Sampling Subgraph Network With Application to Graph Classification

Jinhuan Wang, Pengtao Chen, Bin Ma, Jiajun Zhou, Zhongyuan Ruan, Guanrong Chen, *Fellow, IEEE*, and Qi Xuan, *Senior Member, IEEE*

***Abstract*—Graphs are naturally used to describe the structures of various real-world systems in biology, society, computer science etc., where subgraphs or motifs as basic blocks play an important role in function expression and information processing. However, existing research focuses on the basic statistics of certain motifs, largely ignoring the connection patterns among them. Recently, a subgraph network (SGN) model is proposed to study the potential structure among motifs, and it was found that the integration of SGN can enhance a series of graph classification methods. However, SGN model lacks diversity and is of quite high time complexity, making it difficult to widely apply in practice. In this paper, we introduce sampling strategies into SGN, and design a novel sampling subgraph network model, which is scale-controllable and of higher diversity. We also present a structural feature fusion framework to integrate the structural features of diverse sampling SGNs, so as to improve the performance of graph classification. Extensive experiments demonstrate that, by comparing with the SGN model, our new model indeed has much lower time complexity (reduced by two orders of magnitude) and can enhance a series of graph classification methods.**

***Index Terms*—network sampling, subgraph network, feature fusion, graph classification, biological network, social network.**

I. Introduction

NETWORKS or graphs are frequently used to capture various relationships that exist in the real world, and thus we witness the emergence of social networks [1]–[3], traffic networks [4]–[6], biological networks [7]–[9], literature citation networks [10], [11], etc. The recently proposed graph representation methods allow us to better understand the structures of these networks and promote the development of various disciplines. Interestingly, the early graph embedding methods were benefited from natural language processing [12], while now the graph neural networks (GNN) are used to successfully deal with visual semantic segmentation [13]. Furthermore, these graph embedding methods have made remarkable achievements in such areas as recommendation systems [14], [15], QA sites [16], [17], and even drug discovery [18], [19]. In fact, network science, together with machine learning (especially deep learning), has made an important contribution to the development of cross-disciplines. For instance, in drug discovery, Li *et al.* [20] introduced a dummy super node that is connected with all nodes in the graph by a directed edge as the representation of the graph and modified the graph operation to help the dummy super node learn graph-level feature, which can handle graph-level classification and regression for discovering small molecule drugs. Gao *et al.* [21] provided biological interpretation using two-way attention mechanism through making predictions of unobserved examples (proteins and drugs) on an evaluation dataset from BindingDB.

Subgraphs or motifs [22], [23], as basic building blocks, can be used to describe the mesoscale structure of a network. The networks constructed by different subgraphs may have vastly different topological properties and functions, and thus could be integrated into many graph algorithms to improve their performances. For instance, after extracting the root subgraph with a modified skip-gram model, Narayanan *et al.* [24] proposed Subgraph2Vec as an unsupervised representation learning method, leading to good performance on graph classification. Ugander *et al.* [25] treated subgraph frequencies in social networks as local attributes and found that subgraph frequencies do provide unique insights for identifying social and graph structures of large networks. Inspired by neural document embedding models, Nguyen *et al.* [26] proposed the GE-FSG method, which adopts a series of frequent subgraphs as the inputs of the PV-DBOW model to obtain the entire-graph embeddings, achieving good performance in graph classification and clustering. These studies focus more on the basic statistics, e.g., the number of subgraphs, but lack analysis of the underlying structure among these subgraphs. The recently proposed subgraph network (SGN) model [27]

Manuscript received February 9, 2021; revised July 29, 2021; accepted September 20, 2021. Date of publication September 24, 2021; date of current version December 9, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61973273, in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LR19F030001, in part by the National Key R&D Program of China under Grant 2020YFB1006104, and in part by the Hong Kong Research Grants Council under the GRF Grant CityU11200317. Recommended for acceptance by Dr. George Iosifidis. *(Corresponding author: Qi Xuan.)*

Jinhuan Wang, Pengtao Chen, Jiajun Zhou, and Zhongyuan Ruan are with the Institute of Cyberspace Security, College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China (e-mail: galateawang1@gmail.com; Pengt.Chen@gmail.com; jjzhou@zjut.edu.cn; zyruan@zjut.edu.cn).

Bin Ma is with the Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA 90007 USA (e-mail: binma@usc.edu).

Guanrong Chen is with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong SAR, China (e-mail: eegchen@cityu.edu.hkLife).

Qi Xuan is with the Institute of Cyberspace Security, College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China, with the PCL Research Center of Networks and Communications, Peng Cheng Laboratory, Shenzhen 518000, China, and also with the Utron Technology Co., Ltd., (as Hangzhou Qianjiang Distinguished Expert), Hangzhou 310056, China (e-mail: xuanqi@zjut.edu.cn).

Digital Object Identifier 10.1109/TNSE.2021.3115104

takes the above issue into consideration and connects different subgraphs to construct a new network at a higher level. This process can be iterated to form a series of SGNs of different orders. It has been proven that SGNs can effectively expand the structural space and further improve the performance of network algorithms.

However, SGN model has the following two shortages. First, the rule to establish SGN is deterministic, i.e., users can generate only one SGN of each order for a network. Such lack of diversity will limit the capacity of SGN to expand the latent structure space. Second, when the number of subgraphs exceeds the number of nodes in a network, the generated SGN can be even larger than the original network, which makes it extremely time-consuming to process SGNs of the higher-order, letting alone integrating these SGNs to design algorithms of better performances. On the other hand, it is noted that network sampling can increase the diversity by introducing the randomness, and meanwhile control the scale, providing an effective and inexpensive solution for network analysis. This merit thus is exactly complementary to the SGN model.

In this paper, we introduce network sampling into the SGN model, and proposes Sampling SubGraph Network (S^2GN). In particular, we utilize the following four network sampling strategies, including random walk, biased walk, link selection, and spanning tree, to sample a subnetwork containing certain numbers of nodes and links, and then map the subnetwork to SGN based on certain rules. Network sampling and SGN construction can be used iteratively, so as to create a series of S^2GN of different orders, whose structural features can then be fused with those of the original network, so as to enhance a number of network algorithms. Specifically, we have the following contributions:

- We propose a new network model, sampling subgraph network (S^2GN), by introducing network sampling into SGN. Compared with SGN, our S^2GN can increase the diversity and decrease the complexity to a certain extent, benefiting the subsequent network algorithms.
- We propose feature fusion to fully utilize the structural information extracted from S^2GNs of different orders, generated by different sampling strategies, to enhance various graph classification algorithms based on manual attributes, Graph2Vec, DeepKernel, and CapsGNN.
- We apply the new method to nine real-world network datasets, and our experimental results demonstrate the effectiveness and efficiency of S^2GN. The fusion of S^2GNs generated by different sampling strategies can increase the performance of graph classification algorithms in 33 out of 36 cases, with a relative improvement of 9.58% on average (4.68% for SGN). This value increases to 12.96% (2.06% for SGN) when only CapsGNN is considered, i.e., the combination of S^2GN-Fusion and CapsGNN achieves the F_1-*Score* 80.58% on average, greatly improving the graph classification performance. More remarkably, compared with SGN, generating S^2GNs needs much less time, reduced by almost two orders of magnitude.

The rest of the paper is structured as follows. In Sec. II, we briefly describe the related work in network sampling and feature extraction. In Sec. III, we introduced the construction method of S^2GN. In Sec. IV, we give several feature extraction methods, which together with S^2GN are applied to nine real-world network datasets. Finally, we conclude the paper and highlight some promising directions for future work in Sec. V.

II. Related Work

In this section, to supply some necessary background information, we give a brief overview of network sampling strategies and graph representation algorithms in graph mining and network science.

A. Network Sampling

Our work is closely related to the line of research in the network analysis based on sampling. Sampling methods in graph mining have two main tasks: generating node sequences and limiting the scale of the network. For the former, many studies utilize sampling strategies to extract node sequences to provide materials for subsequent network representation. Random walk [28] is one of the most famous node sampling methods, which has a wide influence in the field of graph mining [29], [30]. For example, DeepWalk [31] combined the random walk with the language model in NLP, which was applied to node classification as a graph embedding method. In addition, Grover and Leskovec [32] designed a biased walk mechanism based on random walk, which had a further improvement in node classification. Breadth-First Sampling [33] is a node sampling algorithm, which is biased to the nodes of high degrees and has been successfully applied in the measurement and topological analysis of OSNs. By limiting the scale of a network, Satuluri *et al.* [34] sparsified graphs and achieved faster graph clustering without sacrificing quality. Moreover, sampling on graphs also has a wide spectrum of applications on network visualization [35]. The sampling method can simplify the network while preserving significant structure information, which is of ultra importance in graph mining.

B. Graph Representation

The most naive network representation method is to calculate graph attributes according to certain typical topological metrics [36]. Early graph embedding methods were considerably affected by NLP. For example, as graph-level embedding algorithms, Narayanan *et al.* proposed Subgraph2Vec [24] and Graph2Vec [37], which achieve good performances on graph classification. Another popular approach is to use graph kernel methods to capture the similarity between graphs. Although representing networks well, they generally have relatively high computational complexity [36]. It is worth mentioning that the WL kernel [38] was used to make the subgraph isomorphism check more effective. On this basis, Yanardag and Vishwanathan [39] proposed an alternative

kernel formulation termed as Deep Graph Kernel (DeepKernel) which achieved good performances on several datasets.

