1

Introduction to Graph and Graph Representation Learning

1.1 Introduction

In today's rapidly evolving, dynamic, and interconnected world, scientists, researchers, businessmen, government, and all humans need a faster and more efficient way of solving problems and challenges with emerging innovations, discoveries, and opportunities. The graph is the most promising and invaluable tool for visualizing and analyzing the business, communicating the research findings and patterns in the data, and gaining knowledge. This chapter provides insight into graphs and graph representation learning (GRL) along with its benefits, and the application in various domains.

1.2 What Is a Graph?

A graph is a mathematical tool for representing data or objects visually and finding associations among objects or pairs of points. It is defined as the network G with an ordered pair of sets (N, E), where N denotes nodes (or vertices), and the links between the nodes N are termed edges E. The nodes are termed entities/features or variables. These nodes represent the individual entities or objects within the system being modeled. For instance, in a social network, nodes might be people; in a road network, cities; in a biological network, proteins or genes. An edge is typically represented as a pair of nodes (u, v), indicating that node u is related to node v. Examples of graphs are as follows:

Example 1.1 In Fig. 1.1 above, A, B, C, D, and E are the nodes, and the relationship among the nodes is shown by a straight line. They are the edges represented as $E = \{(A, B), (B, C), (C, D), (D, E), (E, A), (E, C)\}$.

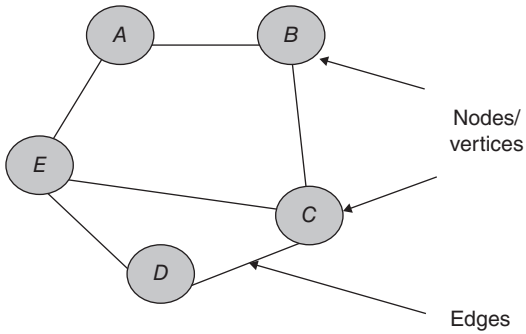

Figure 1.1 Sample graph example.

Example 1.2 ***Entrepreneurs Working in the Healthcare Field***
Entrepreneurs who are working in the healthcare field depend on new technologies, services, and business models to improve patient care, increase accessibility, and reduce costs. They address critical needs in areas such as telemedicine, medical devices, health data analytics, and personalized medicine, driving advancements in the healthcare industry.

In Example 1.2, each node represents an entrepreneur, and edges connect entrepreneurs who work in the healthcare industry. In this graph, each entrepreneur working within the healthcare sector is represented as a distinct node. An edge labeled "health" connects these entrepreneurial nodes, signifying their shared work and impact on the healthcare domain. This is represented in Fig. 1.2.

Example 1.3 ***Entrepreneurs Working in Similar Fields***
Example 1.3 shows the entrepreneurs who share identical fields as given in Fig. 1.3.

Each node represents an entrepreneur. Relationships are represented by edges connecting entrepreneurs who share similar fields (three types of industries such as healthcare, retail, and textile).

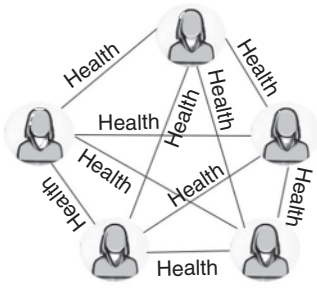

Figure 1.2 Entrepreneurs working in the healthcare industry.

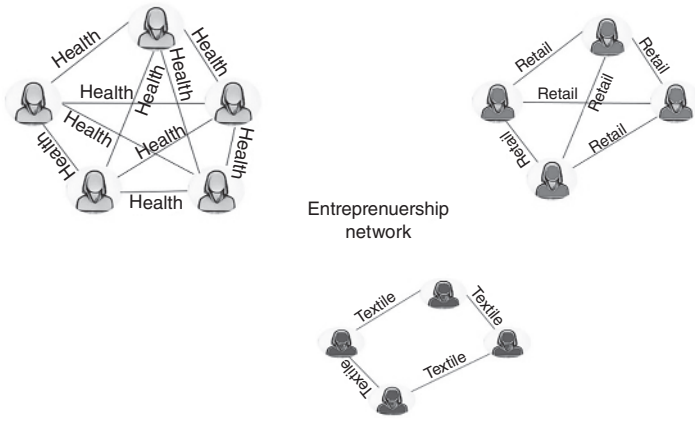

Figure 1.3 Entrepreneurs working in similar fields.

1.3 Importance of Graph

- Before the rise of GRL, traditional machine learning models struggled significantly with complex, interconnected data. These conventional methods relied heavily on manual feature engineering, either by completely overlooking inherent relationships or painstakingly crafting them by hand. This approach severely hampered both scalability and adaptability. While effective for structured data such as tables, sequences, or grids, these methods simply couldn't capture the rich, non-Euclidean structures intrinsic to real-world networks such as social connections, biological systems, and knowledge graphs.
- Graphs are essential tools for making sense of complex, interconnected data.
- They offer a visual framework that simplifies real-time information, serving us to discover patterns, trends, and relationships that remain hidden in raw datasets.
- By effectively representing complicated connections, graphs become crucial for extracting hidden insights, conducting thorough analyses, and clearly communicating outcomes.
- They naturally foster network connections, proving vital for forecasting, predictive modeling, and building robust decision support systems. Furthermore, graphs are an excellent way to share research findings with both experts and the public.
- Their universal utility stretches across numerous real-world applications, from healthcare and finance to social media, computer vision, and recommender systems, solidifying their role as a fundamental data structure for complex systems.

- They help identify patterns, trends, and relationships that might be difficult to distinguish from raw data.
- They allow the users to get connected to the network.
- They are used for forecasting, predicting, and supporting decision-making systems.
- Graphs are a universal framework and a widely used data structure for representing complex systems.
- They provide a statistical technique for finding interactions among the entities and interpreting the connections.

1.4 Types of Graphs

Graphs can be categorized into various types based on their properties [1], structures, and connectivity. Broadly, graphs are classified as directed or undirected depending on whether their edges have a direction. They can also be weighted or unweighted, where weighted graphs assign values to edges, representing costs or distances. Some graphs are simple, with no loops or multiple edges, while multigraphs allow parallel edges. Specific types like bipartite graphs have nodes divided into two disjoint sets, whereas complete graphs connect every pair of nodes. Trees represent hierarchical structures, and planar graphs can be drawn without edge crossings. Other classifications include sparse and dense graphs, cyclic and acyclic graphs, each suited for different applications in network analysis and machine learning. Each type of graph is discussed in this section, and the domain area where they are applied are also discussed.

1.4.1 Directed Graph

A graph where the edges between nodes are represented with arrow lines indicating direction is called a directed graph [1]. In a directed graph, the flow of process or path flows is in the direction of the arrow pointing to the other edge. Consider the author citation graph example: "Author A's Paper is cited by Author B" (Fig. 1.4). This is represented as a directed graph by having nodes as Author A's Paper and edges as a "cited" relationship between papers. An author citation graph can be represented as a directed graph, as shown in Fig. 1.4.

The diagram shown in Fig. 1.4 depicts that Author "A" Paper cited Author "B" Paper and so on. The directed graph provides details about the nodes, edges, and graph properties. Nodes ensure the author's name; edges ensure that the relationship is directional; and weight can be assigned to each edge by simply counting the number of citations. Graph properties such as in-degree and out-degree can be used to find the number of citations received by an author and outdegree to

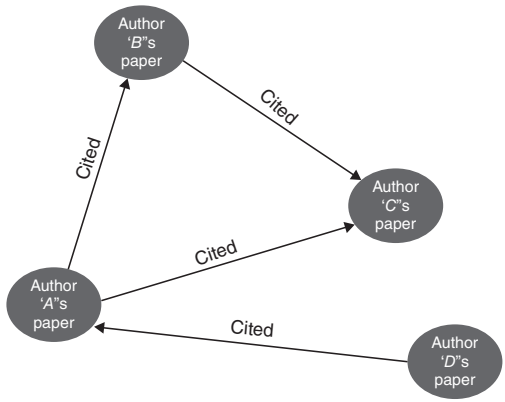

Figure 1.4 Directed graph for author citation.

