IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 30, 2020, accepted May 12, 2020, date of publication May 18, 2020, date of current version May 29, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2995406

Graph Convolutional Architectures via Arbitrary Order of Information Aggregation

CHUNPENG ZHOU[1,*], BENYUN SHI[2,*], HONGJUN QIU[1], AND JIMING LIU[3]

[1]School of Cyberspace, Hangzhou Dianzi University, Zhejiang 310018, China
[2]School of Computer Science and Technology, Nanjing Tech University, Nanjing 211800, China
[3]Department of Computer Science, Hong Kong Baptist University, Hong Kong

Corresponding authors: Benyun Shi (benyunshi@outlook.com) and Jiming Liu (jiming@comp.hkbu.edu.hk)

∗Chunpeng Zhou and Benyun Shi contributed equally to this work.

This work was supported in part by the Hong Kong Research Grants Council under Grant RGC/HKBU12201619 and Grant 12201318, and in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LQ19F030011.

ABSTRACT Graph representation learning (GRL) has recently drawn a lot of attention due to its advantage in solving various machine learning tasks on graphs/networks, ranging from drug design to recommendation systems. One typical GRL approach is graph embedding, the purpose of which is to learn a map that encodes or represents network elements as points in a low-dimensional vector space so that downstream machine learning methods can be easily implemented. Initially, most graph embedding algorithms learn such a map independently from subsequent machine learning tasks. Therefore, they have limitations in solving supervised machine learning tasks on networks. Later, a great deal of graph convolutional networks (GCNs) have been proposed to learn node representations in an end-to-end manner based on different information aggregation mechanisms. By treating network structure as a computational layer in a GCN, the associated information of nodes with higher-order proximity can be aggregated by increasing the number of layers (i.e., depth) of the GCN. As a consequence, the computational overhead will increase and the representations will be projected towards a steady state. To solve this problem, in this paper, we propose a multi-channel graph convolutional network (MCGCN) that allows higher-order information aggregation by enriching the number of input channels. Based on the notion of Katz index, our model can further achieve an arbitrary order of information aggregation without increasing the computational overhead. Comprehensive experiments on several benchmark networks demonstrate the effectiveness of the proposed architecture by comparing it with the-state-of-art GRL methods in terms of node classification and computational efficiency.

INDEX TERMS Graph representation learning, graph convolutional networks, information aggregation, node classification.

I. INTRODUCTION

Networks or graphs are a ubiquitous data structure and have been extensively employed to capture interactions (i.e., edges) between individual units (i.e., nodes) of complex systems in biology, neuroscience, engineering, and social science. Along this line, machine learning tasks on networks have attracted lots of attention with applications ranging from drug design to recommendation systems in social networks [1]–[3]. However, due to the nonlinearity and high dimensions of network structure, it is difficult to directly implement classical machine learning methods on networks. To solve this problem, various graph representation learning (GRL) methods have been proposed in recent years [4]–[6]. One typical GRL method is graph embedding, the purpose of which is to learn a map that encodes network nodes in a low-dimensional vector space such that certain structural properties of the network can be preserved [7]–[9]. In doing so, the obtained node embeddings/representations can then be treated as feature inputs of downstream machine learning methods to solve specific network analytic tasks, such as community mining [10], [11], node classification [12], and link prediction [13]. Nevertheless, there are still two shortcomings. First, most existing graph embedding algorithms focus merely on preserving structural properties of networks, such as structural proximity [14], [15], equivalence [16], [17], and identity [18], [19]. They are limited in learning node representations of attributed

The associate editor coordinating the review of this manuscript and approving it for publication was Juan Wang.

networks, where each node is associated with additional heterogeneous information, such as node attributes and labels. Second, most graph embedding algorithms learn the map function independently from subsequent machine learning tasks. In other words, the representations are learned without the supervision of downstream machine learning outputs (e.g., node labels).

Because unsupervised GRL methods do not leverage label information in the learning process, they have natural defects in solving (semi)supervised machine learning tasks on networks. In the past, to solve the node classification problem, many community-preserving embedding algorithms have been designed with the assumption that densely connected nodes tend to have the same label [15]. However, in reality, it is difficult to determine in advance what structural properties are related to node labels [17]. The situation becomes even more complicated for attributed networks. Therefore, to solve (semi)supervised machine learning tasks on attributed networks, the primary challenge lies in how to find a supervised way to learn representations of network nodes and their associate attributes in an end-to-end manner.

