Complex Graph Analysis and Representation Learning: Problems, Techniques, and Applications

Xinjun Pei, Xiaoheng Deng, *Senior Member, IEEE,* Neal N. Xiong, *Senior Member, IEEE,* Shahid Mumtaz, *Senior Member, IEEE,* and Jie Wu *Fellow, IEEE*

***Abstract*—Graph representation learning (GRL) has become a new learning paradigm, supporting a wide range of tasks such as node classification, link prediction, and graph classification. However, the effectiveness of graph analysis heavily depends on the quality of data representation. While existing GRL methods have made significant progress in learning from simple graphs, addressing the challenges posed by complex graph structures remains an active area of research. In many real-world scenarios, graph data usually exhibits characteristics such as complexity, heterogeneity, and dynamicity, where objects and their interactions may be multi-type, multi-modal, and even multi-dimensional, posing challenges to graph-related analysis. To tackle these challenges, GRL has been developed and widely used to model more complex and powerful graphs. In this survey, we provide a comprehensive and structured analysis of the existing literature on GRL from two clear points of view of simple graph and complex graph. We begin by providing a detailed and thorough analysis of state-of-the-art GRL techniques and classify them according to their underlying learning mechanisms. Furthermore, we systematically investigate GRL from the perspective of complex graphs to address the challenges posed by graph complexity. We emphasize the need for specialized GNN models that can handle the complexity of such systems. Finally, we highlight several promising directions for future research.**

***Index Terms*—Graph Representation Learning, Graph Neural Networks, Heterogeneous Graph, Multi-dimensional Graph, Signed Graph, Hyper Graph, Dynamic Graph.**

I. Introduction

In recent years, Graph Representation Learning (GRL) has received significant research attention from academia and industry, due to the ubiquity of graphs in a large spectrum of real-world applications, ranging from citation graphs [1, 2], social graphs [3–6] to recommendation systems [7, 8]. The analysis of information graphs heavily depends on how the graphs are represented [9, 10], which involves modeling the underlying graph's vertex attributes and the essential and relevant relations among vertices. More specifically, GRL can encode a variety of graph semantic and topology structure information [11–13]. Recent GRL methods use state-of-the-art deep graph neural networks to learn latent graph semantic and topology structure information. For example, in traffic networks, real-time traffic conditions can be predicted by comprehensively analyzing the spatio-temporal correlations of traffic flow.

To extract useful information from graph data, early graph analysis methods used graph embedding methods to project the graph into a low-dimensional vector space to create new features for dimensionality reduction, while preserving the essential characteristics of the original data. This makes the original graph more tractable. Subsequently, traditional vector-based machine learning methods can easily complete the graph analysis tasks. Although the low-dimensional vector representations obtained in this way make the graph learning models or algorithms easier to extract useful information, such methods usually suffer high computation and space overhead. To alleviate this issue, graph neural network (GNN) algorithms [14] have attracted recent attention to automatically capture high-level vertex representations and graph topology information from the given original low-level graph-structured data [15]. Unlike traditional graph embedding methods, GNNs operate directly on the graph structure, allowing for the seamless integration of topological information into the learning process. This enables GNNs to learn more useful information while reducing the computational and space overhead associated with traditional graph embedding techniques.

In many real-world scenarios, the application of GNN-based GRL faces a more complicated situation, because the graphs can be more complicated, e.g., heterogeneous graphs [1, 2], multi-dimensional graphs [4], signed graphs [3, 5], hyper graphs [7, 8], dynamic graphs [6], etc. However, most existing GNN-based GRL methods focus on learning simple or homogeneous graphs containing only one type of node and edge, which may be difficult to adapt to the various characteristics of complex graph structures. Understanding these complex structures has required not just new graph models but also novel analytical tools. As a result, there is a growing need for GRL methods that can handle the complexity of real-world graphs, facilitating more accurate and robust graph representations. This evolution signifies a paradigm shift in how we model graphs, advancing from static and simple representations to dynamic and complex models that reflect real-world systems more accurately. Future GRL methods should be capable of capturing information

Corresponding author: Xiaoheng Deng.

Xinjun Pei and Xiaoheng Deng are with the School of Computer Science and Engineering, Central South University, changsha, hunan, 410083, China (e-mail: pei_xinjun@163.com, dxh@csu.edu.cn).

Neal N. Xiong is with the Department of Computer Science, Colorado Technical University, CO 4348, USA (e-mail:xiongnaixue@gmail.com).

Shahid Mumtaz is with the Department of Applied Informatics, Silesian University of Technology, Akademicka 16 44-100 Gliwice, Poland, and also with the Department of Computer Sciences, Nottingham Trent University, Nottingham, U.K.(e-mail:dr.shahid.mumtaz@ieee.org).

Jie Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122 USA (e-mail:jiewu@temple.edu).

from various entity types, effectively modeling higher-level structural relationships, and dynamically adjusting to changes in graph topology over time. The survey aims to explore how the GRL methodologies address the challenges posed by the evolution of graph types and their inherent complexity.

A. Challenges

Due to the complexity of the graph, the research field of GRL presents some unique challenges for researchers. This complexity arises from the diversity of graph types. There is a growing need to develop GRL methods that are specifically designed to deal with the complexity of heterogeneous, multi-dimensional, signed, hyper, and dynamic graphs. In this survey, we identify and explore five key challenges in applying GNN models to these complex graphs. Each challenge highlights a specific aspect of graph complexity that GNN models must evolve to handle.

1) Heterogeneity in Graphs: Real-world graphs often exhibit heterogeneity, with multiple types of nodes and edges, each representing diverse entities and relationships. Traditional GNN methods may struggle to effectively capture and distinguish these diverse entity types and relationships.

2) Modeling Multi-Dimensional Structures: In scenarios where graphs exhibit multi-dimensional structures, GRL methods need to learn not only from each layer's unique features but also from the inter-layer dependencies. The interaction between different types of relationships can have a significant impact on the learning process of GNN.

3) Polarity Recognition in Graphs: In signed graphs, relationships are characterized by positive or negative edges, such as trust/mistrust in social graphs. Standard GNNs lack mechanisms to distinguish between these polarities, which are critical for understanding the complicated social dynamics. The sign-aware process enables GNNs to identify the sentiment or quality of the relationships in a graph, which can significantly affect the propagation of information.

4) Higher-Level Structural Relationships: In hyper graphs, relationships extend beyond dyadic interactions to encompass connections among multiple entities simultaneously, which may not be adequately captured by traditional GRL approaches. These higher-level structural relationships could be essential for understanding the underlying patterns and dynamics of the graph.

5) Adapting to Graph Evolution: In scenarios where graphs undergo temporal changes in their topology, with nodes and edges continuously evolving over time, the challenge for traditional GRL methods is to encapsulate not just the structural but also the temporal dynamics of these graphs.

B. Our Contributions

Tremendous efforts have been made to address these challenges posed by complex graphs. The adopted learning models and training strategies also vary greatly, covering a wide range of domains from homogeneous graphs to heterogeneous graphs, from single-dimensional graphs to multi-dimensional graphs, from unsigned graphs to signed graphs, from pairwise graphs to hyper graphs, and from static graphs to dynamic graphs. However, little effort has been made to systematically summarize the differences among these diverse complex graph architectures. This survey aims to bridge the knowledge gap by providing a comprehensive review of GNN methods, focusing on their application to both simple and complex graphs. We explore the fundamentals of GNNs, elucidating their various design architectures, training strategies, and applications in both simple and complex graphs. In addition, we track the evolution of GNNs from their basic applications to more sophisticated uses in complex graph scenarios and highlight key milestones and breakthroughs. Our goal is to provide researchers with a thorough understanding of GRL techniques and their applications in various domains.

- We fill the knowledge gap by conducting a comprehensive review of GNN methods that are applicable to both simple and complex graphs.
- We provide a detailed and thorough analysis of state-of-the-art GRL algorithms and develop a unified conceptual framework to emphasize and bridge the conceptual differences between various GRL algorithms.
- We discuss the challenges faced by different types of complex graphs such as heterogeneous graphs and dynamic graphs, and explore the applicability and transformativeness of GRL techniques in these complex graphs.

C. Related Surveys

An early version of the Graph Representation Learning (GRL) survey was proposed by Luis et al,. [16]. They provided several key concepts of GRL, such as graph embedding and related deep learning architectures, and summarized some representative GRL technologies. There are several surveys [13, 17] that cover graph embedding [11, 12] and GRL [16] methods and summarize their applications in graph analysis from different perspectives. For example, Zhang et al. [18] provided a detailed review of five variant architectures of GNNs, elucidating their respective model structures and highlighting the differences between them. Moreover, several attempts have been made to survey heterogeneous graphs [19], large-scale graph processing systems [20, 21], or knowledge graphs [22, 23]. Wang et al. [19] have made a significant contribution to the field by providing a comprehensive overview of heterogeneous graphs. Their work provides insights into tools and frameworks for efficiently processing and analyzing heterogeneous graph data, addressing the unique challenges associated with such graphs. McCune et al. [20] provided a comprehensive overview of large-scale graph processing systems. They introduced the innovative concept of "think like a vertex" (TLAV), in which user-defined programs are implemented from the viewpoint of individual vertices rather than the entire graph. This paradigm shift offers new opportunities for efficient and scalable graph processing. Additionally, Vatter et al. [21] summarized several distributed systems for large-scale GNN models, emphasizing the importance of scalable and distributed approaches for training and deploying GNN models in real-world settings. Furthermore, recent research [22, 23] efforts have focused on techniques related to knowledge graph construction, aiming

to enhance the quality and utility of knowledge graphs in various applications. These studies facilitate more accurate and comprehensive representations of structured knowledge. However, these methods have neither attempted to specifically discuss other complex graphs faced in real-world scenarios, nor provide a broader view for analyzing other complex graphs, such as multi-dimensional graphs, signed graphs, hyper graphs, and dynamic graphs. There is a lack of summary on in-depth analysis of the state-of-the-art GRL techniques for handling more complex graphs. Unlike previous surveys, which may focus on a specific branch or aspect of GRL, our survey covers a wide range of GRL technologies and their applicability to various types of graphs. Specifically, we provide the most comprehensive survey from the perspective of complex graphs. We believe that our unified framework is timely and necessary, which provides some insights into the GRL research, and inspires more studies on complex graphs that are ubiquitous in the real world.

D. Organization of the Survey

The rest of the paper is organized as follows. We first introduce the formal definition of GRL as well as the related concepts in Section II. Section III provides a detailed and comprehensive study of the state-of-the-art GRL algorithms from the perspective of simple graphs. Section IV comprehensively presents an overview of the recent development on GRL techniques for more complex graph representations. Section V provides experimental evidence of the effectiveness of the GRL methods. In Section VI, we highlight some promising directions for future research. Finally, Section VII concludes the paper.

II. Preliminaries

A. Problem Definitions

In a graph, vertices can represent real-world objects or concepts, and edges can represent their relationships. We give the formal definition of the graph representation learning as follows.

Definition 2.1 (Graph Representation). *Given a graph* $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{A}, \mathbf{X})$*, where* $\mathbf{V} = \{v_1, ..., v_n\}$ *denotes the set of vertices, and* $\mathbf{E} = \{e_{ij}\}_{i,j=1}^{n}$ *denotes the set of edges.* $\mathbf{A} \in \mathbb{R}^{|V|\times|V|}$ *represents the adjacency matrix, where* $|V|$ *indicates the number of vertices.* $\mathbf{X} \in \mathbb{R}^{|V|\times d}$ *represents the attribute matrix, where* d *indicates the number of attributes. Each vertex* $v_i \in \mathbf{V}$ *has a d-dimensional vector* $\mathbf{x}_i = \{x_{i,1}, ..., x_{i,d}\}$*. This is a general notation for unweighted/weighted graph or undirected/directed graph.*

B. Generic Paradigm

Graph is a ubiquitous data structure employed in a variety of applications across many disciplines, such as social graphs, recommender systems, and biological graphs. Data representation plays a crucial role in graph analysis, which is useful for learning structured and relational knowledge. Generally speaking, traditional machine learning methods usually encode graph-structured data into feature vectors. In this case, graph topological dependency may be lost. To address these challenges, graph neural network (GNN) has been proposed, which can be considered as an extension of random walk models [24, 25]. The success of GNN is based on the neighborhood aggregation mechanism, in which the vertices update their states and exchange information according to the topological relationships among the vertices. Inspired by [26], we provide a canonical and ubiquitous paradigm for graph modeling and learning, covering most well-known GNN algorithms. Let G be a set of graphs, and $\mathcal{N}$ be a subset of their vertices. Then, we have that:

$$\begin{aligned}\mathcal{D} = \{(\boldsymbol{G}_i, v_{i,j}, \boldsymbol{l}_{i,j}) \mid, \boldsymbol{G}_i = (\boldsymbol{V}_i, \boldsymbol{E}_i) \in \mathcal{G}; v_{i,j} \in \boldsymbol{V}_i; \\ \boldsymbol{V}_i \in \mathcal{N}; \boldsymbol{y}_{i,j} \in \mathbb{R}^m, 1 \le i \le |\mathcal{G}|, 1 \le j \le s_i\}\end{aligned} \quad (1)$$

where $v_{i,j}$ denotes the j-th vertex in $\boldsymbol{V}_i$, $\boldsymbol{y}_{i,j}$ is the label associated to $v_{i,j}$, and s_i denotes the number of supervised vertices in $\boldsymbol{G}_i$. Next, we denote a local transition function ϕ_w that captures the dependency between a vertex and its adjacent neighbors, and a local output function $\psi_{\boldsymbol{w}}$ that produces the output. Then, we have $\boldsymbol{o}_v = \psi_{\boldsymbol{w}}(\boldsymbol{h}_v, \boldsymbol{f}_v)$, where $\boldsymbol{h}_v = \phi_{\boldsymbol{w}}\left(\boldsymbol{f}_v, \boldsymbol{f}_{\mathrm{ne}[v]}, \boldsymbol{x}_{\mathrm{ne}[v]}, \boldsymbol{f}_{\mathrm{co}[v]}\right)$. $\boldsymbol{h}_v \in \mathbb{R}^d$ and $\boldsymbol{o}_v \in \mathbb{R}^d$ denotes the state and output of v, respectively. $\boldsymbol{f}_v$ denotes the feature of v, $\boldsymbol{f}_{\mathrm{ne}[v]}$ denotes the features of its neighbors, $\boldsymbol{x}_{\mathrm{ne}[v]}$ denotes the states, and $\boldsymbol{f}_{\mathrm{co}[v]}$ denotes the features of its edges.