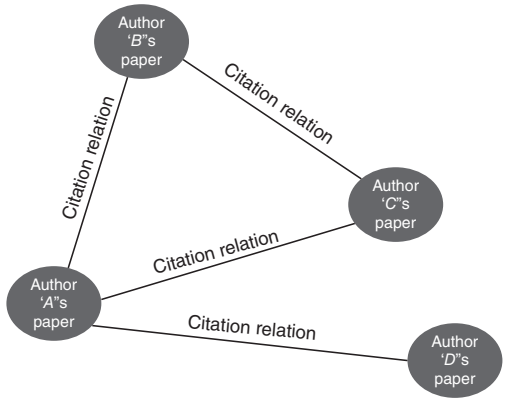

Figure 1.5 Undirected graph example for author citation.

see the number of citations made by an author, respectively. In a directed graph, analysis can be made to identify the authors whose papers are most cited. This can be done by finding the highest in-degree.

1.4.2 Undirected Graph

An undirected graph is a type of graph where edges have no direction. Each edge connects two vertices (nodes) symmetrically. The process flow or path can be done in both directions. Undirected graphs are useful in situations where directions do not matter, and it's simpler to use. For the same example given above, "There is a citation relationship between Author A and Author B." The graph now represents citation activities between authors, but it does not indicate who cited whom. If two authors cite each other, the edge indicates mutual influence or collaboration.

Figure 1.5 shows an example of an undirected graph using an author citation graph with authors and papers as nodes.

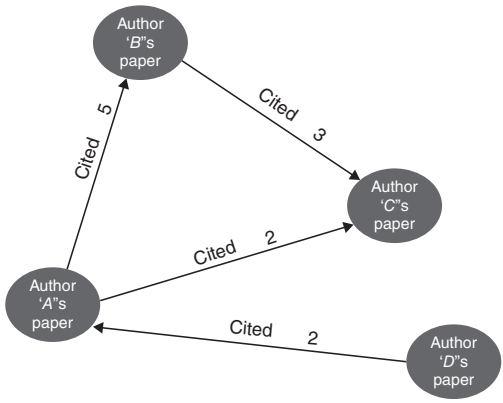

Figure 1.6 Author citation example using weighted graph.

The above undirected graph can be used for identifying clusters of authors that frequently cite each other. It could be used to visualize collaboration or shared research interests, where the focus is on connections rather than specific citations.

1.4.3 Weighted Graph

The other type of graph is a weighted graph, where edges are associated with weights, which represent some kind of value or cost of the connection between nodes. These weights can indicate distances, costs, capacities, strengths, or other quantitative measures. Weighted graphs can be directed or undirected, depending on whether edges have a direction. The main purpose of a weighted graph is that the weights provide additional information about the nature of the connections.

In the context of the author citation graph given in Fig. 1.6, transforming it into a weighted graph adds extra measurable information to the edges, making the representation richer and more meaningful. Weights can signify the strength, frequency, or importance of the relationship between two authors. Let us take the weight information as the frequency of citation between two authors' papers. For example, in the above example, "Author B's Paper has been cited five times by Author A" and "Author C's Paper has been cited three times by Author B." These graphs are used to identify how frequently an author cites another, help to identify influential authors based on their total citation weight, and highlight the quality of the authors' papers. We can group the authors based on their mutual citation.

1.4.4 Multigraph

A multigraph is a kind of graph that allows multiple edges (parallel edges) between the same pair of vertices (nodes). These edges may or may not have associated

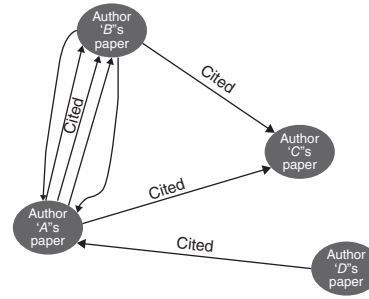

Figure 1.7 Author citation example using a multigraph.

weights. Repeated or redundant interactions between the same entities occur in multigraphs. For the same example of the author citation graph, a multigraph would represent multiple citation instances between the same pair of authors. In Fig. 1.7, Author A's Paper cites three different papers of Author B, and there will be three edges between Authors A and B. Similarly, Author B's Paper cites author A's paper two times.

A multigraph extends the concept of a simple graph by allowing multiple edges between the same pair of nodes. It is a powerful tool for modeling real-world systems with complex relationships. A multigraph may also contain loops, which are edges that connect a node to itself.

1.4.5 Complete Graph

A complete graph is another type of graph where every pair of distinct nodes is connected by a unique edge. In a complete graph, every node interacts with every other node directly. It is an undirected and connected graph where there must be at least one path between any two nodes.

In the author citation graph example given in Fig. 1.8, a complete graph is a graph where every author is directly connected to every other author. Each author cites every other author's paper at least once. The graph is fully connected, which shows that none of the authors is left out of the citation loop.

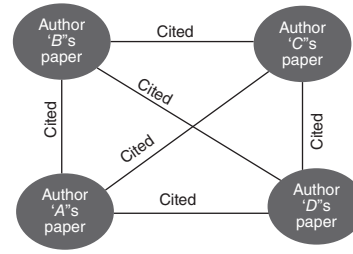

Figure 1.8 Example of complete graph.

1.4.6 Planar and Bipartite Graph

A planar graph is a type of graph that can be drawn on a two-dimensional plane without any edges crossing each other. The key feature of planar graphs is that their edges only intersect at their endpoints (vertices). If a graph can be arranged this way, it is said to be embeddable in the plane.

For example:

- A triangle graph is planar, as its three edges can be drawn without any crossings.
- The complete graph with four nodes with edges connecting every pair is also planar, but higher-order complete graphs with a crossing of lines are not planar.

In the author citation graph, nodes represent a small group of researchers, and directed edges indicate one author citing another author's work within a specialized domain. The graph's simple structure allows it to be drawn on a plane without any citation links crossing, reflecting a clear, nonoverlapping flow of knowledge or collaboration among closely related works. The graph in Fig. 1.8 is nonplanar, as there is a crossing of lines.

Bipartite Graph

A bipartite graph shown in Fig. 1.9 is a two-set graph in which the nodes can be divided into two disjoint and independent sets S_1 and S_2, such that every edge connects a node from S_1 to a node in S_2. Edges only exist between nodes in different sets. This special type of graph is useful for analyzing the relationship between two distinct groups of nodes. Consider the author citation graph as a bipartite graph where the set of nodes in the S_1 and S_2 sets represents authors and papers, respectively. The edges represent relationships between authors who cite papers in set S_1 and papers that are cited by authors in set S_2.

