FF Sampling: The FF sampling method shows the lowest EMD with Cluster-GAE (0.221) among all comparisons, suggesting that FF and Cluster-GAE produce more similar similarity matrices compared with RW.

Cluster-GAE: Cluster-GAE demonstrates the lowest EMD value when compared to No Sampling, with an EMD of 0.081. This indicates that Cluster-GAE is highly effective in preserving the overall graph structure. Additionally, the EMD of 0.221 when compared to FF further supports the notion that Cluster-GAE maintains a more comprehensive view of the graph. This is because Cluster-GAE samples from different partitions of the whole graph during the training of the GAE, thereby capturing a broader representation of the graph. In contrast, the EMD value of 0.431 when compared to RW suggests that RW captures a distinctly different structure, highlighting the unique aspects of the graph that RW emphasizes.

No Sampling: Serving as the baseline, the No Sampling method allows us to measure how much the other methods deviate from the original graph structure. Cluster-GAE shows the smallest divergence with an EMD of 0.081, followed by FF with 0.277. RW exhibits the highest divergence with an EMD of 0.481, highlighting its focus on local structures.

Moreover, the values in the table III represent the Frobenius norms of the differences between similarity matrices of protein embeddings generated by different sampling methods. These values provide a measure of how different the similarity matrices are from each other. A lower Frobenius norm indicates that the similarity matrices (and hence the protein embeddings) are more similar, while a higher Frobenius norm indicates greater dissimilarity.

In this case, the pair **FF vs. RW** with a Frobenius norm of 543.328 indicates that these two methods produce the most similar similarity matrices. On the other hand, the highest Frobenius norm values indicate the most dissimilar similarity matrices. **RW vs. No Sampling** and **RW vs. Cluster-GAE** show the highest values, 926.857 and 921.285 respectively, indicating that these two methods produce the most dissimilar similarity matrices.

In summary, according to the table II and table III we conclude that Cluster-GAE is the most effective sampling method. It achieved the lowest Earth Mover's Distance (EMD) and Frobenius Norm compared to no sampling, indicating that it preserves the graph structure very well. Additionally, FF and RW sampling methods were found to be less effective for learning embeddings. While training GAE without any sampling method is infeasible for large graphs due to memory constraints, Cluster-GAE provides a practical and efficient solution, maintaining the integrity of the graph's structure while enabling scalable training.

TABLE III: Comparison of Frobenius Norm of Differences Between Similarity Matrices of Protein Embeddings Across Different Sampling Methods for Dimension $d = 64$

Model 1	Model 2	Frobenius Norm
FF	RW	543.328
FF	No Sampling	754.069
FF	Cluster-GAE	581.145
RW	No Sampling	926.857
RW	Cluster-GAE	921.285
No Sampling	Cluster-GAE	730.927

According to Table V, IV, Cluster-GAE consistently performs the best across both embedding dimensions (64 and 128) in terms of Silhouette and Calinski-Harabasz scores for both $K = 2$ and $K = 3$. Additionally, it achieves the lowest Davies-Bouldin score at an embedding dimension of 128. Comparing no sampling with other sampling methods clearly demonstrates that sampling methods significantly enhance clustering quality. Furthermore, higher embedding dimensions (128) yield similar clustering metrics to those observed with an embedding dimension of 64, indicating that increased dimensionality does not degrade performance and does not necessarily lead to better clustering performance.

Table VI indicates that both the "Cluster-GAE" and "No Sampling" methods exhibit higher and relatively stable mutual information scores across different values of K, suggesting these methods produce more consistent and reliable clustering results. A higher mutual information score between the K-Means and Agglomerative clustering results indicates a greater degree of similarity between the clusters produced by these two different algorithms, reflecting the robustness of the clusters.

The RW method shows a significant increase in mutual information from $K = 2$ to $K = 3$, suggesting improved performance with a greater number of clusters. Similarly, the FF method also shows improvement from $K = 2$ to $K = 3$, though not as pronounced as the improvement seen with RW. Despite their slightly lower scores, the RW and FF methods still demonstrate reasonable consistency in clustering results between the two algorithms for $K = 3$. This indicates that while they may not be as robust as "Cluster-GAE" and "No Sampling," they still maintain a reasonable degree of reliability in producing consistent clustering outcomes with more clusters.