Let $\boldsymbol{h}, \boldsymbol{o}, \boldsymbol{f}$, and $\boldsymbol{f}_{\boldsymbol{V}}$ be the vectors constructed by stacking all of the states, outputs, features, and vertex features, respectively. Then, we have $\boldsymbol{h} = \Phi_{\boldsymbol{w}}(\boldsymbol{f}, \boldsymbol{x})$ and $\boldsymbol{o} = \Psi_{\boldsymbol{w}}(\boldsymbol{h}, \boldsymbol{f}_{\boldsymbol{V}})$, where $\Phi_{\boldsymbol{w}}$ and $\Psi_{\boldsymbol{w}}$ are the global transition function and the global output function, respectively. Finally, the loss function can be minimized by $L_{\boldsymbol{w}} = \sum_{i=1}^{|\mathcal{G}|}\sum_{j=1}^{s_i}\left(\boldsymbol{y}_{i,j} - \varphi_{\boldsymbol{w}}(\boldsymbol{G}_i, v_{i,j})\right)^2$.

III. Simple Graph Representation Learning

Given the success of deep learning, increasing efforts have been made to generalize deep learning to graph. Specifically, deep graph neural networks gained enormous popularity in graph representation learning. This encourages us to dedicate more efforts to this research area. In this section, we introduce the graph neural networks for graph-structured data, and group them into two types: 1) the spectral domain techniques, which filter certain frequencies in the graph signal; and 2) the spatial domain techniques, which aims to utilize the graph structure. We review representative algorithms for each group.

A. Spectral Method

Spectral GNN was first proposed by [27], which relies on the spectral analysis theory. In this work, Bruna et al. [27] generalize CNN to signals, which performs localized filtering on the spatial and spectral domains. Moreover, Henaff et al. [28] develop an extension of spectral graphs. However, the matrix multiplication involved in Graph Fourier transform (GFT) often leads to demandingly high computational costs. To deal with this challenge, Defferrard et al. [29] propose a Chebyshev expansion of the graph Laplacian [30], and provide a strict control over the local support of filters.

Moreover, Kipf et al. [31] simplify the Chebyshev expansion to alleviate the problem of overfitting. Li et al. [32] propose a deep learning framework DCRNN to model the traffic flow, which incorporates the spatial and temporal dependency information. Besides, some progress is made on graph parallelization by modificating filter functions in the spectral domain, such as CayleyNet [29], AGCN [33], DualGCN [34], GWNN [35], MSSGs [36], PFME [37], and WGGP [38]. These studies generalize the convolutional graph to the signals through the GFT technology. Following the work [39], we give a definition of the spectral method as follows.

Definition 3.1 (Spectral Method) *Let $\mathbf{X} \in \mathbb{R}^N$ be a signal on a graph $\mathbf{G}$. The Graph Fourier transform (GFT) is used to obtain the graph Fourier coefficients $\hat{\mathbf{X}}$ of the signal $\mathbf{X}$, which can be defined as $\hat{\mathbf{X}} = \mathbf{U}^\top \mathbf{X} \in \mathbb{R}^{N \times N}$. Then, the graph convolution operation can be defined in the Fourier domain [39] such that $f_1 * f_2 = \mathbf{U}\left[\left(\mathbf{U}^\top f_1\right) \odot \left(\mathbf{U}^\top f_2\right)\right]$, where f_1 and f_2 are two signals defined on vertices, and $\odot$ represents the element-wise product. Specifically, the vertex signal f_2 is set to $\mathbf{X}$, which can be filtered by the spectral signal $\hat{f}_1 = \mathbf{U}^\top f_1 = \mathbf{g}$. Then, we have:*

$$\mathbf{Z} = \mathbf{g}(\tilde{\mathbf{L}})\mathbf{X} = \mathbf{U}\left[\mathbf{g}(\mathbf{\Lambda}) \odot \left(\mathbf{U}^\top \mathbf{X}\right)\right] = \mathbf{U}\mathbf{g}(\mathbf{\Lambda})\mathbf{U}^\top \mathbf{X} \quad (2)$$

where $\mathbf{Z}$ is the updated vertex representations, $\mathbf{\Lambda}$ is the diagonal matrix, and $\mathbf{g}$ is known as frequency response function. The goal is to learn a function $\mathbf{g}(\cdot)$.

Moreover, the spectral method can be further categorized into three subcategories below.

(1) Linear Approximation (LA): LA aims to adjust weights on frequency components during aggregation [31, 39, 40].

$$\mathbf{Z} = \left(\sum_{i=0}^{l} \theta_i \lambda_i \mathbf{u}_i \mathbf{u}_i^\top\right)\mathbf{X} = \mathbf{U}\mathbf{g}_\theta(\mathbf{\Lambda})\mathbf{U}^\top \mathbf{X} \quad (3)$$

where θ is parameters, λ gathers the lowest frequency components, and $\mathbf{u}$ is the eigenvector.

(2) Polynomial Approximation (PA): When more higher order eigenvalues are added [29, 41], LA is extended to PA. Then, the polynomial approximation of the frequency response function [39] can be written as:

$$\mathbf{Z} = \left(\sum_{i=0}^{l}\sum_{j=0}^{k} \theta_j \lambda_i^j \mathbf{u}_i \mathbf{u}^\top{}_i\right)\mathbf{X} = \mathbf{U}\mathbf{P}_\theta(\mathbf{\Lambda})\mathbf{U}^\top \mathbf{X} \quad (4)$$

where $\mathbf{g}_\theta(\mathbf{\Lambda}) = \mathbf{P}_\theta(\mathbf{\Lambda})$ is a polynomial function of eigenvalues.

(3) Rational Approximation (RA): RA approximates frequency response with rational function [39], which can deal with non-smooth signals. Formally, RA can be written as:

$$\mathbf{Z} = \left(\sum_{i}^{l} \frac{\sum_{j=0}^{k} \theta_j \lambda_i^j}{\sum_{m=1}^{n} \phi_m \lambda_i^m + 1} \mathbf{u}_i \mathbf{u}_i^\top\right)\mathbf{X} = \mathbf{U}\frac{\mathbf{P}_\theta(\mathbf{\Lambda})}{\mathbf{Q}_\phi(\mathbf{\Lambda})}\mathbf{U}^\top \mathbf{X} \quad (5)$$

where $\mathbf{P} and \mathbf{Q}$ are two independent polynomial functions, and $\mathbf{g}(\cdot) = \frac{\mathbf{P}_\theta(\cdot)}{\mathbf{Q}_\phi(\cdot)}$ is a rational function.

B. Spatial Method

In addition to the above spectral domain methods, there are some spatial methods that aggregate graph signals within the vertex neighborhood by performing convolution on graph-structured data. DCNN [42] introduces a diffusion-convolution operation to learn the latent representation for graph-structured data. PATCHY-SAN [43] directly performs convolution on locally connected regions of the input graph, where each region contains a fixed-length ordered sequence of vertices. Moreover, DGCNN [44] develops a SortPooling layer to sort the vertex features. Similar to DGCNN, LGCN [45] transforms graph-structured data into grid-like structures by selecting a fixed number of neighboring vertices. PGC-DGCNN [46] gives a new definition of graph convolutional filter. In this work, the 1-D convolutions are regarded as special cases of graph convolutions. In [47], the author proposed a LightGCN that aims to simplify GCN. Recently, a serises of method based on neighborhood aggregation has been proposed [48–51]. For example, GraphSAGE [52] learns the topological structure information and the distribution of vertex features in a inductive way. Xu et al. [40] propose a theoretical framework GIN to analyze the expressive power of GNN. They emphasize that GIN is as powerful as the Weisfeiler Lehman (WL) test. Besides, some progress is made on spatial-based GNNs, such as ClusterGCN [53], KCGN [54], Neo-GNNs [48], GNN-Retro [49], MST-GNN [50], and GHNN [51]. We now give a definition of the spatial method as follows.

Definition 3.2 (Spatial Method) *Given a graph $\mathbf{G}$ with vertex features $\mathbf{X}$, the vertex representations $\mathbf{Z}$ can be updated by:*

$$\mathbf{Z} = f(\mathcal{G})\mathbf{X} \quad (6)$$

where $f(\cdot)$ is a vertex aggregation function that learns how to aggregate vertex features.

Based on the vertex aggregation, spatial-based GNNs can be categorized into three subcategories.

(1) Linear Propagation (LP): LP aims to learn the weights for the vertex and its first-order neighbors [31, 39, 40, 53, 54]. The general form of the linear function of LP can be defined as follows.

$$\mathbf{Z} = (\phi\mathbf{I} + \psi\tilde{\mathbf{A}})\mathbf{X} \quad (7)$$

where ϕ and ψ are the weights, and $\tilde{\mathbf{A}}$ indicates the normalized $\mathbf{A}$.

(2) Polynomial Propagation (PP): Many works [42] [55] [56] involve high-order neighbors. When more neighbors of a higher order are added, LP is extended to PP which collects richer local structure. Formally, PP can be represented as:

$$\mathbf{Z} = \left(\phi\mathbf{I} + \sum_{j=1}^{k} \psi_i \left(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\right)^j\right)\mathbf{X} = \mathbf{P}(\tilde{\mathbf{A}})\mathbf{X} \quad (8)$$

where $\mathbf{P}(\cdot)$ is a polynomial function.

(3) Rational Propagation (RP): In the case of GNNs, each adjacent pair of vertices exchange information with each other via a message-passing mechanism. When the order number is large, this may cause over-smoothing issues. To cope with this

challenge, some recent works [57, 58] consider the reverse propagation method that applies a rational function on the adjacency matrix. Formally, RP can be represented as:

$$\mathbf{Z} = \mathbf{P}(\tilde{\mathbf{A}})\mathbf{Q}(\tilde{\mathbf{A}})^{-1}\mathbf{X} = \frac{\mathbf{P}(\tilde{\mathbf{A}})}{\mathbf{Q}(\tilde{\mathbf{A}})}\mathbf{X} \tag{9}$$

where $\mathbf{P}$ and $\mathbf{Q}$ are two independent polynomial functions.

C. Bridging Spectral and Spatial GNNs

Although the above two methods have been a great success, there is a lack of comprehension of the representational capabilities of GNN. To deal with this problem, some works [26, 39, 59] tried to establish a general framework to bridge the gap between spectral and spatial domains for GNN. Spectral methods require Fourier transform and inverse Fourier transform. In many cases, this leads to high computational costs compared to spatial methods. Designing new graph signal filters (e.g., CheyNet [41] and CayleyNet [29]) can alleviate this limitation. However, most spectral methods are only applicable to a single graph structure (i.e., node-level and edge-level tasks), because the weights cannot be shared among graphs with different structures, even if the graphs have the same size. This limits the application of the spectral-based GNNs at graph-level tasks. Instead of spectral-based graph convolutions, spatial methods perform information propagation by the convolution opearations in a batch of vertices. Therefore, it can be easily generalized to new graphs (e.g., GraphSAGE [52]), while the spectral methods cannot. It should be noted that the graph convolutions defined in the spatial domain are motivated directly by the spatial relationship between vertices and other vertices, which is similar to the traditional CNN on images. By incorporating graph inputs into aggregation functions, spatial-based GNN can flexibly handle multi-source graphs, such as signed graphs and heterogeneous graphs.

In short, the spatial methods are more popular than the spectral methods due to its greater efficiency, generality and flexibility. Formally, we define a general framework for information propagation on graph as belows.

$$\mathbf{H}^{(l+1)} = \sigma\left(\sum_s \mathbf{C}^{(s)}\mathbf{H}^{(l)}\mathbf{W}^{(l,s)}\right), \tag{10}$$

where $\mathbf{C}^{(s)}$ represents $s\text{-}th$ graph convolutions at $l\text{-}th$ hidden layer $\mathbf{H}^{(l)}$, which determines how the vertex aggragates information from its neibors. Specifally, at $0\text{-}th$ layer, $\mathbf{H}^{(0)}$ is equal to $\mathbf{X}$. Eq. 10 generalizes a variety of GNNs by supporting different graph convolutions $\mathbf{C}$. For spatial method, we extend Eq. 6 as $\mathbf{H}^{(l+1)} = \sigma\left(\mathbf{Z}^{(l)}\mathbf{W}^{(l)}\right) = \sigma\left(f(\mathcal{G})^{(l)}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right)$, where $\mathbf{f}(\mathcal{G}) = \sum_s C^{(s)}$. For spectral method, Eq. 2 can be regarded as a special case of this framework in Eq. 10. According to Eq. 2, we have that $\mathbf{Z}^{(l)} = \mathbf{H}^{(l)} = \left[\mathbf{h}_1^{(l)}, \ldots, \mathbf{h}_{f_l}^{(l)}\right]$. For each $\mathbf{h}_j^{(l)}$, we have that:

$$\begin{aligned}\mathbf{h}_j^{(l+1)} &= \sigma\left(\sum_{i=1}^{f_l}\mathbf{U}\,\mathrm{diag}\left(\sum_{s=1}^{S}\mathbf{W}_{i,j}^{(l,s)}\mathbf{\Phi}_s(\boldsymbol{\lambda})\right)\mathbf{U}^\top\mathbf{h}_i^{(l)}\right)\\ &= \sigma\left(\sum_{s=1}^{S}\sum_{i=1}^{f_l}\mathbf{W}_{i,j}^{(l,s)}\mathbf{U}\,\mathrm{diag}\left(\mathbf{\Phi}_s(\boldsymbol{\lambda})\right)\mathbf{U}^\top\mathbf{h}_i^{(l)}\right).\end{aligned} \tag{11}$$

Let $\mathbf{C}^{(s)} = \mathbf{U}\,\mathrm{diag}\left(\mathbf{\Phi}_s(\boldsymbol{\lambda})\right)\mathbf{U}^\top$, we can have that

$$\begin{aligned}H^{(l+1)} =& \sigma\left(C^{(1)}h^{(l)}W^{(l,1)} + \cdots + C^{(S)}h^{(l)}W^{(l,S)}\right)\\ =& \sigma\left(\sum_{s=1}^{S}C^{(s)}h^{(l)}W^{(l,s)}\right),\end{aligned} \tag{12}$$

This framework covers most of the well-known GNN models. Specifically, spectral methods and spatial methods can work in the same way. The only difference is that the graph convolution is either designed in the spectral domain or in the spatial domain.