In Fig. 1.9, the authors' nodes ("Author A," "Author B," "Author C") are in set S_1, and the Papers' nodes ("Paper 1," "Paper 2," "Paper 3") are in set S_2. The citation relationship is represented as Author A cites Paper 1 and Paper 2, Author B cites Paper 2 and Paper 3, and Author C cites Paper 1. The purpose of using a bipartite graph is to analyze the relationship between different types of nodes more easily

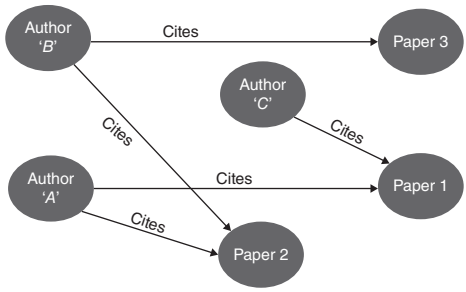

Figure 1.9 Bipartite graph for author citation.

by dividing the nodes into two sets. Also, we can find the shared citation patterns and similar citation behaviors.

The main usage of bipartite graphs in author citation graphs is to recommend papers to authors based on similar citation patterns or topics of interest, to analyze which papers are frequently cited by many authors, and to identify clusters of authors citing similar papers, suggesting shared research interests or collaborations.

1.4.7 Hypergraph

A hypergraph is a generalization of a traditional graph where edges, known as hyperedges, can connect more than two nodes. Unlike traditional graphs, where edges link only two nodes, hyperedges in a hypergraph can connect a subset of nodes, enabling the representation of group relationships. Hypergraphs are suitable for capturing complex, multi-way interactions that standard traditional graphs cannot represent. They are used for tasks requiring group-level relationships. They are also designed to capture higher-order relationships.

For the author citation graph, authors (nodes) are connected by citations (edges). Typically, this could be represented as a directed graph, where an edge from node A (Author A) to node B (Author B) means that Author A cites Author B in their work.

However, in a hypergraph model, we can represent more complex relationships such as a group of authors collaborating on a paper and citing each other. In this case, a hyperedge can represent the citation of a paper written by a group of authors, rather than just a single citation relationship between two authors.

Figure 1.10 represents a hypergraph for the author citation example. Nodes represent the author's paper, and the hyperedge represents collaborative relationships. A hyperedge connects a set of vertices (authors) that are involved in a specific citation or collaboration. One hyperedge $H1$ connects Authors A, B, and C, signifying that they are involved in a collaborative paper or a citation relationship where B and C co-author a paper and cite A. Hyperedge $H2$ $\{A, B\}$

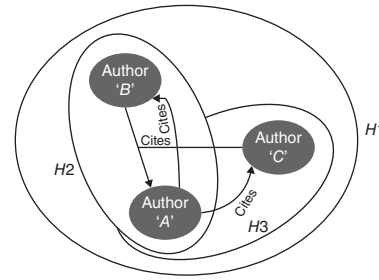

Figure 1.10 Hypergraph for author citation.

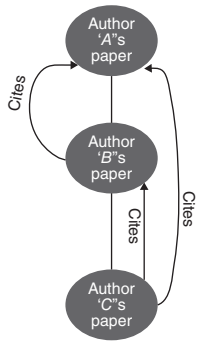

Figure 1.11 Hierarchical graph for author citation.

indicates that Author A cites Author B. Hyperedge $H3\{A, C\}$ indicates that Author A cites Author C.

A hypergraph allows to represent more complex relationships in a citation graph, such as group collaborations or joint citations, which would be difficult to represent using simple pairwise edges. If multiple authors contribute to a paper and cite each other, a hypergraph captures this collaboration better by using a single hyperedge instead of multiple pairwise edges.

1.4.8 Hierarchical Graphs

Another type of special graph is a hierarchical graph, which organizes nodes in layers or levels, forming a hierarchy. Nodes in one layer can be connected to nodes in the adjacent layer. This graph is used to identify the dependencies between nodes in a hierarchical manner. This type of graph helps in the decision-making process.

In the author citation graph shown in Fig. 1.11, the author whose papers are highly cited appears in the top-level node (Author A's Paper is highly cited, as shown in the figure), the author whose papers are moderately cited appears in the next level (Author B's Paper is moderately cited and appears in the middle-level node), and the author whose papers are less cited appears in the bottom-level node (Author C's Paper is less cited, and it is in the lower level).

The graph helps identify influential authors (top-level nodes) and how their work propagates through the hierarchy. It helps the institutions to prioritize the faculty members (researchers) based on their paper publications. This type of graph facilitates analysis of relationships between influential and emerging authors.

1.4.9 Homogeneous and Heterogeneous Graphs

A homogeneous graph is a special kind of graph in which all nodes (vertices) and edges are of the same type. The graph represents relationships or interactions

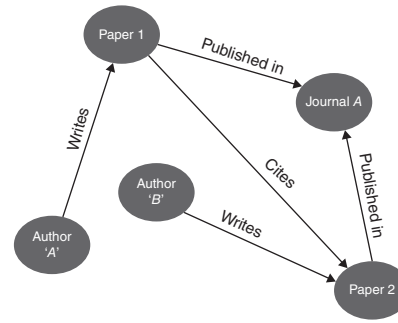

Figure 1.12 Heterogeneous graph for author citation.

between nodes that are fundamentally similar in nature. Homogeneous graphs provide a simpler structure by focusing on one type of relationship or entity. A single type of entity (Author) is represented in nodes. Homogeneous graphs are easier to process computationally because the relationships are uniform. This allows for focused analysis of specific types of interactions or relationships.

In the author citation graph, each node represents an author in the citation network. All nodes represent the same type of entity (authors). All edges represent the same type of relationship (citations). The graph in the figure is a homogeneous graph.

1.4.9.1 Heterogeneous Graph

A heterogeneous graph is also a special kind of graph that involves multiple types of nodes (entities) and/or multiple types of edges (relationships). It is designed to model complex systems where different entities interact in various ways [6]. Nodes are the different types of entities. Here, different types of relationships between entities appear.

A heterogeneous graph given in Fig. 1.12 for a citation graph might include nodes, authors (*A* and *B*), papers (Paper 1 and Paper 2), and a journal (Journal A). The edges in the graph represent relationships like "writes," "cites," and "published in." The relationships can include "author writes a paper," "paper is published in a journal," and "paper cites another paper."

1.5 Overview of Graph Representation Learning (GRL)

Graphs are powerful tools for modeling relationships. However, many graph algorithms are computationally expensive. Large-scale graphs (e.g., social networks, biological networks) have millions or billions of nodes and edges, making storage and processing challenging. Representing different graph structures and their relationships effectively is very difficult. There is no proper method for capturing information and knowledge from various graphs and communication between

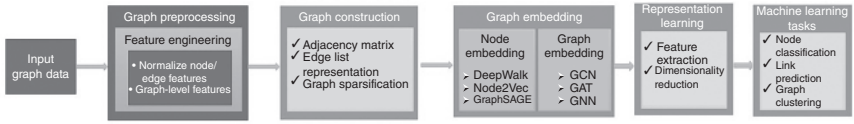

Figure 1.13 Graph representation learning flow of process.

graphs. Traditional graph-based algorithms often struggle with scalability and generalization, requiring manual feature engineering to extract meaningful information. To overcome these challenges, GRL, which represents graph structure in low-dimensional vectors, enables one to learn automatically from complex graphs. GRL has techniques to work on tasks of machine learning models and advanced neural network models. GRL bridges the gap between graph theory and deep learning and helps in easily analyzing the graph data.

1.5.1 Definition of GRL

GRL is the process of converting graph-structured data into low-dimensional [3], continuous embeddings that preserve the graph's structural and feature information.

The flow of the GRL process starts with the input graph (Fig. 1.13), where nodes and edges are defined with data and its attributes. The input data may be noisy, so it needs to be cleaned. The preprocessing step contains a data-cleaning process along with feature engineering. Feature engineering enhances the node and edge representations by normalizing the node and edge features and helps in analyzing the graph connectivity. The next step is graph construction. In this, an adjacency matrix A is created, and the edge list creates pairs of edges with source, target nodes, and weights [2]. In graph sparsification, the less important edges are moved by preserving the overall structure.