By extending the idea of deep learning on networks, graph neural networks (GNNs) have been recognized as a useful framework to tackle supervised network analytic problems (see detailed surveys [6], [9], [20], [21]). In essence, GNNs treat the network structure as a computational graph, and train the whole neural network model in an end-to-end manner [22]. For example, inspired by convolutional neural networks in the field of computer vision, a variety of graph convolutional networks (GCNs) have been proposed in recent years [23]–[26]. By adopting appropriate message passing mechanisms in each convolutional layer of a GCN, each node can aggregate attribute information from its neighboring nodes in the network. However, as the depth of a GCN increases, nodes will aggregate information from other nodes with higher-order proximity. In doing so, node representations will be projected towards a steady state after several aggregation steps [27]. As a result, the depth of the existing GCNs cannot be too large.

In many real-world applications, node labels are relevant to their structural roles in a network [28]–[30]. Moreover, nodes with the same/similar structural roles may be far away from each other in the network [17]. In this case, GCNs with limited depth cannot aggregate information of nodes with similar roles but far away from each other. Therefore, it would be desirable to develop a computationally efficient convolutional architecture such that information of nodes with higher-order of proximity can be exploited through appropriate aggregators while keeping their heterogeneity. In this paper, instead of increasing the depth of GNNs, we aim to develop a novel graph convolutional architecture by enriching the information channels to support arbitrary order of information aggregation through the network.

Specifically, in this paper, we focus mainly on how to develop a graph convolutional architecture to solve the semi-supervised node classification problem on attributed networks in an end-to-end manner. The novelty and contributions of this paper are summarized as follows:

1) We propose a multi-channel graph convolutional network (MCGCN) that allows higher-order of information aggregation by enriching the number of input channels. Based on the notion of Katz index, the proposed model can further achieve an arbitrary order of information aggregation without increasing the computational overhead.
2) We introduce a shared weight mechanism to assess the relative importance of different attributes, which are weighted and aggregated among nodes with a certain order of proximity.
3) We carry out experiments on several benchmark datasets to evaluate the performance of the proposed MCGCN architecture, by comparing with the state-of-the-art GRL methods in terms of classification accuracy and computational efficiency.

The remainder of this paper is organized as follows. In Section II, we briefly review the related work of this paper. In Section III, we formulate the semi-supervised node classification problem on attributed networks. In Section IV, we present a multi-channel convolutional architecture that can aggregate information from nodes with arbitrary order of proximity. In Section V, we carry out experiments to evaluate the performance of the proposed MCGCN method by comparing it with several state-of-the-art methods. Finally, we conclude this work in Section VI.

II. RELATED WORK

In recent years, the graph embedding approach has been proposed with the purpose of *automatically* representing, or encoding network elements into a low-dimensional vector space by preserving certain network properties [4], [5], [7], [8]. Generally, existing node embedding methods can be classified into three categories: factorization-based approach [14], [31], [32], random walk-based approach [33]–[35], and deep learning-based approach [16]. The first two categories focus mainly on encoding network elements by preserving the structural properties of networks. For example, concerning community-based embeddings [15], the basic idea is to learn embeddings of each node such that the inner product between any two learned vectors approximates certain measures of structural proximity. Once node embeddings (or representations) are obtained, they can be treated as feature inputs for downstream machine learning methods to solve specific network analytic tasks. In doing so, such methods are considered as an unsupervised graph representation learning approach, where the representations are learned independently from the downstream machine learning tasks.

The deep learning-based approach usually learns the encode/decode functions by involving an end-to-end learning process [36], [37]. To deal with (semi)supervised machine learning problems on attributed networks, graph neural network models have been proposed, where the network

structure is treated as a computational graph in each layer of a GNN [6], [20], [21]. By leveraging the information about nodes labels and attributes, they can achieve much better performance than hand-engineered network analytic methods and unsupervised GRL methods [38]. For example, by adopting the message passing mechanisms, GNNs can aggregate node/edge attributes from neighboring nodes with a certain order of proximity in the network. By training model parameters or aggregators with the supervision of label information, GNNs have the potential to deal with both inductive and transfer learning tasks on networks.