Discussion. Spectral methods provide a strict mathematical framework for learning graph representation. Spectral graph theory provides a solid foundation for understanding the spectral properties of graphs, facilitating the development of principle algorithms and theoretical analysis. However, these spectral methods often involve computationally expensive operations, such as the eigendecomposition of the graph Laplacian matrix. The computational complexity of spectral methods scales with the size of the graph, which makes them less efficient for large-scale graphs with millions or billions of nodes and edges. In addition, spectral methods are sensitive to changes in the graph structure, such as node permutations and edge additions or deletions. Since the eigenvectors of the Laplacian matrix are particularly susceptible to perturbations in the graph, small changes in the graph topology can lead to significant variations in the learned embeddings. Specifically, spectral-based methods lack explicit mechanisms for localized information aggregation and propagation, potentially hindering their ability to efficiently capture local neighborhood information.

Unlike spectral methods, spatial methods offer translation-invariant properties, enabling them to learn representations that are insensitive to node order and graph perturbations. Spatial methods can effectively extract features that are invariant to spatial transformations through local convolution operations. However, spatial methods are susceptible to over-smoothing, especially in deep architectures with multiple layers. As information propagates through multiple layers of spatial convolutions, the features of neighboring nodes may become overly smoothed or indistinguishable, which leads to the deterioration of discrimination ability and the degradation of model performance. In addition, spatial convolutional operations involve aggregating information from neighboring nodes, potentially leading to significant computational overhead, especially when dealing with dense or highly interconnected graphs.

D. Extensions to GNNs

Apart from the above mentioned spectral- and spatial-based GNN, several new variants of GNN have been developed in recent years, including (but are not limited to) attention-based GNN, gate-based GNN, graph autoencoders (GAE), and graph generative models (GGM). These variants are derived from the spectral and spatial methods. In the following, we provide a brief review of various variants of GNN.

1) Attention-based Method: One of the earliest graph attention network is [60], which neither requires complex matrix operations, nor does it need to know the structure of the graph upfront. Based on the assumption that the neighboring vertices have different contributions to the central vertex, the multi-attention mechanism is adopted in the propagation step to specify different weights to neighboring vertices. Moreover, the attention-based GNNs can be easily generalized to inductive learning problems, such as [61, 62], etc.

2) Gate-based Method: Many gate-based methods introduce a gated recurrent unit (GRU) into the GNN, and then generate an output for each vertex based on its state. The key idea behind attention-based GNN is to update the hidden state of a vertex based on its previous and neighboring hidden states. Specifically, some other gate-based GNN models (e.g., CCRNN [61], GAAN [62]) use attention mechanism to determine which vertices are important in the current decision.

3) Graph Autoencoder (GAE): y As a generalization of multi-layer perceptron (MLP), GAE encodes vertex features into a latent embedding space, and then decodes graph information from this space. By stacking multiple layers of GAE, it can learn the graph embedding to preserve the topological information. For example, SDNE [63] preserves the vertex first-order proximity and second-order proximity by using two loss functions on the encoder output and the decoder output, respectively. It is worth noting that the variational graph autoencoder (VGAE) [64] can be regarded as a variant of GAE, which combines dimensionality reduction with generative models to learn the distribution of graph data. In addition, in some autoencoder-based methods, GNN is often considered as an encoder, such as VGAE [64], GC-MC [65], etc.

4) Graph Generative Model (GGM): GGM enables a wide range of applications: from discovering new chemical structures to constructing knowledge graphs [66]. Early works like [15] formulate graph generation as walk generation. For example, based on a stochastic neural network, NetGAN [15] uses the random walk method to generate graphs, which shows promising results in teams of generating graphs. However, the selection of vertices by random walk may lead to inconsistent results. Some works such as [24] and [25] consider the generation order of edges and vertices, formulating the graph generation as a sequential decision process.

IV. Complex Graph Representation Learning

In previous sections, we have discussed simple graphs, where the graphs are homogeneous, with one type of vertices and edges. In many real-world applications, graphs are often dynamic and complex, with multiple types of vertices and edges. Thus, the aforementioned graph-based learning models cannot handle more complicated graphs with intricate patterns well. Although there have been several variants of GNNs, these methods have neither attempted to specifically discuss more complicated graphs [67]. It is necessary to design more robust neural graphs. In the following sections, we will briefly describe popular complex graphs with formal definitions, and extend the graph-based learning models to capture more complicated patterns.

A. Heterogeneous Graphs

At present, many popular GNN models have been conducted under implicit homophily assumptions, i.e., most connections happen among vertices in the same class. This leads to the poor performance of existing GRL models in heterogeneous graphs (HeG) [2] that composes of multiple types of vertices (e.g., author and paper) and edges (e.g., cite, mention and publish), as shown in Fig. 1 (a). Moreover, most GRL models rely heavily on contextual vertex features in the information propagation process. However, in many cases, contextual information is often insufficient or incomplete. According to predefined rules, a heterogeneous graph can be extracted from complex interactive systems. Formally, we give a defination of the heterogeneous graph as follows.

Definition 4.1 (Heterogeneous Graphs) *Given a heterogeneous graph* $\mathbf{HeG} = \{\mathbf{V}, \mathbf{E}, \mathbf{X}, \varphi, \psi\}$ *with multiple types of vertices and edges. Each vertex* $v_i \in \mathbf{V}$ *and each edge* $e_{ij} \in \mathbf{E}$ *are associated with a vertex type mapping function* $\varphi(v_i) : V \to RV$ *and a edge type mapping function* $\psi(e_{ij}) : E \to RE$*, respectively, where* RV *indicates the vertex types,* RE *indicates the edge types, and* $|RV|+|RE| > 2$*.* $\mathbf{X}$ *represents the feature matrix. Specifically, when* $|RV| = |RE| = 1$*, the graph is homogeneous.*

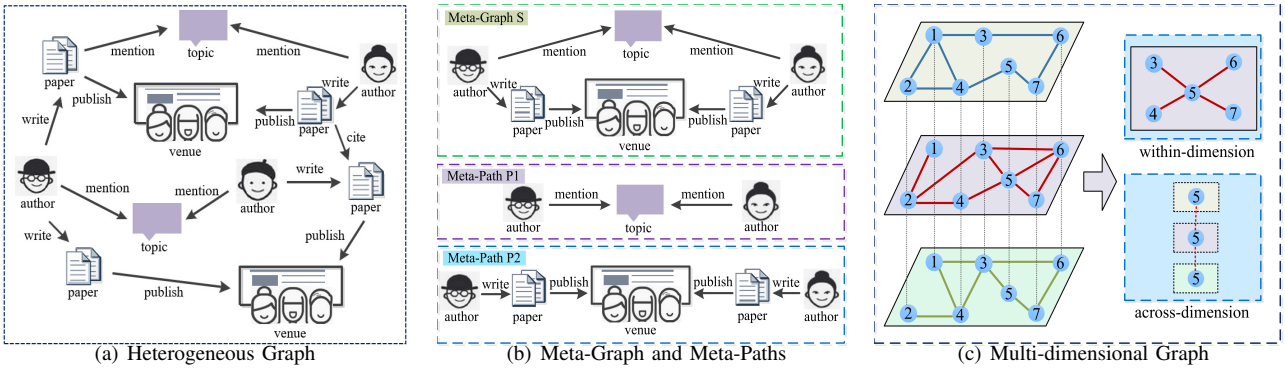
Recently, many meta-path-based methods have been proposed [1, 2], which measures the similarity and relevance between vertices by utilizing the graph structural information. Fig. 1 (b) gives two examples of meta-paths [1], one of the meta-paths P_1: $author \to topic \leftarrow author$ indicates that their paper discusses the same topic, and the other meta-path P_2: $author \to paper \to venue \leftarrow paper \leftarrow author$ indicates that the two authors published their papers in the same venue. Next, we give a defination of metapath.

Definition 4.2 (Meta-Paths) *Given a heterogeneous graph* $\mathbf{HeG}$*, a metapath* P *is defined in the form of* $v_1 \xrightarrow{e_1} v_2 \xrightarrow{e2} \cdots \xrightarrow{v_l} e_{l+1}$*, which describes a composite relation* $\mathbf{E} = e_1 \circ e_2 \circ \cdots e_{l-l} \circ e_l$ *between vertices* v_1 *and* v_l*, where* $\circ$ *represents the composition operatation.*

In order to further capture complex structural informationin, researchers have proposed meta-graph-based methods, which composes of at least two metapaths, as shown in Fig. 1 (b). We give a defination of meta-graph as follows.

Definition 4.3 (Meta-Graphs) *Given a heterogeneous graph* $\mathbf{HeG}$*, a meta-graph* S *is defined as* $S = (\mathbf{V}, \mathbf{E}, n_s, n_t)$*, where* n_s *and* n_t *represent the source vertex (with 0 in degree) and the target vertex (with 0 out degree) in the meta-graph [68], respectively.*

The heterogeneous graph embedding is a foundation research problem. There are different types of vertices (e.g., images or texts) in a heterogeneous graph, which associated with vertex features with different attributes and dimensions. Moreover, heterogeneous graph embedding maps different types of vertices into a common lower-dimension space while preserving the heterogeneous graphical semantics and



(a) Heterogeneous Graph (b) Meta-Graph and Meta-Paths (c) Multi-dimensional Graph

Fig. 1: Three graphs to be used as running examples throughout the survey. (a) A heterogenous graph consisting of multiple types of vertices and edges. (b) Two meta-paths (P_1 and P_2) is involved in the meta-graph S, i.e., P_1: $author \rightarrow topic \leftarrow author$ and P_2: $author \rightarrow paper \rightarrow venue \leftarrow paper \leftarrow author$. (c) A multi-dimensional graph where vertices interact with each other in the within- and across-dimensions, such as vertex 5 in this subgraph.

structure information. Traditionally, matrix factorization is a widely used method for learning heterogeneous graph embeddings. However, such methods usually require expensive calculation costs due to the decomposition of large-scale matrices.

Besides, due to the existence of different types of relations, the local structure of heterogeneous graph is usually different, and semantic dependent, e.g., metapath structure. The key issue is to consider what kind of information or domain knowledge should be integrated into the heterogeneous graph embedding, benefiting various downstream applications. The classic idea is to process them in different metric spaces. PME [69] is a representative work, which uses a relation-specific matrix to convert vertices into different metric spaces, where vertices connected by different types of edges are close to each other. Then, the heterogeneity of the graph can be easily captured.

It is challenging to design effective GRL methods for heterogeneous graphs to utilize neighborhood information while overcoming the heterogeneity of the graph. Recent GRL methods use Heterogeneous Graph Neural Networks (HGNN) (e.g., [1, 2]) to learn complex graph structures and vertex attributes. Moreover, it is also crucial for heterogeneous graphs to learn higher-order relation that describes more complex semantic information. Meta-paths are suitable for dealing with the heterogeneity of the graph, as it can capture various higher-order relations between vertices with rich semantics. Specifically, meta-paths split a heterogeneous graph into several simple heterogeneous graphs, where a distinct meta-path schema is adopted for each type of meta-path. Moreover, the graph filtering can be used to capture different local semantic information, which then are combined to generate the final vertex representations. In HAN [70], authors proposed a heterogeneous graph attention network to aggregate vertex features from meta-path based neighbors. This process can be described as follows.

$$\beta = \frac{\exp\left(\frac{1}{|V|}\sum_{i\in V} \boldsymbol{q}^T \tanh\left(\boldsymbol{W}\mathbf{z}_i^{\Phi} + \boldsymbol{b}\right)\right)}{\sum_{\mathcal{M}'} \exp\left(\frac{1}{|V|}\sum_{i\in V} \boldsymbol{q}^T \tanh\left(\boldsymbol{W}\mathbf{z}_i^{\Phi'} + \boldsymbol{b}\right)\right)} \tag{13}$$

where $\mathbf{z}_i^{\Phi} = \sigma\left(\sum_{j\in\mathcal{N}_i^{\Phi}} \alpha_{ij}^{\Phi} \cdot \mathbf{h}_j'\right)$, $\boldsymbol{q}$ is the semantic-level attention vector, and V is the set of meta-path based neighbors. Specifically, α and β represent the nomalized weight coefficient and the weights of a meta-path Φ_i, respectively. Then, the final embedding can be obtained by $\mathbf{Z} = \sum_{i=1}^{P} \beta_{\Phi_i} \cdot \mathbf{z}_i^{\Phi}$.

