Subgraph Networks With Application to Structural Feature Space Expansion

Qi Xuan, *Member, IEEE*, Jinhuan Wang, Minghao Zhao, Junkun Yuan, Chenbo Fu, Zhongyuan Ruan, and Guanrong Chen, *Life Fellow, IEEE*

Abstract—Real-world networks exhibit prominent hierarchical and modular structures, with various subgraphs as building blocks. Most existing studies simply consider distinct subgraphs as motifs and use only their numbers to characterize the underlying network. Although such statistics can be used to describe a network model, or even to design some network algorithms, the role of subgraphs in such applications can be further explored so as to improve the results. In this article, the concept of subgraph network (SGN) is introduced and then applied to network models, with algorithms designed for constructing the 1st-order and 2nd-order SGNs, which can be easily extended to build higher-order ones. Furthermore, these SGNs are used to expand the structural feature space of the underlying network, beneficial for network classification. Numerical experiments demonstrate that the network classification model based on the structural features of the original network together with the 1st-order and 2nd-order SGNs always performs the best as compared to the models based only on one or two of such networks. In other words, the structural features of SGNs can complement that of the original network for better network classification, regardless of the feature extraction method used, such as the handcrafted, network embedding and kernel-based methods.

Index Terms—Subgraph, motif, network classification, structural feature, learning algorithm, biological network, social network

1 Introduction

Many real-world systems can be naturally represented by networks, such as biological networks [1], [2], collaboration networks [3], [4], software networks [5], [6], and social networks [7], [8]. Studying the substructure of a network, e.g., its subgraphs, is an efficient way to understand and analyze the network [9]. In fact, subgraphs are basic structural elements of a network, and distinct sets of subgraphs are usually associated with different types of networks. In retrospect, as shown in [10], frequent appearance of subgraphs can reveal topological interaction patterns, each of which performs precisely some specialized functions, therefore they can be used to distinguish different communities and various networks.

Up to now, a number of studies on network subgraphs for graph classification have been reported. Ugander *et al.* [11] treated subgraph frequency as a local property in social network and found that subgraph frequency can indeed provide unique insights for identifying both social structure and graph structure in a large network. Similarly, Vohra [12] summarized the network by stacking subgraph frequencies into a vector as a global network property and then classified networks into different groups, where these frequency statistics are implemented through two schemes [11], [13]. Moreover, in the study of biological networks, Grochow *et al.* [14] proposed a novel algorithm for identifying larger network elements and functional motifs, revealing the clustering properties of motifs through subgraph enumeration and symmetry-breaking. Without any interaction dependencies between them, these studies simply acquired a sequence of discrete motif entities with feature information such as counting, weight, etc. to describe the underlying network. Except for subgraph frequency statistics, Benson *et al.* [15] obtained the corresponding embedding representation through laplacian matrix analysis method. Moreover, in [16], an incremental subgraph join feature selection algorithm was designed, which forces graph classifiers to join short-pattern subgraphs so as to generate long-pattern subgraph features. Similarly, Yang *et al.* [17] proposed the NEST method which combined the motifs and convolutional neural network.

The studies mentioned above try to reveal subgraph-level patterns, which can be considered as network building blocks of particular functions, to capture mesoscopic structure. However, most of them ignored the interaction between these subgraphs, which could be of particular importance to represent the global structure of subgraph-level. In order to address this, we propose a method to establish *Subgraph Networks* (SGNs) of different orders. It can be expected that such SGNs can capture the structural features of different aspects and thus

- *Q. Xuan, J. Wang, C. Fu, and Z. Ruan are with the Institute of Cyberspace Security, College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China. E-mail: {xuanqi, JinhuanWang, cbfu, zyruan}@zjut.edu.cn.*
- *M. Zhao is with the Fuxi AI Lab, NetEase Inc., Hangzhou 310052, China. E-mail: zhaominghao@corp.netease.com.*
- *J. Yuan is with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China. E-mail: yuanjk@zju.edu.cn.*
- *G. Chen is with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong SAR, China. E-mail: eegchen@cityu.edu.hk.*

Manuscript received 13 Sept. 2018; revised 28 June 2019; accepted 24 Nov. 2019. Date of publication 5 Dec. 2019; date of current version 29 Apr. 2021. (Corresponding authors: Qi Xuan and Zhongyuan Ruan.)
Recommended for acceptance by P. Cui.
Digital Object Identifier no. 10.1109/TKDE.2019.2957755

may benefit the follow-up tasks, such as network classification. Briefly, there are three steps to build an SGN from an original network: first, detect subgraphs in the original network; second, choose appropriate subgraphs for a task; third, utilize the chosen subgraphs to build an SGN. Line graph [18] thus can be considered as a special SGN, where a link connecting two nodes in the original network is considered as a subgraph, and two subgraphs are connected in the SGN if the corresponding two links share a same terminal node. Clearly more complicated subgraphs can be considered, e.g., three nodes with two links, so as to get a higher-order SGN, as will be further discussed in Section 3. The key point here is that the SGN extracts the representative parts of the original network and then assembles them to reconstruct a new network that preserves the relationship among subgraphs. Our method thus implicitly maintains the higher-order structures under the premise of providing the information of local structures. And, the network structure of SGN can complement the original network and, as a result, the integration of their features will benefit the subsequent structure-based algorithms design and applications.

The main contributions of this work are summarized as follows.

- A new concept of SGN is introduced, along with algorithms designed for constructing the 1st-order and 2nd-order SGNs from a given network. These algorithms can be easily extended to construct higher-order SGNs.
- SGN is used to obtain a series of handcrafted structural features which, together with the features automatically extracted by using some advanced network-embedding methods, kernel-based methods and depth model, provide complementary features to those extracted from the original network.
- SGN is applied to network classification. Experiments on seven groups of networks are carried out, showing that integrating the features obtained from SGN can indeed significantly improve the classification accuracy in most cases, as compared to the same feature extraction and classification methods based only on the original networks.

The rest of the paper is organized as follows. In Section 2, some related work about subgraph and network representation methods are briefly introduced. In Section 3, the definition of SGN is provided and algorithms for constructing the 1st-order and 2nd-order SGNs are designed. In Section 4, handcrafted structural features are characterized, for both the original network and SGNs. In Section 5, several automatic feature extraction methods are discussed, whereas SGNs are applied to graph classification for some real-world networks. Finally, Section 6 concludes the investigation, with a future research outlook.

2 Related Work

In this section, we review the related work of subgraph in graph mining applications and the network representation methods combined with depth models in recent years.

2.1 Subgraph in Graph Mining

Recently, subgraphs have been widely applied in the study between entities in networks. For example, in [19], [20], [21], different algorithms were designed for detecting network subgraphs. In [22], a method for detecting strong ties was proposed using frequent subgraphs in a social network, where it was observed that frequent subgraphs as network structural features could lead to good performances in alleviating the sparse problem for detecting strong ties on the network. By adding time stamp to the topology, temporal frequent subgraphs [23], [24], [25] were studied for some time-dependent networks, such as social and communication networks, as well as biological and neural networks. Furthermore, subgraphs were also applied to graph clustering. In [26], a graph clustering method was developed based on frequent subgraphs, which can effectively detect communities in a network. Network subgraphs deeply depict the local structural features of the network and have important research value in the application of graph mining.

2.2 Network Representation

The combination of subgraph structures and depth models enriches the research methods of the network and brings inspiration to researchers. With the rapid development of deep learning, many graph mining and representation methods have been proposed and tested, with practical applications to, e.g., drug design (through studying chemical compound and proteins data) [27], [28] and market analysis (through purchase history) [29]. Methods like word2vec [30] and doc2vec [31] have shown good performances in natural language processing (NLP), bringing some new insights to the field of graph representation. Inspired by these algorithms, graph2vec [32] was proposed, which was shown to be outstanding for graph representation. Among the existing graph mining methods, graph kernel [33], [34], [35] has obtained unanimous praise in recent years, whereas the bottleneck is its high computational cost. As a winner from competitions on a plenty of machine learning problems, convolutional neural network (CNN) has attracted lots of attention, especially in the area of computer vision [36], and it has been reformulated by the new convolution operator for graph structure data [37]. It was put forward in [38], referred to as graphconv, the first trial of an analogy of CNN on graphs. Then, graph convolutional network (GCN), designed in [39] as an extension to the k-localized kernel, resolved the problem of over localization as compared with graphconv. Based on graph neural network (GNN) and capsule, Zhang *et al.* [40] designed the CapsGNN, which can generate multiple embeddings for each graph to capture network properties from different aspects. This method was extensively tested, and achieve the state-of-the-art results.

Network algorithms benefit from graph embedding by automatically extracting features of arbitrary dimensions. However, such methods still largely rely on the original network, and thus may ignore important hidden structural features. To bridge the gap, we map the original network to different structural spaces, in terms of different SGNs. Different from those existing subgraph-based methods [12], [15], [17] that only enumerate a set of motifs as functional building blocks and then match them in the original network for

subsequent representation, our SGN model maps the subgraphs in the original network to the nodes in a higher-order structural space, addressing the connections between the subgraphs. Therefore, it can be considered that SGN provides a general framework to expand the structural feature space, which can be naturally integrated into many graph representation methods to further improve their effectiveness.

3 Subgraph Networks

Generally, SGN can be considered as a mapping in network space, which maps the original node-level network to subgraph-level networks. In this section, SGN is first introduced, followed by algorithms for constructing the 1st-order and 2nd-order SGNs.