GRL techniques embed graph data into a low-dimensional space by preserving the feature information. The techniques are DeepWalk, Node2Vec, graph neural networks (GNN), graph convolution networks (GCN), GraphSAGE, and graph attention networks (GAT). Dimensionality reduction and feature extraction are done during the learning process. Finally, the graph is used for various tasks like node classification, link prediction, and graph clustering.

1.5.2 Benefits of GRL

GRL has key benefits:

1) Helps in automatically learning features of nodes, edges, and graph structures.
2) Facilitates the identification and capture of structural relationships such as neighborhood relationships and community associations.

3) Helps in working with irregular, unstructured, and non-grid-structured graph data.
4) Handles efficiently large-scale graphs by reducing their complexity while maintaining essential information.
5) Preserves the structure and pattern of dynamically growing graphs.
6) Applies techniques to machine learning tasks such as link prediction, node classification, and clustering.

1.6 Overview of Graph Connectivity

Graph connectivity is essential for gathering information about the entities in the domain fields. It plays a vital role in determining the information flow between entities and finding the relationship between entities. In GRL, graph connectivity helps in learning or understanding the meaningful pattern from the entire graph structure.

1.6.1 Definition of Graph Connectivity

Graph connectivity refers to the extent to which edges connect the vertices (or nodes) in a graph. It highlights the strength to navigate between nodes within the graph and represents a core concept in graph theory.

Graph connectivity in an undirected graph examines whether a path exists between any two vertices/nodes. In a directed graph, connectivity considers the direction of edges.

1.6.2 Importance of Graph Connectivity

The main concept of graph connectivity in GRL revolves around how the structural and relational information in graphs can be captured and leveraged to learn meaningful representations for nodes, edges, or entire graphs. Connectivity helps as the foundation for defining the relationships and dependencies among graph components, directly influencing the learned embeddings and their effectiveness in downstream tasks. Graph connectivity is so important because it ensures that communication or flow is possible between nodes, even if some nodes/edges fail. Any fault or failure can be detected, as it may indicate malicious attacks and help in evaluating critical points or bottlenecks.

Example ***Author Citation Graph***
In an author citation graph, nodes represent authors and edges indicate citations between them. Graph connectivity is crucial for determining how research knowledge spreads across academic communities.

For example, suppose an author publishes an innovative paper. In that case, its impact is higher if the citation graph is well-connected, meaning researchers from

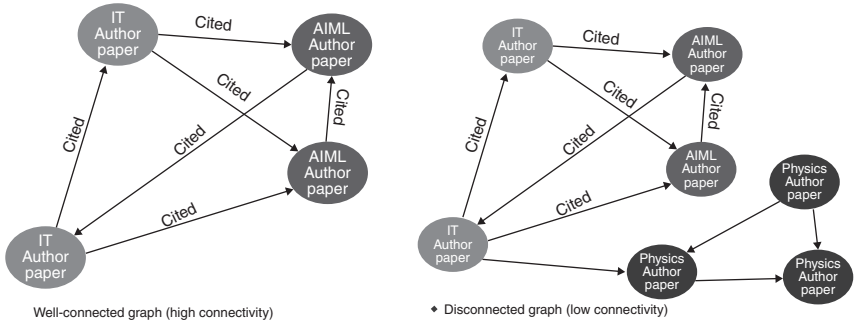

Figure 1.14 Well-connected and disconnected graphs.

different fields can reference and build upon the work. However, if the graph is disconnected, knowledge remains trapped within isolated groups, limiting interdisciplinary collaboration.

A well-connected citation network:

- Enable more sharing of research insights across domains.
- Enables authors to gain more recognition and have a greater impact.
- Allows better collaboration opportunities between research groups.

In a well-connected citation graph (high connectivity) as shown in Fig. 1.14, authors actively cite each other's work, ensuring seamless knowledge transfer across research communities. This fosters innovation, accelerates discovery, and enhances the impact of scientific contributions. For example, a researcher in IT citing work from Artificial Intelligence and Machine Learning can lead to AI-driven healthcare advancements, where deep learning techniques improve disease prediction models. Such interdisciplinary connections drive technological progress and enable novel applications.

Conversely, in a disconnected citation graph (low connectivity) as shown in Fig. 1.14, research communities remain isolated, limiting the exchange of ideas and slowing innovation. Citations are often confined within small, specialized groups, restricting the broader adoption of valuable insights. For instance, researchers in the physics stream may rarely cite work in IT or AIML, missing out on computational breakthroughs. This lack of connectivity reduces the potential for cross-pollination of ideas, delaying scientific progress and innovation.

1.6.3 Role of Graph Connectivity

Graph connectivity plays a vital role in various real-world applications, ensuring efficient communication, robustness, and network reliability. It provides network reliability even if some parts of the network fail. A well-connected internet backbone ensures users can access websites even if some servers go down. A strongly

connected graph provides strong connectivity between nodes with its neighborhood nodes and within the community network. Graph connectivity allows the transmission of data/information by optimizing information flow and communication within and outside the network. It ensures that the network can be secured from cyberattacks or any failure in the network. It is important to support efficient navigation. It plays an important role in finding the shortest route between locations using Google Maps.

Graph connectivity helps in identifying the identical patterns in the network, which facilitates customers' similar buying patterns, and friends in social media, and connects people around the world. Some of the applications where graph connectivity is so important are social media, finding the shortest route for traveling salesman problems, fraud detection, cybersecurity, stable infrastructure, banks to find the suspicious transaction patterns, and electrical connections to ensure good connectivity and identify power failures.

It helps in connecting the patients with the doctors, which will facilitate the immediate and proper treatment.

1.7 Foundation on Graph Neighborhood

1.7.1 Introduction

In GRL, a set of nodes that are directly or indirectly connected within a graph is referred to as neighborhood nodes. Consider a graph $G = (V,E)$, where $v_i \in V$ and $e_i \in E$. The neighborhood [2] is defined as $N(v_i)$ of a node v_i such that

$$N(v_i) = \left\{ v_j \in V \mid (v_i, v_j) \in E \text{ and } i \neq j \right.$$

where v_j is the adjacent node to v_i and (v_i, v_j) is the direct edge between v_i and v_j.

In Fig. 1.15, the dark black-color node at the center has direct neighborhood nodes, which are colored in dark gray shade and indirect neighborhood nodes, which are colored in light gray. Neighborhood nodes help in understanding local structures and connectivity patterns and influence propagation within a network. We can gain knowledge about the neighbor nodes and their relationship or association in various situations.

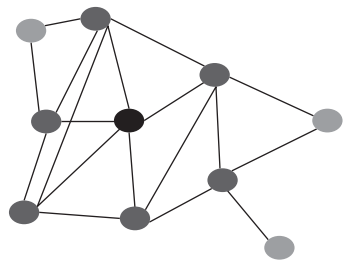

Figure 1.15 Illustration graph for direct neighborhood nodes.

1.7.2 Properties of Graph Neighborhoods

1.7.2.1 *Properties*

1) Most important characteristics such as the node's features, edge's features, and graph's structure are used for learning the insights obtained from the neighboring nodes.
2) Node degree helps to understand a node's influence on its neighbors and also to determine how much information it is gathering from its neighboring nodes. The interaction and influence of a node with its neighbor nodes are high for higher degree nodes and low for lower degree nodes.
3) We can measure how important a node is within a network.
4) Neighboring nodes show similar structural properties that can enhance representation learning.
5) Graph neighborhoods within two graphs are examined to identify similar communities, communication between the communities, and similar patterns.