D. Qualitative Analysis

To complement the quantitative metrics, qualitative analyses were also conducted, including:

- **t-SNE Visualization**: The t-distributed Stochastic Neighbor Embedding (t-SNE) technique [31] was used to visualize the high-dimensional protein embeddings in a two-dimensional space. This visualization aids in identifying patterns and clusters within the protein data, offering a more intuitive understanding of the relationships and structures present in the dataset.
- **Functional Enrichment Analysis**: This analysis was applied to the clustered data to identify overrepresented biological functions and pathways within each cluster. By understanding the functional roles of the proteins in each cluster, we can gain insights into the underlying biological mechanisms and their relevance to the conditions studied.

The t-SNE visualization shown in Figure 4 illustrates the clustering of protein embeddings derived using the Cluster-GAE sampling method, followed by the K-Means clustering algorithm with $K = 3$. This plot reduces the high-dimensional protein embedding space into two dimensions, enabling a visual assessment of clustering performance and cluster separation. The clusters are well-separated, indicating that the protein embeddings learned by the Cluster-GAE method effectively capture distinct groups within the data. However, Some points on the borders might appear closer to other clusters, which can indicate overlapping features between clusters.

The clear separation of clusters in the t-SNE plot demonstrates that the Cluster-GAE method successfully learns meaningful protein embeddings, which can be grouped into distinct clusters. This finding shows the robustness of the Cluster-GAE method in capturing the underlying structure of the protein data, providing a reliable foundation for further biological analysis and interpretation. To be more specific, the clusters may represent groups of proteins with similar biological functions or pathways.

This analysis can provide new insights into protein functions and interactions, potentially uncovering novel biological pathways and mechanisms. Additionally, the distinct clustering of proteins can assist in identifying potential biomarkers for diseases, thereby guiding the development of targeted therapies.

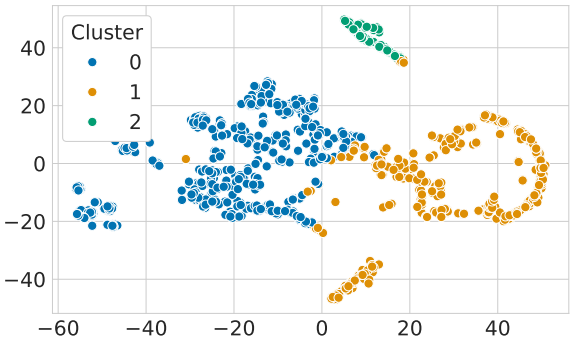

Fig. 4: t-SNE visualization of protein embeddings clustered using K-means ($K = 3$) with Cluster-GAE.

The result of functional enrichment analysis is shown in Figures 5 and 6. The bar plot in Figure 5 shows the top enriched pathways for each cluster derived from the K-Means clustering method. The x-axis represents the -log10 p-value, which indicates the significance of enrichment, with higher values suggesting more significant enrichment. The y-axis

