

# EXPLORING GRAPH NEURAL NETWORKS FOR CLUSTERING AND CLASSIFICATION

by

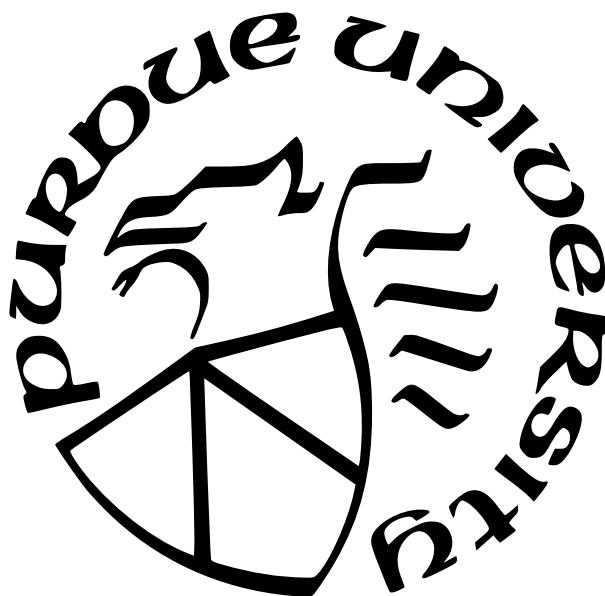
Fattah Muhammad Tahabi

A Thesis

*Submitted to the Faculty of Purdue University*

*In Partial Fulfillment of the Requirements for the degree of*

Master of Science



Department of Electrical and Computer Engineering

Indianapolis, Indiana

December 2022

**THE PURDUE UNIVERSITY GRADUATE SCHOOL  
STATEMENT OF COMMITTEE APPROVAL**

**Dr. Xiao Luo, Co-Chair**

Department of Computer and Information Technology

**Dr. Brian King, Co-Chair**

Department of Electrical and Computer Engineering

**Dr. Lingxi Li**

Department of Electrical and Computer Engineering

**Approved by:**

Dr. Brian King

## ACKNOWLEDGMENTS

I am indebted to the thesis committee members, Dr. Xiao Luo, Dr. Brian King, and Dr. Lingxi Li, for their assistance and mentoring. I would like to thank Sherrie Tucker for her regular reminders about important academic events since the start of the program. Finally, I want to express my gratitude to my family members for their unwavering support.

# TABLE OF CONTENTS

LIST OF TABLES . . . . .	8
LIST OF FIGURES . . . . .	9
ABSTRACT . . . . .	10
1 INTRODUCTION . . . . .	11
2 GNN FUNDAMENTALS . . . . .	13
2.1 A Brief Introduction to Graph Neural Network . . . . .	13
2.2 Graph Learning Strategies . . . . .	14
2.3 Some GNN Models Used in Our Research . . . . .	14
2.3.1 Weighted Node2Vec . . . . .	15
2.3.2 GCN . . . . .	17
2.3.3 GAT . . . . .	17
2.3.4 GRAPHSAGE . . . . .	18
2.3.5 BiLSTM with Attention Mechanism on GNN . . . . .	18
3 CLUSTERING AND CLASSIFICATION ON GRAPHS . . . . .	20
3.1 What is Clustering and Classification on Graphs . . . . .	20
3.2 Difference between Classification and Clustering . . . . .	20
3.3 Common Techniques for Node Clustering on Graphs . . . . .	21
3.3.1 Hierarchical Clustering . . . . .	21
3.3.2 Graph Partitioning . . . . .	21

3.3.3	Spectral Clustering . . . . .	22
3.3.4	Edge Betweenness Clustering . . . . .	23
3.3.5	Partitional Clustering . . . . .	23
3.4	Common Techniques for Node Classification on Graphs . . . . .	24
3.4.1	Community-Based Node Classification . . . . .	25
3.4.2	Role-Based Node Classification . . . . .	26
3.5	A Simple Network for Graph Analysis . . . . .	26
4	BACKGROUND AND RELATED WORKS . . . . .	30
4.1	Graph Clustering Related Works in Clinical Domain . . . . .	30
4.1.1	Symptoms and Symptom Clusters Extraction . . . . .	31
4.1.2	Graph Models and Their Application to Clinical Data . . . . .	32
4.2	Node Classification Related Works on Dynamic Graphs . . . . .	33
5	CLUSTERING ON SYMPTOM GRAPH . . . . .	35
5.1	Node Clustering to Solve Problems in Healthcare . . . . .	35
5.2	Objective . . . . .	35
5.3	Methodology . . . . .	36
5.3.1	Model Pipeline . . . . .	36
5.3.2	Symptom Extraction . . . . .	37
5.3.3	Semantic Grouping . . . . .	38
5.3.4	Symptom Graph Construction . . . . .	38

5.4	Symptom Graph Clustering . . . . .	40
5.4.1	Symptom Embedding Generation . . . . .	40
5.4.2	Symptom Clustering . . . . .	40
5.5	Experimental Results . . . . .	42
5.5.1	Data Set . . . . .	42
5.5.2	Extracted Symptoms and Construction of Symptom Graph . . . . .	44
5.5.3	Symptom Graph Clustering Results . . . . .	46
5.5.4	Statistical Analysis and Evaluation of the Symptom Clusters . . . . .	50
5.6	Discussion . . . . .	53
5.7	Summary . . . . .	54
6	NODE CLASSIFICATION ON DYNAMIC GRAPHS . . . . .	55
6.1	Objective . . . . .	55
6.2	Methodology . . . . .	55
6.2.1	Problem Statement . . . . .	55
6.2.2	G2B Architecture and Edge Importance Interpretation . . . . .	56
6.3	Experimental Results . . . . .	58
6.3.1	Datasets and Baselines . . . . .	58
6.3.2	Performance Results . . . . .	59
6.3.3	Model Analysis . . . . .	59
6.3.4	Dynamic Edge Interpretation . . . . .	61

6.4	Summary . . . . .	62
7	CONCLUSION . . . . .	63
	REFERENCES . . . . .	65

## LIST OF TABLES

5.1	Snippets of Clinical Notes and Extracted Symptoms . . . . .	40
5.2	Extracted Symptoms from Clinical Notes . . . . .	45
5.3	Hyperparameters of Weighted Node2Vec . . . . .	47
5.4	Statistics of the Patients with Symptoms of Each Cluster . . . . .	52
6.1	Summary of Datasets . . . . .	59
6.2	Performance Comparison Against the Baselines . . . . .	60



## LIST OF FIGURES

2.1	Pipeline of GNN . . . . .	13
2.2	Edge Weight Transition between Nodes . . . . .	16
3.1	Two Networks with Graph Partitioning. The Minimum Bisection Solution Indicated by the Dashed Lines . . . . .	22
3.2	A Graphical Depiction of the Members of Zachary’s Karate Club Network with Two Ground Truth Communities in Different Colors . . . . .	27
3.3	The Loss Curve for Zachary’s Karate Club Network Converges After 5500 Epochs . . . . .	27
3.4	(a) Zachary’s Karate Club Network, (b) Corresponding Node2Vec Embedding Representation . . . . .	28
3.5	(a) Clustering Algorithms Executed on Node2Vec Embedding, (b) Clustering Produces Four Communities in Different Colors . . . . .	29
5.1	System Framework of SymptomGraph . . . . .	37
5.2	Symptom Phrases Detection Using UMLS MetaMap . . . . .	38
5.3	Graph Built Based on Clinical Notes in Table 5.1 . . . . .	39
5.4	Top 10 Frequent Types of Clinical Notes . . . . .	43
5.5	Distribution of the Clinical Parameters . . . . .	43
5.6	Symptom Graph Built Using Our Data Set . . . . .	44
5.7	t-SNE Visualization of Symptom Distribution . . . . .	46
5.8	Dunn Index of Different Clustering Models . . . . .	48
5.9	Dunn Index of Different HC Methods . . . . .	48
5.10	Dendrogram of HC Ward Linkage Method with Threshold at 0.85 Indicated by Dashed Vertical Line . . . . .	49
5.11	Symptom Clustering Result . . . . .	49
6.1	System Framework of G2B . . . . .	57
6.2	F1 of the Model Analysis . . . . .	60
6.3	ACC of the Model Analysis . . . . .	61
6.4	AUC the Model Analysis . . . . .	61
6.5	Edge Importance Interpretation . . . . .	62

## ABSTRACT

Graph Neural Networks (GNNs) have become excessively popular and prominent deep learning techniques to analyze structural graph data for their ability to solve complex real-world problems. Because graphs provide an efficient approach to contriving abstract hypothetical concepts, modern research overcomes the limitations of classical graph theory, requiring prior knowledge of the graph structure before employing traditional algorithms. GNNs, an impressive framework for representation learning of graphs, have already produced many state-of-the-art techniques to solve node classification, link prediction, and graph classification tasks. GNNs can learn meaningful representations of graphs incorporating topological structure, node attributes, and neighborhood aggregation to solve supervised, semi-supervised, and unsupervised graph-based problems. In this study, the usefulness of GNNs has been analyzed primarily from two aspects - clustering and classification. We focus on these two techniques, as they are the most popular strategies in data mining to discern collected data and employ predictive analysis.

# 1. INTRODUCTION

Plenty of modern problems can be categorized as classification and clustering tasks of machine learning. Innovative ideas are being developed in research communities to enhance the robustness of these techniques for development purposes. In this study, we encounter these two categories from a graphical point of view, adopting Graph Neural Networks.

Classification is the systematic approach of recognizing, understanding, and grouping objects into given categories. Class labels are predicted for the examples of input data. Classification algorithms leverage input training data for anticipating the likelihood or probability that new data fall into one or more of the respective and relevant categories. Finally, they assign predefined tags to instances based on features. Classification algorithms are massively used in filtering spam emails, identifying health problems, face and speech recognition, sentiment analysis, object detection, and whatnot.

Clustering is an unsupervised learning approach for similar grouping of instances depending on their features without using the class labels. To generate the clusters, there is no need for a training dataset; hence, it is often less complex than classification problems. Clustering is used in several industries, such as market segmentation, social network analysis, medical imaging, anomaly detection, image segmentation, etc.

Even though Classification and Clustering problems have many solutions in Euclidean space, often, the scenarios model to the non-Euclidean domain. Since graph structures best represent such phenomena, graph analysis techniques come into the picture. Graphs are complicated objects that are not conducive to conventional learning. To understand the problem specification, we need to map the problem to the graph domain. Then different graph neural network tools can be used to generate node embeddings taking care of their attributes and edge interactions with their neighborhood. After a sufficient amount of training, the models converge to represent the graph information best. In simple words, with proper embeddings of graphs, the problem can be projected to the feature space to apply different classification and clustering methods.

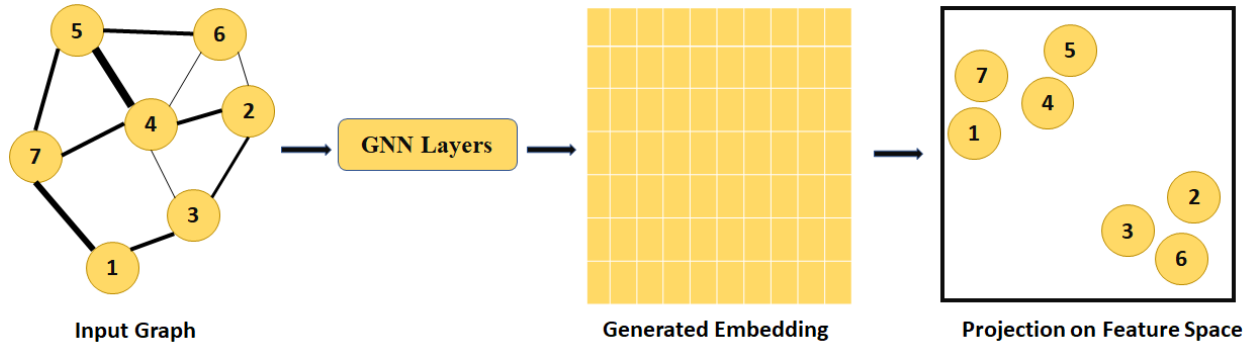
When dealing with graphs, they can either be static or dynamic. Static graphs do not allow the change of the topology and attributes over time, whereas nodes and edge connections

and features can be altered on different time stamps in dynamic graphs. Both classification and clustering techniques exhibit different behaviors in various types of graphs. In the case of temporal graphs, the results can be different at different time stamps. In this research, we apply node clustering in a static symptom graph built from Electronic Health Record (EHR) clinical notes of patients to determine symptom clusters. Later we focus on employing node classification on temporal graphs where node and edge features vary over time.

## 2. GNN FUNDAMENTALS

### 2.1 A Brief Introduction to Graph Neural Network

A GNN is an optimizable representation that may be applied to all of the characteristics of the graph while still conserving the permutation invariances, also defined as graph symmetries. With GNN, each node uses the feature vectors of its neighbors to generate a new feature vector in a recursive neighborhood aggregation (or message passing) method, which occurs mainly in three phases. The network embeddings of the nodes next to a given node (or messages) are determined. All the messages are assembled with the aid of an aggregation operation. All of the merged data is then updated using a function. After certain iterations of the preceding steps, the altered feature vector of a node is used to represent the node. Message-passing GNN layers can be stacked on top of each other to provide a single node information access from all other nodes in the graph; for instance, after four levels, a node has the understanding of the nodes that are four layers far from it. With pooling, for instance, by adding together feature vectors across all nodes in the graph, we can generate a representation of the whole graph [1].



**Figure 2.1.** Pipeline of GNN

Figure 2.1 presents a simple overview of the GNN pipeline. A well-GNN model generates the most accurate embedding of the nodes in the feature space. It considers the node attributes, edge weights, and relative positions of the nodes. A successful embedding generation results in perfect graph analysis.

## 2.2 Graph Learning Strategies

Graph Neural Networks adopt three different learning approaches in machine learning - supervised, semi-supervised and unsupervised techniques. In supervised learning, we know some presumptions of the target outcome. The model’s primary goal is to either map the output with the input labels or predict the output labels given the labeled input data. These algorithms are supervised in their work by the datasets provided. They can do so because of the incorporated feedback systems. On the other hand, the purpose of unsupervised learning is to explore contextual patterns in a dataset without labels. Unsupervised learners can benefit from supervised learning because it does not rely on a feedback system to improve results. Sometimes, we also have very few labeled but many unlabeled ones. In such cases, semi-supervised learning comes into the picture. While supervised learning presupposes that the whole dataset to be instructed on a task includes labels for each input, that may not be the case. Labeling is a time-consuming processing job because input data is generally unpaired.

## 2.3 Some GNN Models Used in Our Research

This study explores node clustering in a static graph and node classification on a dynamic graph. First, we employed a weighted Node2Vec model to construct symptom embeddings based on the symptom graph described in detail in Chapter 5. In the second part of this research, We develop a model to compare to some of the state-of-the-art baselines, including GCN[2], GAT[3], GraphSAGE[4] in the field of dynamic node classification which is elaborated in Chapter 6. Combinations of these basic GNNs derived some other custom GNN models like GC-LSTM[5], EGCN[6], RNNGCN[7], TRNNGCN[7] and G2B (our model) used for dynamic node classification. Below we briefly describe these different types of popular GNNs to provide enough background knowledge to help understand the research details.

### 2.3.1 Weighted Node2Vec

Node2Vec [8] is one of the most advanced unsupervised GNNs with an identity feature matrix that can be trained to build node embeddings, i.e., each node learns its vector representation.

$$P\left(\frac{y}{c}\right) = \begin{cases} \frac{\mu_{cy}}{N} & \text{if } e_{cy} \in G \\ 0 & \text{otherwise} \end{cases} \quad (2.1)$$

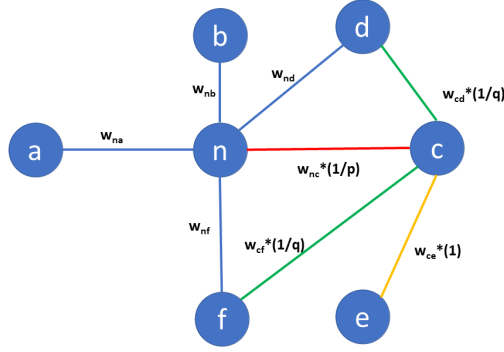
Node2Vec utilizes breadth-first Sampling (BFS) and Depth-first Sampling (DFS) to explore neighborhoods using a biased random walk approach. The BFS investigates the immediate neighbors of the source node. In contrast, the DFS provides information on nodes placed at a greater distance from the source node, providing a global perspective of the graph structure. Node2Vec interpolates between BFS and DFS using a fixed-length random walk algorithm.

Given a graph  $G = (V, E)$ , the walk from node  $c$  to  $y$  is defined by the probability density function  $P$  in Equation 2.1, where  $\frac{\mu_{cy}}{N}$  is the transition probability between node  $y$  and  $c$ .  $e_{cy}$  is the edge in  $G$  that enable the connection between node  $y$  and  $c$ .  $N$  is the normalizing constant.