With the rise of spectral analysis of graph data in recent years, graph convolutional neural network (GCN) has been developed. It uses the Laplace decomposition of graphs to achieve convolutional operation in the spectral domain. Inspired by the idea of localized first-order approximation of spectral graph convolutions, Kipf *et al.* [40] presented a scalable approach for semi-supervised learning on graph-structured data (citation networks such as Citeseer, knowledge graphs such as NELL, etc.) which achieved good performances. Later, mathematical analysis on GCN went further and proved that the Laplacian decomposition used by GCN and Laplacian smoothing on images have mathematically equivalent forms [41]. At the same time, GCNs in the spatial domain have also been proposed. Inspired by the idea of convolution kernels in CNN, Mathias *et al.* [42] proposed the method of PATCHY-SAN, which can determine the direction of the convolutions and the order of the nodes in the convolution window, and this model also achieved good results in graph classification. In this way, GCN treats the obtained information without weighting, i.e. the information of important neighbors and non-important neighbors will be put into the convolution layer in an unbiased manner. GAT overcomes this shortage by supplementing a self-attention coefficient before the convolution layer [43]. Based on the newly proposed capsule network architecture, Zhang *et al.* [44] designed a CapsGNN to generate multiple embeddings for each graph, thereby capturing the classification-related information and the potential information with respect to the graph properties at the same time, which achieved the good performance.

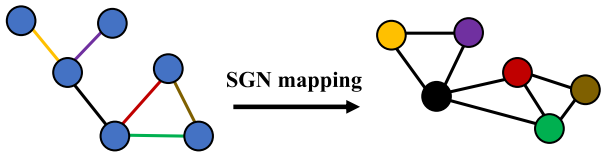
Although the above graph representation methods have relatively high expressiveness and learning ability, largely improving the performance of graph classification, they do not have good interpretability, and in addition, they only rely on a single network structure, limiting their ability to exploit the latent structural space. Therefore, we generate multiple S^2 GNs to fully expand the latent structural space, so as to enhance the network algorithms. Our experiments have demonstrated that S^2 GNs can be naturally integrated with many graph representation methods by our feature fusion framework for the further improvement of their effectiveness.

III. Methodology

We first briefly review SGN and the four network sampling methods. Then we introduce the framework to establish S^2GN.

A. Subgraph Network

Subgraph network (SGN) [27] is considered as a mapping function in network space. It provides a scalable model that transforms the original node-level network into a subgraph-level network. As shown in Fig. 1, the SGN in Fig. 1(b) can be obtained by SGN mapping from the original network in Fig. 1 (a). One can see that the edges of different colors in (a) are mapped into the corresponding nodes in (b), which are naturally connected depending on whether they share the same node in the original network.



(a) Original network (b) SGN

Fig. 1. Schematic diagram of SGN construction.

Formally, given an undirected network $G = (V, E)$ as an original network, where V and E are the node and edge sets, respectively. Let $V_i \subseteq V$ and $E_i \subseteq E$. Then, $g_i = (V_i, E_i)$ is a subgraph of G. The SGN, denoted by $G_s = \mathscr{S}(G)$, is a mapping from G to $G_s = (V_s, E_s)$, where the node and edge sets are denoted by $V_s = \{g_i | i = 0, 1, 2, \ldots, n\}$ and $E_s \subseteq (V_s \times V_s)$. If $g_a \cap g_b \neq \emptyset$, i.e., $g_a \cap g_b \in V$, in the original network, then they are connected in the SGN, i.e., $(g_a, g_b) \in E_s$. It can be seen that the construction of SGN has three steps: (i) detect subgraphs $\{g_i\}$ from the original network; (ii) clear and define the connection rules between subgraphs; (iii) build SGN by leveraging the subgraphs.

For simplicity, here for the case of 1st-order SGN, denoted by $\text{SGN}^{(1)}$, pairwise linked nodes are chosen as building units, and the adjacent node pairs are connected. In this case, $\text{SGN}^{(1)}$ is equivalent to the line graph [45], which reveals the topological interaction between edges of the original network. Fu *et al.* [3] used this method to map the original network to an SGN, and then used the node centrality in SGN to predict the weights of edges of the original network. As the SGN gradually maps to the higher-order network space, one can observe more abundant feature information. For example, the 2nd-order subgraph network, denoted by $\text{SGN}^{(2)}$, is obtained by repeating the mapping process on the $\text{SGN}^{(1)}$. The building unit of $\text{SGN}^{(2)}$ is a 2-hop structure (open triangle), which maintains the 2nd-order interactive information of the edge structures and can provide more insights about the local structure of a network [46]. To reduce the density of SGN, in the case of $\text{SGN}^{(2)}$, two building units are connected when they share the same edge. The latent structural information provided by higher-order SGNs may steadily diminish as the order increases. Therefore, SGN generally works best with the first two orders [27].

B. Network Sampling Strategies

In this paper, we adopt the following four sampling strategies, including random walk, biased walk, link selection, and spanning tree, to design our S^2GN.

Random Walk: Random walk [47] can be used to obtain the co-occurrence relationship between nodes during network sampling. A node in a network can be described by the wandering sequence starting from it. The wandering sequence obtained from the node contains both local and higher-order neighbors. When the wandering scope is extended to the graph level, one can peek into the topology of the whole network. In

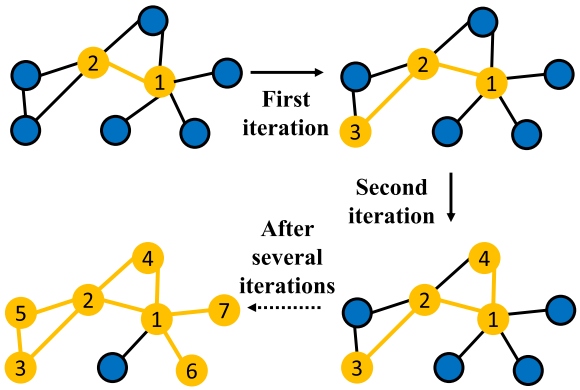


Fig. 2. Illustration of the walk procedure in link selection.

our model, given a network $G = (V, E)$, the random walk algorithm is described as follows:

- *Start with an initial node* $v^0 \in V$.
- *At step i, choose one neighbouring node* $u \in \mathcal{N}(v^{i-1})$.
- *Let* $v^i \leftarrow u$ *be the next node and get the edge* $\widehat{E} \leftarrow \widehat{E} + \{(v^{i-1}, v^i)\}$.
- *Repeat the steps until* $|\widehat{E}| = |V|$.

Node v^i is generated by the following distribution:

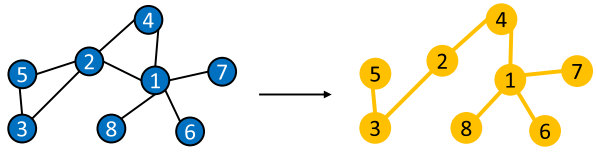
$$P(v^i = x | v^{i-1} = m) = \begin{cases} \frac{\alpha}{N}, if(m, x) \in E \\ 0, otherwise \end{cases}$$

where α is the transition probability between nodes m and x, and N is the normalizing constant. One can follow the above steps to simulate a random walk and get the final substructure $\widehat{G} = (\widehat{V}, \widehat{E})$.

Biased Walk: In the field of network science, biased walk [48] is different from the random walk where the probability of a potential new state is independent of external conditions. When the network is too large to be analyzed by statistical methods, the directed walk extracts the flexible neighborhood of the undirected network according to certain rules, which provides an effective method for structural analysis. The concept of the biased walk has attracted considerable attention, especially in the fields of transportation and social networks [49]. Here, we adopt the walking mechanism of Node2Vec [32], where the homophily and structural equivalence of nodes are preserved by integrating the depth-first search and breadth-first search. Specifically, we adopt the 2nd-order random walk with parameters p and q, which takes into account the topological distance between the next node and the previous node as well as the connectivity of the current node. Thus, the transition probability α between v^i and v^{i+1} is determined by

$$\alpha_{(v^i, v^{i+1})} = \omega_{pq}(v^{i-1}, v^{i+1}) = \begin{cases} \frac{1}{p}, d_{(v^{i-1}, v^{i+1})} = 0 \\ 1, d_{(v^{i-1}, v^{i+1})} = 1 \\ \frac{1}{q}, d_{(v^{i-1}, v^{i+1})} = 2 \end{cases}$$

where v^{i-1}, v^i, and v^{i+1} are the previous, current, and next nodes, respectively, and $d_{(v^{i-1}, v^{i+1})} \in (0, 1, 2)$ indicates the shortest path between v^{i-1} and v^{i+1}. Note that α is equal to ω_{pq} when the network is unweighted. Various substructures of network can be obtained by controlling p and q.



(a) Original network (b) A spanning tree

Fig. 3. The substructure obtained by spanning tree.

Link Selection: We also propose a new edge-based sampling method, namely link selection. Given a network $G = (V, E)$, we first sample an initial edge $e^0 = (v^0, v^1)$, and then randomly select a node of this edge as the source node of the next sampling edge. The nodes of all the sampled edges form the source node pool V_{pool} for the next sampling. The sampling process will not terminate until the stop condition is met. The substructures after this sampling strategy are obtained by a diffuse search from a central edge, which ensures the acquisition of important network structures. As shown in Fig. 2, the node pair (1,2) is selected as the initial edge and then we can get the substructure that contains nodes (1,2,3) after one iteration through node "2" and get an expanding substructure that contains nodes (1,2,3,4) after second iteration through another node "1". After several iterations, one can get the final substructure, which contains 7 nodes and 8 edges while the program satisfies the stop condition.