Understanding the properties of neighborhood nodes is essential for designing efficient GRL models. These properties influence how information propagates, how embeddings are formed, and how well a model performs on graph-based tasks.

1.7.3 Types of Node/Graph Neighborhoods

1.7.3.1 Direct/Local Neighborhood (1-Hop Neighborhood)

Nodes that are directly connected using an edge are direct or local neighbors. They are immediate neighborhood nodes and are also called 1-hop neighborhood nodes. The above figure is an example of a direct neighborhood connection.

These types of nodes are used for identifying close friends in social media (e.g., Facebook, LinkedIn) and help in signifying products, jobs, and friends that can be directly connected in recommendation systems. It is also helpful for identifying fraud easily through direct interaction or connection with people, transactions, and so on. Direct neighbors are used to represent direct protein interactions in biological networks. In healthcare, patients having the same symptoms for a disease can be identified by direct connection and can be used for further analysis of the cause and significance.

1.7.3.2 K-Hop Neighborhood

K-hop neighborhood refers to the number of nodes that are reachable from a given node to more than one edge. "k" refers to the number of edges involved in the connection. The graph given in Fig. 1.16 shows a 2-hop and 3-hop neighborhood where two and three edges are involved in the node's neighborhood, respectively.

These types of nodes help determine how far the news spreads on social media and measure how far the product marketing reaches the people. This type is also

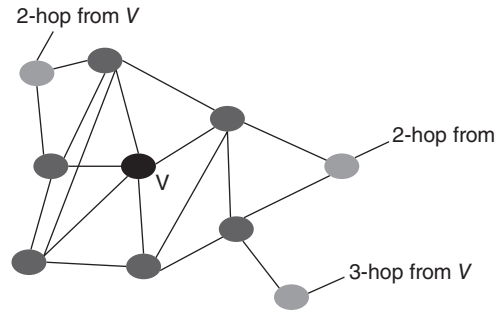

Figure 1.16 K-hop neighborhood example graph.

useful in GRL techniques for node embedding, and in healthcare, we can keep track of the spread of disease globally. We can also analyze the interconnection in telecommunication networks and routing networks.

1.7.3.3 Ego Neighborhood

Ego neighborhood is a subgraph with a central node (ego) and its 1-hop neighborhood nodes (alters) along with edges among them. The equation for the ego neighborhood of a node is

$$Ego(V) - \left\{N(V) \in E + U \in E'\right\}$$

where V is the central node (ego), $N(V)$ are the 1-hop neighbor nodes (alters), and U are the remaining nodes in edges E' other than the edges near V.

These types of nodes are used for community detection, anomaly detection by analyzing the subgraph structure, and user profiling.

1.7.3.4 Attention-Based Neighborhoods

In attention-based neighborhoods, nodes assign different importance (weights) to their neighbors during information aggregation. This approach is widely used in GATs to dynamically learn which neighbors contribute more to a node's representation.

In Fig. 1.17, node "c" is the central node, and the neighborhood nodes are assigned weights $\{w1, w2, w3, w4\}$. In attention-based neighborhood nodes, the most commonly used aggregation function is the attention aggregation function, often implemented using self-attention mechanisms [3]. This function assigns varying attention weights to each neighbor, allowing the model to prioritize more relevant or influential neighbors when aggregating their features.

These types of neighborhood nodes are used for assigning higher importance to the neighboring nodes.

1.7.3.5 Structural Neighborhood (Graphlets and Motifs)

Nodes that are connected in specific patterns are termed as motifs (recurrent subgraphs) or graphlets (small subgraphs).

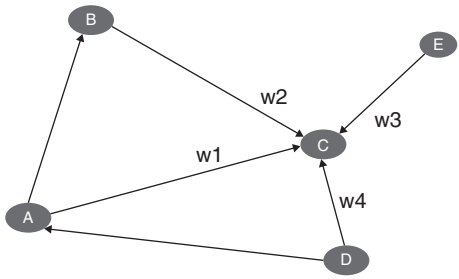

Figure 1.17 Graph used for graph attention networks.

Motifs The motifs help identify meaningful structural patterns in networks. Star motifs, triangle motifs, chain motifs, bipartite motifs, square motifs, and two-star motifs are different types of motifs.

In star motifs, a central node is connected to several other nodes. The other nodes are not connected. In triangle motifs, three nodes are connected one after the other. In chain motifs, nodes are arranged in a linear sequence, each node connected to two others, except for the end nodes. In bipartite motifs, the graph can be divided into two sets of nodes, with all edges connecting nodes from one set to the other and no edges within a set. In two-star motifs, there are two central nodes, each connected to a set of nodes forming two distinct stars. In square motifs, four nodes form a cycle with a diagonal edge connecting two nonadjacent nodes.

Let $G = (V, E)$ be a graph with a set of nodes $\{v1, v2, v3, v4, v5, v6\}$ in V and a set of edges in E (Fig. 1.18). The mathematical representation of all motifs is as follows:

Star motif:

$$\{(v1, v2), (v1, v3), (v1, v4), (v1, v5)\}$$

Triangle motifs:

$$\{(v1, v2), (v2, v3), (v3, v1)\}$$

Chain motifs:

$$\{(v1, v2), (v2, v3)\}$$

Square motifs:

$$\{(v1, v2), (v2, v3), (v3, v4), (v4, v1), (v1, v3)\}$$

Bipartite motifs:

$$\{(v1, v4), (v2, v5), (v3, v6)\}$$

Two-star motifs:

$$\{(v1, v3), (v1, v4), (v2, v5), (v2, v6)\}$$

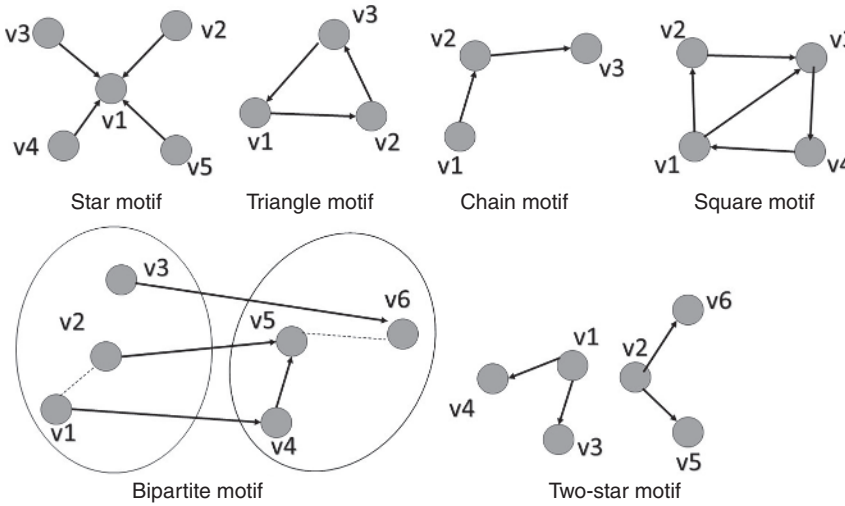

Figure 1.18 Different types of motifs.

1.7.3.6 Graphlets

Graphlets are small, connected, non-isomorphic subgraphs of a larger network that capture diverse topological structures without considering their frequency. A non-isomorphic subgraph has a unique structural arrangement that cannot be transformed into another subgraph through node relabeling within the same or a different graph, ensuring that each graphlet represents a distinct local pattern in the network. Graphlets are used to compare two networks and to find the structural patterns.