Definition 1 (Network). *An undirected network is represented by $G(V, E)$, where V and $E \subseteq (V \times V)$ denote the sets of nodes and links, respectively. The element (v_i, v_j) in E is an unordered pair of nodes v_i and v_j, i.e., $(v_i, v_j) = (v_j, v_i)$, for all $i, j = 1, 2, \ldots, N$, where N is the number of nodes, namely the size of the network.*

Definition 2 (Subgraph). *Given a network $G(V, E)$, $g_i = (V_i, E_i)$ is a subgraph of G, denoted by $g_i \subseteq G$ if and only if $V_i \subseteq V$ and $E_i \subseteq E$. The sequence of subgraphs is denoted as $g = \{g_i \subseteq G | i = 1, 2, \ldots, n\}$, $n \leq N$.*

Definition 3 (SGN: Subgraph Network). *Given a network $G(V, E)$, the SGN, denoted by $G^* = L(G)$, is a mapping from G to $G^*(V^*, E^*)$, with the sets of nodes and links denoted by $V^* = \{g_j | j = 0, 1, \ldots, n\}$ and $E^* \subseteq (V^* \times V^*)$, respectively. Two subgraphs g_i and g_j are connected if they share some common nodes or links in the original network, i.e., $V_i \cap V_j \neq \emptyset$. Similarly, the element (g_i, g_j) in E^* is an unordered pair of subgraphs g_i and g_j, i.e., $(g_i, g_j) = (g_j, g_i)$, $i = 1, 2, \ldots, n$ with $n \leq N$.*

According to the definition of SGN, one can see that: (i) subgraph is a part of the original network; (ii) SGN is derived from a higher-order mapping of the original network G; (iii) the connecting rule between two subgraphs needs to be clarified. Following the approach of [41], where the problem of graph representation in a domain with higher-order relations is discussed, constructing sets of nodes as p-chains, corresponding to points (0-chains), lines (1-chains), triangles (2-chains), etc., here the new framework constructs subgraphs as 1st order, 2nd order, etc. For clarity, three steps in building the new framework are outlined as follows.

- Detecting subgraphs from the original network. Networks are rich of subgraph structures, with some subgraphs occurring frequently, e.g., motifs [20]. Different kinds of networks may have different local structures, captured by different distributions of various subgraphs.
- Choosing appropriate subgraphs. Generally, subgraphs should not be too large, since in this case SGN may only contain a very small number of nodes, making the subsequent analysis less meaningful. Moreover, the chosen subgraphs should be connected to each other, i.e., they should share some common part (nodes or links) of the original network, so that higher-order structural information can emerge.
- Utilizing the subgraphs to build SGN. After extracting enough subgraphs from the original network, connections among them are established following certain rules so as to build SGN. Here, for simplicity, consider two subgraphs. They are connected if and only if they share the same nodes or links from the original network. There certainly can be other connecting rules, leading to totally different SGNs, which will be discussed elsewhere in the future.

In this paper, the most fundamental subgraphs, i.e., line and triangle, are chosen as subgraphs, since they are simple and relatively frequently appearing in most networks. Thus, two kinds of SGNs of different orders are constructed as follows.

3.1 First-Order SGN

In the case of first-order, a line, or a link, is chosen as a subgraph, based on which SGN is built, denoted by $\text{SGN}^{(1)}$. The 1st-order SGN is also known as a line graph, where the nodes are the links in the original network, and two nodes are connected if the corresponding links share a same end node.

Algorithm 1. Constructing $\text{SGN}^{(1)}$

Input: A network $G(V, E)$ with node set V and link set $E \subseteq (V \times V)$.
Output: $\text{SGN}^{(1)}$, denoted by $G'(V', E')$.

```
Initialize a node set V' and a link set E';
for each v ∈ V do
    get the neighbor set Ω of v;
    for each ω ∈ Ω do
        ℓ = sorted([v, ω]);
        ℓ_str ← merge the nodes in list ℓ into a string;
        add the new node ℓ_str into Ṽ;
    end
    for i, j ∈ Ṽ and i ≠ j do
        add the link (i, j) into E';
    end
    add Ṽ into V';
end
return G'(V', E');
```

The process to build $\text{SGN}^{(1)}$ from a given network is shown in Fig. 1. In this example, the original network has 6 nodes connected by 6 links. First, extract lines as subgraphs, labeled them by their corresponding end nodes, as shown in Fig. 1b. These lines are treated as nodes in SGN. Then, connect these lines based on their labels, i.e., two lines are connected if they share one same end node, as shown in Fig. 1c. Finally, obtain SGN with 6 nodes and 8 links, as shown in Fig. 1d. A pseudocode of constructing $\text{SGN}^{(1)}$ is given in Algorithm 1. The input of this algorithm is the original network $G(V, E)$ and the output is the constructed $\text{SGN}^{(1)}$, denoted by $G'(V', E')$, where V' and E' represent the sets of nodes and links in the $\text{SGN}^{(1)}$, respectively.

3.2 Second-Order SGN

Now, construct higher-order subgraphs by considering the connection patterns among three nodes. There are more diverse connection patterns among three nodes than the case of two nodes. In theory, there are 13 possible non-

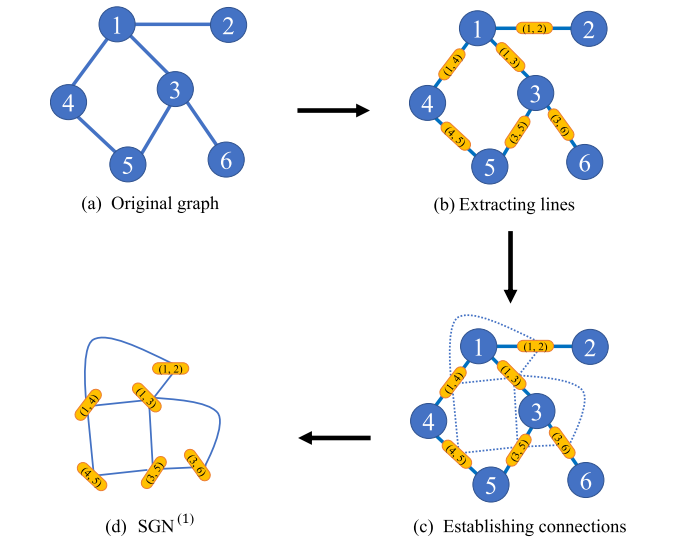

Fig. 1. The process of building $\text{SGN}^{(1)}$ from a given network: (a) The original network, (b) extracting lines as subgraphs, (c) establishing connections among these lines, and (d) forming $\text{SGN}^{(1)}$.

isomorphic connection patterns among three nodes [20] in a directed network, as shown in Fig. 2a. This number decreases to 2 in an undirected network, namely only open and closed triangles, as shown in Fig. 2b. Here, only connected subgraphs are considered, while those with less than two links are ignored. Compared with lines, triangles can provide more insights about the local structure of a network [42]. For instance, in [43], the evolution of triangles in a Google+ online social network was studied, obtaining some valuable information during the emerging and pruning of various triangles.

The open triangles are defined as subgraphs to establish the 2nd-order SGN, denoted by $\text{SGN}^{(2)}$. Here, second-order means that there are two links in each open triangle, and two open triangles are connected in $\text{SGN}^{(2)}$ if they share a same link. Note that *same link* rather than *same node* is used here to avoid obtaining a very dense $\text{SGN}^{(2)}$. This is because a dense network, with each pair of nodes connected with a higher probability, tends to provide less structural information in general.

The iterative process to build $\text{SGN}^{(2)}$ from an original network is shown in Fig. 3. First, extract lines, labeled by their corresponding end nodes, as shown in Fig. 3b, to establish $\text{SGN}^{(1)}$. Then, in the line graph $\text{SGN}^{(1)}$, further extract lines to obtain open triangles as subgraphs, labeled

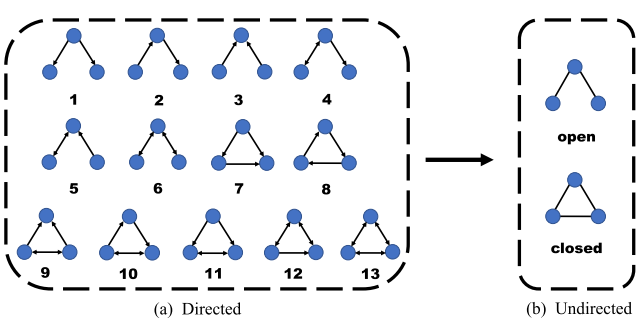

Fig. 2. The connection patterns among three nodes for (a) directed and (b) undirected networks.

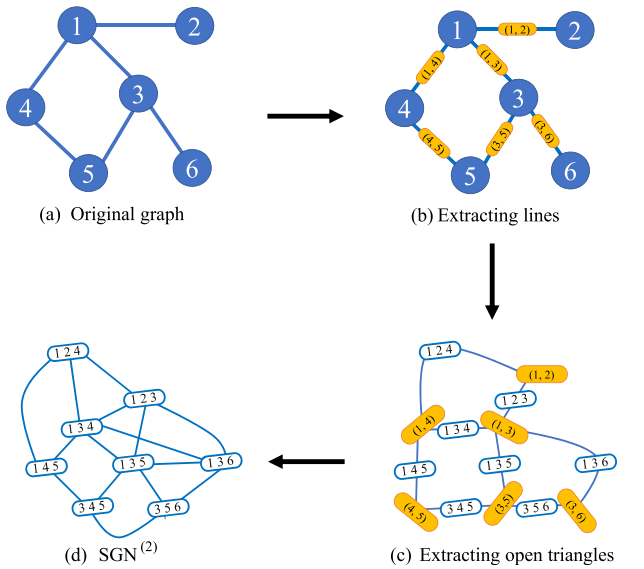

Fig. 3. The process to build $\text{SGN}^{(2)}$ from a given network: (a) The original network, (b) extracting lines, (b) building $\text{SGN}^{(1)}$ and extracting open triangles as subgraphs, and (d) establishing connections among these open triangles to obtain $\text{SGN}^{(2)}$.

by their corresponding three nodes, as shown in Fig. 3c. Finally, obtain $\text{SGN}^{(2)}$ with 8 nodes and 14 links, as shown in Fig. 3d. A pseudocode of constructing $\text{SGN}^{(2)}$ is given in Algorithm 2. The input of this algorithm is the original network $G(V, E)$ and the output is the constructed $\text{SGN}^{(2)}$, denoted by $G''(V'', E'')$, where V'' and E'' represent the sets of nodes and links in the $\text{SGN}^{(2)}$, respectively.