Recently, meta-graphs have been proposed to capture more complex structural relationships, which can reflect the high-order similarity between vertices, i.e., more meta-graph instances between two vertices indicate a closer relationship. In MetaGraph2Vec [68], the meta-graph-guided random walk is first used to generate a sequence of heterogeneous vertices, and then captures structural information and high-order similarity between distant vertices. Instead of such meta-graph-based representation learning, most of HGNN learn the heterogeneous vertex embeddings through the message-passing scheme.

Discussion. Unlike homogeneous graphs, which are limited to representing binary relationships between homogeneous entities, heterogeneous graphs can capture multi-modal data and incorporate domain-specific attributes associated with various types of nodes and edges. Approaches such as HAN and MetaGraph2Vec have been developed that employ meta-path-based random walks or graph convolutional networks to learn more comprehensive and informative representations. Despite their effectiveness, there are still several challenges in determining the optimal meta-paths or graph convolutional architectures. In addition, in real-world scenarios, certain node and edge types may be under-represented or under-sampled. This data sparsity and imbalance can lead to biased representations and suboptimal performance in downstream tasks. The diversity of entities and relationships in heterogeneous graphs can introduce semantic ambiguity and noise, potentially affecting the quality of learned representations. Heterogeneous graphs often contain a wide range of node and edge types, each with its own semantic meaning and contextual significance. This complexity makes it difficult for GNNs to distinguish meaningful relationships from noise.

B. Multi-Dimensional Graphs

Although existing GNN-based methods have demonstrated the superior performance on many graph-level and node-level tasks, most of them are designed for single-dimensional graphs, where one type of relation exists between a pair of vertices [4, 71–73]. In many real-world applications, a graph has multiple types of relations, i.e., a pair of vertices are connected by multiple type of relation simultaneously. Fig. 1 (c) gives an example of multi-dimensional graph (MDG). For example, in the video-sharing site YouTube, users can interact with each other via various types of actions, e.g., "sharing" or "commenting". Therefore, it is natural for such a graph to be modeled as a multi-dimensional graph, where each type of relationship is a dimension. The rich interactions between dimensions pose a huge challenge for the design of graph neural networks. Now, we give a defination of multi-dimensional graph as follows.

Definition 4.4 (Multi-Dimensional Graphs) *Given a multi-dimensional graph* $\mathbf{MDG} = \{\mathbf{V}, \mathbf{E}, D\}$*, where* $\mathbf{V}$ *denotes a set of* N *vertices* $\{v_1, v_2, \ldots, v_N\}$*, and* $\mathbf{E}$ *denotes* D *sets of edges* $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_D\}$*. There are* D *types of edges in total, and each edge set* $\mathbf{e}_d$ *describes the* d*-th type of relation between vertices in the corresponding* d *dimensions with a adjacency metrix* $\mathbf{a}_d \in \mathbf{A}$*.*

Intuitively, for a dimension, its graph structure is independent of other dimensions. To capture the within- and across-information, multi-dimensional graphs use the mapping functions to construct the co-occurrence relations in all dimensions [4]. For each vertex v_i, MDG learns its representations across all dimensions, and then integrates the information extracted from all dimensions to generate an overall representation. Besides, in order to update the representation of the vertex v_i in the dimension d, each vertex aggregates information from its within-dimension neighbors directly connected to the vertex v_i in the same dimension, and also aggregates the information extracted from its across-dimension neighbors that can be seen as the "copies" of the vertex v_i in other dimensions.

For a multi-dimensional graph, the graph structures in different dimensions are quite diverse since each dimension can describe one form of interaction for a common set of vertices. Moreover, the edges between dimensions can be complementary. Existing methods usually leverage two common characteristics, i.e., diversity that enables information aggregation within a single dimension, and collaboration that enables information aggregation cross d dimensions. Moreover, a single dimensional graph may be sparse and noisy. Therefore, it is beneficial to take advantage of the multiple dimensions to learn more robust representations.

For a large graph with millions of vertices, it is computationally expensive to consider all of the pairs of vertices. To solve this issue, mGCN [4] adopts the negative sampling approach to randomly sample n vertices in dimension d, and then calculates the representations for those sampled vertices. Moreover, in the within- and cross-dimension aggregation step, some advanced combination methods have also been considered, e.g., weighted averages, nonlinear functions and even feedforward neural networks. The within-dimension aggregation can be represented by $\mathbf{H}_{wd} = \mathbf{H}_d \cdot \hat{\mathbf{A}}_d$, where $\hat{\mathbf{A}}_d$ represents the row normalized adjacency matrix. Moreover, the cross-dimension aggregation can be represented by $\mathbf{H}_{ad} = \sum_{g=1,\ldots,D} b_{g,d} \cdot \sigma(\mathbf{W}_g \cdot \mathbf{H})$, where $\mathbf{H}_{ad}$ and $\mathbf{H}_{wd}$ are the within- and cross-dimension aggregation, respectively. Specifically, $b_{g,d}$ is the normalized importance score, which models the importance of dimension g to dimension d.

$$b_{g,d} = \frac{\operatorname{tr}\left(\mathbf{W}_g^T \mathbf{M} \mathbf{W}_d\right)}{\sum_{g=1}^{D} \operatorname{tr}\left(\mathbf{W}_g^T \mathbf{M} \mathbf{W}_d\right)} \tag{14}$$

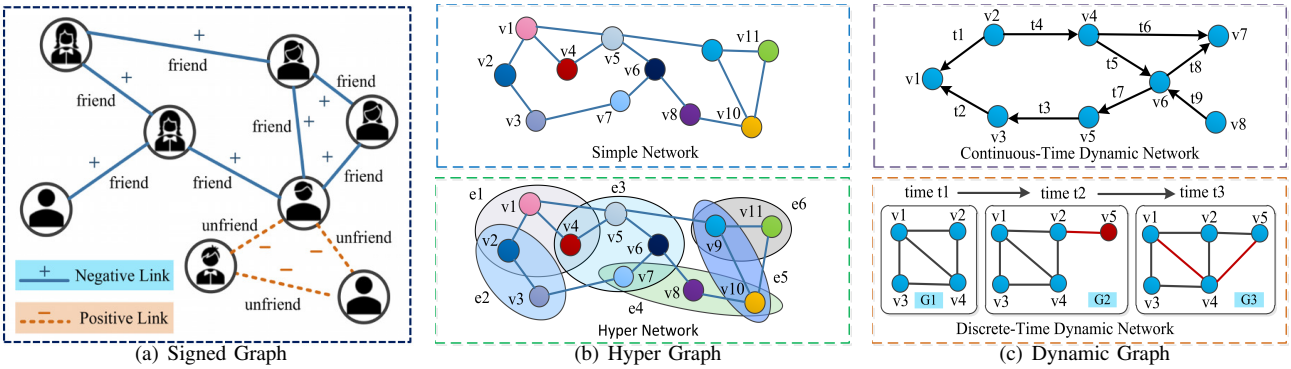
Then, we can obtain the final representation $\mathbf{H}_d$ by combining the within- and cross-dimension aggregations. This process can be described as $\mathbf{H}_d^k = (1-\alpha) \cdot \mathbf{H}_{w}{}_d^k + \alpha \cdot \mathbf{H}_{a}{}_d^k$, where α is the hyper-parameter.

In another study, Wang et al., [71] studied multi-view graphs, and proposed a multi-view network embedding method I^2MNE, which consists of an intra-view attention that aggregates vertex features in the single view, and an inter-view attention that integrates vertex representations across different views. Moreover, Zhang et al., [72] propose a scalable multiplex network embedding (MNE), which maps multi-type relations into a unified embedding space, and captures the distinct property of each sub-graph. In [73], authors extended the MNE method [72], and proposed a general attributed multiplex heterogeneous network embedding (GATNE), which can support both transductive and inductive embeddings learning.

Discussion. Multi-dimensional graphs provide a powerful framework for modeling and analyzing complex relationships and dependencies in real-world systems. By considering multiple dimensions of node and edge features, these graphs can capture rich and diverse information, leading to improved performance in a wide range of tasks, such as link prediction, node classification, and recommendation. However, multi-dimensional graphs often exhibit higher computational complexity compared to traditional graphs, especially when dealing with large-scale datasets with dense and interconnected relationships. The presence of multiple attributes or features associated with nodes and edges can result in high-dimensional representations that increase the computational burden of learning and inference tasks. This increased computational complexity can pose challenges in terms of resource requirements and scalability, especially for real-time applications and resource-constrained environments. Additionally, the curse of dimensionality may result in overfitting, especially in the presence of noisy or redundant features.

C. Signed Graphs

With the development of online social graphs [5], such as Facebook and Twitter, signed graphs (SigG) with positive and negative links have encouraged researchers to pay more and more attention on leveraging machine learning techniques. Fig. 2 (a) shows an illustrative example of a signed graph that is from a social graph. The relations among users can be represented by positive and negative links, where the positive links indicate friendships, followers and trust, while the



(a) Signed Graph (b) Hyper Graph (c) Dynamic Graph

Fig. 2: Three graphs to be used as running examples throughout the survey. (a) A signed graph, where users are uniformly modeled as vertices, while the "unfriend" and "friend" relations are modeled as the "negative" and "positive" edges, respectively. (b) In a simple graph, each edge only connects two vertices, which is denoted by a solid line. In a hyper graph, each hyperedge connects more than two vertices, which is denoted by a colored ellipse. (c) **Top:** A continuous-time dynamic graph where each node (or edge) is associated with a timestamp. **Bottom:** A discrete-time dynamic graph with topological evolution, which consists of three snapshots.

negative links indicate foes, distrust, blocked and antagonism [5]. For a social graph with hundreds of thousands of users and millions of links, the graph is often sparse and noisy. The formal definition of a signed graph is given below:

Definition 4.5 (Signed Graphs) *Let* $\mathbf{SigG} = \{\mathbf{V}, \mathbf{E}^+, \mathbf{E}^-\}$ *be a signed social graph, where* $\mathcal{V}$ *is the set of vertices,* $\mathbf{E}^+$ *denotes the set of positive edges, and* $\mathbf{E}^-$ *denotes the set of negative edges. Note that* $\mathbf{E}^+ \cap \mathbf{E}^- = \emptyset$*, in other words, a pair of vertices can only be either positive or negative links simultaneously. There is a signed adjacency matrix* $\mathbf{A}$*, where* $\mathbf{A}_{ij} = 1$ *indicates a positive edge between vertex* v_i *to vertex* v_j*. Similarly,* $\mathbf{A}_{ij} = 0$ *and* $\mathbf{A}_{ij} = -1$ *indicate a negative edge and a missing edge, respectively.*

Previous work [31] has mostly focused on the analysis and mining of unsigned graphs consisting of only positive links. However, since the algorithms and theories for unsigned graphs cannot be simply extended to signed graphs with negative links, it is necessary to develop more dedicated methods for signed graphs. Therefore, mining signed graphs still faces tremendous challenges. The existence of negative links also brings unprecedented opportunities for signed graphs, as a few negative links could significantly improve the performance of various analytical tasks, such as positive link prediction and recommendation on social media.

As described in Definition 4.5, a signed graph contains positive and negative edges. Heider et al. [74] first introduced the structural balance theory, an important social theory, and then conducted research on individual perceptions and attitudes. The theory is further developed by [75], which introduces the concept of balanced signed graph to capture the forbidden patterns in social graphs. Recently, based on structural balance theory, Wang et al. [76] propose a signed graph embedding algorithm SiNE, which preserves the relative relations between "friends" and "foes", i.e., the mapping function maps "friends" closer than "foes" in the embedding space. Moreover, SNE [77] uses a log-bilinear model to capture the positive or negative relationship for each edge by incorporating two signed-type vectors.

The growing interest for the signed graphs has led to a deep search for ever better GNN methods over signed graphs. However, the graph filters designed for simple graphs cannot be directly applied signed graphs due to the existence of negative edges. Therefore, dedicated efforts are required to design specific graph filters for signed graphs. A naive approach is to treat a signed graph as two independent unsigned graphs, but each of which contains only positive or negative edges. Then graph filters can be applied to learn the representations of these two unsigned graphs. In this setting, the complex interactions between the positive and negative edges are ignored. Hence, it is a challenge to handle the two types of edges in a single coherent model. The balance theory provides a insight for understanding their interactions in the complex signed graphs [5].

In [5], the graph filters are designed to aggregate information from balanced and unbalanced neighbors as follows.

$$\mathbf{h}_i^{B(l)} = \sigma\left(\mathbf{W}^{B(l)}\left[\sum_{j\in\mathcal{N}_i^+}\frac{\mathbf{h}_j^{B(l-1)}}{|\mathcal{N}_i^+|}, \sum_{k\in\mathcal{N}_i^-}\frac{\mathbf{h}_k^{U(l-1)}}{|\mathcal{N}_i^-|}, \mathbf{h}_i^{B(l-1)}\right]\right)$$
$$\mathbf{h}_i^{U(l)} = \sigma\left(\mathbf{W}^{U(l)}\left[\sum_{j\in\mathcal{N}_i^+}\frac{\mathbf{h}_j^{U(l-1)}}{|\mathcal{N}_i^+|}, \sum_{k\in\mathcal{N}_i^-}\frac{\mathbf{h}_k^{B(l-1)}}{|\mathcal{N}_i^-|}, \mathbf{h}_i^{U(l-1)}\right]\right) \quad (15)$$

where σ is a non-linear activation function, $\mathbf{W}^{B(l)}$ and $\mathbf{W}^{U(l)}$ are learning parameters. $\mathbf{h}_i^{B(l)}$ and $\mathbf{h}_i^{U(l)}$ denote the balanced and unbalanced representations, respectively.