A random walk for a weighted network is meant to use the edge weights to determine the transition probabilities between two nodes. As shown in Figure 2.2, if the random walk came through edge  $(n, c)$  from node  $n$  to  $c$ , the transition probability  $\mu_{nc}$  on edge  $(n, c)$  is calculated as Equation 2.2, where  $\beta_{pq}(n, y)$  is determined by distance ( $d_{ny}$ ) between  $n$  and a neighbor ( $y$ ) of  $c$ , and  $w_{cy}$  is the weight on the edge between a neighbor ( $y$ ) and  $c$ .

$$\mu_{cy} = \beta_{pq}(n, y) \cdot w_{cy} \quad y \in \{d, e, f, n\} \quad (2.2)$$

There are three types of neighbors in the calculation. Based on the graph shown as Figure 2.2, weights are assigned differently to each type, shown as Equation 2.3. Both  $p$  and  $q$  determine the pace of the walk's investigation and departure from the beginning node's neighborhood.  $p$  determines whether a node may be instantly re-explored throughout the



**Figure 2.2.** Edge Weight Transition between Nodes

walk, whereas  $q$  enables the search to differentiate between inner and outward nodes. The edge weights affect the likelihood of reaching the following node from the preceding node.

$$\beta_{pq}(n, y) = \begin{cases} \frac{1}{p} & \text{if } d_{ny} = 0, \text{ when } y = n \\ 1 & \text{if } d_{ny} = 1, \text{ when } y = e \\ \frac{1}{q} & \text{if } d_{ny} = 2, \text{ when } y = d \text{ or } f \end{cases} \quad (2.3)$$

The random walks are used to construct node embeddings after sampling the maximum number of random walks with maximum length  $L$ . The goal of the embedding generation process is to maximize the logarithmic probability of detecting a network neighborhood  $M_s(u)$  for a given node  $u$ , depending on its feature representation.

Equation 2.4 represents the objective function of the Node2Vec training process, whereas  $P(\frac{M_s(u)}{f(u)})$  represents the probability of exploring neighboring nodes of node  $u$  from its embedded space.

$$\max_f \left( \sum_{u \in V} \log P\left(\frac{M_s(u)}{f(u)}\right) \right) \quad (2.4)$$

For the training of the graph neural network, the skip-gram negative sampling model is utilized to generate pairs of input and context nodes with a predefined window size.



### 2.3.2 GCN

Graph Convolutional Networks, also known as GCNs, are extensions of graph neural networks that use convolutional layers to learn from graph data. Each graph node in a GCN network obtains the characteristics from its neighbors and then iteratively combines those features utilizing an aggregation function such as the average. The aggregation function needs to use an isotropic aggregation, meaning that each neighbor should contribute the same amount to updating the representation of the central node. The compiled features are then introduced into a neural network with convolutional layers. The neural network generates the new vector representation of the node. The entire procedure is then repeated while utilizing the newly added node characteristics. GCNs have strong performance in various contexts regarding tasks involving node classification.

### 2.3.3 GAT

Graph Attention Networks (GATs) are the other subset of GNNs [9]. To address the limitations of Graph Convolutional Networks (GCNs), GATs are neural architecture networks that process data in a graph format and include masked self-attentional layers. GAT uses stacked layers, in which individual nodes can be constituted of characteristics that are shared with their companions. When attention is paid to these nodes, the entire network begins to provide distinct weights to the various nodes that are present in the vicinity only. The neighbors of each node are given different weights in a GAT model, according to their relative contributions, using the attention layers. As a result, in a GAT, the edges carry a greater or lesser amount of weight depending on the importance of the nodes they link. Applying this strategy, the network will be able to perform using just the key data of the nodes that are of relevance to it. Attention is responsible for giving these weights, which permits us to pay more attention to nodes that have a higher significance. We update the reflection of each node with a weighted sum of adjacent representations.

### 2.3.4 GRAPHSAGE

The GraphSAGE [4] neural network solution is a convolutional graph neural network. The philosophy of the approach is to develop a module that builds deep node features by sampling and averaging feature representations from a node’s surroundings. The GraphSAGE procedure may be used to infer the embeddings of previously unseen nodes by learning a function that can trigger the embedding of a node. It’s commonly called “inductive learning,” when information is obtained in this way.

To explain GraphSage, let’s assume we have a graph  $G = (V, E)$ , where  $V$  represents the vertices and  $E$  represents the edges with  $m$  aggregators used to move information across layers. Each node  $u$  collects the representation  $h_m$  of nodes from its nearest neighbor  $N(u)$ , a function of the representation created in the previous iteration. If  $f$  is an input feature, then the initial representation is  $h_0 = f_0$ . The aggregate function is given below in Equation 2.5

$$h_m = AGGREGATE(h_u, \forall u \in N(u)) \quad (2.5)$$

The representation of the vertex is concatenated with the aggregated representation of the surrounding. In designing the representation needed for the subsequent stage, the concatenated representation is first routed through a fully connected layer, and then a non-linear activation function is incorporated into it [10].

### 2.3.5 BiLSTM with Attention Mechanism on GNN

Bidirectional Long Short-term Memory (BiLSTM) models are sequence processing models comprised of two LSTMs, one of which processes the input in a forward direction and the other in a backward direction [11]. With a regular LSTM, we can only direct the input to move forward or backward. Forward and backward passes are made across the unfolded network over time. This is similar to how forward and backward passes are made in a normal network, except that BiLSTM must unfold the hidden forward states and the backward hidden states for each time step. Backpropagation over time is used to train the BiLSTM networks (BPTT) [12], [13]. It is well known that BiLSTM can address the issue of gradient

vanishing problem that can happen in RNN [14]. BiLSTM is an effective technique because it expands the amount of data available to the network and improves the contextual information available to the algorithm. BiLSTM is used in emerging graph neural networks to process sequences of changing duration, as seen, for instance, in dynamic graphs. It goes beyond the limitations of conventional LSTM by making use of contextual knowledge from future data to examine local behavior. Using bidirectionality, we can make sure that information is not lost. In addition, the BiLSTM layers with an attention mechanism applied to them reveal useful insights into the network.

### 3. CLUSTERING AND CLASSIFICATION ON GRAPHS

#### 3.1 What is Clustering and Classification on Graphs

It is not a simple process to maximally embed the graph structure and node attributes into the low-dimensional feature space, which is the objective of graph clustering based on embedding. Most current sophisticated approaches for clustering network nodes separate graph embedding technology and clustering algorithm, ignoring their possible correlation. For node classification and representation learning on graphs in general, various techniques focusing on graph kernels or graph neural networks have already been made in recent years.

In node classification, a node receives a label based on its resemblance to other nodes in the network. The similarity between nodes may be defined using several criteria, including homophily, influence diffusion/maximization, and other node attributes [15]. Homophily is the tendency for people to communicate effectively with those who are pretty similar to themselves[16].

Often the objective of graph analysis is to find similar groups and separate dissimilar ones in the network. This task is called node clustering, also known as community detection. Here we find patterns from the available features of the nodes. The most important fact here is that graph-based node clustering does not use labels in the training phase.

#### 3.2 Difference between Classification and Clustering

The predictive analysis employs classification and clustering techniques to understand data sets and split them following certain segmentation rules or the interconnections between items. With the help of the available training data, classification assigns labels to the data. Instead of using a single similarity metric to group data, clustering employs numerous. Objects can be divided into classes by one or more attributes using two distinct types of learning strategies: classification and clustering. Although deceptively similar, these procedures are actually rather distinguishable when applied to the field of data mining. Broadly, clustering is used in unsupervised learning to group instances with similar characteristics or qualities. In contrast, classification is used in supervised learning to give labels to examples based on

their properties (thus the name). Learning a model to focus exclusively on distinguishable categories of data is known as classification. There are two important phases to the process: training and labeling. During the learning phase, a classification classifier is developed, and later, during the classification phase, that model is deployed to infer the class labels for the unlabeled data.

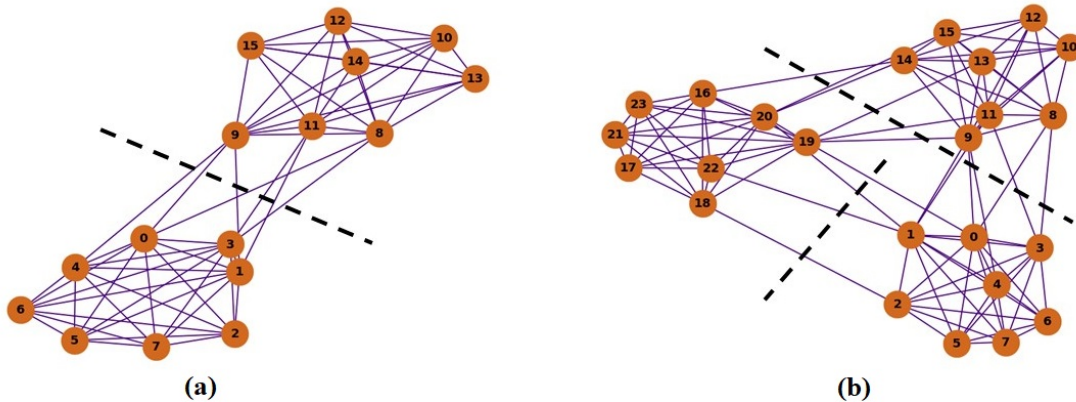
### **3.3 Common Techniques for Node Clustering on Graphs**

#### **3.3.1 Hierarchical Clustering**

The vertices in a graph may be arranged in a hierarchy, with smaller groups included within bigger ones, which in turn may be contained within still-larger ones, and so on. For example, social networks frequently have a hierarchical layout. Hierarchical clustering methods can be useful in such situations since they expose the graph’s multi-tiered structure [17]. The goal of hierarchical clustering is to find groups of nodes that are very similar to one another. Determining a vertex-to-vertex similarity measure is the first step in any hierarchical clustering technique. When a measure is selected, the similarity between any two vertices is calculated, regardless of whether they are linked. After completing this procedure, a new matrix, the similarity matrix, is produced. Traditionally, there is the agglomerative method, in which clusters are periodically merged if their similarity is strong enough. In a second variation of the hierarchical clustering algorithm, called a divisive method, clusters are repeatedly broken up by cutting off connections between nodes that aren’t very similar.

#### **3.3.2 Graph Partitioning**

The task of graph partitioning consists of splitting the vertices into smaller groups of a preset size in such a way that the number of edges that lie between the groups is reduced to the smallest possible value. The term “cut size” refers to the number of edges that may be found passing between clusters. In Figure 3.1(a) for a network that has sixteen vertices with two groups, whereas in Figure-3.1(b), the network has three clusters in its twenty-four vertices.



**Figure 3.1.** Two Networks with Graph Partitioning. The Minimum Bisection Solution Indicated by the Dashed Lines

When creating a partition, providing the desired number of clusters is essential. The solution is easy, with all vertices falling into the same cluster if one imposes a partition with the smallest cut size and leaves the number of clusters unconstrained. It is also essential to provide the size of the graph; without doing so, the most likely option would be to cut off the vertex with the lowest degree from the remainder of the graph.

It is common practice for algorithms to “bisect” the graph. Iterative bisection is commonly used to achieve partitions into clusters. The requirement that the clusters be of similar size is typically imposed. The minimal bisection issue is a fairly difficult NP-hard problem. Graph partitioning has several important uses, including in scientific computing, separating VLSI design phases, and scheduling tasks over many processors.

### 3.3.3 Spectral Clustering

When traditional clustering methods like K-Means fail to provide valuable results due to the presence of outliers, spectral clustering might be a useful alternative. However, it requires setting the similarity threshold and the predicted number of clusters. Knowledge from the eigenvalues of particular matrices constructed from the graph or data set is used in spectral clustering. If there is a nonzero vector  $x$  and a positive real number such that  $Ax = \lambda x$ , then

$x$  is an eigenvector of the matrix  $A$ , where  $\lambda$  is the eigenvalue [18]. The matrix  $A$  converts the previous vectors into new vectors. When the  $A$  is applied to most vectors, those vectors will almost always end up in an entirely new location. The magnitude of an eigenvector is the only thing that changes. If a line connects the origin and the eigenvector, the eigenvector will continue to be on the line even after the mapping has been performed. The magnitude of the vector scaling along the line depends on the value of  $\lambda$  [19].

Spectral clustering is linked to random walks on graphs. In fact, one may encourage random walkers to concentrate within clusters and less time traveling between clusters by reducing the number of edges connecting them. In a bipartition, the normalized cut is the product of the two probabilities of a random walker going from one cluster to the other. Finding a partition that minimizes the frequency of interactions between clusters is what we mean when we talk about reducing the normalized cut.

### 3.3.4 Edge Betweenness Clustering

Girvan and Newman concentrated on the idea of betweenness, which is the sum of all the shortest paths between adjacent vertex pairs. It expresses the magnitude of edges' involvement in a process, referred to as edge betweenness. It should be no surprise that inter-cluster edges have a high value of the edge betweenness. This is due to the fact that many of the shortest pathways linking the vertices of different clusters will cross through these edges. Edge Betweenness is calculated as Equation 3.1.

$$c_B(e) = \sum_{p,q \in V} \frac{\sigma(\frac{p,q}{e})}{\sigma(p,q)} \quad (3.1)$$

where  $V$ ,  $\sigma(p,q)$  and  $\sigma(\frac{p,q}{e})$  are the set of nodes, number of shortest  $(p,q)$ -paths and number of those paths passing through edge  $e$  [20] respectively.

### 3.3.5 Partitional Clustering

The partitional clustering algorithm is also a widely used technique. Let's suppose that  $K$  is the predetermined number of clusters. Each vertex is a point within the metric space,

and distances between any two points are quantified. It's a way to quantify how different two vertices are from one another. The objective is to partition the points into  $K$  clusters in such a way as to maximize or minimize a cost function specified in terms of distances between points and/or from points to centroids. The following are examples of frequently-used features:

In “Minimum K-clustering” the longest distance between any two points in the cluster is used as the cost function. The K-cluster sizes are ordered so that the biggest one has the shortest feasible diameter. The objective is to maintain relatively compact clusters. “K-Clustering Sum” is similar to “Minimum K-clustering” but uses the average distance between each pair of points in a cluster rather than the diameter. “K-Center” refers to the process in which, for each cluster,  $j$  one determines a reference point  $R_j$ , also known as the centroid, and then computes the maximum of the distances  $D_j$  that separate each cluster point from the centroid. The clusters and centroids are selected in such a way as to minimize the maximum value of  $D_j$ , which is done in a self-consistent manner. The method “K-Median” is synonymous with “K-Center”, with the exception that the maximum distance from the centroid is substituted by the average distance.

In the research that has been done, “K-means clustering” has emerged as the most common partitional approach [21]. The total intra-cluster distance, often known as the total squared error function, is the cost function as in Equation 3.2 [22] in this scenario.

$$\sum_{j=1}^k \sum_{y_j \in S_j} (y_j - c_j)^2 \quad (3.2)$$

where for  $j$ -th cluster  $y_j$  represents the position of data points,  $c_j$  denotes the centroid of that cluster, and  $S_j$  denotes the subset of points that make up that cluster.

### 3.4 Common Techniques for Node Classification on Graphs

Deep Learning based techniques for node classification may roughly be grouped into two classes: node classification based on (I) community membership [8], [23] and (II) role of the nodes [24]. When looking at networks, it is commonly seen that nodes belonging to the same communities are more likely to be connected. In contrast, nodes with similar responsibilities



in the network are generally found to be far from one another. Significant work has been done in both community and role identifications, and various techniques have been established to determine whether or not a node is a member of a certain community or plays a particular function in a given classification. The classification of nodes based on their communities separates this network into several cohesive groups. In contrast, the classification of nodes based on their roles reveals that nodes performing functions analogous to one another in the network structure are paired together. We discuss different methods for node classification under these two subcategories.

### 3.4.1 Community-Based Node Classification

Recent research trends may be identified, as well as the expansion, relevance, and inter-connectedness of research respondents, by applying community-detection practices and principles to citation networks. Brain networks [25] expose anatomical and functional isolation of brain areas utilizing this method [26]. This method leverages different community detection techniques.

**Graph Convolution Network-Based Strategy:** Graph Convolution Networks [2] (GCNs) are a prevalent form of graphical neural network technique [94] for learning node representations, and their influential findings in unsupervised and supervised node classification methods have inspired interest in the field among many academics. Multiple methods have been developed to use its potential to extract high-dimensional data from complicated networks for community discovery.

**Auto-Encoder-Based Strategy:** Auto-encoders are able to comprehend a novel network representation through unsupervised means by combining the decoder and encoder modules. They have a symmetrical architecture with various hidden levels where the output of one layer is fed into the next. Their primary focus is on decreasing the variation in reconstruction and reorganization errors.

**Random Walk-Based Strategy:** Word2Vector [27] was constructed to efficiently represent words as dense, compact vectors. Word2vector's core pretense is to reconstruct a neighborhood word vector based on the frequency with which nearby words appear together. Several random

walk-based embedding algorithms for different networks take inspiration from Word2vector by treating each network node as a word and the randomly generated path across them as a sentence. Word2Vector’s model was then replicated by other network embedding technological innovations such as DeepWalk [28] and Node2Vec[8].