Figure 1.19 shows a graph with two-node graphlets with only one edge between the nodes, three-node graphlets in orange color, and four-node graphlets in green color. The number of *i*th node graphlets in the graph can be calculated using the

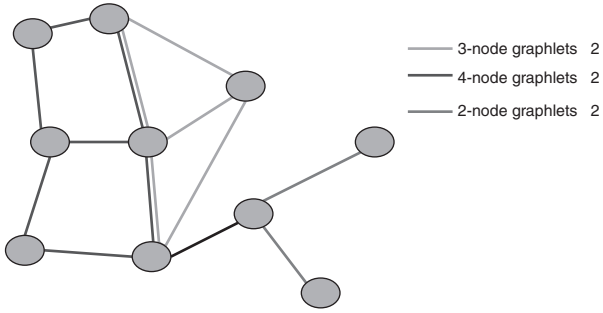

Figure 1.19 Graphlets example.

mathematical formula:

$$G_i = \Sigma i(V) \text{ where } i(V) \text{ is the count of } ith \text{ node graphlets.}$$

Importance of neighborhoods in graph theory Neighbor nodes play a crucial role in graph analysis, as they define the local structure and interactions within a network. Their importance includes the following:

1) It determines the flow of information from one node to another node. In social network examples, the users share their interests, ideas, or habits with other users.
2) It can influence the characteristics or features of nearby nodes. In the author citation graph example, one author's paper can influence the information or knowledge in another author's paper.
3) It can be used to perform various machine learning tasks such as node classification, clustering, and link prediction. In healthcare example, the patient nodes that are nearby have similar symptoms, and they can be grouped as affected patients and non-affected patients.
4) It helps understand the structure of the graph and do topology analysis.

Neighbor nodes play a fundamental role in graph analysis, as they shape the structure, connectivity, and functionality of a network. They are crucial in information propagation, where data, influence, or signals spread through neighboring nodes, enabling applications in social networks, telecommunications, and biological systems. The properties of a node's neighbors also determine its centrality and significance within the network, helping identify key influencers, hubs, or bottlenecks in traffic and communication networks. In GRL, neighborhood aggregation is essential for embedding techniques and GNNs, where a node's representation is learned from its neighbors, improving tasks such as node classification, anomaly detection, and clustering.

Also, neighborhood structures assist in community detection, as closely connected nodes tend to form dense subgroups, which is vital for understanding social structures, protein interactions, and financial fraud detection. Link prediction and recommendation systems also heavily rely on neighborhood information, as common neighbors increase the likelihood of future connections—this is widely applied in friend suggestion algorithms, e-commerce recommendations, and citation network analysis. Additionally, neighbor nodes are key in graph traversal algorithms such as Breadth First Search (BFS) and Depth First Search (DFS), which are used in pathfinding, web crawling, and network routing.

Overall, neighborhood nodes are integral to both theoretical and practical graph-based applications, influencing diverse domains such as biology, computer networks, finance, transportation, and artificial intelligence. Their study enables better insights into network behaviors, enhances predictive modeling, and improves decision-making in complex systems.

1.8 Applications of Graph

Graphs are powerful mathematical structures that capture relationships and interactions within complex systems. Nodes and edges of the graph represent data across diverse domains. From social networks to molecular biology, graphs enable the modeling of intricate dependencies that would be difficult to analyze using traditional methods.

Their applications range from optimizing traffic flow in smart cities and predicting molecular properties in drug discovery to detecting fraud in financial transactions and enhancing recommendation systems in e-commerce platforms. Their main purpose is for prediction, classification, detection, and learning. Following are some of the applications of graph.

1.8.1 Traffic Prediction in Transportation

Traffic prediction is a crucial application of GRL in transportation. Road networks can naturally be modeled as graphs, where intersections of roads are represented as nodes and roads as edges. Graph-based methods excel at capturing spatial dependencies in these networks, making them powerful for traffic forecasting.

Traffic prediction involves forecasting traffic conditions, including vehicle volume and travel times, for specific areas or roadways. This task is vital in optimizing transportation systems and easing traffic congestion. For example, estimating vehicle density, speed, or flow at specific road segments for the next few minutes or hours for traffic flow prediction. Graph-based models predict traffic flow by considering both temporal patterns and spatial dependencies.

1.8.2 Pattern Recognition

Graphs are used in pattern recognition extensively due to their ability to model complex relationships between objects, making them useful in various real-world applications. By representing data as graphs, pattern recognition systems can efficiently identify structures, similarities, and anomalies.

In image and video analysis, graphs are used to represent image structures, where pixels, regions, or objects are treated as nodes, and relationships between them as edges. Graph-based feature extraction and classification improve image segmentation, object recognition, and scene understanding. For example, face recognition can be done by extracting similar features. Signature verification systems use graph matching to detect forgery. Graph-based neural networks analyze time-series data in speech signals for pattern classification. For example, virtual assistants (e.g., Siri, Alexa, and Google Assistant) use graph-based representations to recognize and classify speech patterns. Graph-based models detect cancerous

patterns in medical imaging, such as MRIs and CT scans. For example, brain network analysis uses graphs to identify neurological disorders like Alzheimer's and epilepsy.

1.8.3 Social Network Analysis

Graphs play a crucial role in social network analysis (SNA) by modeling relationships between nodes such as individuals, groups, and organizations. Their interaction is represented using edges. Facebook and LinkedIn use graphs to identify friends and academicians doing research in a similar area. Twitter uses graphs to identify communities based on shared interests. Instagram, YouTube, and TikTok use graph-based influence analysis to recommend content and identify top influencers. Researchers who have a common interest can be linked together. Customer service platforms such as Amazon use feedback from customers and perform sentiment analysis and opinion mining using graph-based models. Twitter, Instagram, and Facebook can detect fake profiles, spam bots, and fraudulent actions by analyzing unusual network structures.

1.8.4 Application of Graphs in Biology

Graphs are fundamental in biology, visualizing and analyzing complicated relationships across diverse fields. They model food webs and species interactions in ecology, illustrating energy flow and habitat connections. In genomics, graphs depict gene regulatory networks, protein–protein interactions (PPIs), and metabolic pathways. Evolutionary biologists use phylogenetic trees to show species relationships, while neuroscientists map brain connectivity through neural networks. Graphs are also vital in epidemiology for modeling disease spread and contact tracing. Furthermore, they support computational biology in protein folding and bioinformatics algorithms, proving indispensable for comprehending biological systems and processes.

1.8.5 Application of Graphs in Chemistry

Graphs play a vital role in chemistry by modeling and analyzing complex chemical systems. In molecular structure representation, chemical graph theory visualizes atoms as nodes and bonds as edges, aiding in isomer identification and reaction pathway analysis. Reaction networks, including catalytic and metabolic pathways, are represented graphically to understand chemical transformations. In quantum chemistry, graphs depict molecular orbitals and calculate properties such as bond orders. Drug discovery leverages graphs for pharmacophore mapping, Quantitative Structure–Activity Relationship (QSAR) models, and molecular

docking. Crystallography and materials science use graphs to model crystal lattices, bonding patterns, and network structures. Chemical informatics employs graph-based methods for database searches and molecular fingerprinting. Spectroscopy utilizes graphs to analyze mass spectrometry fragmentation and Nuclear Magnetic Resonance (NMR) interactions effectively. Polymer chemistry uses graphs to study polymer networks and topologies, while environmental chemistry models pollutant transport and chemical interactions. Theoretical chemistry applies graphs in topology optimization and reaction dynamics, making them indispensable for bridging theoretical and experimental insights in chemistry.

1.9 Case Studies of GRL

In the previous section, we explored various applications of graphs, focusing on how traditional graph algorithms help in areas such as social networks, road navigation, and network security. While these applications rely on well-established techniques such as Dijkstra's algorithm, BFS, and PageRank, they often fall short in capturing the deeper, more complex relationships hidden within large-scale graphs. This is where GRL comes into play, transforming the way we analyze and utilize graph data.