Discussion. Signed graphs provide a richer representation of social dynamics compared to unipartite graphs, allowing for the modeling of complex phenomena such as social balance, homophily, and polarization. By capturing both positive and negative relationships, signed graphs enable the representation of conflicting opinions, trust issues, and adversarial relationships, providing a richer and more comprehensive view of social interactions. However, obtaining labeled data for signed graphs can be challenging, as it often requires manual annotation or expert judgment, resulting in small and imbalanced datasets. The scarcity of labeled data hinders

the development and evaluation of signed graph analysis algorithms, limiting their effectiveness and generalization. Unlike unipartite graphs, where positive edges typically represent favorable interactions and negative edges represent unfavorable interactions, the interpretation of signed edges in signed graphs depends on context and domain-specific factors. This contextual dependence can lead to inconsistencies and inaccuracies in edge annotations. In addition, signed graphs may exhibit heterogeneity and noise in edge signs, particularly in real-world datasets with diverse social interactions and ambiguous relationships. The presence of noise in edge signs can affect the performance of signed graph analysis algorithms, leading to suboptimal results and unreliable predictions.

D. Hyper Graphs

In the past few years, hyper graphs have attracted widespread attention, which can accurately represent the underlying structure, while reducing the overall number of links compared to regular graph representations that only encode pairwise information via edges. In a hyper graph, the relationships among data points could involve multiple objects, represented by a hyperedge. Fig. 2 (b) shows an example of hyper graph, which describes the relations between papers in the citation graph. In this case, an author may publish multiple papers. To encode higher-order relations, the author in the citation graph can be regarded as a hyperedge connecting multiple papers (vertices). Formally, the definition of hyper graph is as follows.

Definition 4.6 (Hyper Graphs) *Let* $\mathbf{HyG} = \{\mathbf{V}, \mathbf{E}, T\}$ *be a hyper graph, where* $\mathbf{V} = \{V_t\}_{t=1}^{T}$ *is a set of vertices with* T *types, and* $\mathbf{E} \in 2^{V}$ *is a collection of hyperedges consisting of non-empty subsets of* $\mathbf{V}$. *A hyper graph* $\mathbf{HyG}$ *degenerates to a simple graph if each edge has exactly two vertices, i.e.,* $|e| = 2$ *for all* $e \in \mathbf{E}$. *Moreover, it could be a heterogeneous hyper graph if the number of vertex types* T *is greater than 1, i.e.,* $T > 1$.

Most of the existing graph embedding methods assume that a pairwise relationship among objects exists in the real-world graph, i.e., each edge links only a pair of vertices, forming a pairwise graph. However, the existence of hyper graphs poses particular challenges to existing graph embedding methods, especially when these hyper graphs are comprised of complex interactions among objects of different modalities. This means that the relationships between vertices could go beyond pairwise. Hyper graphs usually can capture those high-order vertex relationships.

A typical approach is to convert a hyper graph into traditional pairwise graphs, and then apply the existing graph embedding methods developed on the pairwise graph for further analysis. There are two representative techniques, i.e., clique expansion and star expansion. Previous works [78] [7] assume that the hyperedges can be constructed by latent similarity, while preserving hyperedge either explicitly or implicitly. Although this assumption is reasonable for homogeneous hyper graphs, it is usually not valid when learning heterogeneous hyper graph embeddings [8], in which indecomposibility of hyperedges is a common property. To deal with indecomposable hyperedges, Tu et al. [7] propose a deep model DHNE to preserve the local and global structural information in the heterogeneous hypergraphs.

There is a growing interest of generalizing these models to hyper graph. The key to build graph neural networks for hyper graphs is to extract the pairwise relations from the hyperedges by applying the graph filters designed for simple graphs. In HGNN [79], the authors use a set of pairwise edges to approximate each hyperedge, where only one representative simple edge for each hyperedge is considered as $(v_i, v_j) := \arg\max_{v_i, v_j \in e} \|\mathbf{h}(v_i) - \mathbf{h}(v_j)\|_2^2$, where $\mathbf{h}(v_i)$ and $\mathbf{h}(v_j)$ represent the vertex features. The (v_i, v_j) will be "small" when $\mathbf{h}(v_i)$ and $\mathbf{h}(v_j)$ are "close" to each other. Then, a graph filter [80] is used to extract pairwise relations as follows.

$$\mathbf{X}^{(l+1)} = \sigma\left(\mathbf{D}_v^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^{\top}\mathbf{D}_v^{-1/2}\mathbf{X}^{(l)}\mathbf{\Theta}^{(l)}\right) \quad (16)$$

where $\mathbf{X}^{(1)}$ is the signal of hyper graph at l layer, and σ is a nonlinear activation function. By the hyperedge convolution operations, the HGNN can effectively deal with complex and high-order data correlations. Moreover, the traditional GNN models designed for simple graphs can be regarded as a special case of HGNN, where the edges are seen as the second-order hyperedges.

For real scenarios with multi-modal data [81], the data correlation modeling is more complex, as the graph structure could be beyond pairwise connections, such as visual connections, text connections, and social connections. To deal with this issue, some studies have introduced the multi-hypergraph structure to assign weights for different sub-hypergraphs, each of which corresponds to a modal. For example, in [82], the authors proposed a multi-hypergraph learning method MHL to utilize the multi-modality data.

Discussion. Hyper graphs offer a distinct advantage over traditional graphs by enabling the modeling of higher-order interactions using hyperedges. Unlike traditional graphs, which are limited to pairwise relationships between nodes, hyper graphs can model complex interactions and dependencies among multiple entities. However, the complex interplay between multiple nodes and hyperedges can obscure the underlying patterns and dependencies in the data, making it challenging to extract meaningful information from learned embeddings. Furthermore, in real-world datasets with dense and interconnected relationships, the presence of hyperedges connecting multiple nodes can result in high-dimensional representations that complicate the learning process. In addition, hyper graphs can be difficult to generalize across different domains. Due to complex higher-order interactions, models trained on one domain-specific dataset may exhibit limited transferability to other domains. Improving the generalization and transferability of GNN models across diverse domains is essential for their broader applicability in real-world settings.

E. Dynamic Graphs

Traditionally, research has been done mostly on static graphs where the connections between vertices are fixed.

However, many applications involve dynamic graphs (DyG) that changes over time as new vertices and edge are continuously emerging. For example, in online social graphs, new users (vertices) and new friendships (edges) appear every day. Each vertex and each edge are associated with a timestamp. An example of dynamic graph is shown in Fig. 2 (c). A dynamic graph allows learning models to leverage the temporal and structural patterns. Formally, we give the definitions of continuous-time dynamic graphs and discrete-time dynamic graphs.

Definition 4.7 (Continuous-Time Dynamic Graph) *Let* $\mathbf{CTDG} = (\mathbf{V}, \mathbf{E}, \varphi_v, \varphi_e)$ *be a continuous-time dynamic graph, where* $\mathbf{V}$ *denotes a set of* N *vertices* $\{v_1^t, v_2^t, \ldots, v_N^t\}$*, and* $\mathbf{E}$ *denotes* D *sets of edges* $\{e_1^t, e_2^t, \ldots, e_D^t\}$*. Each vertex and each edge are mapped to their emerging timestamps* t *through the mapping functions* φ_v *and* φ_e*, respectively.*

Definition 4.8 (Discrete-Time Dynamic Graph) *The discrete-time dynamic graph can be characterized in the form of a time series, denoted by* $\mathbf{DTDG} = \{\mathbf{G}^1, ..., \mathbf{G}^T\}$*. Then, the graph at time step* t *can be denoted by* $\mathbf{G}^t = \{\mathbf{V}^t, \mathbf{E}^t\}$.

The discrete representations of a dynamic graph at different time intervals are referred to as snapshots. Then, the static graph analysis methods can be used for each snapshot, as shown in Fig. 2 (c). Moreover, the static graphs can be regarded as coarse-grained, while the dynamic graphs that are constantly evolving can be regarded as fine-grained. As the temporal granularity increases, the model complexity also increases. Therefore, the dynamic graphs have carried the most information but is also the most complex.

Dynamic graph embedding includes various of methods such as tensor decomposition and random walks, which maps the dynamic graph into a latent space. Tensor factorization is similar to matrix factorization [83], where time is an additional dimension. In other words, the dynamic graph structure can be represented by a set of adjacency matrices $\mathcal{A} = \{\mathbf{A}^1, \ldots, \mathbf{A}^T\}$ where $\mathcal{A} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}| \times T}$. T is the timestamp. Then, tensor decomposition can be applied to these adjacency matrices. STWalk [84] performs the space-walk and time-walk on a temporal graph at the current and previous time-steps to learn effective vertex trajectory representations. Moreover, Nguyen et al. [85] use a deep learning model based on random walk to incorporate the temporal dependencies in the vertex embeddings.

In the discrete time dynamic graph, each snapshot can be regarded as a static graph. Then, the learning model designed for static graph can be applied to each snapshot, and the entire dynamic graph can be processed. In general, the time window is used to split a discrete graph into multiple snapshots, which makes the graph structure appear in each snapshot. Although reducing the size of the time window or the interval between snapshots can increase the temporal granularity, this may produce a snapshot without graph structure. Meanwhile, the running time will increase linearly with the number of snapshots. Compared with the discrete representations, the continuous graphs can offer superior temporal granularity, avoiding the loss of information.

Recently, there is a growing interest of generalizing GNN models to discrete-time dynamic graphs consisting of a set of snapshots. By combining the GNN model with the deep time-series model (such as an RNN or a GRU), the dynamic graph neural network can learn time series across graph snapshots over time. In this setting, the GNN and deep time-series model are used to model graph topology and time dependency, respectively. Both models provide great flexibility. In [6], authors propose a deep learning model, EvolveGCN, which adapts the GNN to aggregate the neighbouring information for each vertex in a graph snapshot, and uses the RNN captures the dynamism of the graph sequence.

Recently, spatio-temporal graph convolutional networks (ST-GCN) [86, 87] have been proposed to learn both the spatial and temporal patterns from graph data. In general, the ST-GCN performs the spatial graph convolution and the temporal graph convolution to extract the spatial information and temporal information, respectively. This process can be described as follows.

$$\Gamma *_{\mathcal{T}} X = P \odot \sigma(Q) \tag{17}$$

$$Y = \sum_{i=0}^{K-1} \theta_i T_i(\tilde{L}) X \tag{18}$$

where θ is the Chebyshev coefficient. $\Gamma *_{\mathcal{T}} X$ denotes the spatial graph convolution operations. P and Q are input of gates in the gated linear units (GLU) respectively. Y denotes the output of the temporal graph convolution layer.

As mentioned earlier, it is difficult for existing GNN to directly act on the continuous-time dynamic graphs (CTDG) due to the inherent essence of GNN. Most existing GNN-based research is limited to the setting of discrete-time dynamic graph (DTDG), where each snapshot in the DTDG can be processed by the static GNN model. Few works can handle any continuous-time dynamic graph, where edges can appear at any time and new vertices can be constantly added to the graph. In order to support the continuous-time scenario, Rossi et al., [88] proposed a generic framework, Temporal Graph Networks (TGNs), to deal with the dynamic graphs represented as sequences of timed events.

Discussion. Unlike static graphs, which assume the relationships between entities are constant, dynamic graphs provide the flexibility to represent time-varying interactions and evolving graph structures. By leveraging historical temporal information and modeling sequential dependencies, dynamic graph models can anticipate future changes in graph structures and interactions, enabling better prediction and decision-making. However, dynamic graphs also present some challenges. In real-world datasets with irregular sampling intervals and incomplete observations, the time nature of dynamic graphs can exacerbate the data sparsity problem. Gaps in time series may lead to unreliable representations of graph dynamics and reduce the accuracy of predictive models. Moreover, dynamic graphs often exhibit higher model complexity compared to static graphs, especially when dealing with long time series. The need to capture time dependencies and evolving relationships can result in complex model architectures and parameterizations that require complex algorithms and computational resources for training and inference. Additionally, dynamic graphs are susceptible to concept drift and model

adaptation challenges, resulting in degraded performance and outdated representations.

V. Evaluation

In this section, we offer a systematic approach to evaluate GRL methods and address the challenges associated with performance evaluation and dataset standardization. First, we summarize some commonly datasets used in GRL. Second, we highlight some well-established benchmarks in the literature, which play a pivotal role in evaluating the effectiveness of existing GRL methods and offer valuable guidance for researchers in this field.

A. Benchmark Datasets

A key aspect of evaluating GRL methods is the availability of standardized benchmark datasets that accurately represent real-world graph scenarios. However, to date, no standard dataset has been established for evaluating GRL methods across different graph types. To address this gap, we provide an overview of commonly used datasets in the literature to provide a summary of applications and use cases of GRL systems. Table I summarizes the selected benchmark datasets, which can be categorized into six groups: simple graphs, heterogeneous graphs, multi-dimensional graphs, signed graphs, hyper graphs, and dynamic graphs. These datasets cover diverse domains and provide valuable resources for evaluating the effectiveness of GNN-based learning algorithms. Each dataset is described in detail, including its characteristics, domain, and applicability for specific GRL tasks. Below, we provide a brief overview of each dataset.

Simple Graphs (SimG). For simple networks, several datasets are widely used for citation graph analysis, including Cora, Citeseer, and Pubmed [89]. In these datasets, vertices represent documents (such as academic papers), and edges represent citations between them. Another dataset, NELL [90], is derived from a knowledge graph, where nodes are interconnected by directed and labeled edges. The PPI dataset [52] contains 24 graphs, each representing interactions between proteins. Researchers leverage this dataset to assess the effectiveness of algorithms in analyzing complex biological graphs. Reddit [52], a popular online discussion forum, offers diverse topical communities where users engage in discussions and share content. In Reddit, posts are linked if the same user comments on both, forming a graph structure that reflects user engagement and content interaction within communities.