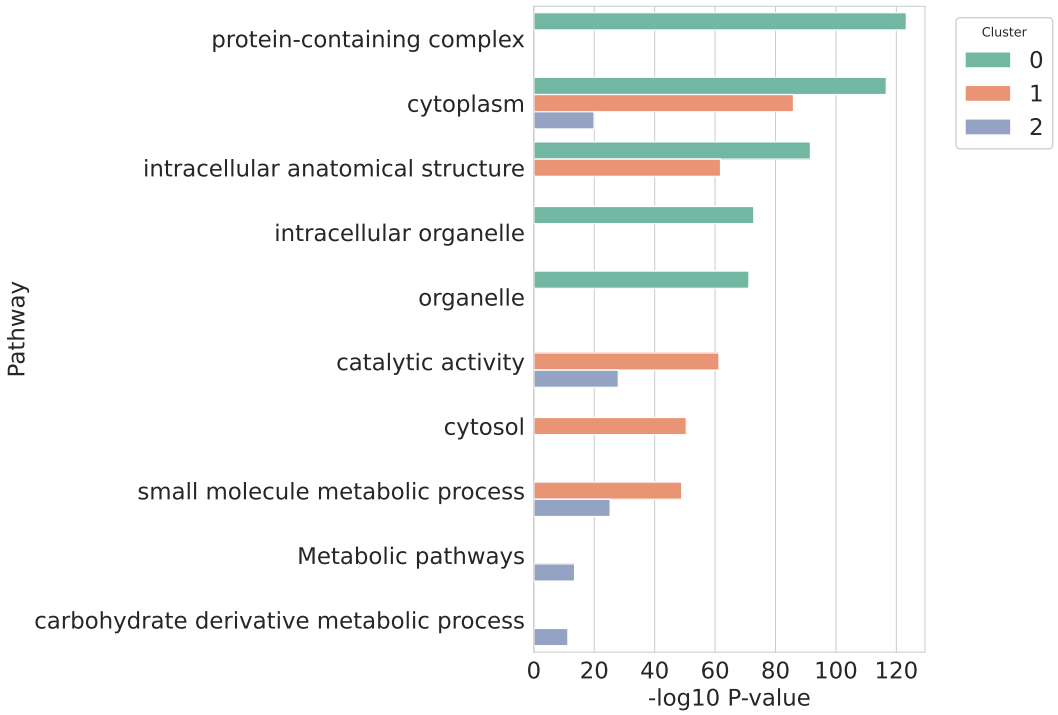

Fig. 5: Top 10 over-represented pathways in the clusters after doing the functional enrichment analysis

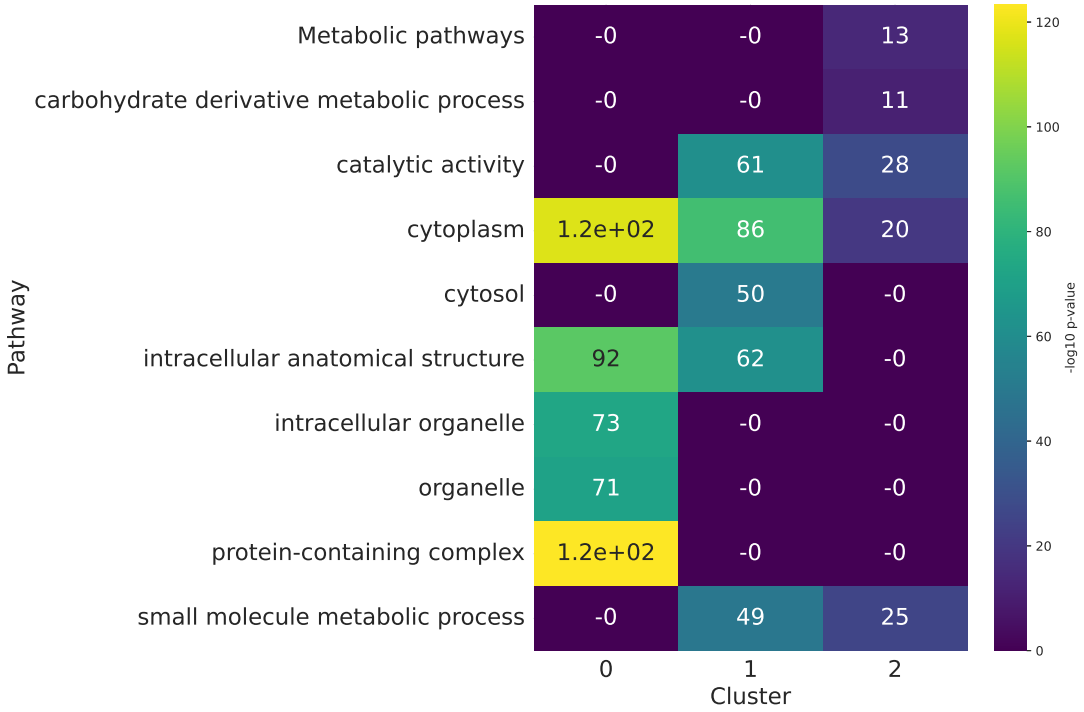

Fig. 6: Top 10 over-represented pathways in the clusters after doing the functional enrichment analysis

TABLE IV: Comparison of Clustering Metrics for Different Embedding Dimensions and Sampling Methods for $K = 2$ using K-means. Metrics include Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index, evaluated for embedding dimensions of 64 and 128.