Algorithm 2. Constructing $\text{SGN}^{(2)}$

Input: A network $G(V, E)$ with node set V and link set $E \subseteq (V \times V)$.
Output: $\text{SGN}^{(2)}$, denoted by $G''(V'', E'')$.

```
Initialize a node set V'' and a link set E'';
for each v ∈ V do
    get the neighbors set Ω of v;
    Ω̃ ← get the full combination of node pairs in the neighbor collection;
    for each (ω1, ω2) ∈ Ω̃ do
        ℓ̃ = [v, ω1, ω2];
        ℓ̃_str ← merge the nodes in list ℓ̃ into a string;
        add the new node ℓ̃_str into Ṽ;
    end
    for i, j ∈ Ṽ and i ≠ j do
        add the edge (i, j) into E'' ;
    end
    add Ṽ into V'';
end
return G''(V'', E'');
```

Clearly, the new method can be easily extended to construct higher-order SGNs by choosing proper subgraphs and connecting rules. For instance, based on Algorithms 1 and 2, for the network shown in Fig. 3d, one can further label each link by the 4 numbers from the end nodes, i.e., these numbers correspond to the 4 nodes in the original network. Then, one can treat each link with a different label as a node, and connect them if they share 3 same numbers, so as to establish the 3rd-order SGN.

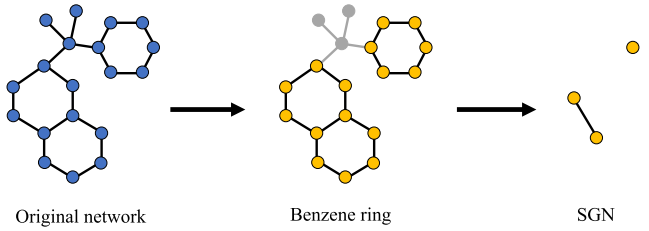

Fig. 4. A compound network, where each node denotes an atom and its corresponding SGN obtained by taking benzene rings as subgraphs.

It is interesting to investigate such a higher-order SGN. However, as subgraphs become too large, the SGN may contain only few nodes, making the network structure less informative. It may be argued that there might be some functional subgraphs in certain networks, which could be better blocks to be used to build SGNs. However, this may not be true. Take the compound networks in chemistry as examples, e.g., benzene ring, and other functional groups such as hydroxyl group, carboxyl group and aldehyde group, which play an important role in the properties of organic substances. In such networks, however, one usually cannot choose the benzene ring as a building block, since most of these networks are of small sizes and contain a small number of benzene rings, as shown in Fig. 4. In this case, if one uses benzene rings as subgraphs, an SGN will be built containing only three nodes, with one isolated from the other two. As such, this SGN can hardly provide sufficient information to distinguish itself from the other substances, hence will not be useful.

4 Network Attributes

Now, besides the original network, denoted by $\text{SGN}^{(0)}$ for simplicity, there are two SGNs, i.e., $\text{SGN}^{(1)}$ and $\text{SGN}^{(2)}$. These networks together may provide more comprehensive structural information for subsequent applications. In this paper, the focus is on its application to network classification. A typical procedure for accomplishing the task consists of two steps: first, extract network structural features; second, design a machine learning method based on these features to realize the classification. In network science, there are many classic topological attributes, which have been widely used in link prediction [44], graph classification [45] and so on. Here, the following handcrafted network features are used to design the classifier.

- *Number of Nodes (N)*: Total number of nodes in the network.
- *Number of links (L)*: Total number of links in the network.
- *Average degree (K)*: The mean value of links connected to a node in the network.
- *Percentage of leaf nodes (P)*: A node of degree 1 is defined as a leaf node. Suppose there are totally F leaf nodes in the network. Then

$$P = \frac{F}{N}\,. \tag{1}$$

- *Average clustering coefficient (C)*: For node v_i, the clustering coefficient represents the probability of a connection between any two neighbors of v_i. Suppose that there are k_i neighbors of v_i and these nodes are connected by L_i links. Then, the average clustering coefficient is defined as

$$C = \frac{1}{N}\sum_{i=1}^{N}\frac{2L_i}{k_i(k_i-1)}\,. \tag{2}$$

- *Largest eigenvalue of the adjacency matrix (λ)*. The adjacency matrix A of the network is an $N \times N$ matrix, with its element $a_{ij} = 1$ if nodes v_i and v_j are connected, and $a_{ij} = 0$ otherwise. In this step, calculate all the eigenvalues of A and choose the largest one.
- *Network density (D)*. Given the number of nodes N and the number of links L, network density is defined as

$$D = \frac{2L}{N(N-1)}\,. \tag{3}$$

- *Average betweenness centrality (C_B)*. Betweenness centrality is a centrality metric based on shortest paths. The average betweenness centrality of the network is defined as

$$C_B = \frac{1}{N}\sum_{i=1}^{N}\sum_{s\neq i\neq t}\frac{n_{st}^{i}}{g_{st}}, \tag{4}$$

where g_{st} is the number of shortest paths between v_s and v_t, and n_{st}^{i} is the number of shortest paths between v_s and v_t that pass through v_i.
- *Average closeness centrality (C_C)*. The closeness centrality of a node in a connected network is defined as the reciprocal of the average shortest path length between this node and the others. The average closeness centrality is defined as

$$C_C = \frac{1}{N}\sum_{i=1}^{N}\frac{n-1}{\sum_{j=1}^{n} d_{ij}}\,, \tag{5}$$

where d_{ij} is the shortest path length between nodes v_i and v_j.
- *Average eigenvector centrality (C_E)*. Usually, the importance of a node depends not only on its degree but also on the importance of its neighbors. Eigenvector centrality is another measure of the importance of a node based on its neighbors, which is defined as

$$C_E = \frac{1}{N}\sum_{i=1}^{N} x_i\,, \tag{6}$$

where x_i represents the importance of node v_i and is calculated based on the following equation:

$$x_i = \alpha\sum_{j=1}^{N} a_{ij}x_j\,, \tag{7}$$

where α is a preset parameter, which should be less than the reciprocal of the maximum eigenvalue of the adjacency matrix A.

TABLE 1
Basic Statistics of the 7 Datasets

Dataset	#Graphs	#Classes	#Positive	#Negative
MUTAG	188	2	125	63
PTC	344	2	152	192
PROTEINS	1113	2	663	450
NCI1	4110	2	2057	2053
NCI109	4127	2	2079	2048
IMDB-B	1000	2	500	500
REDDIT-B	2000	2	1000	1000

#Graphs is the number of graphs. #Classes is the number of classes. #Positive and #Negative are the numbers of graphs in the two different classes.

- *Average neighbor degree (D_N).* Neighbor degree of a node is the average degree of all the neighbors of this node, which is defined as

$$D_N = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{k_i}\sum_{v_j\in\Omega_i} k_j\,, \tag{8}$$

where Ω_i is a set of the neighbors of node v_i, and k_j is the degree of node $v_j \in \Omega_i$.

Note that, among the above 11 features, number of nodes (N), number of links (L), average degree (K) and network density (D) are the most basic properties of a network [46]. Average clustering coefficient (C) [47] is also a very popular metric to quantify the link density in ego networks. The percentage of leaf nodes (P) can distinguish whether a network is tree-like or rich with rings. The largest eigenvalue of the adjacency matrix (λ) is chosen since the eigenvalues are the isomorphic invariant of a graph, which can be used to estimate many static attributes, such as connectivity, diameter, etc. Average neighbor degree (D_N) captures the 2-hop information. Also, centrality measures are indicators of the importance (status, prestige, standing, and the like) of a node in a network, therefore, we also use average betweenness centrality (C_B), average closeness centrality (C_C), and average eigenvector centrality (C_E) to describe the global structure of a network.

4.1 Datasets

Experiments were conducted on 7 real-world network datasets, as introduced in the following, with each containing two classes of networks. The first 5 datasets are about bio- and chemo-informatics, while the last two are social networks. The basic statistics of these datasets are presented in Table 1.

- *MUTAG*. This dataset is about heteroaromatic nitro and mutagenic aromatic compounds, with nodes and links representing atoms and the chemical bonds between them, respectively. They are labeled according to whether there is a mutagenic effect on a special bacteria [48].
- *PTC*. This dataset includes 344 chemical compound graphs, with nodes and links representing atoms and the chemical bonds between them, respectively. Their labels are determined by their carcinogenicity for rats [49].

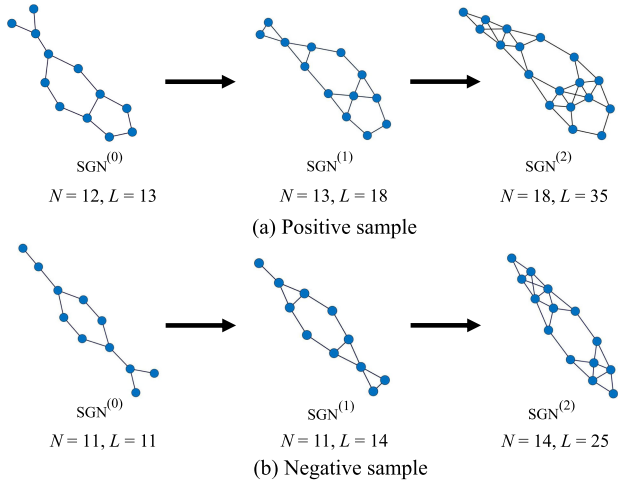

Fig. 5. $\mathrm{SGN}^{(0)}$, $\mathrm{SGN}^{(1)}$ and $\mathrm{SGN}^{(2)}$ as well as the numbers of nodes and links for (a) positive and (b) negative samples in the MUTAG dataset.

- *PROTEINS*. This dataset comprises of 1,113 graphs. The nodes are Secondary Structure Elements (SSEs) and the links are neighbors in the amino-acid sequence or in the 3D space. These graphs represent either enzyme or non-enzyme proteins [50].
- *NCI1 & NCI109*. These two datasets comprise of 4,110 and 4,127 graphs, respectively. The nodes and links represent atoms and chemical bonds between them, respectively. They are two balanced subsets of the datasets of chemical compounds screened for the activities against non-small cell lung cancer and ovarian cancer cell lines, respectively. The positive and negative samples are distinguished according to whether they are effective against cancer cells [2].
- *IMDB-B*. This dataset is about movie collaboration, which is collected from IMDB, containing lots of information about different movies. Each graph is an ego-network, where nodes represent actors or actresses and links indicate whether they appear in the same movie. Each graph is categorized into one of the two genres (Action and Romance) [3].
- *REDDIT-B*. This dataset is crawled from Reddit, which is composed of submission graphs from popular subreddits. Each graph corresponds to an online discussion thread, where nodes are users, and there is an link between two nodes if one of them responded to the other's comments. The four popular subreddits are IAmA, AskReddit, TrollXChromosomes and atheism. There are also two categories of graphs: IAmA and AskReddit are two QA-based subreddits and TrollXChromosomes and atheism are two discussion-based subreddits [35].