Heterogeneous Graphs (HeG). For heterogeneous graphs, there are several widely used heterogeneous graph datasets from different domains. The DBLP [91] dataset is collected from a computer science bibliography website, where the node features are the terms related to papers, authors, and conferences respectively. IMDB-He [91] is an online database containing information about movies and television programs, including details such as cast, production crew, and plot summaries. ACM [91] is a citation graph dataset, where papers are labeled according to the conference of publication, and features are constructed by the keywords. In Yelp [92], nodes are labeled based on their category, and node features are represented as elements of a bag-of-words derived from keywords.

Multi-Dimensional Graphs (MDG). Several widely used multi-dimensional graph datasets from different domains are used to evaluate the performance of GNNs. The DBLP-MD [4] is a co-authorship multi-dimensional graph extracted from the DBLP dataset, where the co-authorship in each year is treated as different relations. Epinions-MD [4] is a comprehensive review site where users can submit product reviews and rate the usefulness of reviews written by other users. In addition, users can establish relationships of trust and distrust within the platform. Due to the scarcity of real-world labeled high-dimensional multiplex graphs, the work [94] collected five datasets from various sources to assess the applicability of HMGE, including BIOGRID-Ext [93], BIOGRID [93], DBLP-Authors [94], IMDB-MD [94], STRING-DB [95]. Moreover, the benchmark datasets, including Amazon [96], YouTube [97], and Twitter [98], were adapted from [73].

Signed Graphs (SigG). For signed graphs, four public real-world benchmark datasets, including Bitcoin-Alpha [99], Bitcoin-OTC [100], Slashdot [101], and Epinions-Sig [101], are commonly used for evaluation. Many signed GRL methods were evaluated on these datasets. Bitcoin-Alpha and Bitcoin-OTC are collected from Bitcoin trading platforms where members can rate each other based on trustworthiness to prevent fraudulent transactions. Slashdot is a well-known technology-related news website with a strong and active user community. In this dataset, users can tag each other as friends (positive links) or foes (negative links). The Epinions-Sig dataset was collected from a consumer review site where members can establish trust relationships with each other. These trust relationships combine to form a Web of Trust.

Hyper Graphs (HyG). Table I provides a set of five available benchmark datasets from the existing literature on hypergraph neural networks. The benchmark datasets, namely Cora-CA and DBLP-CA, consist of co-authorship graphs and are sourced from the study by Sen et al. [89]. In Cora-CA and DBLP-CA, a hyperedge is constructed to connect all documents cited by an author. The 20News and Zoo datasets are sourced from the UCI Categorical Machine Learning Repository [102]. In the 20News dataset, the node features are represented using Term Frequency-Inverse Document Frequency (TF-IDF) representations of news messages, while the Zoo dataset contains a mix of categorical and numerical measurements describing different animals. Moreover, NTU2012 [103] is a dataset in the field of computer vision and graphics, specifically focusing on three-dimensional (3D) data.

Dynamic Graphs (DyG). In Table I, we provide four benchmark datasets commonly used in dynamic neural networks research. The Reddit-Dy [104] dataset collects data on active users and their posts under subreddits. When a user creates a post in the subreddit, the interaction is formed and the user post is transformed into a feature vector. In Wikipedia-Dy [104], popular edit pages and active users are nodes, and each edit is treated as a link. In Twitter [88], users are nodes and retweets are edges. Features are vector representations based on BERT. MOOC [104] is a dataset

TABLE I: A summary of commonly used benchmark datasets.

Dataset	$\#\mathcal{V}$	$\#\mathcal{E}$	$\#\mathcal{E}$-type	$\#\mathcal{E}^+$	$\#\mathcal{E}^-$	#Features	#Class	SimG	HeG	MDG	SigG	HyG	DyG
Cora [89]	2708	5429	-	-	-	1433	7	✓	-	-	-	✓	-
Citeseer [89]	3327	4732	-	-	-	3703	6	✓	-	-	-	✓	-
Pubmed [89]	19717	44338	-	-	-	500	3	✓	-	-	-	✓	-
NELL [90]	65755	266144	-	-	-	5414	210	✓	-	-	-	-	-
PPI [52]	56944	818716	-	-	-	50	121	✓	-	-	-	-	-
Reddit [52]	232965	114615892	-	-	-	602	41	✓	-	-	-	-	-
ACM [91]	8994	25922	4	-	-	1902	-	-	✓	-	-	-	-
DBLP-He [91]	18405	67946	4	-	-	334	-	-	✓	-	-	-	-
IMDB-He [91]	12772	37288	4	-	-	1256	-	-	✓	-	-	-	-
Yelp [92]	3913	77176	6	-	-	82	-	-	✓	-	-	-	-
DBLP-MD [4]	138072	2015650	-	-	-	20	10	-	-	✓	-	-	-
Epinions-MD [4]	15108	485154	-	-	-	15	5	-	-	✓	-	-	-
BIOGRID-Ext [93]	4503	311645	-	-	-	28	4	-	-	✓	-	-	-
BIOGRID [93]	4211	280979	-	-	-	15	4	-	-	✓	-	-	-
DBLP-Authors [94]	5124	33250	-	-	-	10	4	-	-	✓	-	-	-
IMDB-MD [94]	3000	224984	-	-	-	8	3	-	-	✓	-	-	-
STRING-DB [95]	4083	4923554	-	-	-	7	3	-	-	✓	-	-	-
Amazon [96]	312320	7500100	4	-	-	-	-	-	-	✓	-	-	-
YouTube [97]	15088	13628895	5	-	-	-	-	-	-	✓	-	-	-
Twitter [98]	456626	15367315	4	-	-	-	-	-	-	✓	-	-	-
Bitcoin-Alpha [99]	3783	14145	-	12729	1416	-	-	-	-	-	✓	-	-
Bitcoin-OTC [100]	5901	21522	-	18390	3132	-	-	-	-	-	✓	-	-
Slashdot [101]	33586	396003	-	295201	100802	-	-	-	-	-	✓	-	-
Epinions-Sig [101]	16992	327227	-	276309	50918	-	-	-	-	-	✓	-	-
Cora-CA [89] [80]	2708	1072	-	-	-	1433	7	-	-	-	-	✓	-
DBLP-CA [89] [80]	43413	22535	-	-	-	1425	6	-	-	-	-	✓	-
Zoo [102]	101	43	-	-	-	16	7	-	-	-	-	✓	-
20News [102]	16242	100	-	-	-	100	4	-	-	-	-	✓	-
NTU2012 [103]	2012	2012	-	-	-	100	67	-	-	-	-	✓	-
Reddit-Dy [104]	11000	672447	-	-	-	172	-	-	-	-	-	-	✓
Wikipedia-Dy [104]	9227	157474	-	-	-	172	-	-	-	-	-	-	✓
MOOC [104]	7135	411749	-	-	-	-	-	-	-	-	-	-	✓
Twitter-Dy [88]	8861	119872	-	-	-	768	-	-	-	-	-	-	✓
Industrial [105]	170243	2135762	-	-	-	100	-	-	-	-	-	-	✓

that comprises actions performed by students on a MOOC online course. In Industrial [105], popular products and active customers from the online grocery shopping website are nodes, and the customer-product purchases are temporal edges.

B. Performance Assessment

A fundamental aspect of successful research is to provide evidence of the effectiveness of GRL approaches. Although many GRL techniques have been proposed to address challenges in complex graphs, evaluating their effectiveness requires systematic comparisons across different graph conditions. After reviewing the various complex graphs, it would be interesting to understand the performance results of the GRL approaches when dealing with different types of graphs. However, evaluating the performance of GRL methods across different graph types remains challenging due to the lack of standardized benchmarks and comprehensive evaluation frameworks in the literature. Table II summarizes the metrics used in the literature to evaluate the performance of the GRL approaches. Our analysis focuses on the following six metrics: Accuracy, Precision, Macro-averaged F1 score (Macro-F1), Micro-average F1 score (Micro-F1), F1 score, and Area Under Curve (AUC). For all metrics, a higher value indicates better performance. It is important to note that different complex graphs are applicable to varying scenarios, which brings many challenges. For instance, heterogeneous graphs, characterized by diverse node types and interactions, present unique challenges that require strategies to address this diversity. Similarly, hyper graphs, characterized by hyperedges connecting multiple nodes, necessitate specialized methodologies to capture their complex interactions. By considering the unique characteristics of various graph types, researchers can gain deeper insights into the performance of GNNs and advance the state-of-the-art in graph analysis and modeling.

Given the complexity and characteristics of each graph type, we conduct a thorough review of GRL methods and analyze their performance in various graph architectures, including simple graphs, heterogeneous graphs, multi-dimensional graphs, signed graphs, hyper graphs, and dynamic graphs. However, conducting a full-fledged benchmarking effort covering a wide range of graph complexities is a daunting task, exceeding the scope of any single survey or study. Therefore, while it may not be feasible to comprehensively evaluate all GRL methods across diverse graph types, it is still valuable to provide insights into the performance of state-of-the-art GRL methods in representative scenarios. By synthesizing insights from existing literature, we identify the most representative GRL methods for each graph type and report the best results presented in their papers in Table II. These results are directly extracted from the literature, providing a reliable basis for comparison. We aim to bridge the gap between GRL research and facilitate future studies in this field. By leveraging these

TABLE II: Summary of some representative algorithms.

Category	Methods	Datasets	Accuracy	Precision	Macro-F1	Micro-F1	F1	AUC
Simple Graph	GCN [31]	Cora [89]	81.5	-	-	-	-	-
		Citeseer [89]	70.3	-	-	-	-	-
		Pubmed [89]	79.0	-	-	-	-	-
		NELL [90]	66.0	-	-	-	-	-
	GraphSAGE [52]	PPI [52]	-	-	-	-	61.2	-
		Reddit [52]	-	-	-	-	50.2	-
	GAT [60]	Cora [89]	83.0	-	-	-	-	-
		Citeseer [89]	72.5	-	-	-	-	-
		Pubmed [89]	79.0	-	-	-	-	-
		PPI [52]	97.3	-	-	-	-	-
Heterogeneous Graph	HAN[70]	ACM [91]	-	-	90.63	90.54	-	-
		DBLP-He [91]	-	-	93.08	93.99	-	-
		IMDB-He [91]	-	-	54.38	58.51	-	-
	HANE[106]	ACM [91]	-	-	93.48	93.37	-	-
		DBLP-He [91]	-	-	91.92	92.77	-	-
		Yelp [92]	-	-	93.55	92.76	-	-
	MAGNN[107]	DBLP-He [91]	-	-	94.10	94.47	-	-
		IMDB-He [91]	-	-	61.44	61.53	-	-
Multi-dimensional Graph	mGCN[4]	DBLP-MD [4]	-	-	67.0	68.0	-	-
		Epinions-MD [4]	-	-	44.0	56.0	-	-
	HMGE[94]	BIOGRID-Ext [93]	-	-	98.17	98.24	-	-
		BIOGRID [93]	-	-	98.75	98.77	-	-
		DBLP-Authors [94]	-	-	57.52	71.76	-	-
		IMDB-MD [94]	-	-	43.02	43.16	-	-
		STRING-DB [95]	-	-	81.38	82.99	-	-
	GATNE[73]	Amazon [96]	-	-	-	-	92.87	97.44
		YouTube [97]	-	-	-	-	76.83	97.44
		Twitter [98]	-	-	-	-	84.96	92.30
Signed Graph	SignedGCN[5]	Bitcoin-Alpha [99]	-	-	-	-	91.7	79.6
		Bitcoin-OTC [100]	-	-	-	-	92.5	82.3
		Slashdot [101]	-	-	-	-	86.4	80.4
		Epinions [101]	-	-	-	-	93.3	86.4
	LightSGCN[108]	Bitcoin-Alpha [99]	-	-	76.4	91.3	-	88.1
		Bitcoin-OTC [100]	-	-	81.6	90.2	-	89.8
		Slashdot [101]	-	-	81.7	86.2	-	91.4
		Epinions [101]	-	-	87.4	93.2	-	95.9
	SIHG[109]	Bitcoin-Alpha [99]	-	-	71.15	92.79	96.14	89.81
		Bitcoin-OTC [100]	-	-	79.49	91.65	95.28	91.54
		Slashdot [101]	-	-	79.34	86.96	91.89	89.50
		Epinions [101]	-	-	82.61	92.47	95.71	92.62
Hyper Graph	HGCN[79]	Cora [89]	81.6	-	-	-	-	-
		Pubmed [89]	80.1	-	-	-	-	-
	UniGCNII[110]	Cora [89]	73.6	-	-	-	-	-
		Citeseer [89]	66.5	-	-	-	-	-
		Pubmed [89]	75.8	-	-	-	-	-
		Cora-CA [89] [80]	76.6	-	-	-	-	-
		DBLP-CA [89] [80]	89.4	-	-	-	-	-
	HSL[111]	Cora [89]	79.88	-	-	-	-	-
		Citeseer [89]	73.79	-	-	-	-	-
		Pubmed [89]	89.86	-	-	-	-	-
		Cora-CA [89] [80]	84.43	-	-	-	-	-
		DBLP-CA [89] [80]	91.83	-	-	-	-	-
		Zoo [102]	98.08	-	-	-	-	-
		20News [102]	81.98	-	-	-	-	-
	AllSet[112]	Cora [89]	78.59	-	-	-	-	-
		Citeseer [89]	73.08	-	-	-	-	-
		Pubmed [89]	88.72	-	-	-	-	-
		Cora-CA [89] [80]	83.63	-	-	-	-	-
		DBLP-CA [89] [80]	91.53	-	-	-	-	-
		Zoo [102]	97.50	-	-	-	-	-
		20News [102]	81.38	-	-	-	-	-
Dynamic Graph	JODIE[104]	Reddit-Dy [104]	-	-	-	-	-	59.9
		Wikipedia-Dy [104]	-	-	-	-	-	83.1
		MOOC [104]	-	-	-	-	-	75.6
	TGNs[88]	Reddit-Dy [104]	-	98.46	-	-	-	65.56
		Wikipedia-Dy [104]	-	98.70	-	-	-	83.69
		Twitter-Dy [88]	-	94.52	-	-	-	92.31
	TGAT[105]	Reddit-Dy [104]	90.73	-	-	-	-	65.56
		Wikipedia-Dy [104]	96.62	-	-	-	-	83.69
		Industrial [105]	72.08	-	-	-	-	92.31

resources, researchers can rigorously evaluate and compare the performance of their GRL methods with those reported in existing literature, contributing to the development of more effective and robust graph analysis techniques.