- *Start with an initial edge* $e^0 = (v^0, v^1) \in E$, *and let* $V_{pool} = \{v^0, v^1\}$, $E_{pool} = \{e^0\}$.
- *At step i, choose one node* $u \in V_{pool}$.
- *Let* $u^i \leftarrow u$ *be the next start node and select an edge* $(u^i, u^{i+1}) \notin E_{pool}$.
- *Update* $V_{pool} \leftarrow V_{pool} + \{u^{i+1}\}$ *and get the edge pool* $E_{pool} \leftarrow E_{pool} + \{(u^i, u^{i+1})\}$.
- *Repeat the above steps until* $|E_{pool}| = |V|$.

Note that (u^i, u^{i+1}) has the same transition probability with the random walk, and V_{pool} and E_{pool} are the node and edge sets of the final substructure $\widehat{G}$. This method differs from random walk in that it can search the network on the basis of the current substructure rather than a single node, which can reduce the appearance of a chain structure to a greater extent.

Spanning Tree: A spanning tree [50] is a minimally connected substructure that contains all nodes in the graph, as shown in Fig. 3. Different spanning trees can be obtained by traversing from different nodes. Here we randomly select a node as the initial node. The maximum and minimum spanning trees are unified without considering the edge weights. In this section, we use the typical Kruskal algorithm [51] to generate spanning trees and the weight values of edges are all set to 1.

C. Framework for Constructing S^{2}GN

Most real-world networks have large scale and complex structure. Typically, SGN could be even larger and denser, making the follow-up network algorithms less efficient. It will introduce extra noisy structural information, disturbing the network representation algorithms. In view of this, we focus on optimizing the SGN model and propose a framework for constructing a sampling subgraph network (S^{2}GN) by integrating different network sampling methods. The pseudocodes

Algorithm 1. Construction of S ^{2} GN

Input: A network $G(V, E)$ with node set V and link set $E \subseteq (V \times V)$;
The order of SGN h.
Output: S ^{2} GN, denoted by $G_s(V_s, E_s)$.

Initialize a temporary object G_s;
if the G is not full-connected **then**
    $G_s \leftarrow$ GetMaxSubstructure(G);
end
else
    $G_s \leftarrow G$;
end
while h **do**
    Initial node u = NodeRanking(V_s);
    Get sampling substructure $\widehat{G_s}$ through executing Algorithm 2;
    G_{sgn} = SGNAlgorithms($\widehat{G_s}$);
    $G_s \leftarrow$ Relabeled(G_{sgn});
    $h = h - 1$;
end
return $G_s(V_s, E_s)$

Algorithm 2. Sampling substructure

Input: A network $G_s(V_s, E_s)$;
Source node u;
Sampling walks $l = |V_s|$.
Output: Sampling substructure, denoted by $\widehat{G_s} = g(\widehat{v}, \widehat{e})$.

Let $v_0 = u$, initial $walk_v$ to $[v_0]$, $walk_e$ to $\emptyset$;
Select first edge e_1 with a given probability of sampling strategy;
Append the $v_1 = dst(e_1)$ to $walk_v$, e_1 to $walk_e$;
for $i = 2$ to $l - 1$ **do**
    $cur_v = walk_v$[-1], $cur_e = walk_e$[-1];
    e_i = GetNextEdgeWithStrategy(cur_v, cur_e);
    Append e_i to $walk_e$, $v_i = dst(e_i)$ to $walk_v$;
end
$\widehat{v} = walk_v$, $\widehat{e} = walk_e$;
return $\widehat{G_s} = g(\widehat{v}, \widehat{e})$

of constructing S^{2}GN and sampling substructures are given in Algorithms 1 and 2, respectively. In Algorithms 1, GetMaxSubstructure() is to obtain the maximally connected substructure of original network if it is not connected; NodeRanking() is to rank the input nodes; SGNAlgorithms() is to construct SGNs. GetNextEdgeWithStrategy() in Algorithms 2 is to get the next edge according to a given sampling strategy.

In general, S^{2}GN can be constructed in three steps: source node selection, sampling substructure and S^2 GN construction, which are introduced in the following.

- *Source node selection*: There are many ways to choose the initial node: (i) Randomly select a node as the source node; (ii) Select an initial node according to its importance measured by closeness centrality [52], K-shell [53], PageRank [54] or others [55]. Our framework can be combined with various central node selection methods. In this paper, we use the K-shell decomposition method to make a coarse-grained division of the importance of nodes in the network, so that it is more likely to capture key structures.
- *Sampling substructure*: After the initial source node is determined, a substructure can be obtained by conducting a certain sampling strategy to extract the main context of the current network. According to different sampling strategies, diverse sampling substructures can be generated, reflecting the different aspects of the original network and further benefiting the subsequent network algorithms.
- *S^{2}GN construction*: Based on the sampling substructure, we use SGN model to construct S^{2}GN. Note that network sampling and SGN are adopted iteratively so as to get the S^{2}GNs of higher orders. This method can control the size of S^{2}GNs and meanwhile increase their diversity. Therefore, compared with SGN, the S^{2}GN could further enhance both efficiency and effectiveness of the subsequent network algorithms.

Now, we use various feature extraction methods to get structural features from S^{2}GNs of different orders, which are first fused and then used to establish the graph classification models. The overall framework of S^{2}GN construction for structural feature space expansion is shown in Fig. 4. Note that, generally, information fusion tries to integrate information from multiple aspects to improve algorithm performance, which has a wide range of applications in practice. For instance, in speech recognition, the visual features of the lip motion are fused with the speech signal features to predict the words expressed [56]. In image recognition, Xuan *et al.* [57] developed a multistream convolutional neural network to automatically merge the features of multi-view pearl images, so as to improve the accuracy of pearl classification. In this paper, we use different sampling strategies to capture the structural features from different aspects. As an example, we visualize different 1st-order and 2nd-order S^{2}GNs generated by the four network sampling strategies on positive and negative samples from the MUTAG dataset, as shown in Fig. 5. It can be seen that the S^{2}GNs generated by different sampling strategies have quite different structures, and the structural difference between the positive and negative samples may be enlarged in S^{2}GNs. Therefore, it can be expected that the fusion of these diverse S^{2}GNs could improve the performance of graph classification.

IV. Experiments

Now, we compare S^{2}GN and SGN models on their abilities to enhance graph classification based on four feature extraction methods. We first introduce the datasets, followed by the feature extraction methods and the parameter setting. After that, we show the experimental results with discussion.

A. Datasets

We test our S^{2}GN method on nine real-world network datasets, as introduced in the following. IMDB-BINARY is about social networks, while the others are about bio- and chemo-informatics networks. The basic statistics of these datasets are presented in Table I.

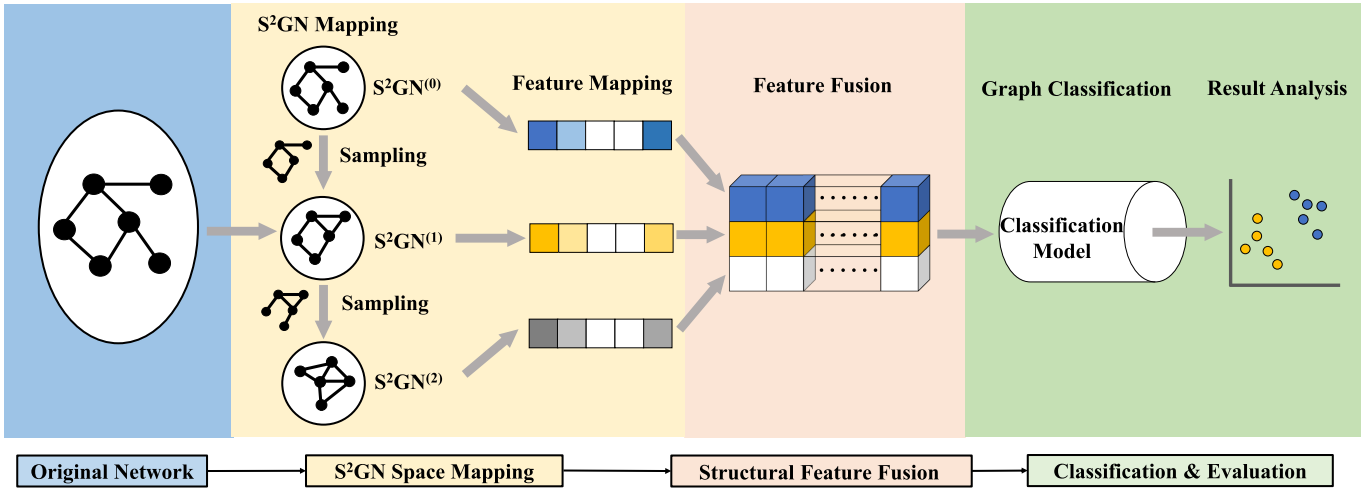


Fig. 4. The overall framework of the S ^{2} GN algorithm for network structural feature fusion.