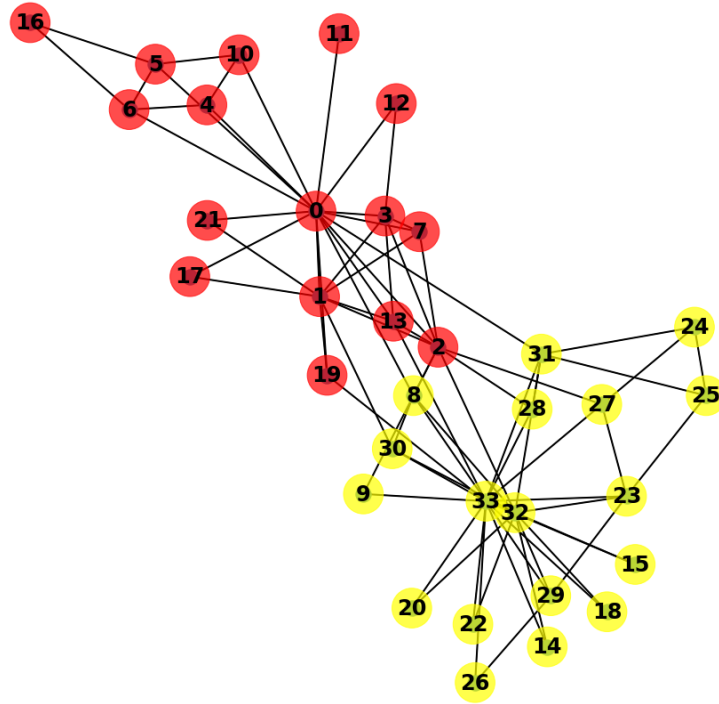
### 3.4.2 Role-Based Node Classification

The community structure is used as the basis for most of the known methods of the node classification scheme, which are in turn, based on the nodes’ bonding to one another. As a corollary, they are unable to grasp the structural knowledge that is conditional on roles [29]. Some of the difficulties associated with researching role-oriented node classification involve establishing clear and appropriate job descriptions for complicated social networks in the real world is a challenging task. Understanding the distribution of nodes with similar jobs and the patterns of interaction between nodes with various responsibilities in a wide social network is an extremely difficult and time-consuming endeavor. Moreover, it’s hard to define an efficient loss function since two differentiated nodes with collaborative effort could be geographically far apart in the network, depending on their responsibilities.

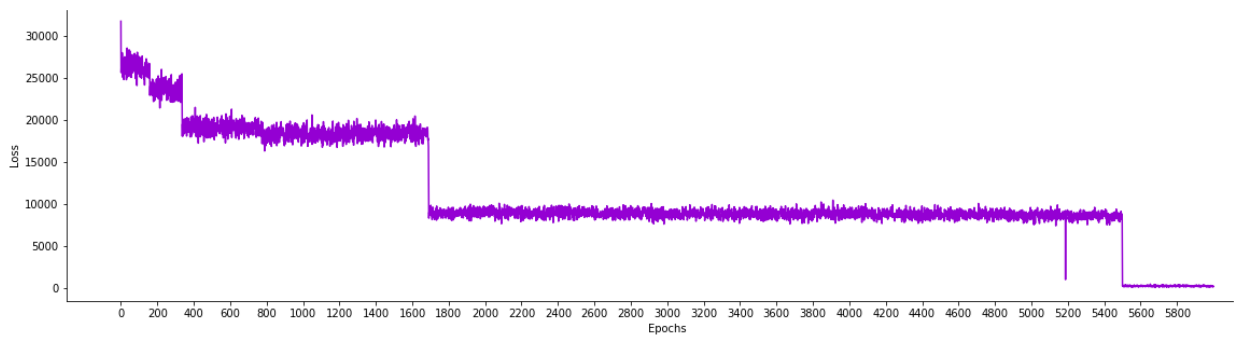
## 3.5 A Simple Network for Graph Analysis

In the field of community detection, a simple but widely used data set is Zachary’s Karate club network, where Wayne W. Zachary investigated the social structure of a karate school [30]. We will use this simple graph to easily explain the step-by-step graph analysis process for solving classification and clustering problems.

Thirty-four people from a karate club are represented in the network, with ties between them showing how they’ve engaged with one another. Two Club leaders had a disagreement that ultimately resulted in the group’s division. Using their interaction information, Zachary correctly predicted which factions the club members would join and which leader after the split. The correct two groups are shown in Figure 3.2 based on the found ground truth community labels.

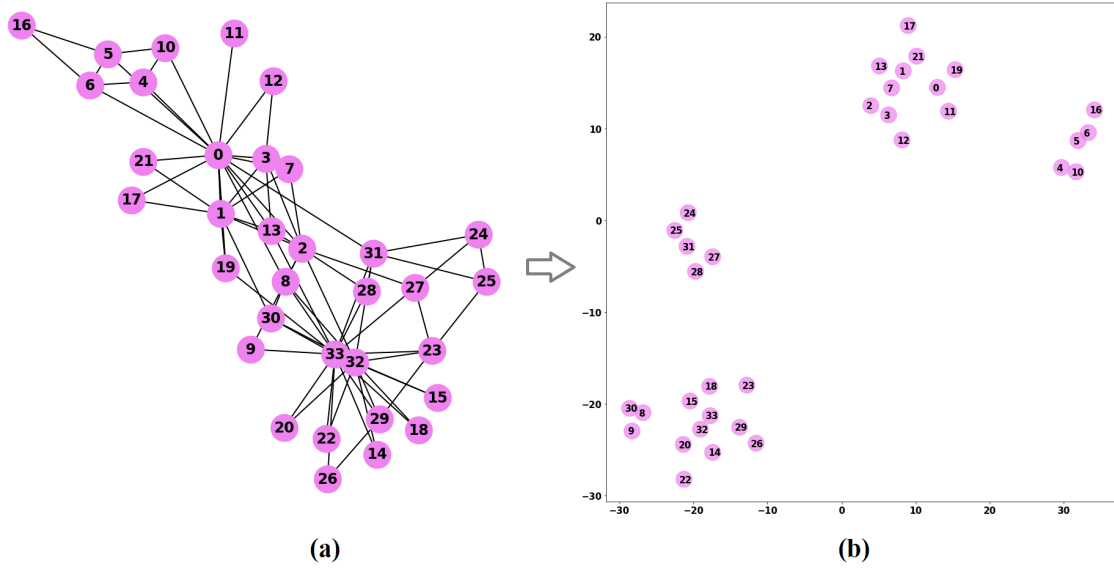


**Figure 3.2.** A Graphical Depiction of the Members of Zachary's Karate Club Network with Two Ground Truth Communities in Different Colors



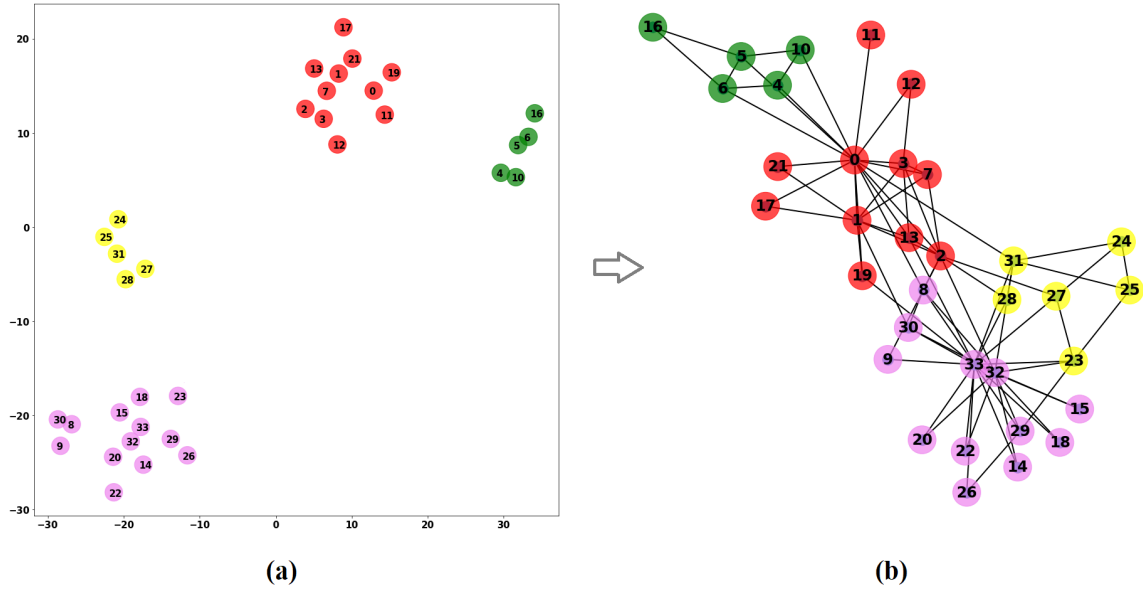
**Figure 3.3.** The Loss Curve for Zachary's Karate Club Network Converges After 5500 Epochs

Now, for node classification, we need to have ground truth labels of the training data. Hence if some of the labels of the nodes of this karate club's network are known, we can use those to train the whole graph and predict labels of the unknown nodes. In the case of node clustering, we try to find communities in the networks, and we are not given labels. In such cases, we apply clustering algorithms to find groups from the network.



**Figure 3.4.** (a) Zachary's Karate Club Network, (b) Corresponding Node2Vec Embedding Representation

Figure 3.4 shows the Node2Vec embedding result while applied on the karate club's network. We observe that the network converges after almost 5500 epochs of node2vec training as in Figure 3.3. From the t-sne visualization, it is clear that even if the ground truth had two class labels, there could be more groups in the networks. So depending on applied clustering methods, other communities can be extracted as shown in Figure 3.5 where we get four groups in the network presented in different colors.



**Figure 3.5.** (a) Clustering Algorithms Executed on Node2Vec Embedding, (b) Clustering Produces Four Communities in Different Colors

Thus the idea is to first encode the graph's nodes into a feature space so that different classification and clustering techniques can be applied to the graph problem. Sometimes the node embedding part is explicitly done, whereas in many cases, it happens internally. Proper embedding should also count these properties if attributes and weights are associated with the nodes and edges.

## 4. BACKGROUND AND RELATED WORKS

### 4.1 Graph Clustering Related Works in Clinical Domain

In nursing science, symptom clusters are defined as two or more symptoms that are related, and that occur together[31]. Symptom understanding and discovery of symptom clusters of chronic diseases, such as cancer, are important for nursing science to enhance scientific knowledge, facilitate communication and improve patient outcomes. Because of the complexity of clinical note mining, the typical symptom research analyzes symptoms through patient surveys [32]–[34]. Limitations of the patient survey approach are that it can be costly and challenging to collect a large amount of data from a large population as well as these surveys might not capture the various symptoms experienced in real-time by the patients and may cause an additional burden for patients to complete. In the EHR clinical notes, healthcare clinicians document clinical diagnosis, medication, and symptoms during a clinical encounter. These symptoms are documented in real clinical scenarios. If these clinician-documented symptoms can be extracted and further analyzed for nursing and clinical research, it will greatly enhance the understanding of symptom prevalence in different diseases and patient populations. Clinicians are also interested in gaining insight into symptoms and clusters of symptoms to inform the development of symptom management strategies. The need to extract clinical signs and symptoms from patient records and further analyze them benefits many other diseases. With the advanced research in natural language process (NLP) and AI, a systemic review [35] found that a number of studies applied NLP techniques to clinical notes in the EHR for symptom analysis. However, most of the existing research on symptom cluster extraction are either using statistical methods, such as factor analysis [36][37][38], PCA [36][39][38], or traditional clustering methods, such as hierarchical clustering, after modeling the occurrence of the symptoms as binary vectors [40][41]. Some other research investigated the semantic clustering of the symptoms[42] while ignoring the symptom co-occurrence, which is the main definition of symptom clusters clinically. One research applied graph models to identify relevant symptoms to a given symptom - symptom expansion[43]. Our research is the first to investigate using graph neural networks to discover symptom clusters based on clinical notes.

#### 4.1.1 Symptoms and Symptom Clusters Extraction

The clinicians apply symptoms or symptom clusters to analyze the etiology of many diseases. However, identifying symptoms or symptom clusters from text or other data types is often challenging and requires interdisciplinary domain knowledge. Researchers have adopted qualitative and quantitative methods to detect symptoms or symptom clusters from various data sources. Kim et al. [44] described various statistical methods to identify and quantify symptom clusters. Canonical correlation, partial correlation and structural equation modeling were the statistical methods used to identify symptom clusters. Sondhi et al. [43] introduced SympGraph, which is a framework to expand a given set of symptoms to other related symptoms by analyzing the underlying graph structure built based on the co-occurrences of the symptoms. Ni et al. [45] built a symptom-disease bipartite network that can simultaneously identify and rank the disease and symptom clusters. They applied domain network clustering and cross-network cluster ranking. Linder et al. [46] used a symptom assessment application that adolescents can use to claim temporary relationships among symptoms themselves. Then, the researchers can find out groups of symptoms. Barsevick [47] described different ways of identifying symptom clusters, such as through expert opinion, group comparisons, evidence of shared variance, identification of subgroups, etc. Gu et al. [42] developed SymptomID to discover COVID-19 symptoms from news reports. The SymptomID used several transformer-based models, such as BERT [48], GPT [49], and XLNet [50], for symptom extraction. Afterward, DBSCAN is used as a clustering method to group the symptom expressions semantically to extract the main symptom expressions. Papachristou et al. [41] applied five different clustering algorithms, including K-modes, Birtch, Spectral, Agglomerative hierarchical clustering, and k-means, then compared to Latent Class Analysis method for symptom clustering using patient registry records. Neijenhuijs et al. [51] performed cluster analysis with HDBSCAN[52], an extension of the DBSCAN [53] clustering algorithm and showed that symptom clusters could make specific interventions among cancer survivors in their recent study. Chow et al. [38] conducted three statistical approaches: Exploratory factor analysis, principal component analysis (PCA), and hierarchical cluster

analysis to detect symptom clusters in non-metastatic breast cancer patients treated by radiation therapy (RT).

#### 4.1.2 Graph Models and Their Application to Clinical Data

Graph clustering is a way of grouping the nodes such that similar nodes stay together and nodes with dissimilar properties stay apart. Graph clustering has been extensively used for community detection problems [23]. In 2016, Grover et al. [8] published the Node2Vec method to learn the continuous feature representations for nodes in networks by mapping nodes to a low-dimensional space of features that maximizes the likelihood of preserving network neighborhoods of nodes. Since then, the Node2Vec model has been applied in different domains, including the clinical domain for node prediction, edge prediction, and other tasks. Shen et al. [54] leveraged Node2Vec to generate node embeddings for the Human Phenotype Ontology (HPO) to assist phenotypic similarity measurement. The results show that the application of the HPO embedding on link prediction achieved 0.81 ROAUC and 0.75 F-measure. Du et al. [55] built a linked graph based on the extracted clinical trial information for registered COVID-19 clinical trials. Each clinical trial is a node in the graph. Then, Node2Vec is applied to learn the embeddings of the clinical trials. The results show that the learned embeddings can assist the clinical trial search and visualization. Kim et al. [56] applied a graph model to understand complex and diverse mechanisms of the biological pathways. The graph was built by extracting genes as nodes and identifying the co-occurrence as edges. The Node2Vec was used to generate the embeddings of the genes. The results showed that the graph model could extract the relationships between genes and identify the gene-gene interactions involved in a type 2 diabetes pathway. Chen et al. [57] investigated the drug-target interactions (DTIs) by using the graph models to understand the mechanism of drug action. They built the molecular associations network and applied Node2Vec to generate the embeddings. Finally, they performed a random forest (RF) classifier to predict potential drug-target pairs. Their results showed that the generated embedding helped to obtain 87.37% accuracy. Lee et al. [58] investigated five embedding generation methods, including Node2Vec, to learn the medical concept embedding based on the Medical concepts



defined by the Observational Health Data Science and Informatics (OHDSI) Observational Medical Outcomes Partnership common data model (OMOP CDM). Their results showed that the embedding learned by Node2Vec performed better than other models when selected parameters were used. Oniani et al. [59] also used Node2Vec to analyze a COVID-19 data set. They built the graph based on the co-occurrences among chemicals, diseases, genes, and mutations by using a linked data set CORD-19-on-FHIR. The DBSCAN was used to identify 63 clusters using silhouette values, and five coronavirus infectious diseases were detected in their corresponding subgroups.

To the best of our knowledge, our innovative research is the first to build a symptom graph based on symptom co-occurrences and apply graph clustering to discover symptom clusters using narrative EHR clinical notes.

## 4.2 Node Classification Related Works on Dynamic Graphs

The conventional method for solving dynamic graph classification issues involves applying static graph classification methods to distinct time steps, merging the results for the final prediction, or merely applying node classification to the last time step. These existing solutions do not leverage the order of the information on the topology and features of graphs.

Recent research on dynamic graph classification has focused on either giving bigger weights to the most recent snapshots [60], [61], introducing a decay rate to the edges of a changing temporal graph [7], [62], or computing the weighted average of features with exponential decay rates to classify nodes [63]–[65]. When working with dynamic graphs, aggregated snapshots are frequently pooled and processed as a static graph representation in order to feed a classification model [65]–[69]. For semi-supervised classification, GCN adopts a layer-wise propagation rule based on a first-order approximation of spectral convolutions on the graph. Even with several layers and nonlinear activation functions, GCNs can inevitably fail [70]. GraphSAGE performs sampling based on the information aggregation of node neighbors in order to provide efficient embedding for unseen nodes. In general, the aggregator architecture consists of mean and pooling-based approaches. However, GraphSAGE-based techniques concentrate primarily on node attributes for node classification and cannot presently take

edge features into account [71]. Some methods [72] have utilized attention mechanisms to assign interpretable weights to node characteristics. Using masked self-attention layers, GAT [3] permits variable attention weights and relevance for distinct nodes in a neighborhood. However, contrary to what a number of scholars assert, historical knowledge is not worthy of consideration [73]. GC-LSTM [5] is a GCN-embedded LSTM model that exploits the temporal properties of all accessible snapshots of the dynamic network, in which linkages between nodes might form, vanish, or remain constant over time. It can learn spatial-temporal characteristics to generate accurate predictions. EGNCN [6] employs the graph convolutional networks (GCN) model to capture the dynamic of graph sequence by updating GCN parameters using RNN. RNNGCN [7] employs an RNN layer to capture the dynamics of graphs and a GCN layer to categorize nodes. This model applies exponential RNN decay weights to snapshots in order to retain the historical data of the node’s characteristics. TRNNGCN [7] is an improved variant of RNNGCN that uses a decay matrix rather than a decay rate. Both RNNGCN and TRNNGCN are created by employing RNNs to lower the input weights gradually over time using a learned decay rate parameter.