As a typical GNN approach, graph convolutional networks are inspired by the powerful convolutional neural networks in the field of computer vision. Recently, researchers have proposed a variety of graph convolutional networks (GCNs), including spectral-based GCNs [24], [39], spatial-based GCNs [26], [27], [40], and graph attention networks (GAT) [25]. The differences lie in how the *filters* are defined to aggregate the information through the network. For example, the Chebyshev method defined graph convolutions using a K-degree polynomial of the Laplacian to avoid the huge computational cost of the Laplacian eigendecomposition [39]. Then, the vanilla GCN simplified graph convolutions with a specified renormalization trick [24]. By extending the vanilla GCN framework, the GraphSAGE method proposed different aggregator architectures [40]. To reduce the computational complexity, the GraphSAGE chose to aggregate information from a fixed number of neighboring nodes, instead of from all neighbors. Meanwhile, the Monet model presented a generalized CNN architecture, which aggregated the local information from networked data [41]. The GAT networks incorporated the well-known attention mechanism into each layer of the GCN framework such that the weight of information from different neighbors can be learned [25]. More recently, the SGC network simplified the vanilla GCN framework by reducing the number of non-linear activations and aggregation layers [26].

From a computational perspective, the spectral-based GCNs, such as the Chebyshev method [39], often have a relative huge computational complexity due to the computation of eigendecomposition. To avoid the computational burden, the spatial-based GCNs have introduced various message passing mechanisms, where each node can aggregate information from its neighbors in the network. Then, the objective is to learn the weight matrix and/or aggregators through an end-to-end learning process [26]. However, the drawback lies in that as the depth of a GCN increases, the learned representations will be projected towards a steady state [27]. The assumption behind is that neighboring nodes in a network tend to have similar representations, which instead limits the depth of GNNs. Paradoxically, if the depth of a GCN is not large enough, nodes cannot aggregate the information of distant nodes. This will limit its implementation on many real-world applications, for example, classifying nodes with respect to their structural roles [28]–[30]. To tackle this problem, it would be desirable to develop novel graph convolutional architectures that can (i) aggregate information from nodes with arbitrary order of proximity, and (ii) reflect the relative importance of information from nodes with different orders of proximity.

III. PROBLEM STATEMENT

In this section, we first introduce some notations and definitions we will use in the remainder of this work. Then, we introduce the semi-supervised multi-class node classification problem on networks.

Let $G = \{V, E, A\}$ be a network, where $V = \{1, \cdots, N\}$ consists of N interconnected nodes, and $E = \{e_{ij} | i, j \in V\}$ is a set of M edges between the nodes. Moreover, denote $A \in \mathcal{R}^{N \times N}$ as the adjacency matrix of G, where $a_{ij} = 1$ if there exists an edge $(i, j) \in E$ and $a_{ij} = 0$ otherwise. If G is an undirected network, A is symmetric, that is, $A_{ij} = A_{ij}$, for $\forall i, j \in V$. Further, we denote I_N as the $N \times N$ identity matrix, and denote D as the diagonal degree matrix, where $D_{ii} = \sum_j A_{ij}$ represents the degree of node i.

Suppose that each node i in G is associated with a d-dimensional feature vector $\mathbf{x}_i \in \mathcal{R}^d$. Then, the attributed graph can be defined as follows:

Definition 1 (Attributed Graph or Network): A graph $G = (V, E, X)$ is called an attributed graph, where each node i is associated with a d-dimensional feature vector $\mathbf{x}_i \in \mathcal{R}^d$. The entire feature matrix $X \in \mathcal{R}^{N \times d}$ stacks N feature vectors on top of one another. In other words, we have $\mathbf{x}_i$ is the i^{th} row of matrix X.

In this paper, we assume that each node i in an attributed graph $G = (V, E, X)$ belongs to one of C classes, which can be denoted as a C-dimensional one-hot vector $\mathbf{y}_i \in \{0, 1\}^C$. For example, if the label of i is c, the we have $\mathbf{y}_i = [0_{(1)}, \cdots, 0_{(c-1)}, 1, 0_{(c+1)}, \cdots, 0_{(C)}]$. When stacking the labels $\mathbf{y}_i$, we have a label matrix $Y \in \{0, 1\}^{N \times L}$, where $\mathbf{y}_i$ is the i^{th} row of Y. Suppose only a subset of nodes $V_L \in V$ in a network have labels Y_L, and the rest of nodes $V_U = V \backslash V_L$ are unlabeled. Formally, the semi-supervised node classification problem can be defined as follows:

Definition 2 (Semi-Supervised Node Classification on Networks): Given an attributed graph $G = (V, E, X)$ and the label $\mathbf{y}_j$ of each node $j \in V_L$, the objective of the node classification problem is to (i) learn a model f such that $f(i; A, X, V_L, Y_L) \rightarrow \mathbf{y}_i$ for $\forall i \in V_L$, and (ii) predict the label of node $j \in V_U$ with $\mathbf{y}_j = f(j; A, X, V_L, Y_L)$.