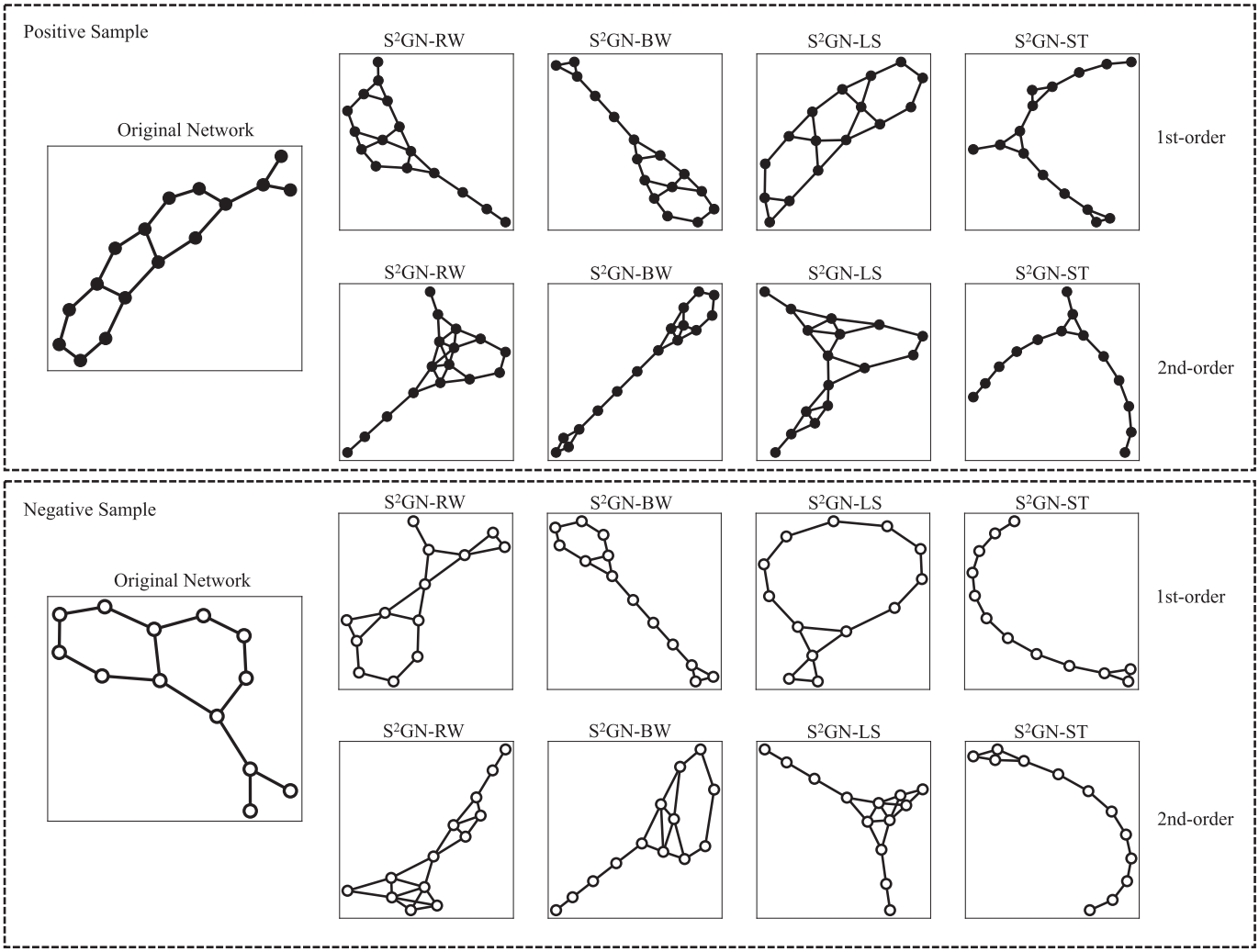


Fig. 5. Visualization of 1st-order and 2nd-order S ^{2} GNs using four network sampling strategies on positive and negative samples from the MUTAG dataset.

- *MUTAG* [58] contains 188 mutagenic aromatic and heteroaromatic compounds, with nodes and edges representing atoms and the chemical bonds between them, respectively. They are labeled according to whether there is a mutagenic effect on a special bacteria.
- *PTC* [59] includes 344 chemical compound graphs, with nodes and edges representing atoms and the

TABLE I
BASIC STATISTICS OF NINE DATASETS. N_G IS THE NUMBER OF GRAPHS, $\#C_{max}$ IS THE NUMBER OF GRAPHS BELONGING TO THE LARGEST CLASS, N_C IS THE NUMBER OF CLASSES, AND #NODES AND #EDGES ARE THE AVERAGE NUMBERS OF NODES AND EDGES, RESPECTIVELY, OF THE GRAPHS IN THE DATASET

Dataset	N_G	$\#C_{max}$	N_C	#Nodes	#Edges
MUTAG	188	125	2	18	20
PTC	344	192	2	14	14
PROTEINS	1113	663	2	39	73
ENZYMES	600	100	6	32	63
NCI1	4110	2057	2	30	32
NCI109	4127	2079	2	30	32
IMDB-BINARY	1000	500	2	20	97
D&D	1178	691	2	284	716
COLLAB	5000	2600	3	75	2458

chemical bonds between them, respectively. Their labels are determined by their carcinogenicity for rats.

- *PROTEINS* [60] comprises of 1113 graphs. The nodes are Secondary Structure Elements (SSEs) and the edges are neighbors in the amino-acid sequence or in the 3D space. These graphs represent either enzyme or non-enzyme proteins.
- *ENZYMES* [61] contains 600 protein tertiary structures, and each enzyme belongs to one of the 6 EC top-level classes.
- *NCI1* & *NCI109* [8] comprise of 4110 and 4127 graphs, respectively. The nodes and edges 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.
- *IMDB-BINARY* [26] is about movie collaboration including 1000 graphs, which is collected from IMDB and contains lots of information about different movies. Each graph is an ego-network, where nodes represent actors or actresses and edges indicate whether they appear in the same movie. Each graph is categorized into one of the two genres (Action and Romance).
- *D & D* [62] contains 1178 graphs of protein structures. A node represents an amino acid and edges are constructed if the distance between two nodes is less than 6 A. A label denotes whether a protein is an enzyme or non-enzyme.
- *COLLAB* [39] is a scientific collaboration dataset of 5000 graphs. Each graph corresponds to a researchers ego-network, where nodes represent the researchers and an edge indicates collaboration relationship between two researchers. A researchers ego-network has three possible labels, i.e., High Energy Physics, Condensed Matter Physics, and Astro Physics, which are the fields that the researcher belongs to.

B. Feature Extraction Methods

We adopt four typical methods to generate graph representation, namely manual attributes, Graph2Vec, DeepKernel, and CapsGNN, which are introduced in the following.

- *Attributes:* Here, we use the same 11 manual attributes as those introduced in [27], including the number of nodes, the number of edges, average degree, network density, average clustering coefficient, the percentage of leaf nodes, the largest eigenvalue of the adjacency matrix, average betweenness centrality, average closeness centrality, and average eigenvector centrality.
- *Graph2Vec* [37]: 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 natural language processing (NLP).
- *DeepKernel* [39]: This method provides a unified framework that leverages the dependency information of sub-structures by learning latent representations. The 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. According to the definition of kernel: $\mathcal{K} = \mathcal{P}\mathcal{M}\mathcal{P}^{\mathrm{T}} = \mathcal{P}\mathcal{V}\mathcal{V}^{\mathrm{T}}\mathcal{P}^{\mathrm{T}} = \mathcal{H}\mathcal{H}^{\mathrm{T}}$, one can use the columns in the matrix $\mathcal{H} = \mathcal{P}\mathcal{V}$ as the inputs to the classifier.
- *CapsGNN* [44]: This method was inspired by CapsNet [63], which adopts 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.

C. Parameter Setting

For source node selection, we choose the node of the largest K-shell [53] as the source node for random walk (RW) and biased walk (BW), and choose the edge of the largest betweenness centrality [64]–[66] as the source edge for link selection (LS). We randomly pick up a node as the source node for the spanning tree (ST) to increase the diversity of S^2GN, since the sampled subnetworks will be quite similar if we fix the source node for this method. Moreover, we set the two parameters of BW as $p = 4$ and $q = 1$.

In this study, for *Graph2Vec*, the embedding dimension is adopted according to [37]. Since the embedding dimension is predominant for learning performances, a commonly-used value of 1024 is adopted. The other parameters are set to default values: the learning rate is set to 0.5, the batch size is set to 512 and the number of epochs is set to 1000. For *DeepKernel*, according to [39], the Weisfelier-Lehman subtree kernel is used to build the corpus and its height is set to 2. Furthermore, the embedding dimension is set to 10, the window size is set to 5 and skip-gram is used for the word2vec

model. 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 Random Forest is chosen as the classification model. Meanwhile, for each feature extraction method, the feature space is first expanded by using S^2GNs, 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 8 folds for training and 2 fold for testing. Here, the $F1$-$Score$ is adopted as the metric to evaluate the classification performance:

$$F_1 = \frac{2PR}{P+R}, \tag{1}$$

where P and R are the precision and recall, respectively. In order to diminish the random effect of the fold assignment to some extent, the experiment is repeated 100 times and then the average F_1-$Score$ and its standard deviation are reported.

We further define the relative improvement rate (RIMP) of SGN or S^2GN model as

$$RIMP = (F1_{model} - F1_{ori})/F1_{ori} \tag{2}$$

where $F1_{ori}$ and $F1_{model}$ refer to the $F1$-$Score$ of the graph classification algorithm without and with the SGN model (or S^2GN-Fusion model), respectively.