## 5. CLUSTERING ON SYMPTOM GRAPH

### 5.1 Node Clustering to Solve Problems in Healthcare

Before and after treatments, cancer and other chronic disease patients typically have varied symptoms. Physical, GI, psychological, cognitive (memory loss), or other problems may occur. This research presents SymptomGraph to detect symptom clusters in EHR narrative text (EHR). Utilizing a collection of natural language processing (NLP) and artificial intelligence (AI) techniques, SymptomGraph is designed to first extract the clinician-documented symptoms from clinical notes. Then, a semantic symptom expression clustering technique is employed to identify a collection of typical symptoms. Symptom co-occurrences are used to construct a graph of symptoms. Finally, a graph clustering method is created to identify clusters of symptoms. SymptomGraph may be used to analyze symptom survey data, notwithstanding its application to narrative clinical notes. We deployed SymptomGraph to a data collection of colorectal cancer patients with and without diabetes (Type 2) to identify clusters of patient symptoms one year after treatment. Our findings indicate that SymptomGraph may detect the usual post-chemotherapy symptom clusters of colorectal cancer patients. In addition, the results suggest that colorectal cancer patients with diabetes frequently exhibit more significant symptoms of peripheral neuropathy, that younger patients exhibit mental dysfunctions due to alcohol or tobacco usage, and that patients with advanced disease exhibit higher symptoms of memory loss. Our approach may be expanded to extract and evaluate symptom clusters of other chronic or acute disorders in addition to COVID-19.

### 5.2 Objective

Electronic Health Records (EHRs) have become a standard tool used in healthcare delivery. Although the original objective of EHRs was to store patient records, the ever-growing clinical data in the EHRs enables artificial intelligence (AI)-based clinical data mining and decision support applications. The heterogeneous EHR data are available in both structured and unstructured clinical notes. The diagnosis, medication, and lab tests are often recorded in the structured fields of the EHR, whereas the symptoms and clinical evidence are usually

written in the clinical notes. Indeed, much clinical information is written in the clinical notes, making it valuable to analyze to understand the clinical information and their associations to support clinical research and build clinical decision support systems.

Expanding upon the methodologies in the literature and our previous research, we examined the following three questions in this research: 1) How to model symptom correlations using graph models?; 2) How to discover the symptom clusters using graph clustering?; 3) How to analyze and evaluate the discovered symptom clusters? To the best of our knowledge, no previous work has applied unsupervised graph clustering on clinical notes to answer these questions.

This research introduces SymptomGraph, a symptom mining system that builds a symptom graph from clinical notes in order to identify symptom clusters and the clinical parameters associated with them. A SymptomGraph is a graph consisting of extracted symptoms as nodes and symptom co-occurrences as edges. In order to facilitate symptom grouping, the graph learning algorithm Node2Vec is taught to offer a generic representation of symptoms. Through a literature search in the clinical area, we compared the suggested SymptomGraph to clinical parameters using symptom clusters discovered by our method.

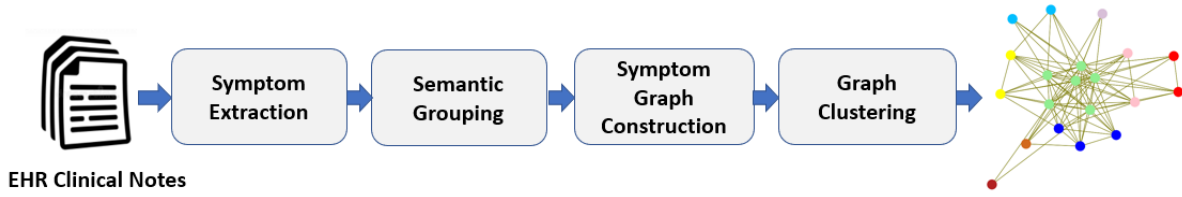
The main contribution of this study includes:

- Develop a graph model based on the co-occurrences of symptoms.
- Apply graph clustering to discover the symptom clusters.
- Evaluate the symptom clusters based on medical knowledge reflected by literature.
- Demonstrate that our model can be applied to narrative text to discover the symptom clusters of other chronic or acute diseases.

## 5.3 Methodology

### 5.3.1 Model Pipeline

Figure 5.1 depicts the four key components of the system used to detect clusters of symptoms in clinical notes available in an electronic health record system. The first component



**Figure 5.1.** System Framework of SymptomGraph

is developed to extract symptom expressions from the raw clinical notes. Since a symptom may have several descriptive names, we developed a component to group the symptom expressions based on their semantic meanings.

Then, each group is represented by a typical symptom of the group. Based on the identified typical symptom, the co-occurrences of the symptoms are used to build a symptom graph. Each symptom is a node on the graph, whereas an edge between two nodes represents the co-occurrences of the two symptoms within a clinical note. After constructing the symptom graph, the state-of-the-art feature learning method for network analysis, Node2Vec, is applied to generate symptom embedding for symptom representation. The clustering algorithms are utilized to generate symptom clusters by working on the symptom embeddings.

### 5.3.2 Symptom Extraction

UMLS MetaMap [74] has been utilized to extract the symptoms from the narrative clinical notes. UMLS MetaMap is an NLP tool that can map phrases in the text to different semantic types using various language sources. The UMLS MetaMap can map phrases in a text into 127 semantic types. This research used two semantic types - Sign or Symptom and Mental and Behavioral Dysfunction to identify symptoms from clinical notes. Figure 5.2 provides an example of clinical notes with highlighted terms mapped into these two semantic types using UMLS MetaMap. UMLS MetaMap includes functions to detect negation and Word Sense Disambiguation (WSD). Therefore, it can exclude most of the negating symptoms. However, because some parts of the clinical notes were not written as sentences,

The patient experienced months of **constipation** and felt **depression**.  
Sign or Symptom                      Mental and Behavioral Dysfunction

**Figure 5.2.** Symptom Phrases Detection Using UMLS MetaMap

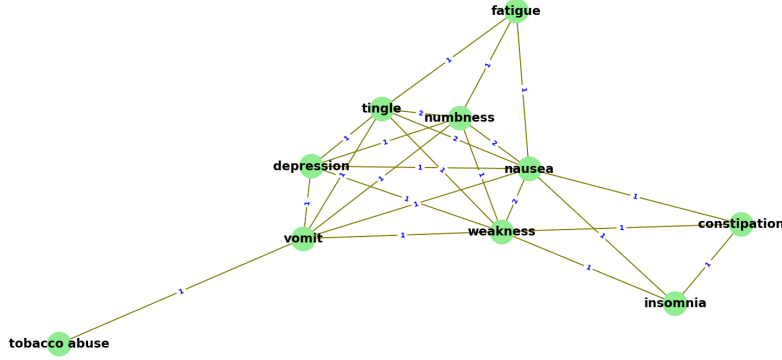
some negation situations cannot be fully detected by UMLS MetaMap. We developed a pattern-based approach to remove those negation cases that UMLS MetaMap cannot detect.

### 5.3.3 Semantic Grouping

After identifying symptoms and extracting from the clinical notes using UMLS MetaMap, we found that many symptoms have similar semantic meanings but are written in different expressions. For example, ‘fatigue’, ‘exhaustion’, ‘general fatigue’, ‘generalized fatigue’, ‘lethargic’, ‘lethargy and ‘tiredness indicate the same symptom - ‘fatigue’ of the patient. These symptoms need to be grouped together to be differentiated from other typical symptoms. To group the extracted symptom expressions with similar semantic meanings, we used universal sentence encoder [75] first to convert symptom expressions that include one or more words into embedding representations. Previous research [76] demonstrated that a universal sentence encoder performs well in identifying similar semantic expressions in clinical text narratives. We then used a hierarchical clustering algorithm to generate clusters of the symptom expressions to group them into different symptoms, such as ‘anxious’, ‘numbness’, ‘fatigue’ etc. The most frequent symptom expression in each cluster represents the group. Through this process, we reduced the various symptom expressions to a set of typical symptoms.

### 5.3.4 Symptom Graph Construction

This research aims to identify the symptoms co-related to each other during the progression of diseases or after treatments. Instead of applying typical principal component analysis or factor analysis, we investigated a graph model to understand the relations between symptoms. We constructed a symptom graph based on the symptom expressions extracted from each clinical note. Any extracted symptom expressions from the clinical notes are converted to the



**Figure 5.3.** Graph Built Based on Clinical Notes in Table 5.1

corresponding typical symptom expression by referring to the symptom semantic grouping. For example, if ‘lethargic’ is described in a clinical note, ‘fatigue’ is used to represent it. It is worth noting that we only consider it once if ‘fatigue’ is mentioned multiple times in a clinical note. The reason is that we want to focus on the co-occurrences of different symptoms.

To construct a symptom graph,  $G = (V, E)$  be an undirected graph,  $V$  is the set of vertices representing the typical symptom expressions, and  $E$  is the set of edges representing the symptoms that occur together in a clinical note. After processing all clinical notes in our data set, the symptom graph was built. Indeed, we generated a weighted graph  $G$ . The weight on each edge indicates the number of co-occurrences between the symptoms. In order to limit the impact of the rare co-occurrences, we removed the edges with a weight less than a predefined threshold  $\theta$ , which means if two symptoms occur simultaneously less than  $\theta$  times in the clinical notes, the co-occurrence is not considered in this research. Table 5.1 shows snippets of five clinical notes with the extracted symptoms. Figure 5.3 shows the symptom graph built based on these clinical notes.

**Table 5.1.** Snippets of Clinical Notes and Extracted Symptoms

Clinical Report Snippets	Symptoms
<i>.... The patient has been having a lot of <b>tingling</b>, <b>numbness</b> with discomfort in hands and feet after each chemotherapy dose.....Review was positive for persistent <b>fatigue</b> .....Recently he has had a feeling of continuous <b>nausea</b>...</i>	fatigue, nausea, numbness, tingle
<i>... her legs are just <b>weak</b>.... She does have some <b>numbness</b> and <b>tingling</b> of her fingers...<b>Depression</b>...OXY-CODONE causes <b>nausea</b> and <b>vomiting</b>.</i>	depression, nausea, numbness, tingle, vomit, weakness
<i>...Difficulty walking and <b>weakness</b>/diffuse atrophy...and <b>constipation</b>...as needed for <b>nausea</b>...as needed for <b>sleeplessness</b>...</i>	constipation, nausea, insomnia, weakness
<i>...He had complaints of <b>vomiting</b> after he had eaten a breakfast about 3 hours ago....<b>Tobacco abuse</b>...</i>	tobacco abuse, vomit

## 5.4 Symptom Graph Clustering

### 5.4.1 Symptom Embedding Generation

Graph Neural Networks (GNNs) [77] have been applied to many graph-based applications for node classification or link prediction in the clinical domain [78], [79]. However, unsupervised GNNs have not been extensively explored for clinical note analysis in the clinical domains, although unsupervised graph clustering can be more resistant to advances in GNNs. We used weighted Node2Vec for symptom embedding generation.

### 5.4.2 Symptom Clustering

Node2Vec is used to project the nodes of the symptoms into the vector space. After that, different clustering algorithms are applied, including K-Means, Hierarchical Agglomerative, Gaussian Mixture Models, and DBScan.



K-Means: In the K-Means clustering algorithm, data points are assigned to exactly one cluster of the predefined  $K$  number of clusters. For these  $K$  clusters, there are  $K$  centroids, and those centroids are updated in such a way that the sum of the squared distance between the data points in a cluster and the corresponding centroid of that cluster is minimum.

Hierarchical Clustering (HC): The HC algorithm hierarchically generates bigger clusters from single data points using the agglomerative approach. Two data points are merged into a single cluster in each step depending on some distance criterion. The type of distance being considered gives rise of different popular hierarchical clustering methods such as the ward, single, average, and weighted methods.

Gaussian Mixture Models (GMMs): Gaussian Mixture Models (GMMs) presumes a definite number of Gaussian distributions, and each of these distributions symbolizes a cluster. GMM prefers to group the data points pertaining to a single distribution together. These are probabilistic models which employ a soft clustering approach for distributing the points in various clusters. For a dataset of  $K$  clusters, there should be a mixture of  $K$  Gaussian Distributions, each having a specific mean and variance. Expectation-Maximization (EM) techniques are used to determine these values.

DBSCAN: DBSCAN refers to Density-Based Spatial Clustering of Applications with Noise (DBSCAN) which organize points that are close to each other depending on a distance measurement and a minimum number of points. It has the ability to detect outliers among the points that are in low-density regions. For a fixed minimum number of points, changing the distance values we generate different numbers of clusters.

To evaluate the clustering methods and identify the optimum number of clusters for the symptom graph, the Dunn Index ( $DI$ ) is calculated for the different numbers of clusters.  $DI$  is the ratio of the smallest inter-cluster distance to the largest intra-cluster distance. Given  $k$  clusters, the Dunn Index is calculated as:

$$DI_k = \frac{\min_{1 \leq i \leq j \leq k} \delta(S_i, S_j)}{\max_{1 \leq n \leq k} \Delta_n} \quad (5.1)$$

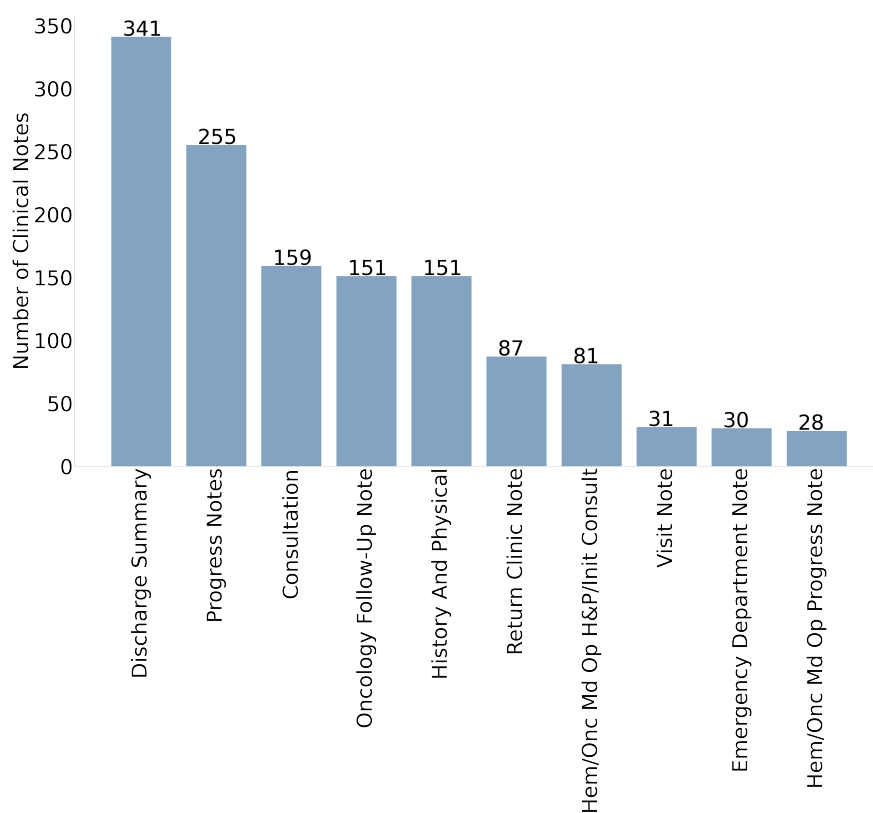
Where the largest intra-cluster distance  $\delta(S_i, S_j)$  of cluster  $S_i$  and  $S_j$  and the smallest inter-cluster distance  $\Delta_n$  consisting of  $n$  elements can be calculated using any of the single,

complete, average or centroid linkage distance functions. A well clustering distribution maintains that the clusters are furthest away from each other and elements in each cluster are closer together. Hence higher  $DI$  is expected for a satisfactory clustering result.