4.2 Benefits of SGN

Here, take the MUTAG dataset as an example to show that SGNs of different orders may capture different aspects of a network structure.

First, a positive sample and a negative one are chosen from the MUTAG dataset, with their $\mathrm{SGN}^{(0)}$, $\mathrm{SGN}^{(1)}$ and $\mathrm{SGN}^{(2)}$ visualized in Fig. 5. To facilitate a comparison, the numbers of nodes and links of these networks are also

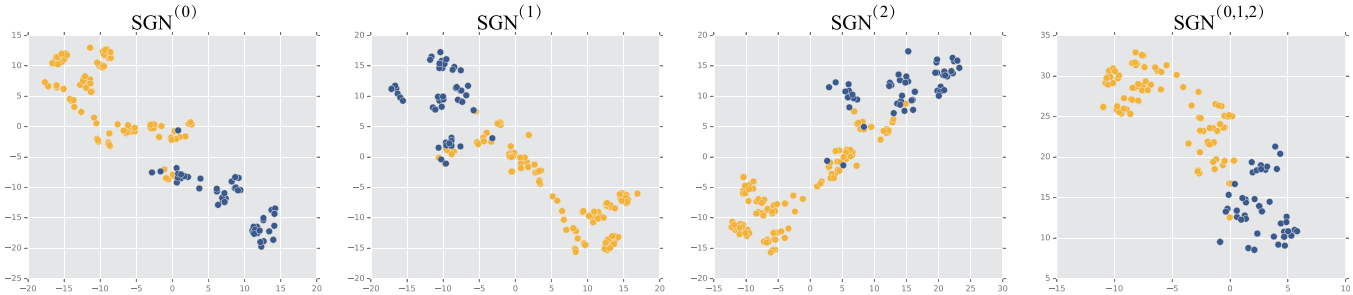

Fig. 6. The t-SNE visualization of handcrafted network features. The same color of points represent the same class of networks in MUTAG.

presented in the figure. Here, a positive sample means that this compound has mutagenic effect on the bacteria; otherwise, it is negative. As can be seen, although the original networks of the two samples have quite similar sizes, their difference is seemingly enlarged in the higher-order SGNs; more precisely, the numbers of nodes and links in SGN increase faster for the positive sample than the negative one as the order increases.

Then, the handcrafted network features are visualized by using t-SNE in Fig. 6, where the networks in MUTAG can indeed be distinguished to a certain extent by these features of the original network, the 1st-order SGN and the 2nd-order SGN, respectively. Moreover, when all the features are put together, it appears that these networks can be better distinguished, indicating that SGNs of different orders and the original network may complement to each other. Therefore, integrating the structural information of all these networks may significantly improve the performances of the subsequent algorithms designed based on network structures.

5 Experiments

With the rapid growth of real-world graph data, network classification is becoming more and more important, and a number of effective network classification methods [51], [52], [53] have been proposed in recent years. Along this line of research, as an application of the proposed SGN, classifiers are designed based on the structural features obtained from SGNs as well as from the original networks.

5.1 Automatic Feature Extraction Methods

Besides those handcrafted features, one can also use some advanced methods, such as network embedding methods, to automatically generate a feature vector of certain dimension from the given network. Under the present framework, such automatically generated feature vectors can also be further expanded based on SGNs.

Network embedding method, graph2vec, and two graph kernel-based methods, subtree kernel WL and deep WL methods, and depth model algorithm CapsGNN, are chosen as automatic feature extraction methods.

- *Graph2vec* [32]. This is the first unsupervised embedding approach for an entire network, which is based on the extending word-and-document embedding techniques that has shown great advantages in NLP. Similarly, graph2vec establishes the relationship between a network and the rooted subgraphs using a similar model to *doc2vec* [31]. Graph2vec first extracts rooted subgraphs and provides corresponding labels into the vocabulary, and then trains a skip-gram model to obtain the representation of the entire network.
- *WL* [34]. This is a rapid feature extraction scheme based on the Weisfeiler-Lehman (WL) test for isomorphism on graphs. It maps the original network to a sequence of graphs, with node attributes capturing both topological and label information. The key idea of the algorithm is to augment the node labels by the sorted set of node labels of neighboring nodes, and compress these augmented labels into new and short labels. These steps are then repeated until the node label sets of the two compared networks differ, or the number of iterations reaches a preset value. It should be noted that, to facilitate the expansion of the new model, the sub-structure frequency vectors, instead of the kernel matrix $\mathcal{K}$, are used as the inputs to the new classifier.
- *Deep WL* [35]. This provides a unified framework that leverages the dependency information of sub-structures by learning latent representations. The differences from the WL kernel generate a corpus of sub-structures by integrating language-modeling and deep-learning techniques [54], where a co-occurrence relationship of sub-structures is preserved and sub-structure vector representations are obtained before the kernel is computed. Then, a sub-structure similarity matrix, $\mathcal{M}$, is calculated by the matrix $\mathcal{V}$ with each column representing a sub-structure vector. Denote by $\mathcal{P}$ the matrix with each column representing a sub-structure frequency vector. Then, according to the definition of kernel

$$\mathcal{K} = \mathcal{P}\mathcal{M}\mathcal{P}^{\mathrm{T}} = \mathcal{P}\mathcal{V}\mathcal{V}^{\mathrm{T}}\mathcal{P}^{\mathrm{T}} = \mathcal{H}\mathcal{H}^{\mathrm{T}}, \tag{9}$$

  one can use the columns in the matrix $\mathcal{H} = \mathcal{P}\mathcal{V}$ as the inputs to the classifier.
- *CapsGNN* [40]: This method was inspired by CapsNet, which adopted the concept of capsules to overcome the weakness of existing GNN-based graph embedding algorithms. In particular, CapsGNN extracts node features in the form of capsules and utilizes the routing mechanism to capture important information at the graph level. The model generates multiple embeddings for each graph so as to capture graph properties from different aspects.

TABLE 2
Classification Results on the 7 Datasets, in Terms of F_1-$Score$, Based on Different Feature Extraction Methods and Combinations of SGNs