D. Experimental Results

We use the four network sampling strategies to generate sampling substructures, and further construct the corresponding 1st-order and 2nd-order S^2GNs, denoted by S^2GN-RW, S^2GN-BW, S^2GN-LS, and S^2GN-ST, respectively[1]. The above S^2GN models integrate the original network, $S^2GN^{(1)}$, and $S^2GN^{(2)}$ in order to be able to compare reasonably and fairly with the SGN model (i.e., SGN $^{(0,1,2)}$). After that, we adopt the four feature extraction methods, namely manual attributes, Graph2Vec, DeepKernel, and CapsGNN, to get structural feature vectors. For each feature extraction method, we fuse the vectors generated from the different S^2GNs to a single vector. Finally, this vector is fed into the Random Forest model to produce the classification result. Note that we also produce the results for a single sampling strategy for a more comprehensive comparison. Here, a ten-fold cross-validation method is used to calculate $F1$-$Score$ of graph classification. To enrich the sampling structures and reduce the probability of sampling repetition, 10 sampling averaging processes were carried out for each sampling strategy.

1) Enhancement on Classification Performance: The experimental results are shown in Table II, where one can see that the four S^2GN models based on a single sampling strategy, i.e., S^2GN-RW, S^2GN-BW, S^2GN-LS, and S^2GN-ST, are comparable with the SGN model, which all produce similar classification results under different datasets and feature extraction methods. Interestingly, S^2GN-BW outperforms SGN in enhancing the classification models based on the four feature extraction methods in most cases, leading to a relative improvement of 4.80% on average. Such results are consistent with the experience that Node2Vec is a powerful method to capture the structural properties of a network. Moreover, since different S^2GNs generated by different sampling strategies can capture the different aspects of a network, as visualized in Fig. 5, one may expect that the fusion of these S^2GNs can produce even better classification results. Indeed, we find that the fusion of S^2GNs increases the performance of the original graph classification algorithms in 33 out of 36 cases, with a relative improvement of 9.58% on average (much better than 4.68% by SGN). The value increases to 12.96% (much better than 2.06% by SGN) when only CapsGNN is considered. This result is quite impressive, since CapsGNN, together with S^2GN, achieves the state-of-the-art performance on PROTEINS and IMDB-BINARY datasets.

Moreover, we find that compared with other sampling methods, the experiments based on spanning tree method on MUTAG have achieved unsatisfactory performance, i.e., the results on S^2GN-ST are overall underperformed than other models. We think, for certain types of networks, the performance of S^2GN is related to the sampling strategy. MUTAG is a dataset of mutagenic aromatic and heteroaromatic compounds containing many benzene rings. Combined with nitroso, chlorine, carbonyl, and other functional groups, these benzene ring structures can be used to determine whether the compound has a mutagenic effect on a bacterium. As we all know, a spanning tree must meet the following two conditions: (1) contains all nodes in a connected graph, and (2) there is one and only one path between any two nodes. That is, a sampling substructure after the spanning tree method does not contain loops. Therefore, the spanning tree method could break the benzene ring structures, which are of great significance for distinguishing the networks in MUTAG. Once the ring structures are broken, S^2GN-ST could not extract the critical features to support the graph classification on MUTAG. One can see that different networks have their own unique properties and the selection of sampling strategies according to networks of different types is an importance and valuable research direction. In future work, we will focus on this point and explore more matching relationship between different sampling strategies and various networks.

To address the robustness of our S^2GN model against the size variation of the training set, the F_1-$Score$ is calculated by using various sizes of training sets (from 10 to 90 percent, within a 20 percent interval). For each size, the training and test sets are randomly divided, which is repeated 100 times with the average result recorded. The results are shown in Fig. 6 for various feature extraction methods on nine datasets. It can be seen that still the curves of S^2GN-Fusion are relatively higher than those of S^2GNs generated by a single sampling strategy in most cases, indicating that the superiority of S^2GN-Fusion is robust

[1] It has been proven that the graph classification models can be significantly enhanced by appropriately using the structural information of the SGNs in the first two orders, while such gain will be reduced soon as more SGNs of higher orders are integrated [27]. This is why we only use the S^2GNs of the first two orders here.