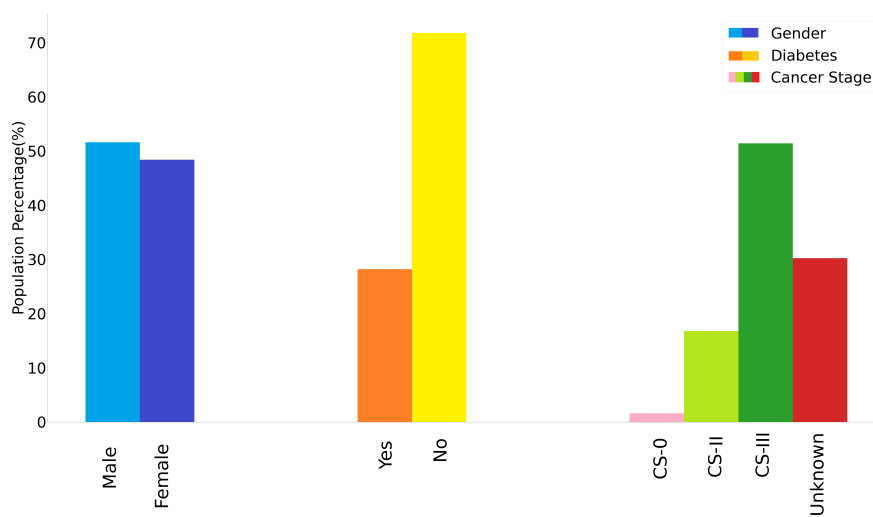
## 5.5 Experimental Results

### 5.5.1 Data Set

The patient cohort in this research consists of patients with a primary diagnosis of colorectal cancer with and without diabetes who have records in a large academic medical center’s EHR system. The colorectal cancer patients were identified using the International Classification of Diseases (ICD). The ICD codes are 153-154 (ICD-9) and C18-C20 (ICD-10). This research focused on investigating the symptom clusters one year after the patients received the first chemotherapy. Through these ICD codes, we identified colorectal cancer patients who received chemotherapy within the ten years of 2007-2017. Among these patients, 996 patients have clinical notes within one year after the chemotherapy treatment. For each patient, we extracted clinical notes to generate a symptom graph for symptom clustering. There are 1675 clinical notes, and over 119 types of clinical notes were included. Figure 5.4 shows the top 10 major types of clinical notes. Discharge Summary and Progress Notes are the major types of clinical notes. Figure 5.5 shows the gender distribution, percentage of patients with diabetes, and percentage distribution of the patients based on the stage of cancer. There is slightly more male in our data set. About 28.2% of patients have diabetes, and 1.6%, 16.8%, 51.4% of patients are in cancer stage 0, II, III, respectively. The pathologic stage of the cancer is unknown for 30.2% of patients.



**Figure 5.4.** Top 10 Frequent Types of Clinical Notes

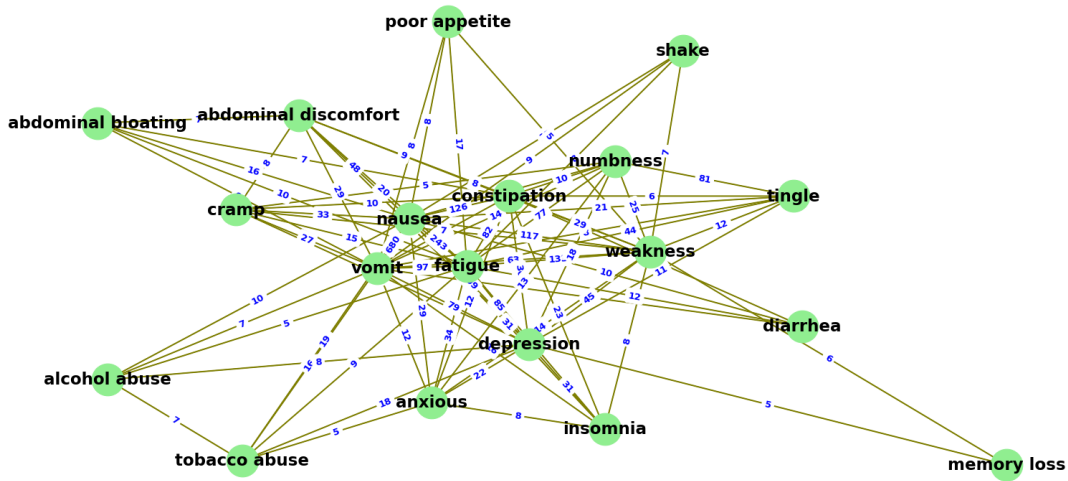


**Figure 5.5.** Distribution of the Clinical Parameters

Other than the clinical notes, we included gender, diabetes status, the pathologic stage of cancer, and age at diagnosis as clinical parameters to understand the relationships between those data points and symptom clusters.

### 5.5.2 Extracted Symptoms and Construction of Symptom Graph

After symptom extraction, 106 unique positive symptom expressions were extracted from the data set. After applying the semantic grouping, we grouped them into 19 typical symptoms. Table 5.2 shows the typical symptoms and the grouped symptom expression along with their document frequency (DF - the number of clinical notes that have these symptom expressions). The symptom extraction results show that ‘nausea’, ‘vomit’, and ‘fatigue’ are typical symptoms reported in more than 700 clinical notes within one year after the chemotherapy. The least reported symptom within the same timeframe was ‘memory loss’.

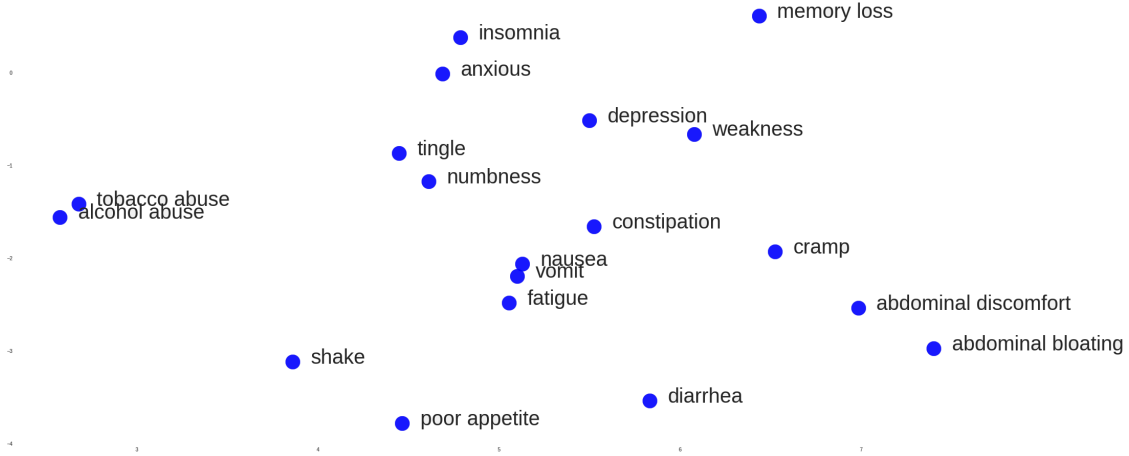


**Figure 5.6.** Symptom Graph Built Using Our Data Set

Based on the extracted symptoms and their co-occurrences within the clinical report and the frequency of the co-occurrences, we constructed the symptom graph  $G$ , shown as Figure 5.6. The weights on the edges represent the frequency of the co-occurrences in the clinical

**Table 5.2.** Extracted Symptoms from Clinical Notes

Symptoms	DF	Symptom Expressions
abdominal discomfort	86	abdominal discomfort, abdominal, abdominal cramping, abdominal tenderness, abdominal symptom, abdominal fullness, abdominal cramp
abdominal bloating	39	abdominal bloating, bloating, bloating symptom
insomnia	88	insomnia, sleeplessness
vomit	751	vomit, persistent vomit, intractable vomit
shake	35	shake, shake chill
anxious	90	anxious, nervousness, nervous, panic attack, anxious behavior
weakness	349	weakness, weak, generalize weakness, feel weakness, general weakness, muscle weakness, generalize muscle weakness
memory loss	15	memory loss, memory problem, forgetful, forgetfulness
poor appetite	29	poor appetite, diminish appetite, reduce appetite
fatigue	731	fatigue, burning, lethargic, lethargy, tiredness, exhaust, exhaustion, combat fatigue, extreme fatigue, general fatigue, generalize fatigue, tire
alcohol abuse	32	alcohol abuse, acute alcoholic intoxication, alcoholism, etoh abuse
tobacco abuse	52	tobacco abuse, abuse, marijuana abuse, substance abuse, drug abuse, medication abuse
constipation	265	constipation, chronic constipation
depression	320	depression, major depressive disorder, depression and anxiety, anxiety and depression, chronic depression, depressive disorder, depression symptom, minimal depression, depressive symptom, depress mood, mental depression, reactive depression, single episode of major depressive disorder, depressed mixed anxiety and depressive disorder
numbness	198	numbness, leg numbness, numbness in finger, hand numbness, numbness in foot, numbness of finger, low extremity numbness, numbness in toe, numbness in leg, numbness of hand, numbness of toe
tingle	112	tingle, have tingle sensation, tingle in finger, tingle finger
cramp	74	cramp, crampy, cramp sensation quality, muscle cramp, acute pain, leg cramp
diarrhea	26	diarrhea, vomit diarrhoea, diarrhea and vomiting, severe diarrhea
nausea	1121	nausea, nausea vomiting, chronic nausea, nausea and vomiting, postoperative nausea, symptom nausea



**Figure 5.7.** t-SNE Visualization of Symptom Distribution

notes. In this research, we set the  $\theta$  introduced in 5.3.4 to be 5, which means we only included the edges with a weight of 5 or more. The symptom graph shows that some symptoms often co-occur together. For example, ‘nausea’ and ‘vomit’ co-occur 680 times, ‘nausea’ and ‘depression’ co-occur 109 times, etc. This symptom graph provides direct visualization of the co-occurrences of the symptoms.

### 5.5.3 Symptom Graph Clustering Results

To generate the symptom clusters, we first applied the weighted Node2Vec algorithm described in Section 2.3.1 to the graph to generate symptom embeddings. The hyperparameters of the weighted Node2Vec are optimized (shown in Table 5.3). In this research, we set the size of the embeddings to 512.

After the symptom embeddings are generated, we applied t-Distributed Stochastic Neighbor Embedding (t-SNE) [80] - an unsupervised, non-linear technique for high dimensional data visualizing, to visualize the symptom distribution. Figure 5.7 shows the t-SNE imaging on the two-dimensional space. From the visualization, it can be noted that some symptoms are closer to each other than others. For example, ‘tobacco abuse’ and ‘alcohol abuse’ are

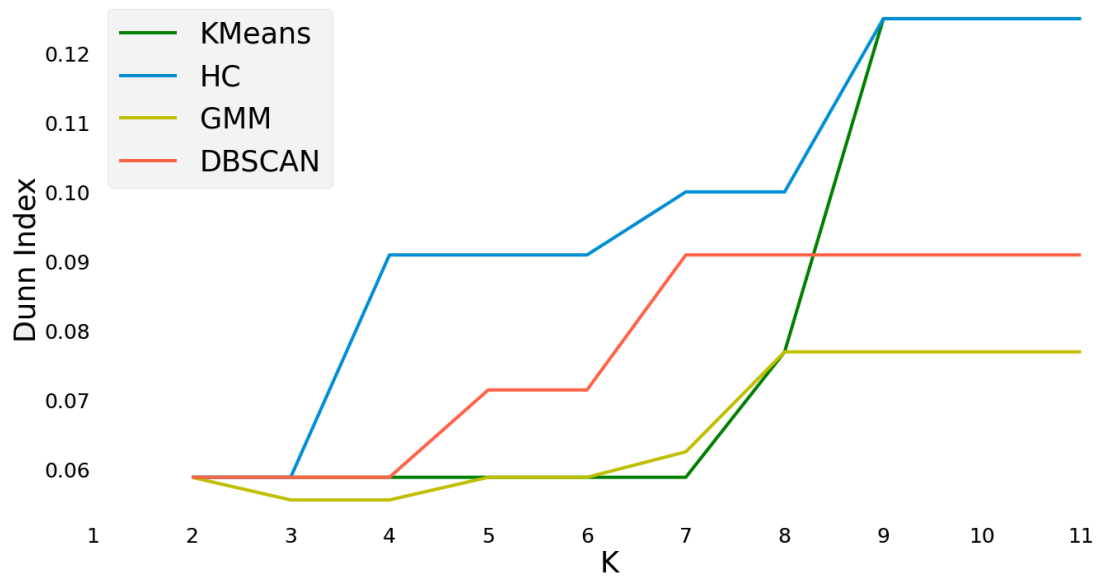
**Table 5.3.** Hyperparameters of Weighted Node2Vec

Parameter	Value
Maximum length of a random walk	100
Number of random walks per root node	10
Probability of returning to source node ( $1/p$ )	2
Probability for moving away from source node ( $1/q$ )	0.5
Random seed	42
Size of embedding	512
Size of window	5
Number of epochs	11000

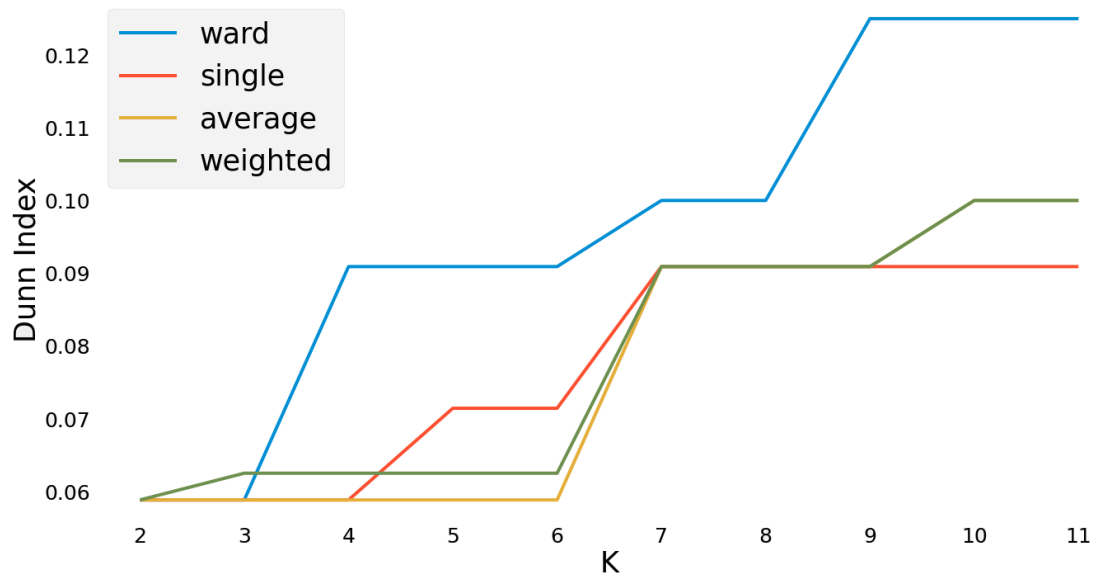
more closely located than the other symptoms, whereas ‘memory loss’ is relatively far from the rest of the symptoms.

We calculated the Dunn Index value of all different clustering algorithms with different  $k$  values. As shown in Figure 5.8, HC gains the highest  $DI$  than the rest of the methods. To further investigate, we applied different HC methods (Figure 5.9) based on the distance calculation and found that ‘ward’ is better than ‘single’, ‘average’ and ‘weighted’ distance calculations. Hence, we determined the optimum number of symptom clusters is 9. The 9 clusters are found from the intersection of the dendrogram of the Ward HC method and linkage threshold line at 0.85 shown in Figure 5.10.

Figure 5.11 shows the clustered symptom graph with symptoms within the same cluster colored the same. The t-SNE visualization also reflects these cluster distributions. There are four clusters with only one symptom in each: ‘memory loss’, ‘shake’, ‘diarrhea’, and ‘poor appetite’. Based on the t-SNE visualization, these single symptom clusters are not close to each other and are far from other symptoms, which infers that these symptoms have less correlation with other symptoms based on our data set. These single symptoms reflect either individual gastrointestinal problems or mental issues. There are two big clusters with four symptoms in each. One of them has ‘anxious’, ‘insomnia’, ‘weakness’, and ‘depression’ that are mainly mood problems after the cancer patients start the chemotherapy [81]–[83].

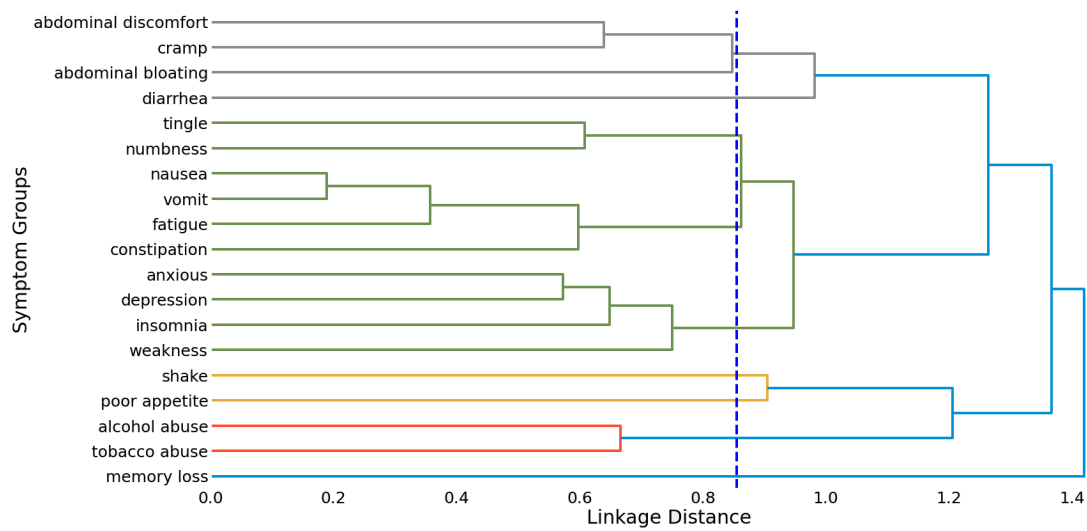


**Figure 5.8.** Dunn Index of Different Clustering Models

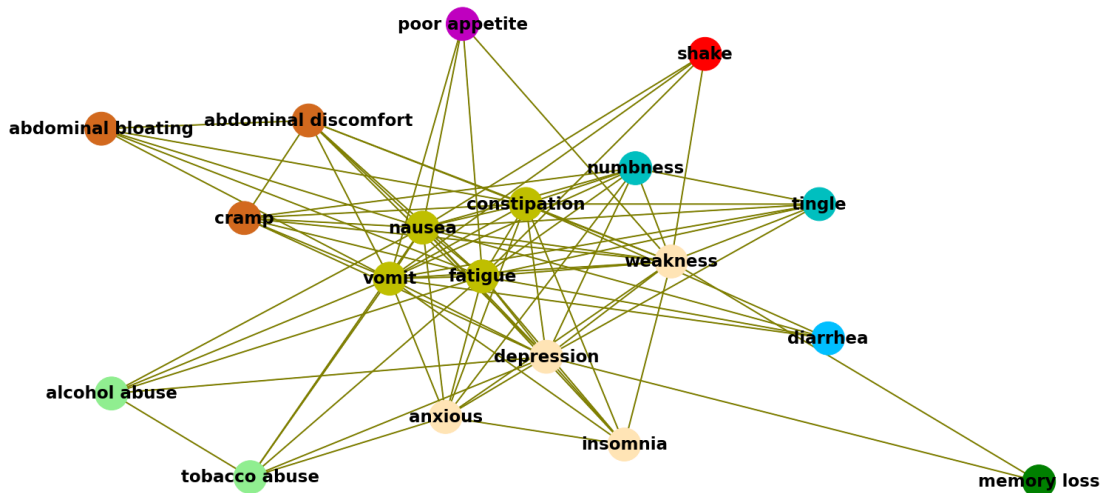


**Figure 5.9.** Dunn Index of Different HC Methods





**Figure 5.10.** Dendrogram of HC Ward Linkage Method with Threshold at 0.85 Indicated by Dashed Vertical Line



**Figure 5.11.** Symptom Clustering Result

The other big cluster includes ‘nausea’, ‘fatigue’, ‘vomit’, and ‘constipation’ which are the typical gastrointestinal distress and physical function symptoms after the colorectal cancer patient receives the chemotherapy [84], [85]. Three clusters include two or three symptoms. ‘Tobacco abuse’ and ‘Alcohol abuse’ are in the same cluster and found to be associated with other symptom expressions, such as significantly higher pain [86]. The research also showed that alcohol abuse is highly prevalent in patients with advanced cancer, and a subgroup of cancer patients are more likely to have a history of chemotherapy or are actively smoking [86]. ‘Tingling’ and ‘Numbness’ are two symptoms linked to the neuropathic side effects commonly reported in patients receiving chemotherapy [87]. Based on the literature [88], tingling or numbness in the fingers and hands and/or toes and feet were the frequently reported peripheral neuropathy symptoms. ‘Abdominal discomfort,’ ‘Abdominal bloating,’ and ‘cramp’ are grouped in one cluster. These are gastrointestinal symptoms often shown in colorectal cancer patients [89].