Unlike traditional graph algorithms that operate on discrete structures, GRL enables us to learn low-dimensional vector embeddings for nodes, edges, or entire graphs. These embeddings preserve the structural and feature-based relationships within a graph, allowing us to apply machine learning models for tasks such as classification, clustering, and prediction. This approach has revolutionized various fields by enabling more scalable, efficient, and accurate graph-based applications.

One of the most prominent applications of GRL is in SNA, where platforms such as Facebook and LinkedIn use node embeddings to recommend friends and professional connections based on learned relationships. These methods go beyond simple mutual friends and leverage the entire network structure to suggest meaningful connections. Similarly, in financial fraud detection, GRL models help identify suspicious transactions by analyzing money flow between accounts and detecting anomalous subgraphs that indicate fraudulent behavior.

In the field of bioinformatics and drug discovery, GRL methods have made significant contributions by learning molecular representations. Instead of treating chemical compounds as strings, GRL converts molecules into graphs and learns representations that predict drug effectiveness, toxicity, and interactions. This has led to breakthroughs in AI-driven drug discovery, reducing the time and cost of finding new treatments.

Another important application lies in cybersecurity, where GRL models detect intrusions and anomalies in network traffic by learning network activity patterns.

Unlike traditional rule-based systems, which rely on predefined attack signatures, GRL-based methods can generalize to identify new and evolving threats.

These applications demonstrate the power of GRL in extracting hidden patterns and enabling data-driven predictions that were previously challenging using conventional graph algorithms. As graphs continue to grow in scale and complexity, GRL will play a pivotal role in advancing research and real-world applications across multiple domains.

1.9.1 Case Study in SNA

Social networks, such as Facebook, Twitter, and LinkedIn, consist of massive graphs where users (nodes) are connected by friendships, follows, or interactions (edges). GRL plays a crucial role in analyzing, predicting, and optimizing various aspects of social networks by capturing the hidden structure of user connections.

1.9.1.1 Friend and Connection Recommendations

Traditional recommendation systems use mutual friends or shared interests to suggest new connections. However, GRL-based models learn richer node embeddings that encode a user's entire network position. By comparing these embeddings, platforms such as LinkedIn and Facebook can recommend highly relevant connections even if users don't share direct mutual friends.

For example, a social network is represented as a graph, where nodes (V) represent users; edges (E) represent friendships or follow relationships; node attributes may include user interests, activity level, location, etc.; and edge attributes may include interaction frequency, message count, or strength of connection. The goal is to predict new edges (friendships) that should exist in the network based on graph embeddings. The benefits of GRL-based friend recommendations [4] are capturing indirect relationships beyond mutual friends, handling large-scale networks efficiently by learning compact embeddings, personalizing recommendations based on user behavior, not just structural links, and continuously updating as new connections and interactions occur.

1.9.1.2 Community Detection and Group Formation

Detecting communities in social networks helps platforms suggest groups, events, and discussions that match user interests. GRL models use learned node representations to identify clusters of users with similar engagement patterns. This improves user engagement by fostering more relevant social interactions.

Community detection is an essential problem in SNA, where the goal is to identify clusters of users who share similar interests, behaviors, or interactions. Traditional graph-based methods rely on network modularity and structural patterns, but GRL enhances this by learning rich, low-dimensional representations that capture hidden relationships and dynamic community structures.

Platforms such as Facebook, Twitter, and LinkedIn use community detection to recommend relevant groups, pages, and discussion forums, significantly enhancing user engagement through personalized social interactions. These social networks are modeled as a graph $G = (V, E)$, where nodes (V) are users and edges (E) represent connections (like friendships, follows, or interactions). User profiles provide node features (e.g., age, location, interests), while edge features can include message frequency or relationship duration.

A community is a subset of nodes highly connected internally but sparsely connected externally. Identifying these groups enables better recommendations, targeted advertisements, and network analysis. The benefits of GRL in community detection include capturing nonobvious connections beyond direct friendships. It handles large-scale and dynamic networks efficiently, identifies overlapping communities by allowing users to belong to multiple groups, and enhances personalized recommendations by improving user engagement.

1.9.1.3 Fake Account and Bot Detection

GRL helps in identifying fake accounts, bots, and fraudulent activities by learning network structures. Bots often exhibit unusual connection patterns, and GRL-based models can detect these anomalies by analyzing structural deviations. This enhances platform security and improves trust in social networks.

Fake accounts and bots are a major issue in social networks, affecting platform integrity, misinformation spread, and fraud. Traditional rule-based and heuristic methods struggle with large-scale bot networks due to their dynamic and developing approaches. GRL provides a powerful approach by using network structure, user interactions, and machine learning models to detect fake users more effectively.

For example, platforms such as Twitter, Facebook, and LinkedIn use GNNs, node embeddings, and anomaly detection algorithms to identify and eliminate suspicious accounts, ensuring a safer and more authentic user experience. Fake accounts and bots in social networks typically follow distinct behavioral patterns, which can be analyzed using graph-based techniques. These include automated bot networks where bots create fake engagement by liking, commenting, and retweeting posts. They follow each other in structured patterns, forming dense clusters. Furthermore, bots are used for spamming, political manipulation, and fake news promotion. Fake profiles send spam links, phishing messages, or impersonate real people.

Fake accounts and bots, typically, have low interaction diversity, mainly engaging with other fake accounts. A social network can be represented as a graph $G = (V, E)$ where nodes (V) represent users (real or fake); edges (E) represent interactions (follows, likes, comments, messages); node features include account age, number of posts, and engagement patterns; and edge features include interaction

frequency, sentiment, and reciprocity. The goal is to detect fake accounts, and its benefits are capturing hidden bot networks beyond simple rule-based detection by dynamically adapting to new bot behaviors using real-time updates, detecting fake accounts before they become active, and preventing large-scale manipulation. This works efficiently on large social networks (Twitter, Facebook, LinkedIn).

1.9.1.4 Sentiment and Opinion Analysis

By modeling social interactions as a graph, GRL enables sentiment analysis by learning how opinions propagate. This is useful for detecting political polarization, misinformation spread, and public sentiment trends, helping policymakers and organizations respond proactively.

Sentiment and opinion analysis play a crucial role in understanding public perception, analyzing trends, and detecting misinformation across social media, news platforms, and product reviews. Traditional natural language processing techniques use text-based sentiment analysis, but they fail to capture network-driven influences, opinion propagation, and contextual relationships between users.

GRL enhances sentiment analysis by having user interactions, discussion networks, and contextual dependencies to improve the accuracy of sentiment classification. Social networks (e.g., Twitter, Facebook, Reddit), e-commerce platforms (e.g., Amazon, Yelp), and news aggregators benefit significantly from graph-based sentiment analysis. A sentiment network is modeled as a graph $G = (V, E)$ where nodes (V) represent users, posts, or discussions; edges (E) represent interactions (likes, retweets, replies, comments); node attributes include sentiment scores, user credibility, and writing style; and edge attributes include interaction polarity (support vs. disagreement), frequency, and sentiment shift over time. The goal is to do sentiment and opinion analysis. The benefit of GRL in sentiment analysis is to capture sentiment propagation beyond isolated text analysis. It handles large-scale, dynamic discussions in real time, identifies opinion leaders and misinformation spreaders, and enhances personalized recommendations based on sentiment trends.