TABLE II
CLASSIFICATION RESULTS MEASURED BY $F1$-$Score$ ON NINE DATASETS BY USING DIFFERENT FEATURE EXTRACTION METHODS

Algorithm	**Classification results** ($F1$-$Score$, %)									
Attributes	MUTAG	PTC	PROTEINS	ENZYMES	NCI1	NCI109	IMDB-BINARY	D&D	COLLAB	Avg.
Original	86.58 ± 3.61	63.52 ± 4.55	78.30 ± 2.49	43.37 ± 2.29	67.48 ± 0.87	67.34 ± 1.25	73.00 ± 3.68	75.85 ± 1.61	$\mathbf{76.53} \pm 0.50$	70.22
SGN	91.58 ± 4.21	67.94 ± 6.36	79.46 ± 2.96	50.22 ± 2.91	69.84 ± 1.59	69.73 ± 1.97	$\mathbf{77.65} \pm 4.50$	76.65 ± 1.59	> 3 days	72.88
$RIMP$-SGN	5.78%	6.96%	1.48%	15.79%	3.50%	3.55%	6.37%	1.05%	-	4.97%
S^{2}GN-RW	90.53 ± 2.11	66.71 ± 1.62	77.76 ± 1.52	52.17 ± 2.36	74.82 ± 0.69	73.96 ± 0.84	71.85 ± 2.74	77.37 ± 2.91	72.84 ± 0.42	73.11
S^{2}GN-BW	93.94 ± 2.37	69.11 ± 2.94	$\mathbf{79.83} \pm 1.46$	53.37 ± 2.78	75.47 ± 0.98	73.99 ± 1.04	76.05 ± 1.29	$\mathbf{77.75} \pm 1.68$	72.89 ± 0.35	74.71
S^{2}GN-LS	89.21 ± 2.48	66.18 ± 2.55	78.57 ± 1.44	49.50 ± 2.14	75.85 ± 1.03	74.65 ± 0.62	71.80 ± 2.53	76.91 ± 1.94	72.57 ± 0.51	72.96
S^{2}GN-ST	90.79 ± 2.12	70.44 ± 2.75	76.28 ± 1.89	45.33 ± 1.29	72.25 ± 1.08	73.26 ± 0.76	77.60 ± 1.50	76.90 ± 2.47	74.22 ± 0.42	72.81
S^{2}GN-Fusion	$\mathbf{94.74} \pm 1.84$	$\mathbf{72.06} \pm 3.29$	79.14 ± 0.84	$\mathbf{55.25} \pm 1.90$	$\mathbf{76.03} \pm 1.32$	$\mathbf{74.89} \pm 1.18$	76.97 ± 1.21	77.03 ± 2.46	73.12 ± 0.52	**75.46**
$RIMP$-Fusion	9.42%	13.44%	1.07%	27.39%	12.67%	11.21%	5.44%	1.56%	-4.4%	7.47%
Graph2Vec	MUTAG	PTC	PROTEINS	ENZYMES	NCI1	NCI109	IMDB-BINARY	D&D	COLLAB	Avg.
Original	83.15 ± 9.25	60.17 ± 6.86	73.30 ± 2.05	45.17 ± 2.73	73.22 ± 1.81	74.26 ± 1.47	62.47 ± 3.99	70.25 ± 2.18	79.68 ± 0.53	69.08
SGN	86.84 ± 5.70	63.24 ± 6.70	74.44 ± 3.09	48.73 ± 2.56	76.64 ± 3.21	74.86 ± 2.76	70.65 ± 5.55	80.42 ± 3.06	> 3 days	70.73
$RIMP$-SGN	4.44%	5.10%	1.56%	7.88%	4.67%	0.81%	13.09%	14.48%	-	4.39%
S^{2}GN-RW	80.26 ± 2.69	61.47 ± 2.06	76.37 ± 1.12	48.67 ± 2.53	76.88 ± 1.35	74.39 ± 1.40	68.35 ± 1.57	81.86 ± 1.80	79.90 ± 0.32	72.02
S^{2}GN-BW	$\mathbf{86.84} \pm 3.07$	$\mathbf{64.71} \pm 2.85$	$\mathbf{77.13} \pm 1.09$	52.33 ± 2.30	77.39 ± 1.12	75.69 ± 1.46	71.64 ± 2.00	82.12 ± 2.22	79.69 ± 0.39	74.17
S^{2}GN-LS	81.05 ± 2.57	62.35 ± 2.88	76.91 ± 2.21	47.68 ± 1.73	$\mathbf{79.18} \pm 1.71$	$\mathbf{77.42} \pm 1.13$	67.25 ± 2.16	81.77 ± 1.98	79.61 ± 0.39	72.58
S^{2}GN-ST	81.84 ± 2.99	63.97 ± 2.39	75.20 ± 2.15	49.87 ± 2.91	76.30 ± 1.21	72.95 ± 0.89	72.49 ± 2.11	74.92 ± 2.89	79.83 ± 0.50	71.93
S^{2}GN-Fusion	81.73 ± 3.37	64.38 ± 2.42	75.10 ± 0.89	$\mathbf{54.78} \pm 2.29$	76.91 ± 0.72	75.72 ± 1.31	$\mathbf{76.43} \pm 2.17$	$\mathbf{82.75} \pm 2.79$	$\mathbf{80.53} \pm 0.32$	**74.26**
$RIMP$-Fusion	-1.71%	7.00%	2.46%	21.28%	5.04%	1.97%	22.35%	17.79%	1.07 %	7.50%
DeepKernel	MUTAG	PTC	PROTEINS	ENZYMES	NCI1	NCI109	IMDB-BINARY	D&D	COLLAB	Avg.
Original	82.95 ± 2.68	59.04 ± 1.09	73.30 ± 0.82	45.04 ± 3.73	67.06 ± 1.91	67.04 ± 1.36	67.50 ± 2.45	75.97 ± 1.91	73.64 ± 0.46	67.95
SGN	93.68 ± 5.15	65.88 ± 5.05	76.78 ± 2.41	45.93 ± 3.75	70.26 ± 1.24	71.06 ± 1.61	75.70 ± 1.55	77.84 ± 2.08	> 3 days	72.14
$RIMP$-SGN	12.94%	11.59%	4.75%	1.98%	4.77%	6.00%	12.15%	2.46%	-	7.29%
S^{2}GN-RW	93.68 ± 5.66	61.76 ± 3.77	75.80 ± 4.21	43.32 ± 3.64	69.15 ± 1.63	69.06 ± 1.70	72.30 ± 2.68	83.47 ± 1.00	78.04 ± 0.56	71.84
S^{2}GN-BW	94.00 ± 5.43	67.35 ± 4.48	76.69 ± 2.97	47.75 ± 2.68	71.51 ± 1.38	69.83 ± 2.05	74.10 ± 3.33	81.57 ± 1.11	77.82 ± 0.52	73.40
S^{2}GN-LS	93.68 ± 4.59	66.18 ± 4.21	76.16 ± 1.92	50.28 ± 3.04	$\mathbf{71.55} \pm 1.15$	70.19 ± 2.26	75.80 ± 3.43	$\mathbf{83.98} \pm \mathbf{1.77}$	77.01 ± 0.72	73.87
S^{2}GN-ST	88.95 ± 3.68	65.29 ± 4.59	74.73 ± 4.54	48.02 ± 3.52	70.77 ± 1.20	$\mathbf{71.04} \pm 1.03$	75.90 ± 2.07	78.94 ± 1.38	74.54 ± 0.52	72.02
S^{2}GN-Fusion	$\mathbf{94.73} \pm 4.07$	$\mathbf{70.88} \pm 4.25$	$\mathbf{77.14} \pm 2.97$	$\mathbf{52.21} \pm 2.24$	71.06 ± 1.01	70.48 ± 1.22	$\mathbf{76.50} \pm 3.75$	83.77 ± 1.87	$\mathbf{78.18} \pm 0.68$	**75.00**
$RIMP$-Fusion	14.20%	20.05%	5.24%	15.92%	5.96%	5.13%	13.33%	10.27%	6.17 %	10.38%
CapsGNN	MUTAG	PTC	PROTEINS	ENZYMES	NCI1	NCI109	IMDB-BINARY	D&D	COLLAB	Avg.
Original	86.32 ± 7.52	62.06 ± 4.25	75.89 ± 3.51	49.78 ± 3.02	78.30 ± 1.80	72.99 ± 2.15	72.71 ± 4.36	67.75 ± 2.57	76.21 ± 0.86	71.34
SGN	89.47 ± 7.44	64.12 ± 3.67	76.34 ± 4.13	50.04 ± 2.70	78.61 ± 1.87	73.72 ± 2.39	76.47 ± 5.74	68.71 ± 1.91	> 3 days	72.19
$RIMP$-SGN	3.65%	3.32%	0.59%	0.52%	0.40%	1.00%	5.17%	1.42%	-	2.06%
S^{2}GN-RW	88.70 ± 4.59	77.81 ± 4.96	84.73 ± 2.09	51.33 ± 1.14	74.23 ± 1.40	75.16 ± 1.39	92.50 ± 3.15	78.05 ± 1.91	77.32 ± 0.41	77.76
S^{2}GN-BW	92.63 ± 4.82	81.91 ± 5.45	84.10 ± 3.72	52.77 ± 2.11	$\mathbf{78.83} \pm 2.35$	75.25 ± 1.69	93.35 ± 1.12	$\mathbf{78.66} \pm 2.32$	$\mathbf{77.66} \pm 0.37$	79.46
S^{2}GN-LS	90.53 ± 4.59	79.11 ± 4.16	84.28 ± 1.96	52.17 ± 1.23	76.57 ± 1.26	75.43 ± 1.46	93.92 ± 1.75	77.03 ± 1.52	77.62 ± 0.32	78.52
S^{2}GN-ST	89.21 ± 5.73	78.67 ± 5.06	84.03 ± 2.58	52.56 ± 1.18	76.52 ± 1.42	75.16 ± 1.60	94.20 ± 1.26	72.31 ± 2.73	77.38 ± 0.43	77.78
S^{2}GN-Fusion	$\mathbf{93.15} \pm 4.11$	$\mathbf{84.12} \pm 6.47$	$\mathbf{85.18} \pm 1.84$	$\mathbf{56.08} \pm 3.15$	78.13 ± 2.27	$\mathbf{78.23} \pm 1.05$	$\mathbf{95.10} \pm 2.30$	77.85 ± 1.95	77.40 ± 0.41	**80.58**
$RIMP$-Fusion	7.91%	35.55%	12.24%	12.66%	-0.22%	7.18%	30.79%	14.90%	1.56%	12.96%

in enhancing graph classification algorithms. In particular, such superiority seems much more significant when enhancing CapsGNN, which is interesting and may indicate that the potential of S^{2}GN-Fusion could be exploited further by connecting a better embedding method or end-to-end graph neural network, and meanwhile there could be much room for further improvement for graph classification.

2) Reduction of Time Complexity: Note that one important motivation to introduce sampling strategies into SGN is to control the network size so as to improve the efficiency of the network algorithms based upon them. Therefore, here to address the computational complexity of our method, we record the average computing time of SGN and S^{2}GN generated by the four sampling strategies on the nine datasets, namely MUTAG, PTC, PROTEINS, ENZYMES, NCI1, NCI109, IMDB-BINARY, D & D, and COLLAB. The results are presented in Table III, where one can see that, overall, the computing time of S^{2}GN is much less than that of SGN for each sampling strategy on each dataset, decreasing from hundreds of seconds to less than 64 seconds. In fact, the computing time of S^{2}GNs generated by different sampling strategies is comparable to each other. Considering that S^{2}GN-Fusion method needs to generate all the four S^{2}GNs, its computing time is close to the sum of individual ones, which is still less than 110 seconds. Such results suggest that, by comparing with SGN, our S^{2}GN model can indeed largely increase the efficiency of the network algorithms.

In fact, we can estimate the time complexity of our model in theory. For random walk, it is a computationally efficient sampling method, which only requires $\mathcal{O}(|E|)$ space complexity to store the neighbors of each node in the graph. As for the time complexity, by imposing graph connectivity in the sample generation process, random walk provides a convenient mechanism to increase the effective sampling rate by reusing