In the following, we will propose a graph convolutional architecture to solve the semi-supervised multi-class node classification problem on networks.

IV. THE PROPOSED GCN ARCHITECTURE

In this section, we build upon graph convolutional networks to learn node representations for semi-supervised node classification in an end-to-end fashion. We first introduce the message-passing architecture in GCNs. Then, we present our higher-order GCN architecture that can aggregate any order of information over the network.

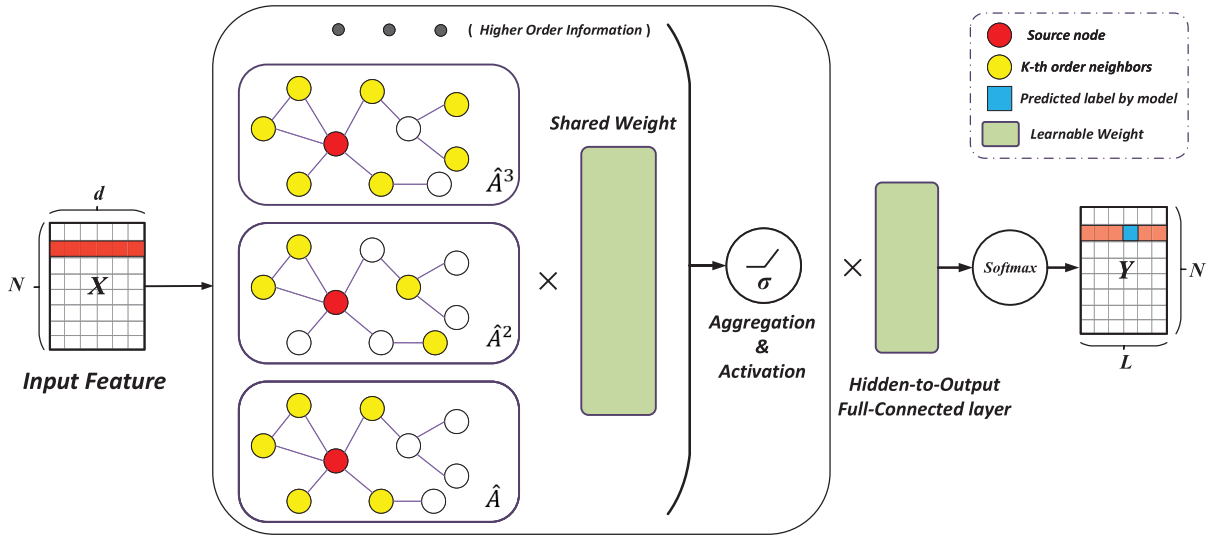


FIGURE 1. A schematic depiction of the proposed higher-order convolutional architecture for semi-supervised node classification task on networks. The model takes the feature matrix X as input, each row of which represents the feature vector of a node. Instead of increasing the number of convolutional layers, the architecture allows higher order of information aggregation by enriching the number of information channels based on $\hat{A}$, $\hat{A}^2$, $\hat{A}^3$, and so on. Each node aggregates its information of all channels through a shared weight mechanism. Finally, a full-connected layer together with the softmax function is used for the multi-class node classification task.

A. A MESSAGE-PASSING MECHANISM

In general, the message-passing architecture can be described as a multi-layer convolutional network. Here, we follow Kipf & Welling to introduce the architecture of GCNs in the context of node classification [24]. Specifically, for the k^{th} convolutional layer in an l-layer GCN, the message-passing model is formulated as follows:

$$H_k = \begin{cases} X, & \text{if } k = 0 \\ \sigma(\hat{A}H_{k-1}W_{k-1}), & \text{if } k \in [1, l] \end{cases} \tag{1}$$

where $H_0 = X$ represents the feature matrix X as the inputs of the model. Moreover, $H_k \in R^{N \times d_k}$ are the output node representations (i.e., "message" or "information") of the k^{th} layer, and subsequently the input node representation of the $(k+1)^{th}$ layer. Consequently, the nodal information will be aggregated through the message-passing model. Here, σ represents the message propagation function for information aggregation over the network. There are many possible implementations of information aggregators. For example, in [24], the information is aggregated through a combination of linear transformations and ReLU activation.