Algorithm	Dataset						
Handcraft	MUTAG	PTC	PROTEINS	NCI1	NCI109	IMDB-B	REDDIT-B
$SGN^{(0)}$	86.58 ± 3.61	63.52 ± 4.55	78.30 ± 2.49	67.48 ± 0.87	67.34 ± 1.25	73.00 ± 3.68	78.68 ± 1.66
$SGN^{(1)}$	88.20 ± 3.62	65.29 ± 6.93	76.79 ± 3.41	65.72 ± 1.41	66.25 ± 2.14	73.30 ± 4.82	76.50 ± 2.73
$SGN^{(2)}$	85.53 ± 4.47	65.00 ± 6.09	75.45 ± 5.04	65.35 ± 2.32	64.15 ± 2.20	74.24 ± 3.38	74.37 ± 3.14
$SGN^{(01)}$	87.89 ± 4.58	66.47 ± 6.73	78.83 ± 3.12	68.76 ± 2.24	68.88 ± 2.17	73.38 ± 3.94	79.15 ± 2.32
$SGN^{(0,2)}$	88.42 ± 4.22	65.59 ± 7.09	78.92 ± 3.17	69.39 ± 1.82	68.09 ± 1.74	75.42 ± 3.34	78.80 ± 2.07
$SGN^{(1,2)}$	88.95 ± 3.37	67.06 ± 6.14	78.21 ± 3.60	68.13 ± 1.30	68.48 ± 1.42	74.94 ± 2.81	77.23 ± 1.98
$SGN^{(0,1,2)}$	**91.58** ± 4.21	**67.94** ± 6.36	**79.46** ± 2.96	**69.84** ± 1.59	**69.73** ± 1.97	**77.65** ± 4.50	**79.23** ± 1.62
Gain	5.78%	6.96%	1.71%	3.50%	3.55%	6.37%	0.70%
Graph2vec	MUTAG	PTC	PROTEINS	NCI1	NCI109	IMDB-B	REDDIT-B
$SGN^{(0)}$	83.15 ± 9.25	60.17 ± 6.86	73.30 ± 2.05	73.22 ± 1.81	74.26 ± 1.47	62.47 ± 3.99	76.00 ± 2.20
$SGN^{(1)}$	63.16 ± 4.68	56.80 ± 5.39	60.27 ± 2.05	54.56 ± 1.38	56.35 ± 1.52	63.06 ± 6.72	75.34 ± 2.55
$SGN^{(2)}$	68.95 ± 8.47	57.35 ± 3.83	59.82 ± 4.11	61.31 ± 2.13	53.54 ± 1.43	64.35 ± 6.63	74.50 ± 2.71
$SGN^{(0,1)}$	83.42 ± 5.40	59.03 ± 3.36	74.12 ± 1.57	73.65 ± 1.38	73.18 ± 1.26	66.59 ± 4.54	77.63 ± 1.25
$SGN^{(0,2)}$	81.32 ± 3.80	61.76 ± 3.73	73.09 ± 1.28	**77.54** ± 2.52	**75.39** ± 1.33	66.53 ± 4.45	77.39 ± 3.10
$SGN^{(1,2)}$	72.63 ± 4.08	59.42 ± 5.84	62.76 ± 3.49	67.47 ± 3.84	68.12 ± 1.86	66.24 ± 2.58	76.00 ± 3.18
$SGN^{(0,1,2)}$	**86.84** ± 5.70	**63.24** ± 6.70	**74.44** ± 3.09	76.64 ± 3.21	74.86 ± 2.76	**70.65** ± 5.55	**78.04** ± 2.61
Gain	4.44%	5.10%	1.56%	5.90%	1.52%	13.73%	2.68%
WL	MUTAG	PTC	PROTEINS	NCI1	NCI109	IMDB-B	REDDIT-B
$SGN^{(0)}$	80.63 ± 3.07	56.91 ± 2.79	72.92 ± 0.56	66.19 ± 0.97	69.26 ± 1.14	70.90 ± 4.18	75.15 ± 2.39
$SGN^{(1)}$	76.05 ± 2.73	61.76 ± 6.17	67.04 ± 1.58	58.24 ± 1.87	57.68 ± 1.28	69.20 ± 3.87	74.83 ± 2.64
$SGN^{(2)}$	74.21 ± 7.33	59.41 ± 5.22	64.01 ± 1.58	52.62 ± 0.53	58.26 ± 0.72	66.10 ± 4.18	74.34 ± 2.65
$SGN^{(0,1)}$	87.11 ± 5.45	62.50 ± 4.15	76.19 ± 2.29	72.49 ± 1.79	69.50 ± 1.76	73.05 ± 4.75	76.90 ± 1.51
$SGN^{(0,2)}$	86.57 ± 4.31	59.71 ± 3.96	74.99 ± 1.56	70.11 ± 1.22	69.67 ± 1.34	72.23 ± 3.13	75.40 ± 4.73
$SGN^{(1,2)}$	76.11 ± 7.75	60.88 ± 3.11	64.81 ± 3.05	56.00 ± 2.35	57.92 ± 1.30	70.45 ± 3.22	74.15 ± 4.17
$SGN^{(0,1,2)}$	**88.94** ± 3.28	**63.53** ± 4.80	**78.08** ± 1.41	**77.03** ± 2.73	**72.92** ± 1.25	**75.00** ± 4.49	**77.03** ± 2.73
Gain	10.31%	11.63%	7.08%	11.63%	5.28%	5.78%	2.50%
Deep WL	MUTAG	PTC	PROTEINS	NCI1	NCI109	IMDB-B	REDDIT-B
$SGN^{(0)}$	82.95 ± 2.68	59.04 ± 1.09	73.30 ± 0.82	67.06 ± 1.91	67.04 ± 1.36	67.50 ± 2.45	77.25 ± 2.52
$SGN^{(1)}$	67.89 ± 6.84	58.53 ± 3.23	69.43 ± 2.57	55.45 ± 1.43	57.63 ± 2.07	73.30 ± 2.38	76.93 ± 3.68
$SGN^{(2)}$	68.42 ± 6.65	62.65 ± 4.17	68.57 ± 2.42	55.22 ± 1.45	55.68 ± 1.12	71.48 ± 2.48	75.29 ± 4.35
$SGN^{(0,1)}$	92.11 ± 5.39	64.41 ± 1.87	74.62 ± 2.51	70.10 ± 1.24	69.39 ± 1.35	73.50 ± 2.87	77.35 ± 2.47
$SGN^{(0,2)}$	93.15 ± 5.28	64.70 ± 4.88	75.89 ± 2.99	70.12 ± 1.31	68.61 ± 1.11	74.36 ± 2.22	76.92 ± 3.13
$SGN^{(1,2)}$	73.68 ± 5.77	64.16 ± 4.92	69.43 ± 2.26	59.95 ± 1.12	56.24 ± 1.62	71.90 ± 2.51	76.20 ± 5.06
$SGN^{(0,1,2)}$	**93.68** ± 5.15	**65.88** ± 5.05	**76.78** ± 2.41	**70.26** ± 1.24	**71.06** ± 1.61	**75.70** ± 1.55	**78.41** ± 1.70
Gain	12.93%	11.58%	4.75%	4.77%	6.00%	11.85%	1.50%
CapsGNN	MUTAG	PTC	PROTEINS	NCI1	NCI109	IMDB-B	REDDIT-B
$SGN^{(0)}$	86.32 ± 7.52	62.06 ± 4.25	75.89 ± 3.51	78.30 ± 1.80	72.99 ± 2.15	72.71 ± 4.36	76.12 ± 3.82
$SGN^{(1)}$	83.68 ± 8.95	61.76 ± 5.00	74.64 ± 3.55	74.70 ± 1.54	69.82 ± 2.24	74.35 ± 4.61	75.64 ± 4.15
$SGN^{(2)}$	82.63 ± 7.08	58.82 ± 3.95	72.39 ± 6.03	69.82 ± 1.89	67.04 ± 2.45	73.64 ± 4.77	72.41 ± 3.97
$SGN^{(0,1)}$	87.37 ± 8.55	63.53 ± 4.40	76.25 ± 3.53	78.42 ± 2.92	73.28 ± 3.11	74.58 ± 4.80	78.49 ± 2.93
$SGN^{(0,2)}$	87.89 ± 5.29	62.20 ± 6.14	73.00 ± 3.17	73.78 ± 2.32	71.52 ± 2.09	74.94 ± 4.56	78.17 ± 5.13
$SGN^{(1,2)}$	78.95 ± 8.49	59.11 ± 5.65	70.09 ± 2.45	70.53 ± 2.45	70.64 ± 2.30	75.29 ± 4.08	75.73 ± 4.97
$SGN^{(0,1,2)}$	**89.47** ± 7.44	**64.12** ± 3.67	**76.34** ± 4.13	**78.61** ± 1.87	**73.72** ± 2.39	**76.47** ± 5.74	**79.68** ± 5.34
Gain	4.65%	3.32%	0.59%	0.40%	1.00%	5.17%	4.87%

In this study, for *graph2vec*, the embedding dimension is adopted according to [32]. Graph2vec is based on the rooted subgraphs which are adopted in the WL kernel. The parameter height of *WL kernel* is set to 3. Since the embedding dimension is predominant for learning performances, a commonly-used value of 1,024 is adopted. The other parameters are set to defaults: the learning rate is set to 0.5, the batch size is set to 512 and the epochs is set to 1,000. For *WL* and *Deep WL*, according to [35], the Weisfelier-Lehman subtree kernel is used to built the corpus and the height of which is set to 2. Then, the Maximum Likelihood Estimation (MLE) is used to compute the kernel in the WL method. Furthermore, the same parameter setting as WL is chosen, with the embedding dimension equal to 10, window size equal to 5 and skip-gram used for the word2vec model in the deep WL method. We adopt the default parameters for *CapsGNN* and flatten the multiple embeddings of each graph as the input.

Without loss of generality, the well-known logistic regression is chosen as the new classification model. Meanwhile, for each feature extraction method, the feature space is first expanded by using SGNs, and then the dimension of the feature vectors is reduced to the same value as that of the feature vector obtained from the original network using PCA in the experiments, for a fair comparison. Each dataset is randomly split into 9 folds for training and 1 fold for testing. Here, the F_1-$Score$ is adopted as the metric to evaluate the classification performance

$$F_1 = \frac{2PR}{P+R}\,, \tag{10}$$

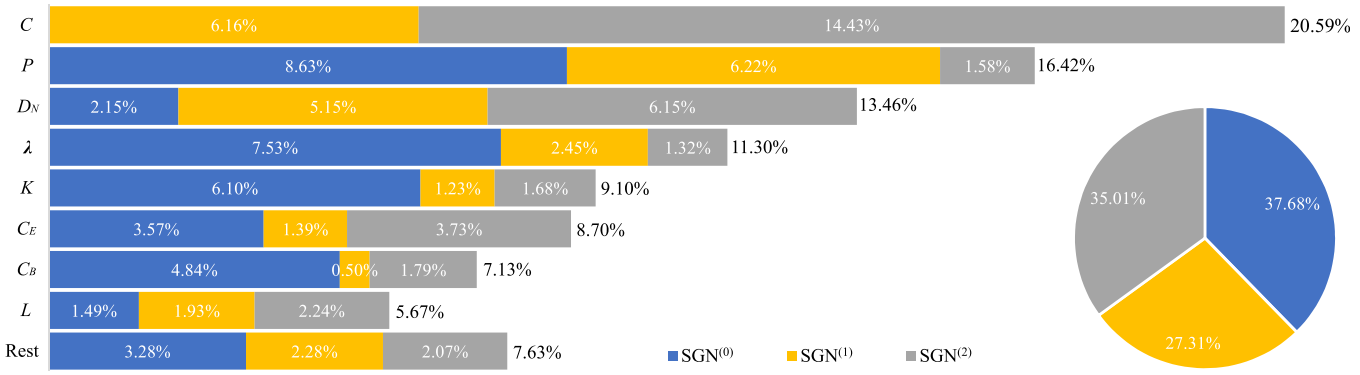

Fig. 7. The importance of handcrafted features in logistic regression model for network classification using $\text{SGN}^{(0)}$, $\text{SGN}^{(1)}$ and $\text{SGN}^{(2)}$ together in MUTAG dataset.

where P and R are the precision and recall, respectively. To exclude the random effect of the fold assignment, experiment is repeated for 500 times and then the average F_1-*Score* and its standard deviation are recorded.

5.2 Computational Complexity

Now, the computational complexity in building SGNs is analyzed. Denote by $|V|$ and $|E|$ the numbers of nodes and links, respectively, in the original network. The average degree of the network is calculated by

$$K = \frac{1}{|V|}\sum_{i=1}^{|V|} k_i = \frac{2|E|}{|V|}\,, \tag{11}$$

where k_i is the degree of node v_i. Based on Algorithm 1, the time complexity in transforming the original network to $\text{SGN}^{(1)}$ is

$$\mathcal{T}_1 = \mathcal{O}(K|V| + |E|^2) = \mathcal{O}(|E|^2 + |E|) = \mathcal{O}(|E|^2)\,. \tag{12}$$

Then, the number of nodes in $\text{SGN}^{(1)}$ is equal to $|E|$ and the number of links is $\sum_{i=1}^{|V|} k_i^2 - |E| \le |E|^2 - |E|$ [18]. Similarly, one can get the time complexity in transforming $\text{SGN}^{(1)}$ to $\text{SGN}^{(2)}$, as

$$\mathcal{T}_2 \le \mathcal{O}((|E|^2 - |E|)^2) = \mathcal{O}(|E|^4)\,. \tag{13}$$

5.3 Experiment Results

As described in Section 3, the proposed SGNs can be used to expand structural feature spaces. To investigate the effectiveness of the 1st-order and the 2nd-order SGNs, i.e., $\text{SGN}^{(1)}$ and $\text{SGN}^{(2)}$, for each feature extraction method, the classification results are compared on the basis of only one network, i.e., $\text{SGN}^{(0)}$, $\text{SGN}^{(1)}$ and $\text{SGN}^{(2)}$, respectively; on the basis of two networks, i.e., $\text{SGN}^{(0)}$ together with $\text{SGN}^{(1)}$ and $\text{SGN}^{(0)}$ together with $\text{SGN}^{(2)}$, denoted by $\text{SGN}^{(0,1)}$ and $\text{SGN}^{(0,2)}$, respectively; and on the basis of three networks, i.e., $\text{SGN}^{(0)}$ together with $\text{SGN}^{(1)}$ and $\text{SGN}^{(2)}$, denoted as $\text{SGN}^{(0,1,2)}$. For a fair comparison, PCA is used to compress the feature vectors to the same dimension for each feature extraction method, before they are input into the logistic regression model.