VI. Future Directions

Although GRL technologies have achieved great success in a number of areas [36–38], it still does not provide satisfactory solutions in more complexed graphs. In the following, we summarize some future directions for GRL.

Scalability. Most real graphs are heterogeneous, dynamic, and sufficiently complex. A key challenge is how to build a scalable model, especially when the model is designed to preserve the local and global properties of complex graphs. Moreover, large-scale graphs also pose challenges to traditional graph analysis tasks. The GRL algorithm needs to be carefully designed to ensure the effectiveness and efficiency of large-scale graphs.

Model Depth. The success of deep learning largely relies on the deep neural network structure, but most of existing state-of-the-art GRL methods (i.e., GNN model) still suffer from a shallow structure problem. This is determined by the inherent essence of the GNNs, i.e., the graph convolutions makes the representations of adjacent nodes similar to each other. In theory, when the number of graph convolutional layers is infinite, the representations of all nodes will eventually converge to one point. How to build a deeper neural network could be a future research direction.

Pre-training Model. Existing deep learning models learn useful patterns and trends from a large number of instances, thereby obtaining a promising reliable performance. However, the labeling efforts for a large amount of data are non-trivial. The idea of pre-training is proposed to obtain pre-training models that are not related to specific tasks from large-scale data. After fine-tuning, this pre-trained model can be used for various downstream tasks.

Interpretability. Interpretability is an important part of putting the deep learning based GRL algorithms into practical application, especially in some decision-critical scenarios, such as disease prediction and chemical reaction prediction. However, the lack of interpretability has drawn widespread criticism of deep learning. Although many existing deep learning based GRL algorithms are promising for modeling complicated graphs, their interpretability and explainability are more challenging than other conventional deep learning models due to the structure-level interconnection between vertices.

Compositionality. Another important research direction is the compositionality of existing models. Many architectural building blocks or functional components can be incorporated into existing neural network models. For example, in [113], authors used various known building blocks (e.g., a graph nerual network layer and a capsule network layer) to construct complex architectures. A key challenge is how to effectively combine disparate architectural building blocks to deliver an agile, efficient, and high-performance architecture.

VII. Conclusion

Over the past few years, graph representation learning has become a practical and powerful tool in both graph analysis and other related areas. Specifically, graph-based deep learning methods like graph neural networks constitute a critical building block in modeling and learning structured graph data, which significantly facilitates the graph analysis. This survey provides a comprehensive review of the state-of-the-art GRL techniques, with a focus on graph neural network methods. We first introduce the formal definition of GRL as well as the related concepts. After that, we provide a taxonomy that groups real-world graphs into two categories: simple graphs and complex graphs, and then summarizes of the state-of-the-art GRL techniques within categories. This survey provides important insights for researchers and practitioners seeking to understand and leverage GRL methods to analyze complex real-world graphs. Finally, we highlight a wide range of important applications of GRL and suggest a few promising and emerging research directions for future work. We believe that our work will inspire the community to further advance the research on GRL.

VIII. Acknowledgement

The authors would like to thank the Editor-in-Chief, the Associate Editor, and the reviewers for their insightful comments and suggestions. This work was supported by the National Natural Science Foundation of China Project (62172441, 62373372), the Joint Funds for Railway Fundamental Research of National Natural Science Foundation of China (U2368201), special fund of National Key Laboratory of Ni&Co Associated Minerals Resources Development and Comprehensive Utilization (GZSYS-KY-2022-018, GZSYS-KY-2022-024), Key Project of Shenzhen City Special Fund for Fundamental Research (JCYJ20220818103200002), the National Natural Science Foundation of Hunan Province (2023JJ30696).

References

[1] X. Wang, Y. Lu *et al.*, "Dynamic heterogeneous information network embedding with meta-path based proximity," *IEEE Trans. Knowl. Data Eng.*, pp. 1–1, 2020.

[2] L. Sun, L. He *et al.*, "Joint embedding of meta-path and meta-graph for heterogeneous information networks," in *Proceedings of ICBK*, 2018, pp. 131–138.

[3] H. Cheng, C. He *et al.*, "Community detection based on directed weighted signed graph convolutional networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 2, pp. 1642–1654, 2024.

[4] Y. Ma, S. Wang, C. C. Aggarwal, D. Yin, and J. Tang, "Multi-dimensional graph convolutional networks," in *Proceedings of SDM*, 2018, pp. 657–665.

[5] T. Derr, Y. Ma, and J. Tang, "Signed graph convolutional networks," in *Proceedings of ICDM*, 2018, pp. 929–934.

[6] A. Pareja *et al.*, "Evolvegcn: Evolving graph convolutional networks for dynamic graphs," in *Proceedings of AAAI*, vol. 34, no. 4, 2020, pp. 5363–5370.

[7] K. Tu, P. Cui, X. Wang, F. Wang, and W. Zhu, "Structural deep embedding for hyper-networks." in *Proceedings of AAAI*, 2017, pp. 426–433.

[8] I. M. Baytas, C. Xiao, F. Wang, A. K. Jain, and J. Zhou, "Heterogeneous hyper-network embedding," in *Proceedings of ICDM*, 2018, pp. 875–880.

[9] E. Y. Sun, H. Wu, S. C. Huang, Y. Kuan, and C. Yung, "Evolutional codes: Novel efficient graph data representation for mobile edge computing," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 1, pp. 1387–1397, 2024.

[10] B. Wang, B. Jiang, and C. Ding, "Fl-gnns: Robust network representation via feature learning guided graph neural networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 1, pp. 750–760, 2024.

[11] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 833–852, 2019.

[12] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge Based Systems*, vol. 151, pp. 78–94, 2018.

[13] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, vol. 6, pp. 3–28, 2020.

[14] X. Pei, X. Deng *et al.*, "Privacy-enhanced graph neural network for decentralized local graphs," *IEEE Trans. Inf. Forensics Secur.*, vol. 19, pp. 1614–1629, 2024.

[15] A. Bojchevski, O. Shchur, D. Zgner, and S. Gnnemann, "Netgan: Generating graphs via random walks." in *Proceedings of ICML*, 2018, pp. 609–618.

[16] L. G. Moyano, "Learning network representations," *European Physical Journal-special Topics*, vol. 226, no. 3, pp. 499–518, 2017.

[17] J. Tang, Y. Chang, C. C. Aggarwal, and H. Liu, "A survey of signed network mining in social media," *ACM Comput. Surv.*, vol. 49, p. 42, 2016.

[18] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 249–270, 2022.

[19] X. Wang, D. Bo, C. Shi, S. Fan, Y. Ye, and P. S. Yu, "A survey on heterogeneous graph embedding: Methods, techniques, applications and sources," *IEEE Trans. Big Data*, vol. 9, no. 2, pp. 415–436, 2023.

[20] R. R. McCune, T. Weninger, and G. Madey, "Thinking like a vertex: A survey of vertex-centric frameworks for large-scale distributed graph processing," *ACM Comput. Surv.*, vol. 48, no. 2, pp. 25:1–25:39, 2015.

[21] J. Vatter, R. Mayer, and H. Jacobsen, "The evolution of distributed systems for graph neural networks and their origin in graph processing and deep learning: A survey," *ACM Comput. Surv.*, vol. 56, no. 1, pp. 6:1–6:37, 2024.

[22] X. Zhu, Z. Li, X. Wang, X. Jiang, P. Sun, X. Wang, Y. Xiao, and N. J. Yuan, "Multi-modal knowledge graph construction and application: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 2, pp. 715–735, 2024.

[23] L. Zhong, J. Wu, Q. Li, H. Peng, and X. Wu, "A comprehensive survey on automatic knowledge graph construction," *ACM Comput. Surv.*, vol. 56, no. 4, pp. 94:1–94:62, 2024.

[24] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. W. Battaglia, "Learning deep generative models of graphs," *arXiv preprint arXiv:1803.03324*, 2018.

[25] J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec, "Graphrnn: Generating realistic graphs with deep auto-regressive models," in *Proceedings of ICML*, 2018, pp. 5694–5703.

[26] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[27] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proceedings of ICLR*, 2014.

[28] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data." *arXiv preprint arXiv:1506.05163*, 2015.

[29] R. Levie, F. Monti, X. Bresson, and M. M. Bronstein, "Cayleynets: Graph convolutional neural networks with complex rational spectral filters," *IEEE Trans. Signal Process.*, vol. 67, no. 1, pp. 97–109, 2019.

[30] C. Hawkins, B. Chen, K. Yazdani, and M. T. Hale, "Node and edge differential privacy for graph laplacian spectra: Mechanisms and scaling laws," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 2, pp. 1690–1701, 2024.

[31] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of ICLR (Poster)*, 2016.

[32] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proceedings of ICLR*, 2018.

[33] R. Li, S. Wang *et al.*, "Adaptive graph convolutional neural networks," in *Proceedings of AAAI*, 2018, pp. 3546–3553.

[34] C. Zhuang and Q. Ma, "Dual graph convolutional networks for graph-based semi-supervised classification," in *The Web Conference*, 2018.

[35] B. Xu, H. Shen *et al.*, "Graph wavelet neural network." in *Proceedings of ICLR*, 2019.

[36] J. Jiang, J. Ma, and X. Liu, "Multilayer spectralspatial graphs for label noisy robust hyperspectral image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 2, pp. 839–852, 2022.

[37] X. Wang and M. Zhang, "How powerful are spectral graph neural networks," vol. 162, 2022, pp. 23 341–23 362.

[38] F. L. Opolka, Y. Zhi, P. Liò, and X. Dong, "Adaptive gaussian processes on graphs via spectral graph wavelets," vol. 151, 2022, pp. 4818–4834.

[39] Z. Chen, F. Chen *et al.*, "Bridging the gap between spatial and spectral domains: A survey on graph neural networks." *arXiv preprint arXiv:2002.11867*, 2020.

[40] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks," in *Proceedings of ICLR*, 2018.

[41] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proceedings of NIPS*, vol. 29, 2016, pp. 3844–3852.

[42] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," in *Proceedings of NIPS*, vol. 29, 2016, pp. 1993–2001.

[43] M. Niepert, M. H. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proceedings of ICML*, 2016.

[44] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification." in *National Conference on Artificial Intelligence*, 2018.

[45] H. Gao, Z. Wang, and S. Ji, "Large-scale learnable graph convolutional networks," in *Proceedings of KDD*, 2018, pp. 1416–1424.

[46] D. V. Tran, N. Navarin, and A. Sperduti, "On filter size in graph convolutional networks," in *IEEE Symposium Series on Computational Intelligence*, 2018.

[47] X. He, K. Deng *et al.*, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *Proceedings of ACM SIGIR*, 2020, pp. 639–648.

[48] S. Yun, S. Kim, J. Lee, J. Kang, and H. J. Kim, "Neo-gnns: Neighborhood overlap-aware graph neural networks for link prediction," in *Proceedings of NIPS*, vol. 34, 2021, pp. 13 683–13 694.

[49] P. Han, P. Zhao *et al.*, "Gnn-retro: Retrosynthetic planning with graph neural networks," in *Proceedings of AAAI*, 2022, pp. 4014–4021.

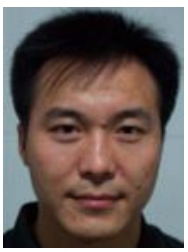
[50] H. Liu, J. Zhang, Q. Liu, and J. Cao, "Minimum spanning tree based graph neural network for emotion classification using EEG," *Neural Networks*, vol. 145, pp. 308–318, 2022.