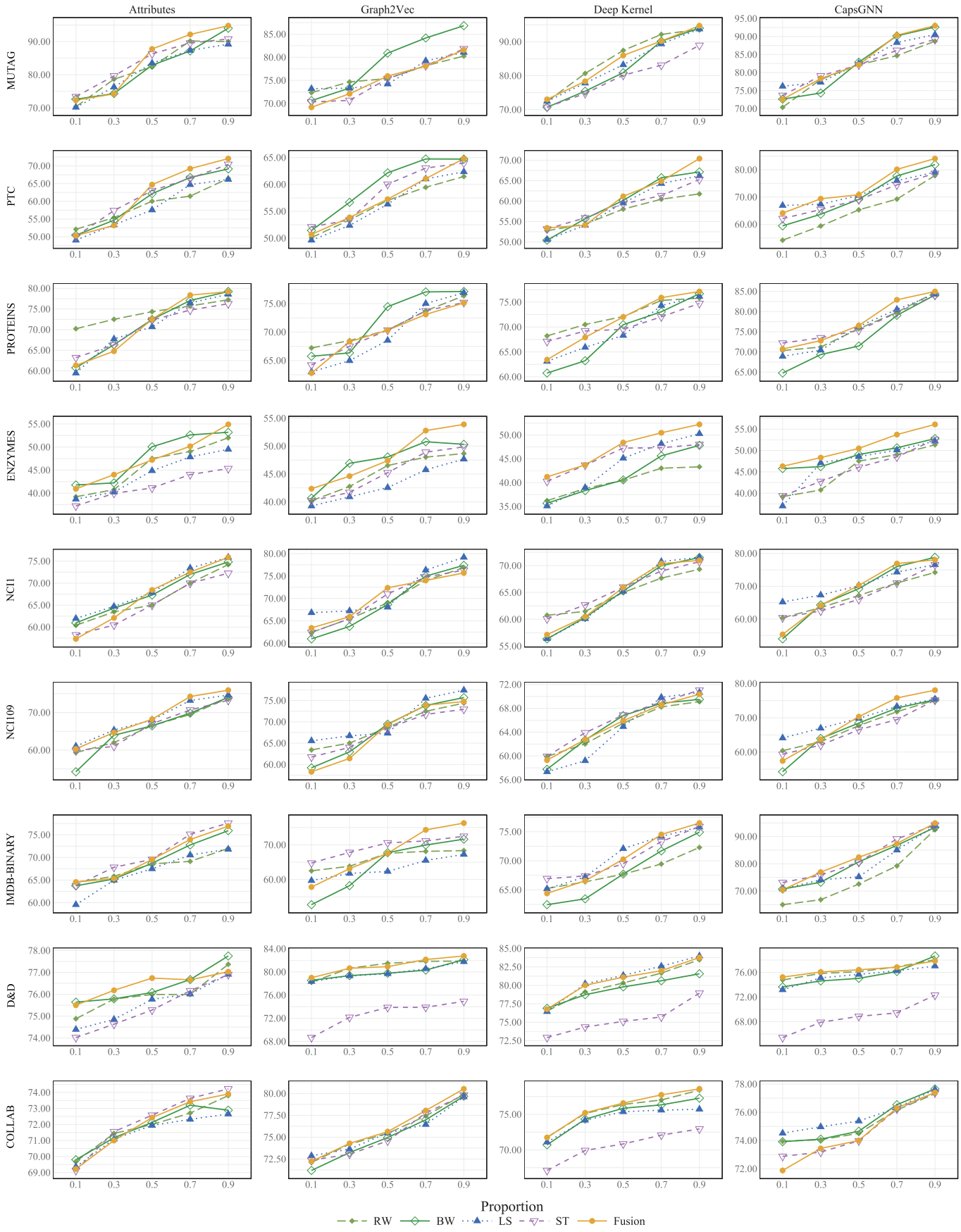


Fig. 6. Average $F1$-*Score* as functions of the training set size (represented by the fraction of samples in the training set), for various feature extraction methods on different datasets, based on RW, BW, LS, ST and Fusion, respectively.

TABLE III
AVERAGE COMPUTING TIME TO ESTABLISH SGN AND S ^{2} GNS BY THE FOUR SAMPLING STRATEGIES ON THE NINE DATASETS

Time (Seconds)	SGN	S^2GN			
		RW	BW	LS	ST
MUTAG	1.58×10^2	0.68	0.25	0.60	0.09
PTC	1.93×10^3	1.22	0.80	1.17	0.61
PROTEINS	3.20×10^3	1.19	1.16	2.02	1.63
ENZYMES	3.97×10^3	1.28	1.23	2.11	1.60
NCI1	1.75×10^2	2.67	2.10	2.48	1.75
NCI109	1.75×10^2	2.68	2.11	2.50	1.75
IMDB-BINARY	1.11×10^4	1.48	1.58	1.26	1.11
D&D	7.90×10^2	2.70	3.16	18.32	0.81
COLLAB	> 3 days	39.60	63.00	5.34	1.02

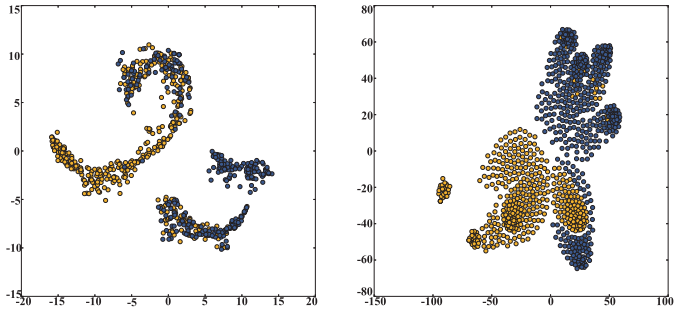


Fig. 7. The t-SNE visualization of structural features using CapsGNN without (left) and with (right) S ^{2} GN-ST. The same color of points represent the same class of graphs in IMDB-BINARY dataset.

samples across different source nodes. For biased walk, we adopt the 2nd random walk mechanism of Node2Vec, where each step of random walk is based on the transition probability α which can be precomputed, so the time consuming of each step using alias sampling is $\mathcal{O}(1)$. Link selection broadens the scope of the start node at each step in the random walk process, thereby accelerating the time to reach the stop condition. Kruskal algorithm to generate spanning trees is a greedy algorithm, which has $\mathcal{O}(|E|log(|E|))$ time complexity. We know that the computational complexity of SGN $^{(1)}$ is $\mathcal{O}(|E|^2)$ and that of constructing SGN$^{(2)}$ is $\mathcal{O}(|E|^4)$. Our S^{2}GN model constrains the expansion of the network scale and reduces the cost of constructing SGNs to the fixed $\mathcal{O}(|E|^2)$. Thus, the time computational complexity $\mathcal{T}$ of our S^{2}GN model is $\mathcal{O}(|E| + |E|^2) \leq \mathcal{T} \leq \mathcal{O}(|E|log|E| + |E|^2|)$ according to the different sampling strategies, which is much lower than that of SGN.

3) Visualization: As a simple case study, we visualize the results of classification on IMDB-BINARY dataset based on CapsGNN method to verify the effectiveness of our S^{2}GN model. Here, we choose S^{2}GN-ST to visualize since this is the best S^{2}GN generated by the single sampling strategy that enhances the classification performance of CapsGNN most. As shown in Fig. 7, the structural features are located in different places by utilizing t-SNE. The left shows the original classification result using CapsGNN without S^{2}GN-ST, while the right depicts the optimized distribution of the same dataset using CapsGNN with S^{2}GN-ST. One can see that the graphs in IMDB-BINARY dataset can indeed be distinguished by the original features of CapsGNN, but it appears that the distinction of graphs could become more explicit after network sampling and SGN mapping, demonstrating the effectiveness of our S^{2}GN model.

V. CONCLUSION

In this paper, we present a novel sampling subgraph network (S^{2}GN) model as well as a structural feature fusion framework for graph classification by introducing network sampling strategies into the SGN model. Compared with the latter, the S^{2}GNs are of higher diversity and controllable scale, and thus benefit the network feature extraction methods to capture more various aspects of the network structure with higher efficiency.

We use different sampling strategies, namely random walk (RW), biased walk (BW), link selection (LS), and spanning tree (ST), to generate the corresponding sampling subgraph networks S^{2}GN-RW, S^{2}GN-BW, S^{2}GN-LS, and S^{2}GN-ST, respectively. The experimental results show that, compared with SGN, S^{2}GN has much lower time complexity, which was reduced by almost two orders of magnitude, and meanwhile they have comparable effects on graph classification. In fact, the network algorithms based on S^{2}GN-BW behave even better than those based on SGN, although each sampling subnetwork is only a part of the original network. More interestingly, when the features of all the four S^{2}GNs are fused and then fed into graph classification models, the classification performance can be significantly enhanced. In particular, when CapsGNN is used to extract the features of these S^{2}GNs, we can achieve the-state-of-the-art results on the PROTEINS and IMDB-BINARY datasets.

In the future, we will try more sampling strategies and then integrate them with SGN to generate more diverse S^{2}GNs; we will also apply our framework to more tasks beyond graph classification, such as link prediction, node classification, etc.

REFERENCES

[1] Q. Xuan, X. Shu, Z. Ruan, J. Wang, C. Fu, and G. Chen, "A self-learning information diffusion model for smart social networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1466–1480, Jul.–Sep. 2020.
[2] 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.
[3] 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.
[4] Z. Ruan, C. Song, X.-H. Yang, G. Shen, and Z. Liu, "Empirical analysis of urban road traffic network: A case study in Hangzhou city, China," *Phys. A, Statist. Mech. Appl.*, vol. 527, 2019, Art. no. 121287.
[5] D. Tang *et al.*, "Predictability of real temporal networks," *Nat. Sci. Rev.*, vol. 7, no. 5, pp. 929–937, 2020.
[6] D. Xu, C. Wei, P. Peng, Q. Xuan, and H. Guo, "GE-GAN: A novel deep learning framework for road traffic state estimation," *Transp. Res. Part C, Emerg. Technol.*, vol. 117, 2020, Art. no. 102635.
[7] 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.
[8] 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.

[9] J. Zhou, J. Shen, S. Yu, G. Chen, and Q. Xuan, "M-Evolve: Structural-mapping-based data augmentation for graph classification," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 190–200, Jan.–Mar. 2021.

[10] M. R. Hosseini, M. Maghrebi, A. Akbarnezhad, I. Martek, and M. Arashpour, "Analysis of citation networks in building information modeling research," *J. Construction Eng. Manage.*, vol. 144, no. 8, 2018, Art. no. 0 4018064.

[11] M. Yasunaga *et al.*, "ScisummNet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 7386–7393.

[12] J. B. Lee, R. A. Rossi, X. Kong, S. Kim, E. Koh, and A. Rao, "Graph convolutional networks with motif-based attention," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 499–508.

[13] Y. Lu, Y. Chen, D. Zhao, and J. Chen, "Graph-FCN for image semantic segmentation," in *Proc. Int. Symp. Neural Netw.*, Springer, 2019, pp. 97–105.

[14] H.-T. Cheng *et al.*, "Wide & deep learning for recommender systems," in *Proc. 1st Workshop Deep Learn. Recommender Syst.*, ACM, 2016, pp. 7–10.

[15] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *Proc. World Wide Web Conf.*, 2019, pp. 3307–3313.

[16] X. Zhang, Y. Li, D. Shen, and L. Carin, "Diffusion maps for textual network embedding," *Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 7587–7597.

[17] C. Fu, Y. Zheng, Y. Liu, Q. Xuan, and G. Chen, "NES-TL: Network embedding similarity-based transfer learning," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1607–1618, Jul–Sep. 2020.

[18] 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, pp. 1–10, 2018.