The results are shown in Table 2, where one can see that, for a single network case, the original network seems to provide more structural information, i.e., the classification model based on $\text{SGN}^{(0)}$ performs better, in terms of higher F_1-*Score*, than those based on $\text{SGN}^{(1)}$ or $\text{SGN}^{(2)}$, in most cases. This is reasonable, because there must be information loss in the processes to build SGNs. However, it still appears to be dependent on the feature extraction method used. For instance, when the Deep WL is adopted, better classification results can be obtained based on $\text{SGN}^{(1)}$ or $\text{SGN}^{(2)}$ than $\text{SGN}^{(0)}$ for 2 datasets, while when handcrafted features are used, even better classification performance is realized based on the 1st-order or 2nd-order SGNs than the original network in 3 datasets. More interestingly, the classification models based on two networks, i.e., $\text{SGN}^{(0,1)}$ and $\text{SGN}^{(0,2)}$, perform better than those based on a single network, while the model based on three networks, i.e., $\text{SGN}^{(0,1,2)}$, performs the best in most cases.

The gain $\mathcal{G}$ on F_1-*Score* is calculated, when all the three networks are used together, i.e., $\text{SGN}^{(0,1,2)}$, compared with that when only the original network is used, i.e., $\text{SGN}^{(0)}$, which is defined to be the relatively difference between their corresponding F_1-*Score*

$$\mathcal{G} = \frac{F_1^{(0,1,2)} - F_1^{(0)}}{F_1^{(0)}} \times 100\%\,. \tag{14}$$

The gains are also presented in Table 2, where one can see that the classification performance is indeed significantly improved in all the 35 cases. Particularly, in 17 cases, the gains are larger than 5 percent, while in 7 cases, they are even larger than 10 percent. These results indicate that the 1st-order and the 2nd-order SGNs can indeed complement the original network regarding the structural information, thus benefiting network classification. Surprisingly, it is found that the chosen handcrafted features based on $\text{SGN}^{(0,1,2)}$ outperforms the other automatically generated features that use more advanced network-embedding or graph-kernel based methods even depth model, in 3 out of 7 datasets, i.e., PTC, PROTEINS and IMDB-B. This phenomenon indicates that, compared with those automatically generated ones, properly chosen traditional structural features are of particular advantage in the proposed framework, in the sense that they are not only more interpretable due to their clear physical meanings, but also equally effective in designing subsequent structure-based algorithms, e.g., for network classification.

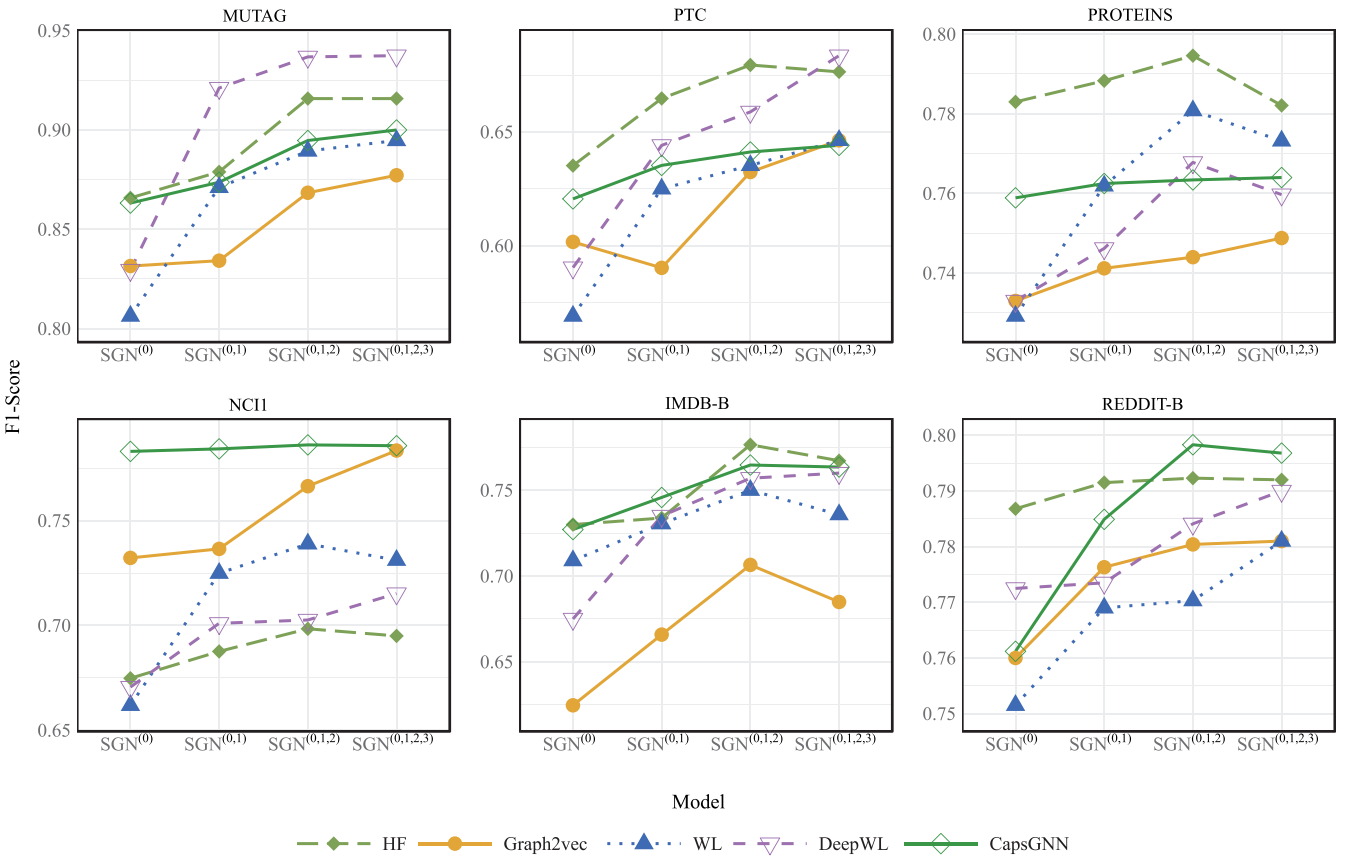

Fig. 8. Average F1-Scores obtained by using different feature extraction methods under different combinations of SGNs.

In addition, the feature importance for the task of network classification is investigated by using logistic regression. Denote by β_i the coefficient of feature x_i in the model, and suppose that there are M features in total. Then, the importance of feature x_i is defined as

$$\mathcal{I} = \frac{|\beta_i|}{\sum_{k=1}^{M} |\beta_k|} \times 100\% . \tag{15}$$

Taking MUTAG for example, the results are visualized in Fig. 7. Overall, the features in $\mathrm{SGN}^{(0)}$ are most important, since they determine 37.68 percent of the model, while the features in $\mathrm{SGN}^{(2)}$ are more important than those in $\mathrm{SGN}^{(1)}$, since they determine 35.01 and 27.31 percent of the model, respectively. When focusing on a single feature, it is found that the clustering coefficient C, the percentage of leaf nodes P, and the average neighbor degree D_N, are the top three most important features, and they together determine more than 50 percent of the model. Interestingly, it appears that different SGNs address different aspects of the network structure in the classification task. For instance, the most important feature in $\mathrm{SGN}^{(2)}$ is the clustering coefficient, while the coefficient for this feature in $\mathrm{SGN}^{(0)}$ is zero since there is no triangle in the networks in MUTAG dataset. Moreover, the largest eigenvalue of the adjacency matrix λ and the average degree K in $\mathrm{SGN}^{(0)}$ are relatively important, while those in $\mathrm{SGN}^{(1)}$ and $\mathrm{SGN}^{(2)}$ have less effect on the model. These results confirm once again that SGNs indeed complement the original network to achieve better network classification performance.

Furthermore, we also visualize the average F1-Scores obtained by using different feature extraction methods under different combinations of SGNs, as shown in Fig. 8. Note that here we also consider the third-order SGNs, in order to present the changing trends of F1-Scores with the number of SGNs more clearly. Indeed, we can find that integrating higher-order SGNs generally helps to capture more structural information, leading to higher classification performance. However, such benefit seems to be shrunk when we go further, i.e., the improvement of F1-Score from $\mathrm{SGN}^{(0,1,2)}$ to $\mathrm{SGN}^{(0,1,2,3)}$ is relatively small, while the

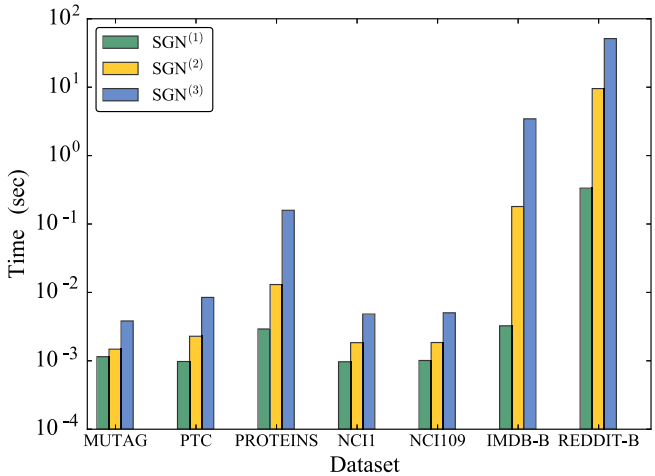

Fig. 9. Average execution time to establish SGNs of different orders on the seven datasets.