In addition, it is noteworthy that $\hat{A}$ is the renormalized adjacent matrix of the graph. Formally, it is calculated as follows:

$$\begin{aligned} \hat{A} &= \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}} \\ \tilde{D}_{ii} &= \sum\nolimits_j \tilde{A}_{ij} \\ \tilde{A} &= A + I_N \end{aligned} \tag{2}$$

where $\tilde{D}$ is the diagonal degree matrix of $\tilde{A}$. The utilization of the renormalization trick can constrain the number of model parameters. In doing so, the model and its variants can address the overfitting problem and reduce the computational overhead of GNNs [24], [39].

With this renormalization trick, every node in the graph can aggregate information from their direct neighbors, also known as the first-order neighbors, in each forward propagation layer of a GCN. Accordingly, if we want to pass messages to, or aggregate information from nodes with high-order proximity, one intuitive way is to increase the number of convolutional layers (i.e., the model depth). Nonetheless, this may result in the overfitting problem as the introduction of more learnable parameters W. Kipf & Welling have shown in their experiments that increasing the model depth will deteriorate the performance of downstream learning tasks, even though the residual connections are used for training between GCN layers [42]. Therefore, the depth of the vanilla GCN and its variants usually have less than three convolutional layers. As a consequence, they cannot aggregate information from distant nodes in the network.

B. A MULTI-CHANNEL CONVOLUTIONAL LAYER

Instead of increasing the number of convolutional layers, in this paper, we propose a multi-channel convolutional architecture by enriching the number of input channels such that higher-order information can be aggregated. As shown in Figure 1, the model takes the feature matrix $X \in \mathcal{R}^{N \times d}$ as input, each row of which represents the feature vector of a node. Then, node information can be separately aggregated in different channels. Specifically, the propagation network in the channel k corresponds to a specific matrix $\hat{A}^k$, which is the k^{th} power of the renormalized adjacency matrix $\hat{A}$. Our forward model then takes the following form:

$$H = \text{AGG}(\hat{A}XW_1, \hat{A}^2XW_2, \hat{A}^3XW_3, \cdots), \tag{3}$$

where $\hat{A}^i X W_i$ represents a high-order GCN channel that captures the information from i^{th} order neighbors. The operator AGG aggregates node-wise information from all channels.

In this paper, we use the SUM operator as the aggregator function. The reasons are twofold: First, the aggregation scheme in the vanilla GCN can be deemed as a class of functions over the sets of neighbor nodes [40], [43]. Among different aggregator functions, such as SUM, MEAN, MAX, only the SUM operator can capture the full multiset (a generalized concept of the set) [43]. Consequently, with respect to the proposed architecture in this paper, the SUM operator is more powerful than others to distinguish diverse network structures. Second, the implementation of the SUM operator can attain a weighted summation over different convolutional channels, which can magnify the relative important information by summation. In doing so, the forward model can be formulated as follow:

$$H = \sum_{i=1}^{k} (\hat{A}^i X W_i), \tag{4}$$

where k represents the total number of channels in the architecture.

From the above equation, the learnable weight W_i in each channel can be regarded as a pre-processing operation on node features [26]. To reduce the number of model parameters and avoid overfitting, in this paper, we use a shared weight matrix W_S among different channels. Further, we use a non-linear function σ to boost the expressive power of our model. In doing so, the feature aggregation rule can be rewritten as:

$$H = \sigma(\sum_{i=1}^{k} [\hat{A}^i X W_S]). \tag{5}$$

Finally, we add a parameter α to make a flexible adjustment among different channels:

$$H = \sigma[\sum_{i=1}^{k} (\alpha\hat{A})^i X W_S]. \tag{6}$$

It determines the weight decay of a channel as the order increase. In doing so, the proposed architecture can aggregate as high as k^{th} order information through a network without increasing the depth of GCNs. We name such a k-channel GCN model as "MCGCN-k" for comparison and evaluation in Section V. The detailed forward process is shown in Algorithm 1.