[19] T. Lane *et al.*, "Comparing and validating machine learning models for mycobacterium tuberculosis drug discovery," *Mol. Pharmaceut.*, vol. 15, no. 10, pp. 4346–4360, 2018.

[20] J. Li, D. Cai, and X. He, "Learning graph-level representation for drug discovery," 2017, *arXiv:1709.03741*.

[21] K. Y. Gao, A. Fokoue, H. Luo, A. Iyengar, S. Dey, and P. Zhang, "Interpretable drug target prediction using deep neural representation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, vol. 2018, pp. 3371–3377.

[22] S.-Y. Liu, J. Xiao, and X.-K. Xu, "Link prediction in signed social networks: From status theory to motif families," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1724–1735, Jul.–Sep. 2020.

[23] 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. 0 52813.

[24] A. Narayanan, M. Chandramohan, L. Chen, Y. Liu, and S. Saminathan, "subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs," in *Proc. Int. Workshop Mining Learn. Graphs*, 2016. [Online]. Available: http://www.mlgworkshop.org/2016/paper/MLG2016_paper_26.pdf

[25] 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.

[26] 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., SIAM*, 2018, pp. 306–314.

[27] Q. Xuan *et al.*, "Subgraph networks with application to structural feature space expansion," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 6, pp. 2776–2789, Jun. 2021.

[28] J. D. Noh and H. Rieger, "Random walks on complex networks," *Phys. Rev. Lett.*, vol. 92, no. 11, 2004, Art. no. 118701.

[29] R. Andersen, F. Chung, and K. Lang, "Local graph partitioning using pagerank vectors," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci.*, 2006, pp. 475–486.

[30] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 355–369, Mar. 2007.

[31] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 701–710.

[32] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 855–864.

[33] M. Kurant, A. Markopoulou, and P. Thiran, "On the bias of BFS (Breadth First Search)," in *Proc. IEEE 22nd Int. Teletraffic Congr.*, 2010, pp. 1–8.

[34] V. Satuluri, S. Parthasarathy, and Y. Ruan, "Local graph sparsification for scalable clustering," in *Proc. ACM SIGMOD Int. Conf. Manage. Data.*, Association for Computing Machinery, 2011, pp. 721–732.

[35] N. M. Devi and S. R. Kasireddy, "Graph analysis and visualization of social network big data," in *Social Netw. Forensics, Cyber Security, Mach. Learn.*. Berlin, Germany: Springer, 2019, pp. 93–104.

[36] G. Li, M. Semerci, B. Yener, and M. J. Zaki, "Graph classification via topological and label attributes," in *Proc. 9th Int. Workshop Mining Learn. Graphs*, San Diego, USA, vol. 2, 2011. [Online]. Available: https://www.cs.purdue.edu/mlg2011/papers/paper_1.pdf

[37] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," in *Proc. Int. Workshop Mining Learn. Graphs*, 2017. [Online]. Available: http://www.mlgworkshop.org/2017/paper/MLG2017_paper_21.pdf

[38] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels," *J. Mach. Learn. Res.*, vol. 12, no. null, pp. 2539–2561, Nov. 2011.

[39] P. Yanardag and S. Vishwanathan, "Deep graph kernels," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 1365–1374.

[40] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl

[41] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3538–3545.

[42] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn.* vol. 48, 2016, pp. 2014–2023. [Online]. Available: http://proceedings.mlr.press/v48/niepert16.html

[43] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[44] X. Zhang and L. Chen, "Capsule graph neural network," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Byl8BnRcYm

[45] 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.

[46] J.-P. Eckmann and E. Moses, "Curvature of co-links uncovers hidden thematic layers in the world wide web," in *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 9, 2002, pp. 5825–5829.

[47] K. Pearson, "The problem of the random walk," *Nature*, vol. 72, no. 1867, pp. 342–342, 1905.

[48] Y. Azar, A. Z. Broder, A. R. Karlin, N. Linial, and S. Phillips, "Biased random walks," in *Proc. 24th Annu. ACM Symp. Theory Comput.*, 1992, pp. 1–9.

[49] K. M. Adal, B. B. Samir, and N. B. Z. Ali, "Biased random walk based routing for mobile ad hoc networks," in *Proc. IEEE Int. Conf. Intell. Adv. Syst.*, 2010, pp. 1–6.

[50] A. Dey *et al.*, "A new algorithm for finding minimum spanning trees with undirected neutrosophic graphs," *Granular Comput.*, vol. 4, no. 1, pp. 63–69, 2019.

[51] L. Najman, J. Cousty, and B. Perret, "Playing with Kruskal: Algorithms for morphological trees in edge-weighted graphs," in *Proc. Int. Symp. Math. Morphology Appl. Signal Image Process.*, Springer, 2013, pp. 135–146.

[52] K. Okamoto, W. Chen, and X.-Y. Li, "Ranking of closeness centrality for large-scale social networks," in *Proc. Int. Workshop Front. Algorithmics*, Springer, 2008, pp. 186–195.

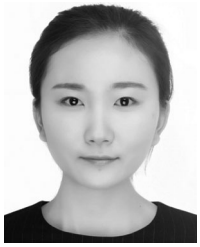
[53] S. B. Seidman, "Network structure and minimum degree," *Social Netw.*, vol. 5, no. 3, pp. 269–287, 1983.

[54] A. N. Langville and C. D. Meyer, "Deeper inside pagerank," *Internet Math.*, vol. 1, no. 3, pp. 335–380, 2004.

[55] L. Lü, D. Chen, X.-L. Ren, Q.-M. Zhang, Y.-C. Zhang, and T. Zhou, "Vital nodes identification in complex networks," *Phys. Rep.*, vol. 650, pp. 1–63, 2016.

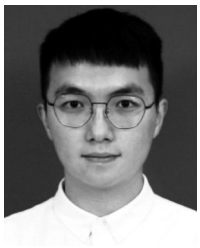
[56] P. Zhou, W. Yang, W. Chen, Y. Wang, and J. Jia, "Modality attention for end-to-end audio-visual speech recognition," in *Proc. ICASSP IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 6565–6569.

[57] 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.
[58] 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. Med. Chem.*, vol. 34, no. 2, pp. 786–797, 1991.
[59] H. Toivonen, A. Srinivasan, R. D. King, S. Kramer, and C. Helma, "Statistical evaluation of the predictive toxicology challenge 2000-2001," *Bioinf.*, vol. 19, no. 10, pp. 1183–1193, 2003.
[60] 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. i 47–i56, 2005.
[61] R. Rossi and N. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. AAAI Conf. Artif. Intell.*, vol. 29, no. 1, 2015, pp. 4292–4293.
[62] P. D. Dobson and A. J. Doig, "Distinguishing enzyme structures from non-enzymes without alignments," *J. Mol. Biol.*, vol. 330, no. 4, pp. 771–783, 2003.
[63] S. Sabour, N. Frosst, and G. Hinton, "Matrix capsules with em routing," in *Proc. 6th Int. Conf. Learn. Representations, ICLR*, 2018, pp. 1–15.
[64] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, no. 2, 2004, Art. no. 0 26113.
[65] J. Yoon, A. Blumer, and K. Lee, "An algorithm for modularity analysis of directed and weighted biological networks based on edge-betweenness centrality," *Bioinformatics*, vol. 22, no. 24, pp. 3106–3108, 2006.
[66] A. Cuzzocrea, A. Papadimitriou, D. Katsaros, and Y. Manolopoulos, "Edge betweenness centrality: A novel algorithm for QoS-based topology control over wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 35, no. 4, pp. 1210–1217, 2012.



Jinhuan Wang received the B.S. and M.S. degree from the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China, in 2017 and 2020, respectively. She is working toward the Ph.D. degree with the School of Information Engineering, Zhejiang University of Technology. Her research interests include social network data mining and machine learning.

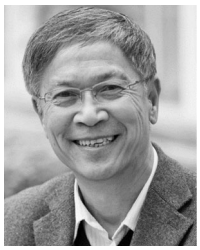
Pengtao Chen is currently working toward the B.S. degree with the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China. His research interests include machine learning and graph mining.



Bin Ma received the B.S. degree from the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China, in 2020. He is studying for the M.S. degree with the Electric and Computer Engineering Department of Viterbi engineering School, University of Southern California. His research interests include data science and machine learning.

Jiajun Zhou received the B.S. degree in automation from the Zhejiang University of Technology, Hangzhou, China, in 2018, where he is currently working toward the M.S. degree in control theory and engineering with the College of Information and Engineering. His current research interests include graph mining and deep learning, especially for network security and knowledge graph reasoning.

Zhongyuan Ruan received the B.Sc. degree in physics from Guizhou University, Guiyang, China, and the Ph.D. degree in physics from the East China Normal University, Shanghai, China, in 2008 and 2013, respectively. He is currently a Lecturer with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, China. His current research interests include complex systems and complex networks.



Guanrong Chen (Life Fellow, IEEE) received the M.Sc. degree in computer science from Sun Yatsen University, Guangzhou, China, in 1981, and the Ph.D. 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 at the 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.

Qi Xuan (Senior Member, IEEE) received the B.S. and Ph.D. 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. He was a Postdoctoral Fellow with the Department of Computer Science, University of California at Davis, CA, from 2012 to 2014, and a visiting scholar with the School of Computer Science, Carnegie Mellon University, PA, in 2019. He is currently a professor with the Institute of Cyberspace Security, Zhejiang University of Technology, Hangzhou. His current research interests include network science, graph data mining, AI security, machine learning, and computer vision.