Sampling Method	Embedding Dimension = 64			Embedding Dimension = 128		
	Silhouette	Calinski-Harabasz	Davies-Bouldin	Silhouette	Calinski-Harabasz	Davies-Bouldin
RW	**0.672**	791.043	0.866	0.417	1130.384	1.083
FF	0.432	1068.283	0.972	0.460	1261.628	0.969
Cluster-GAE	0.551	**2138.868**	**0.736**	**0.534**	**2026.599**	**0.775**
No Sampling	0.369	784.169	1.281	0.226	394.942	1.922

TABLE V: Comparison of Clustering Metrics for Different Embedding Dimensions and Sampling Methods for $K = 3$ using K-means. Metrics include Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index, evaluated for embedding dimensions of 64 and 128.

Sampling Method	Embedding Dimension = 64			Embedding Dimension = 128		
	Silhouette	Calinski-Harabasz	Davies-Bouldin	Silhouette	Calinski-Harabasz	Davies-Bouldin
RW	0.502	1141.947	0.777	0.420	904.776	0.943
FF	0.513	1433.257	**0.707**	0.497	1129.500	0.805
Cluster-GAE	**0.576**	**1794.486**	0.770	**0.556**	**1615.266**	**0.761**
No Sampling	0.438	864.572	0.918	0.270	454.419	1.493

TABLE VI: Mutual Information Scores between Clustering Algorithms ($K = 2$ and $K = 3$)

Sampling Method	K-Means vs Agglomerative ($K = 2$)	K-Means vs Agglomerative ($K = 3$)
RW	0.231	0.688
FF	0.481	0.667
Cluster-GAE	**0.568**	**0.715**
No Sampling	**0.596**	**0.742**

lists the pathways, and the different colors represent the three clusters (0, 1, and 2).

The heatmap in Figure 6 provides a detailed visualization of the enrichment levels for each pathway across the three clusters, with the color scale representing the -log10 p-value. Lighter colors indicate higher enrichment significance.

Both visualizations indicate that in Cluster 0 ”Cytoplasm” and ”protein-containing complex” pathways exhibit the highest -log10 p-values, indicating very strong enrichment. Other pathways such as ”intracellular anatomical structure”, ”intracellular organelle”, and ”organelle” also show high enrichment levels in this cluster.

In Cluster 1, the ”catalytic activity” pathway shows significant enrichment, along with ”cytoplasm”, ”intracellular anatomical structure” and ”cytosol”.

Cluster 2 exhibits lower enrichment levels compared to clusters 0 and 1. The top enriched pathways in this cluster include ”catalytic activity,” ”small molecule metabolic process,” and ”cytoplasm.”

Understanding these enriched pathways helps to explain the functional landscape of the protein clusters, offering potential targets for further biological investigation and research.

VI. Conclusion and Future Works

This thesis explored the application of GNNs and graph sampling methods to analyze biological sample datasets, particularly focusing on proteins with limited sample sizes. Our proposed method, **Cluster-GAE**, combined graph sampling techniques with GNNs to effectively learn representations in large graph structures.

We evaluated four sampling methods: RW, FF, Cluster-GAE, and No Sampling (baseline). Metrics such as Earth Mover’s Distance, Frobenius Norm, Mutual Information Score, and various clustering metrics were used to assess the similarity and quality of protein embeddings. Cluster-GAE consistently preserved graph structure most effectively, demonstrated by low EMD and Frobenius Norm values, and had the highest Mutual Information Score and the best results in clustering metrics, indicating reliable clustering. Additionally, higher embedding dimensions yielded clustering metrics similar to those observed with an embedding dimension of 64, suggesting that it does not necessarily improve clustering quality. Notably, Cluster-GAE is particularly advantageous for larger graph sizes, maintaining its effectiveness in handling complex data structures. Training very large graphs with GAE without any sampling method is impractical, as it cannot fit into memory, highlighting the necessity and efficiency of Cluster-GAE for scalable training.

Qualitative evaluations with t-SNE visualizations and functional enrichment analysis supported our quantitative findings. t-SNE provided intuitive clustering insights, while functional enrichment analysis identified over-represented pathways in each cluster, highlighting the biological relevance of our method. Our approach enhances the ability of biological researchers to uncover meaningful patterns and interactions within protein data, offering valuable insights for future research.