1.9.2 Case Study in Fraud Detection for Financial Transactions

In financial organizations, there were increasing cases of fraud involving online transactions. Traditional systems were not able to detect obvious fraud, such as large transactions from unusual locations or credit card fraud [5]. However, more subtle and well-organized fraudulent acts were going unnoticed.

To address this, the organizations decided to utilize GRL. They began by viewing all transactions and participants as a network of connected entities. Each person, account, transaction, device, and location was treated as a point (or "node") in the network, and the relationships between them, such as who used which device or which account sent money to whom, were considered as connections (or "edges").

By using GRL, the bank could automatically learn hidden patterns and relationships in this network. It became possible to identify suspicious links, like a single device being used by multiple unrelated accounts or accounts frequently connected to known fraudulent ones.

This graph-based view helped the team detect fraud not by looking at transactions individually, but by understanding how everything is connected. Even if a single transaction looked normal, its links to other suspicious nodes raised red flags.

As a result, the institution significantly improved its ability to spot fraud earlier and more accurately. GRL allowed them to see the bigger picture of how fraud spreads through complex connections.

1.9.3 Case Study in Molecular Biology

Case Study 1.1

Identifying Disease Genes Through PPI Networks Using GRL

Overview

In molecular biology, PPI networks represent the physical and functional interactions between proteins within a cell. These networks are modeled as graphs, where nodes represent proteins and edges denote interactions. Understanding these networks is critical for identifying key proteins (or genes) involved in diseases such as cancer, Alzheimer's, and genetic disorders.

Traditional methods to identify disease genes often depend on direct genetic mutations or expression studies, which may overlook important indirect interactions and network-level patterns. Moreover, the complexity and scale of PPI networks pose challenges for manual analysis or conventional statistical models. Researchers have developed graph-based machine learning models, particularly using GRL techniques such as GCNs and node embedding (e.g., node2vec, DeepWalk), to analyze PPI networks. These methods allow for automatic feature extraction from graph structure, enabling efficient and accurate prediction of potential disease-associated genes.

Workflow

Data Preparation:

The PPI network is constructed from databases such as BioGRID, STRING, or IntAct. Known disease genes are labeled using resources such as Online Mendelian Inheritance in Man (OMIM) or DisGeNET.

(Continued)

Case Study 1.1 (Continued)

Graph Embedding:

Each protein (node) is encoded into a low-dimensional vector based on its topological and relational properties using algorithms such as DeepWalk or GCNs.

Training and Prediction:

A machine learning model (e.g., logistic regression and neural network) is trained on the embedded features to classify proteins as disease-related or not.

Validation:

The model's predictions are validated against known disease genes and experimentally verified pathways.

Results

Studies show that GRL-based models significantly outperform traditional statistical or similarity-based methods. For example, in cancer genomics, GCNs have uncovered previously unknown oncogenes by leveraging indirect interactions that were overlooked in gene expression analysis. In neurobiology, GRL helped identify proteins implicated in neurodegenerative disorders through their network proximity to known disease genes.

Impact

This approach has revolutionized drug target discovery, biomarker identification, and personalized medicine. By uncovering hidden patterns in large-scale biological networks, graph-based models enhance our ability to understand complex disease mechanisms and identify novel therapeutic strategies.

Case Study 1.2

Identifying Disease Genes Through PPI Networks Using ModulePred and Discriminative Network Embedding

Overview

Identifying genes associated with specific diseases is a cornerstone of biomedical research. PPI networks offer an invaluable framework for this challenging task, as they inherently capture the complex web of relationships between proteins within a cell. Increasingly, GRL, particularly through GNNs, has emerged as a powerful technique to model the intricate topologies of these PPI networks, paving the way for the discovery of novel gene–disease associations.

One significant innovation in this space is ModulePred [7], a deep learning framework designed to predict disease–gene associations. As described in studies like those found on BioMed Central and PubMed Central (PMC), ModulePred masterfully integrates diverse biological information, including known disease–gene links, identified protein complexes, and augmented protein interactions. It does this by constructing a heterogeneous module network. Through sophisticated graph augmentation and embedding techniques, ModulePred learns robust representations for each node (protein or gene) within this network. This capability allows it to accurately predict disease–gene associations, with experimental results consistently demonstrating its superiority over traditional methods. This underscores the profound efficacy of leveraging functional modules and graph augmentation in disease gene prediction.

Another noteworthy advancement is the discriminative network embedding (DNE) framework, detailed in a study published in *Science*. DNE employs a self-supervised network embedding approach that excels at capturing both the localized and global topological information present in PPI networks. What makes DNE particularly impactful is its ability to learn robust node embeddings without relying on extensive labeled data. This is a critical advantage in biological research, where comprehensive annotations are often scarce, making DNE highly valuable for enhancing the identification of disease genes even in data-limited scenarios.

These breakthroughs in GRL methodologies highlight the immense potential of leveraging PPI networks for identifying disease genes. By effectively capturing intricate biological interactions and seamlessly integrating diverse data sources, GRL approaches such as ModulePred and DNE are fundamentally reshaping our understanding of disease mechanisms, leading to more accurate and comprehensive insights.

1.10 Conclusion

Graphs serve as a powerful and intuitive way to represent complex relationships and interactions in real-world systems. From the fundamental understanding of graph types—such as directed, undirected, weighted, and hierarchical graphs—to the exploration of connectivity and structure, this chapter has laid the foundation for understanding how data can be structured beyond traditional formats.

GRL has emerged as a transformative approach, enabling machines to learn meaningful representations from graph-structured data. The benefits of GRL such as capturing relational patterns and enabling predictive insights are evident in

various domains, including social networks, transportation systems, biology, and financial fraud detection. The case studies presented demonstrate the versatility and real-world impact of GRL in solving complex problems.

As data continues to grow in complexity and interconnectivity, the role of graph theory and representation learning will become increasingly vital. By equipping readers with foundational concepts and practical insights, this chapter sets the stage for deeper exploration into graph-based methods in artificial intelligence and data science.

References

1 R. Kaur, R. Saini, and Sanjivani, "Introduction to graph theory and its types," *J. Emerg. Technol. Innovat. Res. (JETIR)*, vol. 6, no. 4, 2019 ISSN-2349-5162, www.jetir.org.

2 V. T. Hoang, H.-J. Jeon, E.-S. You, Y. Yoon, S. Jung, and O.-J. Lee, "Graph Representation Learning and Its Applications: A Survey," *Sensors*, vol. 23, no. 8, p. 4168, 2023, doi: 10.3390/s23084168.

3 W. L. Hamilton, R. Ying, and J. Leskovec, *Representation Learning on Graphs: Methods and Applications*. IEEE Computer Society Technical Committee on Data Engineering, 2017. https://www-cs.stanford.edu/people/jure/pubs/graphrepresentation-ieee17.pdf.

4 W. Tayyab, "Graph representation learning using AI, ML, methods and applications," *Int. J. Adv. Eng. Technol. Innov*, vol. 1, no. 01, 2021.

5 Yao Zou, Dawei Cheng, Effective high-order graph representation learning for credit card fraud detection, Proc. Thirty-Third Int. Joint Conf. Artificial Intelligence (IJCAI-24), Jeju 03.08.24–09.08.24, pp. 7581–7589, https://www.ijcai.org/proceedings/2024/0839.pdf.

6 J. Gao, J. Wu, and J. Ding, "Heterogeneous Graph Condensation," *IEEE Trans. Knowl. Data Eng*, vol. 36, no. 7, pp. 3126–3138, 2024, doi: 10.1109/TKDE.2024.3362863.

7 X. Jia *et al.*, "A deep learning framework for predicting disease-gene associations with functional modules and graph augmentation," *BMC Bioinformatics*, vol. 25, p. 214, 2024.