#### 5.5.4 Statistical Analysis and Evaluation of the Symptom Clusters

To further evaluate the symptom clusters discovered in our data set, we applied statistical analysis to the symptoms in each one of the clusters to identify the associated clinical parameters. Table 5.4 shows the number of patients with at least one symptom in each symptom cluster along with the mean and standard deviation of age, gender distribution, diabetes patient percentage, and patients in each cancer stage. The colored circles in Table 5.4 match the symptom nodes in Figure 5.11. The t-test was used to determine the patient age differences between patients of any two symptoms clusters. Pearson’s chi-squared test was used to determine whether there was a statistically significant difference between the two clusters regarding gender, diabetes status, and cancer stage. The patients in both compared clusters are excluded when calculating the t-test and Pearson’s chi-squared test. It means there is no overlap between compared two clusters.

The median age of patients in cluster C0 is 57.48. The t-test analysis shows that the patients who have symptoms of ‘tobacco abuse’ or ‘alcohol abuse’ are statistically younger than those having symptoms in either C1 ( $p=0.04$ ) or C3 ( $p=0.03$ ). Patients with either ‘tobacco

abuse’ or ‘alcohol abuse’ are younger than patients who have ‘depression’ or ‘poor appetite’. The literature also demonstrates patients with advanced cancer who have alcoholism, tobacco, or illegal abuse are relatively younger [86], which is consistent with our findings. The t-test results also show that patients in cluster C1 are statistically older (mean aged 63.57) than those in C6 ( $p<0.01$ ). Research indicates loss of appetite is especially serious in elderly patients who are critically ill, such as cancer patients [90], taste and smell changes occur with advancing age, which can lead to poor appetite [91]. Table 5.4 shows that relatively more patients with diabetes experience symptoms in C5, C7, and C8. The chi-squared test results show that more patients with diabetes experienced symptoms ‘numbness’ or ‘tingling’ in fingers, hands, or lower extremities than symptoms in C1 ( $p<0.02$ ), C4 ( $p<0.03$ ), and C6 ( $p<0.01$ ). Although our study data set includes 996 CRC patients, it extends similar results found in the recent research [92], [93], specifically, that CRC patients with diabetes experienced milder to severe neuropathic symptoms, such as tingling or numbness in fingers or hands. From the cancer stage prospect, the percentage of patients in cancer stage III is high in clusters C5, C7, and C8. The statistical analysis results show that, the number of patients in later cancer stage in C5 is significant than those in C0 ( $p<0.01$ ), C1 ( $p<0.01$ ), C2 ( $p<0.01$ ), C3 ( $p<0.01$ ), C4 ( $p<0.01$ ) and C6 ( $p<0.01$ ), which means later cancer stage patients experience neuropathic symptoms. A recent study on Korean cancer patients shows that stage IV cancer and a history of chemotherapy were identified as predictors of neuropathic cancer pain [94]; this is similar to our finding. However, our data set only included up to stage III cancers. The patients who experience ‘memory loss’ had mostly in stage III cancer; the statistical analysis shows that more stage III patients experience ‘memory loss’ than symptoms in C0 ( $p<0.01$ ), C1 ( $p<0.01$ ), C2 ( $p<0.01$ ), C3 ( $p<0.01$ ), C4 ( $p<0.01$ ) and C6 ( $p<0.01$ ) after the chemotherapy. Although we have not identified associated literature for this finding based on our data set, it could be a valuable research hypothesis for further exploration from the clinical side.

**Table 5.4.** Statistics of the Patients with Symptoms of Each Cluster

	C0 	C1 	C2 	C3 	C4 	C5 	C6 	C7 	C8 
	tobacco abuse alcohol abuse	poor appetite	shake	depression weakness anxious insomnia	vomit nausea fatigue constipation	numbness tingle	cramp abdominal discomfort abdominal bloating	diarrhea	memory loss
<b># of patients</b>	58	28	30	414	917	156	148	26	14
<b>age</b>									
mean	57.48	63.57	54.5	60.4	59.19	58.03	54.55	60.27	61.07
(std)	(9.09)	(14.28)	(15.48)	(12.44)	(12.57)	(10.97)	(12.58)	(14.84)	(16.75)
<b>Gender</b>									
Female (%)	60.34	50	66.67	47.34	51.36	53.21	51.35	57.69	42.86
Male (%)	39.66	50	33.33	52.66	48.64	46.79	48.65	42.31	57.14
<b>Diabetes (%)</b>	25.86	17.86	30	34.3	27.92	34.62	22.3	38.46	50
<b>Cancer Stage</b>									
Unknown (%)	53.45	35.71	40	34.06	29.88	19.87	31.76	26.92	14.29
0 (%)	0.0	0.0	0.0	0.72	1.64	1.28	2.03	0.0	0.0
II (%)	15.52	21.43	20	17.15	17.23	14.1	19.59	7.69	0.0
III (%)	31.03	42.86	40	48.07	51.25	64.74	46.62	65.38	85.71

## 5.6 Discussion

This innovative research developed an NLP+AI system first to identify symptom expressions from the EHR clinical notes. We then developed a graph network-based clustering approach to identify the symptom clusters based on symptom co-occurrences in each clinical note. This system is relatively easy to implement, and the graph visualization can assist clinicians in identifying symptom co-occurrences using clinical data. Compared to other research in symptom cluster generation, our study demonstrates how to generate symptom graphs based on the narrative text of the clinical notes in the EHR to identify the symptom clusters. Although this research analyzed and evaluated the SymptomGraph using a data set of colorectal cancer patients, the system can be applied to other patient cohorts to discover the symptom clusters experienced by patients before or after treatments. The symptom cluster findings can assist clinicians in discovering hidden symptom clusters, and developing and validating clinical hypotheses. Based on the symptom clusters and their associations with the clinical parameters, personalized treatments or symptom management strategies can be investigated by clinicians. The statistical analysis of the symptom clusters shows that the clinical parameters, such as comorbidities like diabetes, age, or cancer stage, can be used as predictors of symptom clusters after a treatment like chemotherapy. Our statistical analysis results in the colorectal cancer patients show that the SymptomGraph can identify similar findings to previous clinical research in the literature as well as new symptom clusters which need further clinical validation and investigation.

There are a few limitations to this research. Although word sense disambiguation (WSD) is enabled in UMLS MetaMap, due to the accuracy of the WSD of UMLS MetaMap, there could be some symptom expressions in the clinical notes that were left out. Also, signs and symptoms written with typos cannot be identified. The uncommon signs and symptoms that manifest in less than five clinical notes were not included due to the fact that this research is centered on the overall performance of the SymptomGraph on discovering the ordinary symptom clusters.

## 5.7 Summary

EHR clinical report is the main instrument for a clinician to document patient symptoms before and after treatment. Extracting and analyzing the documented symptoms of chronic or acute diseases is essential for improving patient care and developing personalized clinical decision systems. We developed a system for symptom cluster identification using both NLP and AI techniques. The system includes NLP components for symptom expression extraction, semantic grouping, and AI components to identify symptom clusters based on graph learning and clustering. The patients with the symptoms in a cluster can be further analyzed with other clinical parameters, such as ethnicity, comorbidities, medication, etc. We intend to construct a large graph containing symptoms, genotypes, and phenotypes in order to determine their correlations. We believe that it is possible to use this model to identify clusters of symptoms for other chronic diseases like Covid-19.

## 6. NODE CLASSIFICATION ON DYNAMIC GRAPHS

While many recent efforts in dynamic graph representation learning take snapshots of the graph into account, they often do so in isolation, missing out on the benefits that temporal information may provide. Techniques are created to address the issue by utilizing recurrent neural network-based solutions; nevertheless, these systems still face challenges due to the traditional limits of vanishing or exploding gradient difficulties and convoluted training methods. A unique graph neural architecture, G2B, is introduced to compute the node representations using attention weights aggregated over the neighborhood, allowing us to overcome these challenges. Based on the dynamic edges and node feature information, our technique imposes attention weights at various time steps to categorize nodes in a supervised manner. Based on results from an evaluation utilizing two reference datasets, the suggested G2B model is superior to seven state-of-the-art baseline models for node categorization in real-time networks. Furthermore, the trained model’s attention weights may be used to better capitalize on connections between nodes in the network.

### 6.1 Objective

The main contributions of this research include: (1) Introducing a new model - G2B for dynamic node classification; (2) Comparing G2B against seven different state-of-the-art models to show the superior of our model performance; and (3) Interpreting the dynamic edge importance using attention weights of GAT.

### 6.2 Methodology

#### 6.2.1 Problem Statement

Let  $D = (G_1, G_2, G_3, \dots, G_T)$  be a dynamic graph consisting of successions of snapshots. Here,  $G_t = (V_t, A_t, F_t, L_t)$  indicates the snapshot at timestamp  $t$  whereas,  $V_t$ ,  $A_t$ ,  $F_t$ , and  $L_t$  are set of vertices, adjacency matrix, feature matrix, and class label matrix respectively at time  $t$ . The vertices and edges are dynamic in the sense that they can either be present or absent at different timestamps. If there are attributes associated with each vertex,  $F_t$

serves as the extra information besides the structure, which can be very significant in many applications. In the absence of such information, the feature matrix can be replaced with an identity matrix. We consider the problem as a supervised training of the dynamic snapshots. Given the training set class label  $L_T^{train}$ , the objective is to properly classify the test class label  $L_T^{test}$  at time  $t$ .

### 6.2.2 G2B Architecture and Edge Importance Interpretation

After receiving the graph structure as input from a GraphSAGE Layer, including the node characteristics of each timestamp, G2B sends the output into a GAT layer. After all of the information corresponding to the timestamps have been fed into a BiLSTM layer, the node embeddings of all the timestamps are then given into the layer sequentially. The final updated node embeddings utilized for node classification may be found in this BiLSTM layer, which supplies those embeddings. The pipeline of G2B is shown in Figure 6.1.

G2B involves a two-layer GraphSAGE for each time stamp. In the first layer, information is collected about the nearest neighbors of each vertex, while in the second layer, information is used that also takes into account the neighbors from the next layer.

GraphSAGE is an isotropic representation learning technique where each neighbor contributes equally towards determining the update operation. G2B uses “mean” aggregation method in which it takes the average of the considering nodes. One purpose of using GraphSAGE is that it can handle large graphs efficiently. Although stacking multiple layers of GraphSAGE involves more computational cost, the two-layer GraphSAGE balances the trade-offs between speed and performance.

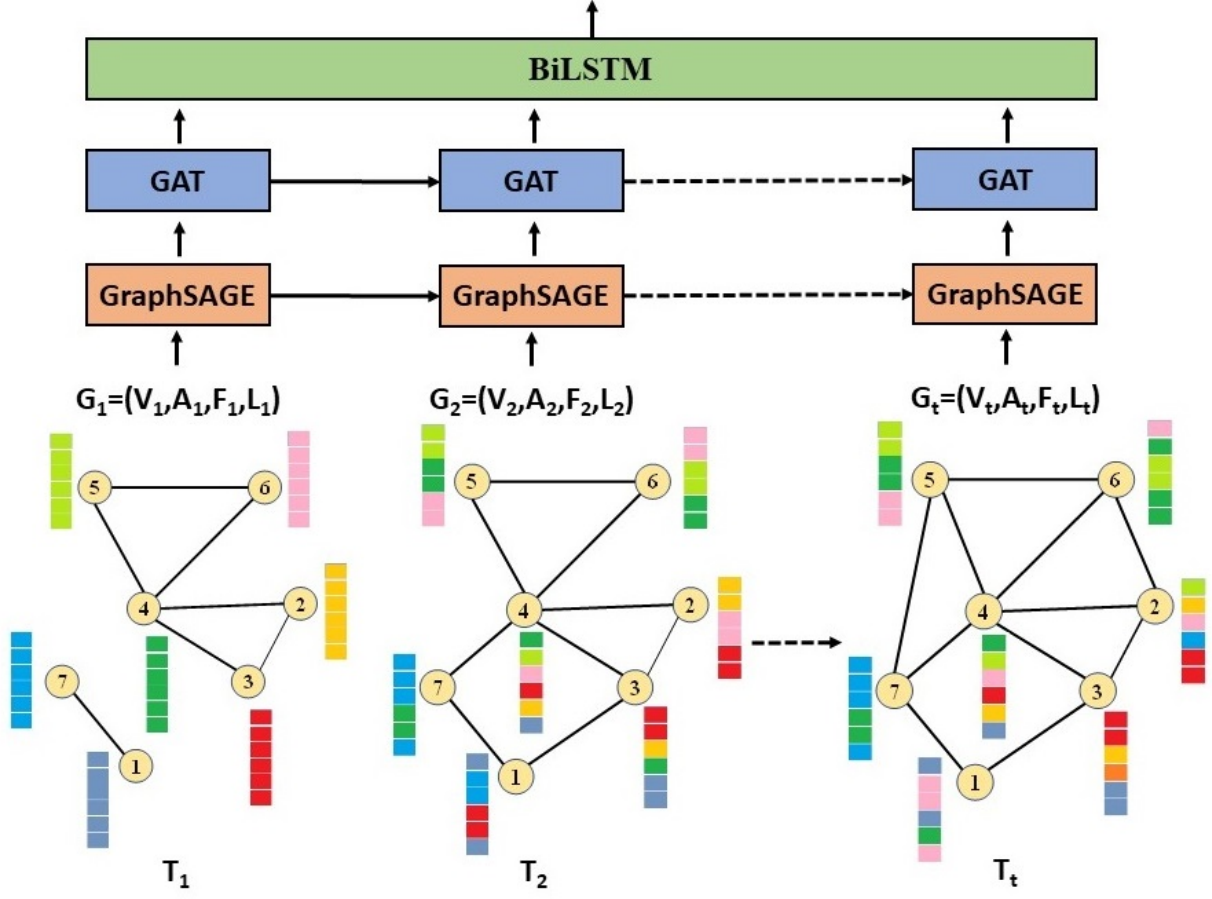
For the target node  $q$  with immediate neighborhood  $N_q$ , the Aggregated node representation  $a_q$  is defined as

$$a_q = F_{aggregate}(\frac{h_p}{p}, p \in N_q) \quad (6.1)$$

The updated node embedding representation of node  $q$  at  $i^{th}$  iteration,

$$h_q^i = F_{update}(a_q, h_q^{i-1}) \quad (6.2)$$





**Figure 6.1.** System Framework of G2B

$F_{aggregate}$  and  $F_{update}$  functions of Equation 6.1 and 6.2 are placeholder for any differentiable functions. After applying the final layer convolution operation of GraphSAGE to these neighbors, an attention mechanism is performed to the output by applying a GAT layer. The GAT layer substitutes the basic aggregation function of the GCN [2] layer and assign different importance to each edge through the attention coefficients calculated by Equation 6.3.

$$e_{pq} = c \left( W \vec{h}_p, W \vec{h}_q \right) \quad (6.3)$$

where  $W$  is the weight matrix and  $c$  is the attentional mechanism such that  $c = \mathbb{R}^N X \mathbb{R}^N \rightarrow R$

The aggregated information from the GraphSAGE layer is passed into a GAT layer, which assigns larger weights to the significant edges via attention coefficients, resulting in more accurate classification of nodes. After conducting GraphSAGE and GAT convolutions on all timestamps, the resultant node embeddings are passed to a BiLSTM layer that captures the long-term relationships between time steps of time series or ordered data. G2B exploits the superior performance of the BiLSTM mechanism and improves the model’s performance on sequence classification challenges.