For future work, several directions can be pursued to build upon our findings. Firstly, expanding the size of the graph dataset by incorporating more proteins from Reactome could provide a richer and more comprehensive analysis. Collaborations with biological researchers will be crucial to validate and interpret the model’s results, ensuring that the discovered

patterns and insights are meaningful and biologically relevant. Additionally, the GAE used in our method could be replaced with a Variational Graph Autoencoder to evaluate its effectiveness in capturing more complex latent structures within the data. These enhancements will further refine our method, potentially leading to novel discoveries and applications in the analysis of biological networks.

Acknowledgment

I would like to thank Daniele Dell'Aglio and Juan Manuel Rodriguez for their invaluable guidance and supervision throughout this project. I also thank Kenneth Kastaniegaard and Alessandro Ranieri at Biogenity company for their helpful discussions and assistance. You can find the code related to this thesis at https://github.com/dellaglio/gnn-pathways

References

[1] Peter Feist and Amanda B. Hummon. Proteomic challenges: Sample preparation techniques for microgram-quantity protein analysis from biological samples. *International Journal of Molecular Sciences*, 16(2):3537–3563, 2015. 2

[2] Jianyu Miao and Lingfeng Niu. A survey on feature selection. *Procedia computer science*, 91:919–926, 2016. 2

[3] Bijay Jassal, Lisa Matthews, Guilherme Viteri, Chuqiao Gong, Pascual Lorente, Antonio Fabregat, Konstantinos Sidiropoulos, Justin Cook, Marc Gillespie, Robin Haw, et al. The reactome pathway knowledgebase. *Nucleic acids research*, 48(D1):D498–D503, 2020. 2, 3

[4] Minoru Kanehisa, Miho Furumichi, Yoko Sato, Mari Ishiguro-Watanabe, and Mao Tanabe. Kegg: integrating viruses and cellular organisms. *Nucleic acids research*, 49(D1):D545–D551, 2021. 2

[5] Uniprot: the universal protein knowledgebase in 2023. *Nucleic acids research*, 51(D1):D523–D531, 2023. 2, 3

[6] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020. 3, 4

[7] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, 2018. 3

[8] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4):1–37, 2021. 3

[9] Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636, 2006. 3, 4, 7

[10] Pili Hu and Wing Cheong Lau. A survey and taxonomy of graph sampling. *arXiv preprint arXiv:1308.5865*, 2013. 3, 6

[11] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE transactions on knowledge and data engineering*, 30(9):1616–1637, 2018. 4

[12] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003. 4

[13] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016. 4, 5, 7

[14] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017. 4

[15] William L. Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159. 4

[16] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 5

[17] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 257–266, 2019. 6

[18] Yazhou Ren, Jingyu Pu, Zhimeng Yang, Jie Xu, Guofeng Li, Xiaorong Pu, Philip S Yu, and Lifang He. Deep clustering: A comprehensive survey. *arXiv preprint arXiv:2210.04142*, 2022. 6

[19] Ka-Chun Wong. A short survey on data clustering algorithms. In *2015 Second international conference on soft computing and machine intelligence (ISCMI)*, pages 64–68. IEEE, 2015. 6

[20] Neo4j. Neo4j graph database & analytics. https://neo4j.com/, 2024. Accessed 24 May 2024. 6

[21] David Adedayo Adeniyi, Zhaoqiang Wei, and Yang Yongquan. Automated web usage data mining and recommendation system using k-nearest neighbor (knn) classification method. *Applied Computing and Informatics*, 12(1):90–108, 2016. 8

[22] Adrian Garcia-Moreno, Raul López-Domínguez, Juan Antonio Villatoro-García, Alberto Ramirez-Mena, Ernesto Aparicio-Puerta, Michael Hackenberg, Alberto Pascual-Montano, and Pedro Carmona-Saez. Functional enrichment analysis of regulatory elements. *Biomedicines*, 10(3):590, 2022. 8

[23] J Reimand, R Kolde, T Arak, P Adler, H Peterson, and J Vilo. g:profiler—a web server for functional interpretation of gene lists (2016 update). *Nucleic Acids Research*, 44(W1):W83–W89, 2016. 8, 10

[24] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004. 9

[25] Scikit learn Developers. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mutual_info_score.html. Accessed: 2024-06-09. 9

[26] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987. 9

[27] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227, 1979. 9

[28] Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974. 9

[29] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40:99–121, 2000. 9

[30] SciPy Developers. https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wasserstein_distance.html. Accessed: 2024-06-09. 9

[31] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 11