C. INFORMATION AGGREGATION WITH ARBITRARY ORDER OF PROXIMITY

Along this line, to aggregate arbitrary order of information for each node, the number of channels should not less than the diameter of the network G. However, for large-scale networks, the diameter will become very large, that is, $k \to \infty$ in Equation 6. Alternatively, we introduce the Katz index as follows:

Algorithm 1 The Forward Process of MCGCN-k Algorithm

Input: The adjacent matrix A, feature matrix X, and parameter α
Output: The predicted label distribution matrix $\overline{Y}$

$\tilde{A} \leftarrow A + I_N$, $\tilde{D}_{ii} \leftarrow \sum_j \tilde{A}_{ij}$;
$\hat{A} \leftarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$;
$A' \leftarrow \alpha\hat{A}$;
if $k > 1$ **then**
  for $i \in [2, k]$ **do**
    $A' \leftarrow \alpha A(I + A')$;
  end
end
$H \leftarrow \sigma(A' X W_S)$;
$\overline{Y} \leftarrow \text{softmax}(H W_F)$;
return $\overline{Y}$

$$\begin{aligned} A_{\text{Katz}} &= \sum_{i=1}^{\infty} \alpha^i \hat{A}^i \\ &= (\alpha\hat{A}) + (\alpha^2\hat{A}^2) + (\alpha^3\hat{A}^3) \cdots \\ &= \alpha\hat{A} \cdot [\, I + (\alpha\hat{A}) + (\alpha^2\hat{A}^2) + \cdots] \\ &= \alpha\hat{A} \cdot (\, I + \sum_{i=1}^{\infty} \alpha^i \hat{A}^i) \\ &= \alpha\hat{A} \cdot (\, I + A_{Katz}). \end{aligned} \tag{7}$$

As a result, the Katz index can formulated as:

$$A_{\text{Katz}} = \alpha\hat{A} \cdot (\, I - \alpha\hat{A})^{-1}. \tag{8}$$

Accordingly, the feature aggregation rule becomes to be:

$$H = \sigma[\alpha\hat{A} \cdot (\, I - \alpha\hat{A})^{-1} X W_S]. \tag{9}$$

Notably, the parameter α should be properly set to make sure the convergence [44]. In doing so, we can aggregate arbitrary order of information with just one-layer GCN. We name this model as "MCGCN-Katz". The detailed forward process is shown in Algorithm 2.

Algorithm 2 The Forward Process of MCGCN-Katz Algorithm

Input: The adjacent matrix A, the feature matrix X, and parameter α
Output: The predicted label distribution matrix $\overline{Y}$

$\tilde{A} \leftarrow A + I_N$ $\tilde{D}_{ii} \leftarrow \sum_j \tilde{A}_{ij}$;
$\hat{A} \leftarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$;
$A_{\text{Katz}} \leftarrow \alpha \cdot \hat{A} \cdot (\, I - \alpha \cdot \hat{A})^{-1}$;
$H \leftarrow \sigma(A_{\text{Katz}} X W_S)$;
$\overline{Y} \leftarrow \text{softmax}(H W_F)$;
return $\overline{Y}$

D. A FULLY CONNECTED LAYER

Once node features are aggregated through the message-passing layer, they are treated as input of a fully connected layer for multi-class node classification (see Figure 1). Formally, the output of the forward propagation model can be estimated as follows:

$$\overline{Y} = \text{softmax}([\sigma(\sum_{i=1}^{k} \hat{A}^i X W_S)]\, W_F), \tag{10}$$

where $W_S \in R^{d \times d_h}$ is a learnable input-to-hidden weight matrix in the convolutional layer, $W_F \in R^{N \times C}$ is a learnable hidden-to-output weight matrix in the fully connected layer, and σ denotes an activation function. Many activation functions can be adopted by σ, such as the rectified linear unit (ReLu) and the leaky ReLU (LReLU). While in this paper, we employ the Exponential Linear Unit (ELU) as the activation function [45]:

$$\sigma(x) = \text{ELU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \beta(e^x - 1), & \text{if } x < 0 \end{cases} \tag{11}$$

Accordingly, the output of the fully connected layer is then normalized by a row-wise softmax activation, which can transform the output into a series of probability for each class:

$$\hat{\mathbf{y}}_i = \text{softmax}(\mathbf{o}_i) = \frac{exp(\mathbf{o}_i)}{\sum_j exp(\mathbf{o}_j)}, \tag{12}$$

where $\mathbf{o}_i$ the output vector of node i after the information propagation and activation. The estimated label distribution $\overline{Y} \in \mathcal{R}^{N \times C}$ stacks N class distribution vectors $\hat{\mathbf{y}}_i$ on top of one another.