The graph attention layer has been used to determine the contribution of different edges at different timestamps. GAT applies the attention mechanism as a replacement for the statistically normalized convolution operation. The equation to get the vertex embedding  $h_i^{(l+1)}$  of layer  $(l + 1)$  from the embeddings of layer  $l$ . The embeddings from neighbors are merged and multiplied by the attention scores as Equation 6.4.

$$h_i^{(l+1)} = \sigma \left( \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)} \right) \quad (6.4)$$

Where  $\alpha_{ij}$  is the attention coefficient between node  $i$  and  $j$ .  $\mathcal{N}(i)$  represents all the neighbors of node  $i$  and  $\sigma$  is the activation function.  $z_i$  is the product of the learnable weight matrix  $W$  and lower layer embedding  $h$  as  $z_i^{(l)} = W^{(l)} h_i^{(l)}$ .

The attention weight  $\alpha$  for each edge indicates the importance of that edge. After the model is trained, the edges are ranked according to the attention weights representing the impact. Then we chose a threshold  $\theta$  for each timestamp to select top-ranked edges for edge importance interpretation. Through this process, the most important edges are determined when they are ranked high consistently throughout the timestamps. This mechanism can also be used to observe the changes in edge importance over time.

## 6.3 Experimental Results

### 6.3.1 Datasets and Baselines

We conducted experiments using two benchmark datasets: DBLP3 and DBLP5, as shown in Table 6.1. Both datasets are collected from the computer science bibliography website

DBLP [7] consisting of a collection of papers from conferences and journals in different domains. Each author is a node in the graph, and co-authorships indicate the edges between nodes. We created two versions of DBLP3 and DBLP5 by splitting the datasets into training, validation, and test sets with ratios 70%, 20%, 10%, and 50%, 20%, and 30%, respectively.

To evaluate the performances, we chose the standard metrics: Area under the ROC curve (AUC), Accuracy (ACC), and F1. The code of the baseline methods and datasets are publicly available<sup>1</sup>.

**Table 6.1.** Summary of Datasets

Dataset	Nodes	Edges	Time Steps	Features	Classes
DBLP3	4257	23540	10	100	3
DBLP5	6606	42815	10	100	5

### 6.3.2 Performance Results

**Hyper-Parameter Setting:** The baseline parameters are implemented using parameters listed in paper [7]. For our G2B model, the Adam optimizer with a learning rate of 0.025 was used. The dropout rate was set to be 0.5. G2B uses two dropout layers in total, one between each of the hidden layers. Training epochs are set to be 30.

From the result, it can be observed that G2B shows better Accuracy, AUC, and F1 scores than most of the baseline models, shown in Table-6.2. For both datasets, when 50% of data is used for training, our model performs better than all baselines. G2B overcomes the limitation of the GC-LSTM model using the BiLSTM layer instead. The performance of GraphSAGE is consistent and competitive compared to GAT and GCN, which demonstrates that using the GraphSAGE layer as the first layer in G2B is a better option.

### 6.3.3 Model Analysis

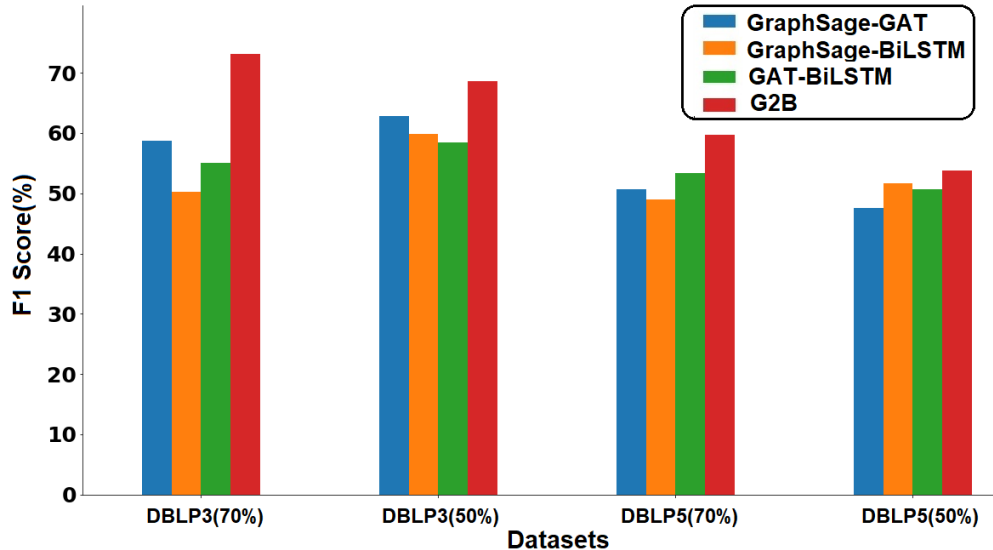
We disabled different components of our model to observe the changes in performance. **GraphSAGE-GAT** is the model with the BiLSTM layer disabled. In this case, we concatenated all the snapshots together into one single timestamp and applied GraphSAGE followed by GAT on the merged graph. **GraphSAGE-BiLSTM** is the model without the

<sup>1</sup><https://github.com/InterpretableClustering/InterpretableClustering>

**Table 6.2.** Performance Comparison Against the Baselines

	DBLP3(70%)			DBLP3(50%)			DBLP5(70%)			DBLP5(50%)		
	ACC	AUC	F1	ACC	AUC	F1	ACC	AUC	F1	ACC	AUC	F1
GCN[2]	74.3	<b>55.2</b>	63.6	70.1	54.4	65.0	57.1	49.9	47.1	54.3	51.7	45.1
GAT[3]	67.3	53.8	54.3	73.5	54.7	64.3	62.8	50.7	51.4	63.8	52.8	52.4
GraphSAGE[4]	71.9	54.1	60.5	72.9	52.2	64.1	64.6	<b>56.8</b>	57.0	61.7	53.3	52.8
GC-LSTM[5]	74.4	51.3	64.2	70.5	51.7	62.2	60.5	50.8	47.6	65.6	47.7	54.0
EGCN[6]	51.2	53.2	49.9	50.8	50.9	47.5	39.8	47.8	35.0	38.4	50.6	32.9
RNNGCN[7]	70.1	46.2	58.1	71.1	52.4	65.4	57.9	49.3	51.0	62.5	49.2	53.5
TRNNGCN[7]	71.6	53.5	59.9	59.5	51.6	52.0	56.1	47.5	48.4	53.6	48.5	46.6
G2B (Our Model)	<b>81.0</b>	50.6	<b>73.1</b>	<b>80.3</b>	<b>54.9</b>	<b>68.6</b>	<b>71.5</b>	53.2	<b>59.7</b>	<b>68.7</b>	<b>55.0</b>	<b>55.2</b>

GAT layer. **GAT-BiLSTM** is the model without the GraphSAGE layer. All models have been executed 30 times. The mean values of the performance metrics were calculated for comparison. The F1, ACC, and AUC comparisons of the models are shown in Figure 6.2, 6.3, and 6.4 respectively. We observed that G2B performs better than the others. It demonstrates the importance of all components in our model. On the DBLP3 dataset, GraphSAGE-GAT works better than the other two. Whereas on the DBLP5 dataset, the models with a BiLSTM component work better.

**Figure 6.2.** F1 of the Model Analysis

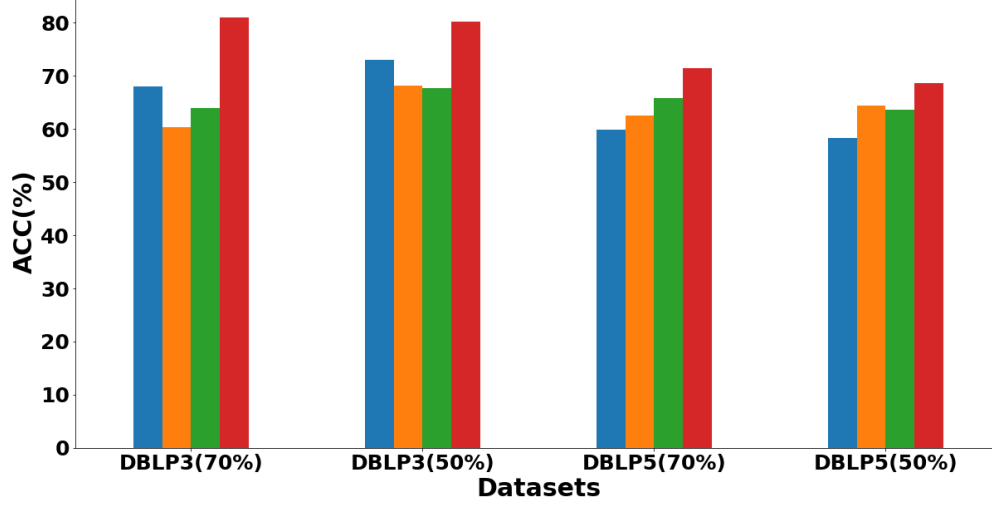


Figure 6.3. ACC of the Model Analysis

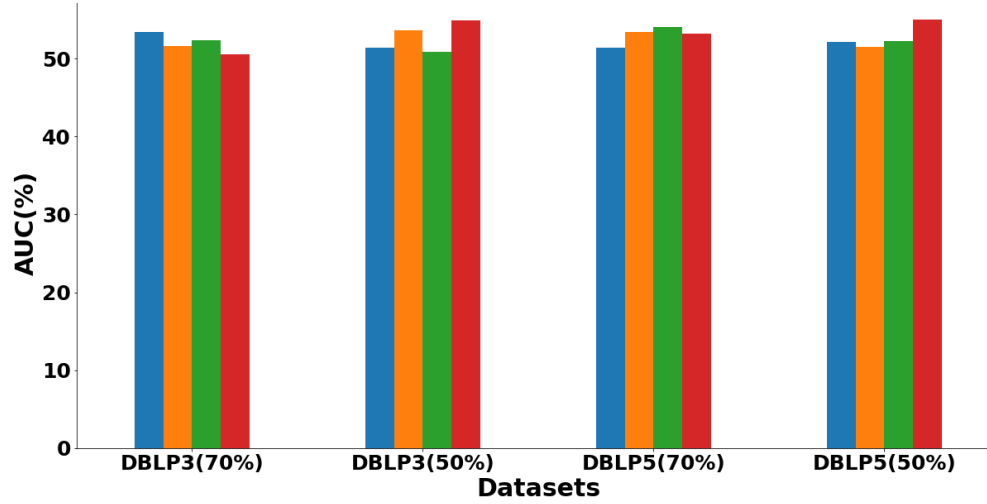
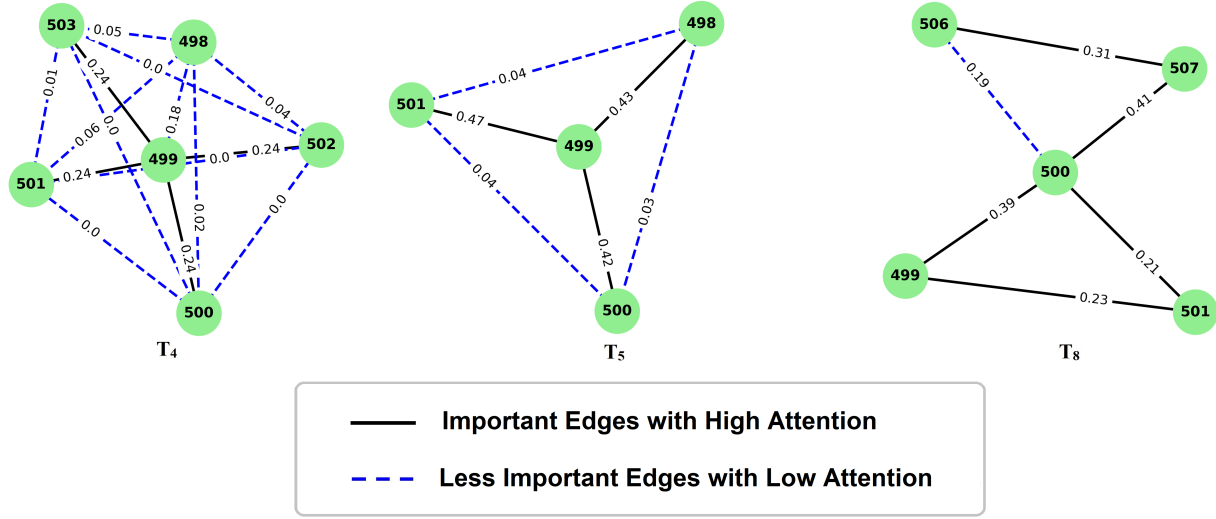


Figure 6.4. AUC the Model Analysis

#### 6.3.4 Dynamic Edge Interpretation

Our G2B model has the ability to determine the significance of edges over many time periods. After the characteristics of all the timestamps had been processed by the BiLSTM layer, we ranked the edges according to the relative attention weights that they had at each timestamp. We took subgraphs with highly ranked edges at each timestamp from the DBLP5(50%) dataset to demonstrate the edge importance interpretation.



**Figure 6.5.** Edge Importance Interpretation

In Figure 6.5, we set the threshold as 0.2 for each timestamp and mark the important edges with the solid lines. The dashed lines represent edges with relatively less importance with lower attention coefficients. In these dynamic subgraphs, edges (499, 500), (499, 501), and (500, 501) are included in the subgraphs corresponding to the timestamps. However, the edge (499, 500) and (499, 501) are consistently significant throughout all three timestamps as attention coefficients are always greater than the threshold. This analysis implies that author 499 consistently co-author papers with authors 500 and 501 over time.

## 6.4 Summary

The G2B model was created in this study to facilitate the classification of dynamic network nodes. As can be seen from the comparison to the baselines, the G2B model outperforms the baselines, especially when less data is used during training. In addition, G2B has the intelligence to understand the contribution of the edges at various stages.

## 7. CONCLUSION

In recent years, the use of graph neural networks to analyze graph data has made significant progress. This thesis investigated the potential real-world applications of Graph Neural Networks. We begin with a discussion of prevalent GNN architectures used in the industry. We attempt to explain the strategy for graph analysis using a simple network. Then We delve into two broad areas of node clustering and classification.

The contributions of this research include: (1) we perform symptom clustering in the clinical domain using a variety of clustering methods, compare them, and select the most effective method based on valid reasons; (2) we conduct statistical analysis to explain the demographics of our study cohort using the symptom clusters; (3) we analyze dynamic graphs where we describe our model to deal with the network’s temporal behavior to solve the problem of node classification.

The future work includes: (1) we intend to analyze symptom graphs from the dynamic graph perspective. Symptoms extracted from new EHR clinical notes at various times will introduce temporal behavior to the symptom graph; (2) we want to examine the emergence and alleviation of various symptoms and their importance across different time periods by the attention mechanism of G2B utilizing the time information associated with the conical notes; (3) we wish to investigate further initializing the parameters with the values learnt from previously trained models for rapid convergence while training the new symptom graph; (4) we aim to apply G2B in clinical data to utilize time-variant attributes and properties associated with the patient; (5) we plan to identify patients in a more vulnerable state with the help of attention-based outputs from G2B so that they can take more preventive steps depending on their condition and situation; (6) We hope to combine the strengths of SymptomGraph and G2B to challenge the state-of-the-art models in the clinical domain those deal with temporal graphs; (7) we would like to investigate GNN contributions in the field of link prediction on graphs, which is widely used in recommendation systems; (8) we are interested to apply these techniques to heterogeneous graphs with nodes of various types, whereas our current work is limited to graphs of homogeneous type.

Graph neural networks cover such a wide range of applications that they can be used to anticipate outcomes in many fields, from medicine and agriculture to retail and transportation to drug discovery and design. As our models perform better than many baseline models, we hope they propose robust solutions to many critical modern problems. Given the pervasiveness of graphs in contemporary data representation, all of these GNN applications appear highly relevant and provide an insightful glimpse into the future.



## REFERENCES

- [1] B. Sanchez-Lengeling, E. Reif, A. Pearce, and A. B. Wiltschko, “A gentle introduction to graph neural networks,” *Distill*, 2021, <https://distill.pub/2021/gnn-intro>. DOI: [10.23915/distill.00033](https://doi.org/10.23915/distill.00033).
- [2] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.
- [3] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” *stat*, vol. 1050, p. 20, 2017.
- [4] W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” *Advances in neural information processing systems*, vol. 30, 2017.
- [5] J. Chen, X. Wang, and X. Xu, “Gc-lstm: Graph convolution embedded lstm for dynamic link prediction,” *arXiv preprint arXiv:1812.04206*, 2018.
- [6] A. Pareja, G. Domeniconi, J. Chen, *et al.*, “Evolvegc: Evolving graph convolutional networks for dynamic graphs,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 5363–5370.
- [7] Y. Yao and C. Joe-Wong, “Interpretable clustering on dynamic graphs with recurrent graph neural networks,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.
- [8] A. Grover and J. Leskovec, “Node2vec: Scalable feature learning for networks,” in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.
- [9] P. Velikovi, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” *arXiv preprint arXiv:1710.10903*, 2017.
- [10] Y. Miao, Y. Xu, and D. Mandic, “Hyper-gst: Predict metro passenger flow incorporating graphsage, hypergraph, social-meaningful edge weights and temporal exploitation,” *arXiv preprint arXiv:2211.04988*, 2022.
- [11] R. Mahajan and V. Mansotra, “Predicting geolocation of tweets: Using combination of cnn and bilstm,” *Data Science and Engineering*, vol. 6, no. 4, pp. 402–410, 2021.