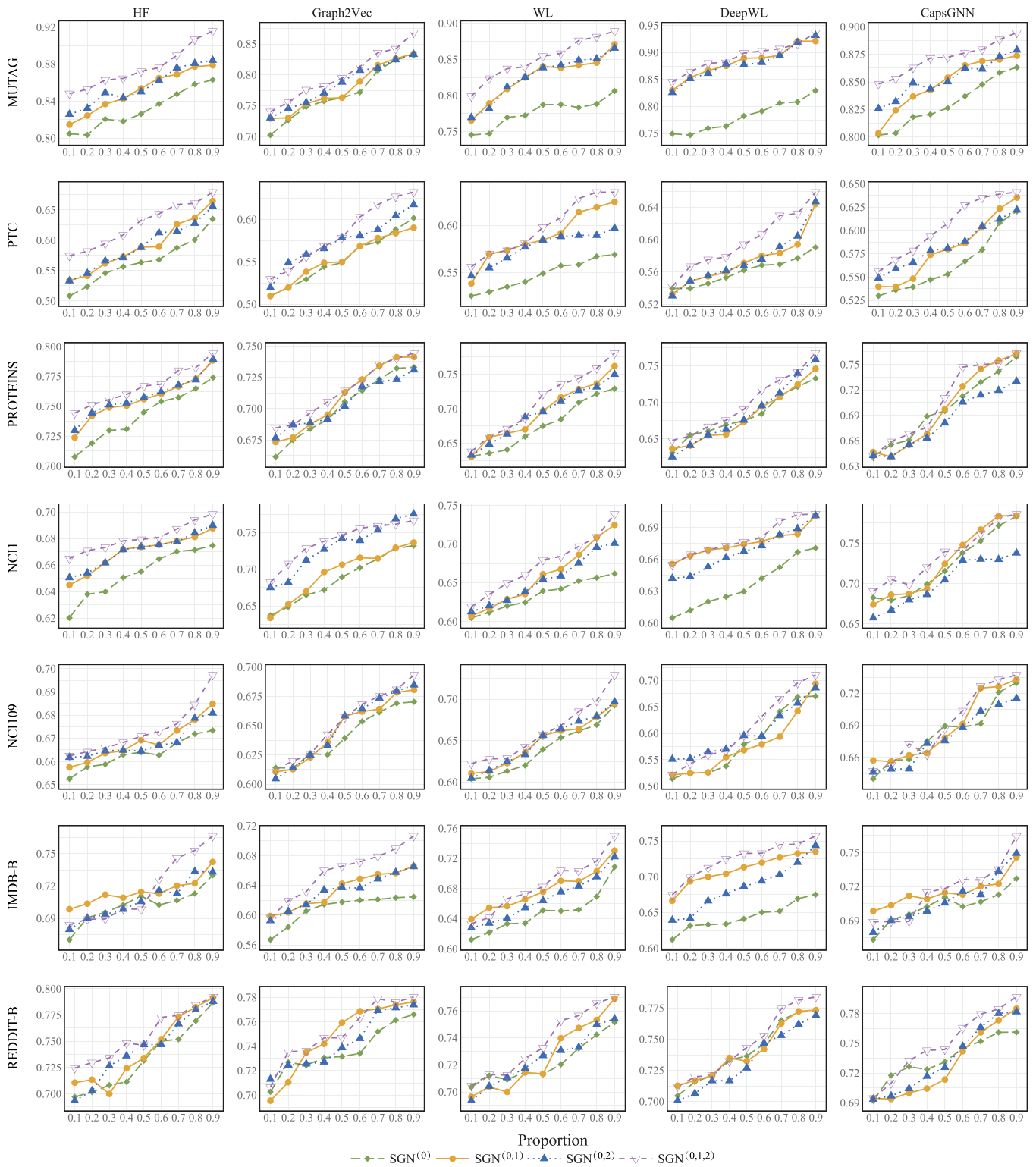

Fig. 10. Average F_1-*Score* as functions of the size of the training set (represented by the fraction of samples in the training set), for various feature extraction methods on different datasets, based on SGN$^{(0)}$, SGN$^{(0,1)}$, SGN$^{(0,2)}$ and SGN$^{(0,1,2)}$, respectively.

computational complexity increases quite fast. And this is the reason why we only consider first-order and second-order SGNs in most parts of this work.

To address the computational complexity of our method, we record the average execution time to establish SGNs of different orders on the seven datasets, including MUTAG, PTC, PROTEINS, NCI1, NCI109, IMDB-B and REDDIT-B. The results are shown in Fig. 9, where we can see that execution time increases fast as the order of SGN and the network size increase. One possible reason is that here the subgraph we chose is relatively simple, making the SGNs of higher-order even more complicated than those of lower-order. Therefore, one way to decrease the computational complexity is to choose more complex subgraphs to establish

simpler higher-order SGNs. Another way to accelerate this process is to adopt parallel computing mechanism, which will be our focus in future work.

To address the robustness of the classification model against the size variation of the training set, the F_1-$Score$ is calculated for the network classification task, using various sizes of training sets (from 10 to 90 percent, within a 10 percent interval). For each size, the training and test sets are randomly divided, which is repeated for 500 times with the average result recorded. The results are shown in Fig. 10 for various feature extraction methods on different datasets. It can be seen that the classification results based on $SGN^{(0)}$, $SGN^{(1)}$ and $SGN^{(2)}$ together are always the best, and the results based on $SGN^{(0)}$ and $SGN^{(1)}$ together, or $SGN^{(0)}$ and $SGN^{(2)}$ together, are always better than those based only on the original network $SGN^{(0)}$. This confirms that the simulation results are quite robust to the variation of the training set size. For further study, our source codes are available online.[1]

6 Conclusion

In this paper, the concept of subgraph network (SGN) is introduced, along with algorithms developed for constructing the 1st-order and 2nd-order SGNs, which can expand the structural feature space. As a multi-order graph representation method, various orders of SGNs can significantly enrich the structural information and thus benefit the network feature extraction methods to capture various aspects of the network structure. Also, the effectiveness of the 1st-order and 2nd-order SGNs are verified. Moreover, the handcrafted features, as well as the features automatically generated by network representation methods including graph2vec and kernel-based methods including Weisfeiler-Lehman (WL) and deep WL methods and CapsGNN method, are used in experiments for network classification on seven real-world datasets.

The experimental results show that the classification model based on the features of the original network together with the 1st-order and 2nd-order SGNs always performs the best, compared with those based only on a single network, either the original one, the 1st-order or the 2nd-order SGN, or those based on a pair of them. This demonstrates that SGNs can indeed complement the original network on structural information and thus benefit the subsequent network classification algorithms, no matter which feature extraction method is adopted. More interestingly, it is found that the model based on handcrafted features performs even better than those based on the features automatically generated by more advanced methods, such as graph2vec, for most datasets. This finding suggests that, in general, properly chosen structural features with clear physical meanings may be effective in designing structure-based algorithms.

Future research may focus on extracting more types of subgraphs to establish SGNs of higher diversity for both static and temporal networks, so as to capture the network structural information more comprehensively, to design consequent algorithms for network classification and perhaps other tasks as well.

1. https://github.com/GalateaWang

Acknowledgments

The authors would like to thank all the members in the IVSN Research Group, Zhejiang University of Technology for the valuable discussion about the ideas and technical details presented in this article. This work was partially supported by the National Natural Science Foundation of China under Grant 61973273 and Grant 61572439, by the Zhejiang Provincial Natural Science Foundation of China under Grant LR19F030001, and by the Hong Kong Research Grants Council under the GRF Grant CityU11200317.

References

[1] M. Walter *et al.*, "Visualization of protein interactions in living plant cells using bimolecular fluorescence complementation," *Plant J.*, vol. 40, no. 3, pp. 428–438, 2004.
[2] N. Wale, I. A. Watson, and G. Karypis, "Comparison of descriptor spaces for chemical compound retrieval and classification," *Knowl. Inf. Syst.*, vol. 14, no. 3, pp. 347–375, 2008.
[3] D. Nguyen, W. Luo, T. D. Nguyen, S. Venkatesh, and D. Phung, "Learning graph representation via frequent subgraphs," in *Proc. SIAM Int. Conf. Data Mining*, 2018, pp. 306–314.
[4] Q. Xuan, Z.-Y. Zhang, C. Fu, H.-X. Hu, and V. Filkov, "Social synchrony on complex networks," *IEEE Trans. Cybern.*, vol. 48, no. 5, pp. 1420–1431, May 2018.
[5] Q. Xuan, A. Okano, P. Devanbu, and V. Filkov, "Focus-shifting patterns of OSS developers and their congruence with call graphs," in *Proc. 22nd ACM SIGSOFT Int. Symp. Found. Softw. Eng.*, 2014, pp. 401–412.
[6] A. Mockus, R. T. Fielding, and J. D. Herbsleb, "Two case studies of open source software development: Apache and Mozilla," *ACM Trans. Softw. Eng. Methodol.*, vol. 11, no. 3, pp. 309–346, 2002.
[7] J. Kim and M. Hastak, "Social network analysis: Characteristics of online social networks after a disaster," *Int. J. Inf. Manage.*, vol. 38, no. 1, pp. 86–96, 2018.
[8] C. Fu *et al.*, "Link weight prediction using supervised learning methods and its application to yelp layered network," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 8, pp. 1507–1518, Aug. 2018.
[9] J. R. Ullmann, "An algorithm for subgraph isomorphism," *J. ACM*, vol. 23, no. 1, pp. 31–42, 1976.
[10] G. Balazsi, A.-L. Barabási, and Z. Oltvai, "Topological units of environmental signal processing in the transcriptional regulatory network of *escherichia coli*," *Proc. Nat. Academy Sci. United States America*, vol. 102, no. 22, pp. 7841–7846, 2005.
[11] J. Ugander, L. Backstrom, and J. Kleinberg, "Subgraph frequencies: Mapping the empirical and extremal geography of large graph collections," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 1307–1318.
[12] Q. Vohra, "Subgraph frequencies and network classification." 2014. [Online]. Available: http://snap.stanford.edu/class/cs224w-2014/projects2014/cs224w-76-final.pdf
[13] M. Jha, C. Seshadhri, and A. Pinar, "Path sampling: A fast and provable method for estimating 4-vertex subgraph counts," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 495–505.
[14] J. A. Grochow and M. Kellis, "Network motif discovery using subgraph enumeration and symmetry-breaking," in *Proc. Annu. Int. Conf. Res. Comput. Mol. Biol.*, 2007, pp. 92–106.
[15] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Sci.*, vol. 353, no. 6295, pp. 163–166, 2016.
[16] H. Wang *et al.*, "Incremental subgraph feature selection for graph classification," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 1, pp. 128–142, Jan. 2017.
[17] C. Yang, M. Liu, V. W. Zheng, and J. Han, "Node, motif and subgraph: Leveraging network functional blocks through structural convolution," in *Proc. IEEE/ACM Int. Conf. Advances Social Netw. Anal. Mining*, 2018, pp. 47–52.
[18] F. Harary and R. Z. Norman, "Some properties of line digraphs," *Rendiconti del Circolo Matematico di Palermo*, vol. 9, no. 2, pp. 161–168, 1960.
[19] M. Thoma *et al.*, "Discriminative frequent subgraph mining with optimality guarantees," *Statist. Anal. Data Mining: ASA Data Sci. J.*, vol. 3, no. 5, pp. 302–318, 2010.
[20] S. Wernicke, "Efficient detection of network motifs," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 3, no. 4, pp. 347–359, Oct. 2006.