[51] W. Ju, X. Luo, Z. Ma, J. Yang, M. Deng, and M. Zhang, "GHNN: graph harmonic neural networks for semi-supervised graph-level classification," *Neural Networks*, vol. 151, pp. 70–

79, 2022.
[52] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proceedings of NIPS*, vol. 30, 2017, pp. 1024–1034.
[53] W.-L. Chiang, X. Liu *et al.*, "Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks," in *Proceedings of KDD*, 2019, pp. 257–266.
[54] C. Huang, H. Xu *et al.*, "Knowledge-aware coupled graph neural network for social recommendation," in *Proceedings of AAAI*, vol. 35, no. 5, 2021, pp. 4115–4122.
[55] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of KDD*, 2016, pp. 855–864.
[56] F. Wu, T. Zhang, A. H. de Souza, C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," in *Proceedings of ICML*, 2019, pp. 6861–6871.
[57] J. Klicpera, A. Bojchevski, and S. Gnnemann, "Predict then propagate: Graph neural networks meet personalized pagerank." in *Proceedings of ICLR*, 2018.
[58] Q. Li, X.-M. Wu, H. Liu, X. Zhang, and Z. Guan, "Label efficient semi-supervised learning via graph filtering," in *Proceedings of CVPR*, 2019, pp. 9582–9591.
[59] M. Balcilar, G. Renton *et al.*, "Analyzing the expressive power of graph neural networks in a spectral perspective," in *Proceedings of ICLR*, 2021.
[60] P. Velickovic, G. Cucurull *et al.*, "Graph attention networks," in *Proceedings of ICLR*, 2018.
[61] J. Ye, L. Sun, B. Du, Y. Fu, and H. Xiong, "Coupled layer-wise graph convolution for transportation demand prediction." in *Proceedings of AAAI*, vol. 35, no. 5, 2021, pp. 4617–4625.
[62] J. Zhang, X. Shi *et al.*, "Gaan: Gated attention networks for learning on large and spatiotemporal graphs," in *Proceedings of CUAI*, 2018, pp. 339–349.
[63] D. Bo, X. Wang *et al.*, "Structural deep clustering network," in *Proceedings of The Web Conference*, 2020, pp. 1400–1410.
[64] T. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv: Machine Learning*, 2016.
[65] R. van den Berg, T. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv: Machine Learning*, 2017.
[66] X. Pei, X. Deng *et al.*, "A knowledge transfer-based semi-supervised federated learning for iot malware detection," *IEEE Trans. Dependable Secur. Comput.*, vol. 20, no. 3, pp. 2127–2143, 2023.
[67] Q. Han, H. Wen, J. Wu, and M. Ren, "Rumor spreading and security monitoring in complex networks," in *Proceedings of CSoNet*. Springer, 2015, pp. 48–59.
[68] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Metagraph2vec: Complex semantic path augmented heterogeneous network embedding," in *Proceedings of PAKDD*, 2018.
[69] H. Chen, H. Yin, W. Wang, H. Wang, Q. V. H. Nguyen, and X. Li, "PME: projected metric embedding on heterogeneous networks for link prediction," in *Proceedings of KDD*, 2018, pp. 1177–1186.
[70] X. Wang, H. Ji *et al.*, "Heterogeneous graph attention network," in *Proceedings of WWW*, 2019, pp. 2022–2032.
[71] Y. Wang, L. Hu, Y. Zhuang, and F. Wu, "Intra-view and inter-view attention for multi-view network embedding," in *Proceedings of PCM*, vol. 11164, 2018, pp. 201–211.
[72] H. Zhang, L. Qiu, L. Yi, and Y. Song, "Scalable multiplex network embedding," in *Proceedings of IJCAI*, 2018, pp. 3082–3088.
[73] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang, "Representation learning for attributed multiplex heterogeneous network," in *Proceedings of KDD*, 2019, pp. 1358–1368.
[74] F. Heider, "Attitudes and cognitive organization." *The Journal of Psychology*, vol. 21, no. 1, pp. 107–112, 1946.
[75] D. Cartwright and F. Harary, "Structural balance: a generalization of heider's theory." *Psychological Review*, vol. 63, no. 5, pp. 277–293, 1956.
[76] S. Wang, J. Tang, C. C. Aggarwal, Y. Chang, and H. Liu, "Signed network embedding in social media," in *Proceedings of ICDM*, 2017, pp. 327–335.
[77] S. Yuan, X. Wu, and Y. Xiang, "Sne: Signed network embedding," in *Proceedings of PACKDD*, 2017, pp. 183–195.
[78] L. Sun, S. Ji, and J. Ye, "Hypergraph spectral learning for multi-label classification," in *Proceedings of KDD*, 2008, pp. 668–676.
[79] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proceedings of AAAI*, vol. 33, no. 1, 2019, pp. 3558–3565.
[80] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. P. Talukdar, "Hypergcn: A new method for training graph convolutional networks on hypergraphs," in *Proceedings of NIPS*, vol. 32, 2019, pp. 1509–1520.
[81] C. Xu, Z. Guan *et al.*, "Recommendation by users multimodal preferences for smart city applications," *IEEE Trans. Ind. Inform.*, vol. 17, no. 6, pp. 4197–4205, 2020.
[82] M. Liu, Y. Gao, P.-T. Yap, and D. Shen, "Multi-hypergraph learning for incomplete multimodality data," *IEEE J. Biomed. Health Inform.*, vol. 22, no. 4, pp. 1197–1208, 2018.
[83] X. Shen, B. Yi, H. Liu, W. Zhang, Z. Zhang, S. Liu, and N. Xiong, "Deep variational matrix factorization with knowledge embedding for recommendation system," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 5, pp. 1906–1918, 2021.
[84] S. Pandhre, H. Mittal, M. Gupta, and V. N. Balasubramanian, "Stwalk: learning trajectory representations in temporal graphs," in *Proceedings of ICDSMD*, 2018, pp. 210–219.
[85] G. H. Nguyen, J. B. Lee, R. A. Rossi, N. K. Ahmed, E. Koh, and S. Kim, "Continuous-time dynamic network embeddings," in *Proceedings of The Web Conference*, 2018, pp. 969–976.
[86] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proceedings of AAAI*, 2018, pp. 7444–7452.
[87] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," in *Proceedings of IJCAI*, 2018, pp. 3634–3640.
[88] E. Rossi, B. Chamberlain *et al.*, "Temporal graph networks for deep learning on dynamic graphs," *CoRR*, vol. abs/2006.10637, 2020.
[89] P. Sen, G. Namata *et al.*, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–106, 2008.
[90] A. Carlson, J. Betteridge *et al.*, "Toward an architecture for never-ending language learning," in *Proceedings of the AAAI*, 2010, pp. 1306–1313.
[91] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," in *Proceedings of NeurIPS*, 2019, pp. 11 960–11 970.
[92] Y. Lu, C. Shi, L. Hu, and Z. Liu, "Relation structure-aware heterogeneous information network embedding," in *Proceedings of AAAI*, 2019, pp. 4456–4463.
[93] R. Oughtred *et al.*, "The biogrid database: A comprehensive biomedical resource of curated protein, genetic, and chemical interactions," *Protein Science*, vol. 30, no. 1, pp. 187–200, 2021.
[94] K. Abdous, N. Mrabah, and M. Bouguessa, "Hierarchical aggregations for high-dimensional multiplex graph embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 4, pp. 1624–1637, 2024.
[95] D. Szklarczyk *et al.*, "String v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets," *Nucleic acids research*, vol. 47, no. D1, pp. D607–D613, 2019.
[96] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proceedings of EMNLP*, 2019, pp. 188–197.
[97] L. Tang, X. Wang, and H. Liu, "Uncoverning groups via

heterogeneous interaction analysis," in *Proceedings of ICDM*, 2009, pp. 503–512.
[98] M. De Domenico, A. Lima, P. Mougel, and M. Musolesi, "The anatomy of a scientific rumor," *Scientific reports*, vol. 3, no. 1, p. 2980, 2013.
[99] S. Kumar, F. Spezzano, V. S. Subrahmanian, and C. Faloutsos, "Edge weight prediction in weighted signed networks," in *Proceedings of ICDM*, 2016, pp. 221–230.
[100] S. Kumar, B. Hooi *et al.*, "REV2: fraudulent user prediction in rating platforms," in *Proceedings of WSDM*, 2018, pp. 333–341.
[101] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg, "Signed networks in social media," in *Proceedings of CHI*, 2010, pp. 1361–1370.
[102] D. Dua and C. Graff, *UCI machine learning repository*. [Online]. Available: http://archive.ics.uci.edu/ml, 2017.
[103] D. Chen, X. Tian, Y. Shen, and M. Ouhyoung, "On visual similarity based 3d model retrieval," *Comput. Graph. Forum*, vol. 22, no. 3, pp. 223–232, 2003.
[104] S. Kumar, X. Zhang, and J. Leskovec, "Predicting dynamic embedding trajectory in temporal interaction networks," in *Proceedings of SIGKDD*, 2019, pp. 1269–1278.
[105] D. Xu, C. Ruan, E. Körpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," in *Proceedings of ICLR*, 2020.
[106] Y. Wang, Z. Duan, B. Liao, F. Wu, and Y. Zhuang, "Heterogeneous attributed network embedding with graph convolutional networks," in *Proceedings of AAAI*, 2019, pp. 10 061–10 062.
[107] X. Fu, J. Zhang, Z. Meng, and I. King, "MAGNN: metapath aggregated graph neural network for heterogeneous graph embedding," in *Proceedings of WWW*, 2020, pp. 2331–2341.
[108] H. Liu, "Lightsgcn: Powering signed graph convolution network for link sign prediction with simplified architecture design," in *ACM SIGIR*, 2022, pp. 2680–2685.
[109] Y. Luo, Z. Huang, H. Chen, Y. Yang, H. Yin, and M. Baktashmotlagh, "Interpretable signed link prediction with signed infomax hyperbolic graph," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2021.
[110] J. Huang and J. Yang, "Unignn: a unified framework for graph and hypergraph neural networks," in *Proceedings of IJCAI*, Z. Zhou, Ed., 2021, pp. 2563–2569.
[111] D. Cai, M. Song *et al.*, "Hypergraph structure learning for hypergraph neural networks," in *Proceedings of IJCAI*, 2022, pp. 1923–1929.
[112] E. Chien, C. Pan, J. Peng, and O. Milenkovic, "You are allset: A multiset function framework for hypergraph neural networks," in *Proceedings of ICLR*, 2022, pp. 1–24.
[113] S. Verma and Z. Zhang, "Graph capsule convolutional neural networks," *CoRR*, vol. abs/1805.08090, 2018.

Xinjun Pei received the master's degree in computer science from Xinjiang University, Wulumuqi, China. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, Central South University, Changsha, China. Since 2017, he has been engaged in the direction of information security. His research interests include wireless communications and networking, edge computing, deep learning, and the Internet of Things.



Xiaoheng Deng (Senior Member, IEEE) received the Ph.D. degrees in computer science from Central South University, Changsha, Hunan, P.R. China, in 2005. Since 2006, he has been an Associate Professor and then a Full Professor with the department of Communication Engineering, Central South University.He is Joint researcher of Shenzhen Research Institue, Central South University. He is a senior member of IEEE, a senior member of CCF, a member of CCF Pervasive Computing Council, a member of ACM. He has been a chair of CCF YOCSEF CHANGSHA from 2009 to 2010. His research interests include network security, edge computing, Internet of Things, and data mining.

Neal N. Xiong (Senior Member, IEEE) is current an Associate Professor (4 year credits) at Department of Computer Science and Mathematics, Sul Ross State University, Alpine, TX 79830, USA. He received his both PhD degrees in Wuhan University (2007, about sensor system engineering), and Japan Advanced Institute of Science and Technology (2008, about dependable communication networks), respectively. Before he attended Sul Ross State University, he worked in Georgia State University, Northeastern State University, and Colorado Technical University (full professor about 5 years) about 10 years. His research interests include Cloud Computing, Security and Dependability, Parallel and Distributed Computing, Networks, and Optimization Theory.

Dr. Xiong published over 200 IEEE journal papers and over 100 international conference papers. Some of his works were published in IEEE JSAC, IEEE or ACM transactions, ACM Sigcomm workshop, IEEE INFOCOM, ICDCS, and IPDPS. He has been a General Chair, Program Chair, Publicity Chair, Program Committee member and Organizing Committee member of over 100 international conferences. He is serving as an Editor-in-Chief, Associate editor or Editor member for over 10 international journals (including Associate Editor for IEEE Tran. on Systems, Man & Cybernetics: Systems, Associate Editor for IEEE Tran. on Network Science and Engineering, Associate Editor for Information Science, Editor-in-Chief for Journal of Internet Technology (JIT), and Editor-in-Chief for Journal of Parallel & Cloud Computing (PCC)), and a guest editor for over 10 international journals, including Sensor Journal, WINET and MONET. He has received the Best Paper Award in the 10th IEEE International Conference on High Performance Computing and Communications (HPCC-08) and the Best student Paper Award in the 28th North American Fuzzy Information Processing Society Annual Conference (NAFIPS2009).

Shahid Mumtaz (Senior Member, IEEE) is an IET Fellow, IEEE ComSoc and ACM Distinguished speaker, recipient of IEEE ComSoC Young Researcher Award (2020), founder and EiC of IET "Journal of Quantum communication," Vice-Chair: Europe/Africa Region- IEEE ComSoc: Green Communications & Computing society and Vice-chair for IEEE standard on P1932.1: Standard for Licensed/Unlicensed Spectrum Interoperability in Wireless Mobile Networks. He is the author of 4 technical books, 12 book chapters, 250+ technical papers (150+ Journal/transaction, 80+ conference, 2 IEEE best paper award in the area of mobile communications. Most of his publication is in the field of Wireless Communication. He is serving as Scientific Expert and Evaluator for various Research Funding Agencies. He was awarded an "Alain Bensoussan fellowship" in 2012. He is the recipient of the NSFC Researcher Fund for Young Scientist in 2017 from China.

Jie Wu (Fellow, IEEE) is the Director of the Center for Networked Computing and Laura H. Carnell professor at Temple University. He also serves as the Director of International Affairs at College of Science and Technology. He served as Chair of Department of Computer and Information Sciences from the summer of 2009 to the summer of 2016 and Associate Vice Provost for International Affairs from the fall of 2015 to the summer of 2017. Prior to joining Temple University, he was a program director at the National Science Foundation and was a distinguished professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, network trust and security, distributed algorithms, and cloud computing. Dr. Wu regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Mobile Computing, IEEE Transactions on Service Computing, Journal of Parallel and Distributed Computing, and Journal of Computer Science and Technology. Dr. Wu is/was general chair/co-chair for IEEE IPDPS08, IEEE DCOSS09, IEEE ICDCS13, ACM MobiHoc14, ICPP16, IEEE CNS16, WiOpt21, and ICDCN22 as well as program chair/cochair for IEEE MASS04, IEEE INFOCOM11, CCF CNCC13, and ICCCN20. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a Fellow of the AAAS and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.