- [12] A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional lstm and other neural network architectures,” *Neural networks*, vol. 18, no. 5-6, pp. 602–610, 2005.
- [13] G. Liu and J. Guo, “Bidirectional lstm with attention mechanism and convolutional layer for text classification,” *Neurocomputing*, vol. 337, pp. 325–338, 2019.
- [14] L. R. Medsker and L. Jain, “Recurrent neural networks,” *Design and Applications*, vol. 5, pp. 64–67, 2001.
- [15] A. Arya, P. K. Pandey, and A. Saxena, “Node classification using deep learning in social networks,” in *Deep Learning for Social Media Data Analytics*, Springer, 2022, pp. 3–26.
- [16] R. S. Solomon, P. Srinivas, A. Das, B. Gamback, and T. Chakraborty, “Understanding the psycho-sociological facets of homophily in social network communities,” *IEEE Computational Intelligence Magazine*, vol. 14, no. 2, pp. 28–40, 2019.
- [17] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.
- [18] J. Aldrich, “Eigenvalue, eigenfunction, eigenvector, and related terms,” *Earliest known uses of some of the words of mathematics*, 2006.
- [19] W. Fleshman, *Spectral clustering foundation and application*, <https://towardsdatascience.com/spectral-clustering-aba2640c0d5b>, Accessed: 2022-11-13, 2019.
- [20] U. Brandes, “On variants of shortest-path betweenness centrality and their generic computation,” *Social networks*, vol. 30, no. 2, pp. 136–145, 2008.
- [21] S. Fortunato, “Community detection in graphs,” *Physics reports*, vol. 486, no. 3-5, pp. 75–174, 2010.
- [22] A. Hagberg, P. Swart, and D. S Chult, “Exploring network structure, dynamics, and function using networkx,” Los Alamos National Lab.(LANL), Los Alamos, NM (United States), Tech. Rep., 2008.
- [23] M. Girvan and M. E. Newman, “Community structure in social and biological networks,” *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

- [24] R. A. Rossi and N. K. Ahmed, “Role discovery in networks,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 4, pp. 1112–1131, 2014.
- [25] O. Sporns and R. F. Betzel, “Modular brain networks,” *Annual review of psychology*, vol. 67, p. 613, 2016.
- [26] P. Chen and S. Redner, “Community structure of the physical review citation network,” *Journal of Informetrics*, vol. 4, no. 3, pp. 278–290, 2010.
- [27] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” *Advances in neural information processing systems*, vol. 26, 2013.
- [28] B. Perozzi, R. Al-Rfou, and S. Skiena, “Deepwalk: Online learning of social representations,” in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.
- [29] R. A. Rossi, D. Jin, S. Kim, N. K. Ahmed, D. Koutra, and J. B. Lee, “On proximity and structural role-based embeddings in networks: Misconceptions, techniques, and applications,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 14, no. 5, pp. 1–37, 2020.
- [30] W. W. Zachary, “An information flow model for conflict and fission in small groups,” *Journal of anthropological research*, vol. 33, no. 4, pp. 452–473, 1977.
- [31] H.-J. Kim, D. B. McGuire, L. Tulman, and A. M. Barsevick, “Symptom clusters: Concept analysis and clinical implications for cancer nursing,” *Cancer nursing*, vol. 28, no. 4, pp. 270–282, 2005.
- [32] D. C. McFarland, K. M. Shaffer, A. Tiersten, and J. Holland, “Physical symptom burden and its association with distress, anxiety, and depression in breast cancer,” *Psychosomatics*, vol. 59, no. 5, pp. 464–471, 2018.
- [33] K. Stavem, W. Ghanima, M. K. Olsen, H. M. Gilboe, and G. Einvik, “Persistent symptoms 1.5–6 months after covid-19 in non-hospitalised subjects: A population-based cohort study,” *Thorax*, vol. 76, no. 4, pp. 405–407, 2021.
- [34] P. Bytzer, N. J. Talley, M. Leemon, L. J. Young, M. P. Jones, and M. Horowitz, “Prevalence of gastrointestinal symptoms associated with diabetes mellitus: A population-based survey of 15 000 adults,” *Archives of internal medicine*, vol. 161, no. 16, pp. 1989–1996, 2001.

- [35] T. A. Koleck, C. Dreisbach, P. E. Bourne, and S. Bakken, “Natural language processing of symptoms documented in free-text narratives of electronic health records: A systematic review,” *Journal of the American Medical Informatics Association*, vol. 26, no. 4, pp. 364–379, 2019.
- [36] H.-J. Kim, “Common factor analysis versus principal component analysis: Choice for symptom cluster research,” *Asian nursing research*, vol. 2, no. 1, pp. 17–24, 2008.
- [37] J.-S. Tsai, C.-H. Wu, T.-Y. Chiu, and C.-Y. Chen, “Significance of symptom clustering in palliative care of advanced cancer patients,” *Journal of pain and symptom management*, vol. 39, no. 4, pp. 655–662, 2010.
- [38] S. Chow, B. A. Wan, W. Pidduck, *et al.*, “Symptom clusters in patients with breast cancer receiving radiation therapy,” *European Journal of Oncology Nursing*, vol. 42, pp. 14–20, 2019.
- [39] S. T. Dong, P. N. Butow, D. S. Costa, M. R. Lovell, and M. Agar, “Symptom clusters in patients with advanced cancer: A systematic review of observational studies,” *Journal of pain and symptom management*, vol. 48, no. 3, pp. 411–450, 2014.
- [40] D. Walsh and L. Rybicki, “Symptom clustering in advanced cancer,” *Supportive care in cancer*, vol. 14, no. 8, pp. 831–836, 2006.
- [41] N. Papachristou, C. Miaskowski, P. Barnaghi, *et al.*, “Comparing machine learning clustering with latent class analysis on cancer symptoms’ data,” in *2016 IEEE Healthcare Innovation Point-Of-Care Technologies Conference (HI-POCT)*, IEEE, 2016, pp. 162–166.
- [42] K. Gu, S. Vosoughi, and T. Prioleau, “Symptomid: A framework for rapid symptom identification in pandemics using news reports,” *ACM Transactions on Management Information Systems (TMIS)*, vol. 12, no. 4, pp. 1–17, 2021.
- [43] P. Sondhi, J. Sun, H. Tong, and C. Zhai, “Sympgraph: A framework for mining clinical notes through symptom relation graphs,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 1167–1175.
- [44] H.-J. Kim and I. L. Abraham, “Statistical approaches to modeling symptom clusters in cancer patients,” *Cancer nursing*, vol. 31, no. 5, E1–E10, 2008.

- [45] J. Ni, H. Fei, W. Fan, and X. Zhang, “Automated medical diagnosis by ranking clusters across the symptom-disease network,” in *2017 IEEE International Conference on Data Mining (ICDM)*, IEEE, 2017, pp. 1009–1014.
- [46] L. A. Linder, S. Ameringer, C. Baggott, *et al.*, “Measures and methods for symptom and symptom cluster assessment in adolescents and young adults with cancer,” in *Seminars in oncology nursing*, Elsevier, vol. 31, 2015, pp. 206–215.
- [47] A. M. Barsevick, “The elusive concept of the symptom cluster,” in *Oncology nursing forum*, vol. 34, 2007.
- [48] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [49] A. Radford and J. Wu, “Rewon child, david luan, dario amodei, and ilya sutskever. 2019,” *Language models are unsupervised multitask learners. OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.
- [50] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” *Advances in neural information processing systems*, vol. 32, 2019.
- [51] K. I. Neijenhuijs, C. F. Peeters, H. van Weert, P. Cuijpers, and I. V.-d. Leeuw, “Symptom clusters among cancer survivors: What can machine learning techniques tell us?” *BMC Medical Research Methodology*, vol. 21, no. 1, pp. 1–12, 2021.
- [52] R. J. Campello, D. Moulavi, and J. Sander, “Density-based clustering based on hierarchical density estimates,” in *Pacific-Asia conference on knowledge discovery and data mining*, Springer, 2013, pp. 160–172.
- [53] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *kdd*, vol. 96, 1996, pp. 226–231.
- [54] F. Shen, S. Liu, Y. Wang, *et al.*, “Constructing node embeddings for human phenotype ontology to assist phenotypic similarity measurement,” in *2018 IEEE International Conference on Healthcare Informatics Workshop (ICHI-W)*, IEEE, 2018, pp. 29–33.
- [55] J. Du, Q. Wang, J. Wang, *et al.*, “Covid-19 trial graph: A linked graph for covid-19 clinical trials,” *Journal of the American Medical Informatics Association*, 2021.

- [56] M. Kim, S. H. Baek, and M. Song, “Relation extraction for biological pathway construction using node2vec,” *BMC bioinformatics*, vol. 19, no. 8, pp. 75–84, 2018.
- [57] Z.-H. Chen, Z.-H. You, Z.-H. Guo, H.-C. Yi, G.-X. Luo, and Y.-B. Wang, “Predicting drug-target interactions by node2vec node embedding in molecular associations network,” in *International Conference on Intelligent Computing*, Springer, 2020, pp. 348–358.
- [58] J. Lee, C. Liu, J. H. Kim, *et al.*, “Comparative effectiveness of medical concept embedding for feature engineering in phenotyping,” *JAMIA open*, vol. 4, no. 2, ooab028, 2021.
- [59] D. Oniani, G. Jiang, H. Liu, and F. Shen, “Constructing co-occurrence network embeddings to assist association extraction for covid-19 and other coronavirus infectious diseases,” *Journal of the American Medical Informatics Association*, vol. 27, no. 8, pp. 1259–1267, 2020.
- [60] L. Bai, L. Yao, S. Kanhere, X. Wang, Q. Sheng, *et al.*, “Stg2seq: Spatial-temporal graph to sequence model for multi-step passenger demand forecasting,” *arXiv preprint arXiv:1905.10069*, 2019.
- [61] C. Su, S. Gao, and S. Li, “Gate: Graph-attention augmented temporal neural network for medication recommendation,” *IEEE Access*, vol. 8, pp. 125 447–125 458, 2020.
- [62] Z. Cui, L. Lin, Z. Pu, and Y. Wang, “Graph markov network for traffic forecasting with missing data,” *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102 671, 2020.
- [63] J. Zhu, Q. Xie, and E. J. Chin, “A hybrid time-series link prediction framework for large social network,” in *International Conference on Database and Expert Systems Applications*, Springer, 2012, pp. 345–359.
- [64] L. Yao, L. Wang, L. Pan, and K. Yao, “Link prediction based on common-neighbors for dynamic social network,” *Procedia Computer Science*, vol. 83, pp. 82–89, 2016.
- [65] Y. Fan, Y. Yao, and C. Joe-Wong, “Gcn-se: Attention as explainability for node classification in dynamic graphs,” in *2021 IEEE International Conference on Data Mining (ICDM)*, IEEE, 2021, pp. 1060–1065.

- [66] D. Liben-Nowell and J. Kleinberg, “The link-prediction problem for social networks,” *Journal of the American society for information science and technology*, vol. 58, no. 7, pp. 1019–1031, 2007.
- [67] U. Sharan and J. Neville, “Temporal-relational classifiers for prediction in evolving domains,” in *2008 Eighth IEEE International Conference on Data Mining*, IEEE, 2008, pp. 540–549.
- [68] R. Hisano, “Semi-supervised graph embedding approach to dynamic link prediction,” in *International Workshop on Complex Networks*, Springer, 2018, pp. 109–121.
- [69] J. Skarding, B. Gabrys, and K. Musial, “Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey,” *IEEE Access*, vol. 9, pp. 79 143–79 168, 2021.
- [70] N. Dehmamy, A.-L. Barabási, and R. Yu, “Understanding the representation power of graph neural networks in learning graph topology,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [71] W. W. Lo, S. Layeghy, M. Sarhan, M. Gallagher, and M. Portmann, “E-graphsage: A graph neural network based intrusion detection system,” *arXiv preprint arXiv:2103.16329*, 2021.
- [72] D. Xu, W. Cheng, D. Luo, X. Liu, and X. Zhang, “Spatio-temporal attentive rnn for node classification in temporal attributed graphs,” in *IJCAI*, 2019, pp. 3947–3953.
- [73] S. Serrano and N. A. Smith, “Is attention interpretable?” *arXiv preprint arXiv:1906.03731*, 2019.
- [74] A. R. Aronson, “Effective mapping of biomedical text to the umls metathesaurus: The metamap program,” in *Proceedings of the AMIA Symposium*, American Medical Informatics Association, 2001, p. 17.
- [75] D. Cer, Y. Yang, S.-y. Kong, *et al.*, “Universal sentence encoder,” *arXiv preprint arXiv:1803.11175*, 2018.
- [76] X. Luo, P. Gandhi, S. Storey, Z. Zhang, Z. Han, and K. Huang, “A computational framework to analyze the associations between symptoms and cancer patient attributes post chemotherapy using ehr data,” *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 11, pp. 4098–4109, 2021.

- [77] J. Zhou, G. Cui, S. Hu, *et al.*, “Graph neural networks: A review of methods and applications,” *AI Open*, vol. 1, pp. 57–81, 2020.
- [78] S. N. Golmaei and X. Luo, “Deepnote-gnn: Predicting hospital readmission using clinical notes and patient network,” in *Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*, 2021, pp. 1–9.
- [79] C. Kang, H. Zhang, Z. Liu, S. Huang, and Y. Yin, “Lr-gnn: A graph neural network based on link representation for predicting molecular associations,” *Briefings in Bioinformatics*, 2021.
- [80] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.,” *Journal of machine learning research*, vol. 9, no. 11, 2008.
- [81] N. S. Redeker, E. L. Lev, and J. Ruggiero, “Insomnia, fatigue, anxiety, depression, and quality of life of cancer patients undergoing chemotherapy,” *Scholarly inquiry for nursing practice*, vol. 14, no. 4, pp. 275–290, 2000.
- [82] D. E. Theobald, “Cancer pain, fatigue, distress, and insomnia in cancer patients,” *Clinical cornerstone*, vol. 6, no. 1, S15–S21, 2004.
- [83] G.-W. Sun, Y.-L. Yang, X.-B. Yang, *et al.*, “Preoperative insomnia and its association with psychological factors, pain and anxiety in Chinese colorectal cancer patients,” *Supportive Care in Cancer*, vol. 28, no. 6, pp. 2911–2919, 2020.
- [84] H. Andreyev, A. Norman, J. Oates, and D. Cunningham, “Why do patients with weight loss have a worse outcome when undergoing chemotherapy for gastrointestinal malignancies?” *European journal of cancer*, vol. 34, no. 4, pp. 503–509, 1998.
- [85] A. Carlotto, V. L. Hogsett, E. M. Maiorini, J. G. Razulis, and S. T. Sonis, “The economic burden of toxicities associated with cancer treatment: Review of the literature and analysis of nausea and vomiting, diarrhoea, oral mucositis and fatigue,” *Pharmacoeconomics*, vol. 31, no. 9, pp. 753–766, 2013.
- [86] R. Dev, H. A. Parsons, S. Palla, J. L. Palmer, E. Del Fabbro, and E. Bruera, “Undocumented alcoholism and its correlation with tobacco and illegal drug use in advanced cancer patients,” *Cancer*, vol. 117, no. 19, pp. 4551–4556, 2011.
- [87] N. P. Staff, A. Grisold, W. Grisold, and A. J. Windebank, “Chemotherapy-induced peripheral neuropathy: A current review,” *Annals of neurology*, vol. 81, no. 6, pp. 772–781, 2017.



- [88] C. Tofthagen, R. D. McAllister, and S. C. McMillan, "Peripheral neuropathy in patients with colorectal cancer receiving oxaliplatin.," *Clinical journal of oncology nursing*, vol. 15, no. 2, 2011.
- [89] M. S. Cappell, "Pathophysiology, clinical presentation, and management of colon cancer," *Gastroenterology Clinics of North America*, vol. 37, no. 1, pp. 1–24, 2008.
- [90] S. Schiffman and B. Graham, "Taste and smell perception affect appetite and immunity in the elderly," *European journal of clinical nutrition*, vol. 54, no. 3, S54–S63, 2000.
- [91] N. de Jong, I. Mulder, C. de Graaf, and W. A. van Staveren, "Impaired sensory functioning in elders: The relation with its potential determinants and nutritional intake," *Journals of Gerontology Series A: Biomedical Sciences and Medical Sciences*, vol. 54, no. 8, B324–B331, 1999.
- [92] P. A. Vissers, F. Mols, M. S. Thong, F. Pouwer, G. Vreugdenhil, and L. V. van de Poll-Franse, "The impact of diabetes on neuropathic symptoms and receipt of chemotherapy among colorectal cancer patients: Results from the profiles registry," *Journal of Cancer Survivorship*, vol. 9, no. 3, pp. 523–531, 2015.
- [93] M. Sempere-Bigorra, I. Julián-Rochina, and O. Cauli, "Chemotherapy-induced neuropathy and diabetes: A scoping review," *Current Oncology*, vol. 28, no. 4, pp. 3124–3138, 2021.
- [94] S. K. Baek, S. W. Shin, S.-J. Koh, *et al.*, "Significance of descriptive symptoms and signs and clinical parameters as predictors of neuropathic cancer pain," *PloS one*, vol. 16, no. 8, e0252781, 2021.