[21] S. Wernicke, "A faster algorithm for detecting network motifs," in *Proc. Int. Workshop Algorithms Bioinf.*, 2005, pp. 165–177.
[22] R. Rotabi, K. Kamath, J. Kleinberg, and A. Sharma, "Detecting strong ties using network motifs," in *Proc. 26th Int. Conf. World Wide Web Companion*, 2017, pp. 983–992.
[23] L. Kovanen, M. Karsai, K. Kaski, J. Kertész, and J. Saramäki, "Temporal motifs in time-dependent networks," *J. Statist. Mech.: Theory Experiment*, vol. 2011, no. 11, 2011, Art. no. P11005.
[24] Q. Xuan, H. Fang, C. Fu, and V. Filkov, "Temporal motifs reveal collaboration patterns in online task-oriented networks," *Phys. Rev. E*, vol. 91, no. 5, 2015, Art. no. 052813.
[25] A. Paranjape, A. R. Benson, and J. Leskovec, "Motifs in temporal networks," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, 2017, pp. 601–610.
[26] C. E. Tsourakakis, J. Pachocki, and M. Mitzenmacher, "Scalable motif-aware graph clustering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 1451–1460.
[27] Y. Jing, Y. Bian, Z. Hu, L. Wang, and X.-Q. S. Xie, "Deep learning for drug design: An artificial intelligence paradigm for drug discovery in the big data era," *AAPS J.*, vol. 20, no. 3, 2018, Art. no. 58.
[28] T. Lane *et al.*, "Comparing and validating machine learning models for *mycobacterium tuberculosis* drug discovery," *Mol. Pharmaceutics*, vol. 15, pp. 4346–4360, 2018.
[29] H.-T. Cheng *et al.*, "Wide & deep learning for recommender systems," in *Proc. 1st Workshop Deep Learn. Recommender Syst.*, 2016, pp. 7–10.
[30] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
[31] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1188–1196.
[32] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," MLG, 2017, *arXiv: 1707.05005*.
[33] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt, "Graph kernels," *J. Mach. Learn. Res.*, vol. 11, pp. 1201–1242, 2010.
[34] N. Shervashidze, P. Schweitzer, E. J. V. Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-Lehman graph kernels," *J. Mach. Learn. Res.*, vol. 12, pp. 2539–2561, 2011.
[35] P. Yanardag and S. Vishwanathan, "Deep graph kernels," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1365–1374.
[36] Q. Xuan *et al.*, "Automatic pearl classification machine based on a multistream convolutional neural network," *IEEE Trans. Ind. Electron.*, vol. 65, no. 8, pp. 6538–6547, Aug. 2018.
[37] D. K. Duvenaud *et al.*, "Convolutional networks on graphs for learning molecular fingerprints," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 2224–2232.
[38] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proc. Int. Conf. Learn. Representations*, 2014.
[39] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.
[40] Z. Xinyi and L. Chen, "Capsule graph neural network," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Byl8BnRcYm
[41] S. Agarwal, K. Branson, and S. Belongie, "Higher order learning with graphs," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 17–24.
[42] J.-P. Eckmann and E. Moses, "Curvature of co-links uncovers hidden thematic layers in the World Wide Web," *Proc. Nat. Academy Sci. United States America*, vol. 99, no. 9, pp. 5825–5829, 2002.
[43] D. Schiöberg, F. Schneider, S. Schmid, S. Uhlig, and A. Feldmann, "Evolution of directed triangle motifs in the Google+ OSN," in *Proc. ACM Web Sci. Conf.*, 2012, pp. 265–274.
[44] P. Wang, B. Xu, Y. Wu, and X. Zhou, "Link prediction in social networks: The state-of-the-art," *Sci. China Inf. Sci.*, vol. 58, no. 1, pp. 1–38, 2015.
[45] G. Li, M. Semerci, B. Yener, and M. J. Zaki, "Graph classification via topological and label attributes," *Statistical Anal. Data Mining: ASA Data Sci. J.*, vol. 5, no. 4, pp. 265–283, 2012.
[46] X. Wang, X. Li, and G. Chen, *Network Science: An Introduction*. Beijing, China: Higher Education Press, 2012, pp. 87–90.
[47] S. N. Soffer and A. Vazquez, "Network clustering coefficient without degree-correlation biases," *Phys. Rev. E*, vol. 71, no. 5, 2005, Art. no. 057101.
[48] A. K. Debnath, R. L. Lopez de Compadre, G. Debnath, A. J. Shusterman, and C. Hansch, "Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity," *J. Medicinal Chemistry*, vol. 34, no. 2, pp. 786–797, 1991.
[49] H. Toivonen, A. Srinivasan, R. D. King, S. Kramer, and C. Helma, "Statistical evaluation of the predictive toxicology challenge 2000–2001," *Bioinformatics*, vol. 19, no. 10, pp. 1183–1193, 2003.
[50] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel, "Protein function prediction via graph kernels," *Bioinf.*, vol. 21, pp. i47–i56, 2005.
[51] T. Joachims, T. Hofmann, Y. Yue, and C.-N. Yu, "Predicting structured objects with support vector machines," *Commun. ACM*, vol. 52, no. 11, pp. 97–104, 2009.
[52] T. Kudo, E. Maeda, and Y. Matsumoto, "An application of boosting to graph classification," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2005, pp. 729–736.
[53] X. Zhao, B. Zong, Z. Guan, K. Zhang, and W. Zhao, "Substructure assembling network for graph classification," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 4514–4521.
[54] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, 2003.

Qi Xuan (M'18) received the BS and PhD degrees in control theory and engineering from Zhejiang University, Hangzhou, China, in 2003 and 2008, respectively. He was a postdoctoral researcher with the Department of Information Science and Electronic Engineering, Zhejiang University, from 2008 to 2010, and a research assistant with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, in 2010 and 2017. From 2012 to 2014, he was a postdoctoral fellow with the Department of Computer Science, University of California at Davis, California. He is currently a professor with the Institute of Cyberspace Security, College of Information Engineering, Zhejiang University of Technology, Hangzhou. His current research interests include network science, graph data mining, cyberspace security, machine learning, and computer vision. He is a member of the IEEE.

Jinhuan Wang received the BS degree from the Zhejiang University of Technology, Hangzhou, China, in 2017. She is currently working toward the MS degree in the School of Information Engineering, Zhejiang University of Technology. Her research interests include data mining in social networks and machine learning.

Minghao Zhao received the BS degree from the Wuhan University of Technology, China, in 2014, and the MS degree from the College of Information Engineering, Zhejiang University of Technology, China, in 2019. He is currently working on Fuxi AI Lab, Netease, Hangzhou, China. His research interests include social networks analysis and machine learning.

Junkun Yuan received the BS degree from the Zhejiang University of Technology, Hangzhou, China, in 2019. He is currently working toward the PhD degree in the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. His research interests include machine learning, data mining, and causal inference.

Chenbo Fu received the BS degree in physics from the Zhejiang University of Technology, in 2007, and the MS and PhD degrees in physics from Zhejiang University, in 2009 and 2013, respectively. He was a postdoctoral researcher with the College of Information Engineering, Zhejiang University of Technology and was a research assistant with the Department of Computer Science, University of California at Davis, in 2014. Currently, he is a lecturer with the College of Information Engineering, Zhejiang University of Technology. His research interests include network based algorithms design, social network data mining, chaos synchronization, network dynamics, and machine learning.

Zhongyuan Ruan received the BSc degree in physics from Guizhou University, Guiyang, China, in 2008, and the PhD degree in physics from the East China Normal University, Shanghai, China, in 2013. He is currently a lecturer with the Institute of Cyberspace Security, College of Information Engineering, Zhejiang University of Technology, Hangzhou, China. His current research interests include complex systems and complex networks.

Guanrong Chen (M89-SM92-F97) received the MSc degree in computer science from Sun Yat-sen University, Guangzhou, China, in 1981, and the PhD degree in applied mathematics from Texas A&M University, College Station, Texas, in 1987. He has been a chair professor and the founding director of the Centre for Chaos and Complex Networks, City University of Hong Kong since 2000. Prior to that, he was a tenured full professor with the University of Houston, Texas. He was awarded the 2011 Euler Gold Medal, Russia, and conferred a Honorary Doctorate by the Saint Petersburg State University, Russia, in 2011 and by the University of Le Havre, Normandy, France, in 2014. He is a member of the Academy of Europe and a fellow of the World Academy of Sciences, and is a highly cited researcher in engineering as well as in mathematics according to Thomson Reuters.