Concerning the multi-class node classification tasks, the cross entropy loss is used to train model parameters:

$$Loss = -\sum_{i \in V_{\text{train}}} \sum_{c=1}^{C} y_{ic} \log \hat{y}_{ic} \tag{13}$$

where V_{train} is the set of labeled nodes used for training, and C is the number of classes. The y_{ic} and $\hat{y}_{ic}$ denote the ground-truth value and the predicted probability of node i with respect to class c. It is worth noting that node set V_{train} is different from the labeled set V_L because there are some nodes with known labels used for validation.

E. COMPUTATIONAL COMPLEXITY

For the sumGCN-k model, the computational complexity of evaluating Equation 6 is $O(k \times |E| \times d)$, where k is the number of channels in the convolutional architecture, $|E|$ is the number of the non-zero elements in the $\hat{A}$, and d is the dimension of node feature. Generally, we have $|E| \ll N^2$ due to the sparsity of real-world networks. Moreover, we have $k \ll |E|$ in the proposed model because k is less than the diameter of the network. For the sumGCN-Katz model, the computational overhead is relatively high because the value of the Katz index requires calculating the inverse of a matrix. Due to the existence of hyperparameter α, the entries in $(\alpha\hat{A})^i$ will become small as the order increases. Therefore, an alternative implementation is to take a relatively large k to attain an approximation of the Katz index, such as $\alpha^k < 0.1$. In summary, the computational complexity of both models is linear to the number of edges in the network.

V. EXPERIMENTS

In this section, we first carry out experiments on several benchmark networks to evaluate the performance of our proposed architecture by comparing it with the state-of-the-art methods with respect to the semi-supervised node classification problem on networks. Then, we conduct experiments to verify the computational efficiency of our architecture. Finally, we show how the different settings of hyperparameter contribute to the performance of our methods in terms of classification accuracy.

A. DATASETS

To evaluate the advantage of higher-order information aggregation, we conduct experiments on two types of network datasets that have been wildly used in the field of graph representation learning. One is the academic paper citation networks, the other is the air-traffic networks.

1) CITATION NETWORKS

For the citation networks, each node represents a published paper and each edge between two nodes indicates the existence of a citation relationship between them. The ground-truth class of each node represents the main subject of this paper. Intuitively, the densely connected modes are more likely to have the same class. In this case, it is not necessary to aggregate information from distant nodes. In our experiments, we adopt the following three citation networks:

1) *Cora Network* [46]: This dataset contains a selection of the Cora database which consists of 2,708 papers in the field of machine learning with 5,229 citation links. All papers have been classified into 7 classes according to their main subject, involving Genetic Algorithms, Neural Networks, Probabilistic Methods, Reinforcement Learning, Rule Learning, and so on. Besides, each dataset contains a binary Bag-of-Words (BoW) feature vector, which is extracted from the paper abstract. The dimension of the feature vector is 1,433.
2) *CiteSeer Network* [46]: This dataset is a part of the Citesser database, which consists of 3,327 papers in the field of machine learning and 4,732 citation links. The class of each node is also generated based on their main subject. Totally, there are 6 classes, involving Agents, Artificial Intelligence, Databases, Information Retrieval, and so on. The feature vector of each node is generated using the Bag-of-Words (BoW) method, The dimension of the feature vector is 3,703.
3) *Pubmed Network* [47]: The Pubmed dataset consists of 19,717 scientific publications and 44,338 citation links from the PubMed database. All papers have been

TABLE 1. Statistic characteristics of the all datasets used in our experiments.

Models	Cora	Citeseer	Pubmed	Brazilian Air	European Air
# Nodes	2,708	3,327	19,717	131	399
# Edge	5,429	4,732	44,338	1,038	5995
# Features	1433	3703	500	*None*	*None*
# Classes	7	6	3	4	4
# Training Nodes	140 (5.2%)	120 (3.6%)	60 (0.3%)	33 (25.2%)	100 (25.21%)
# Validation Nodes	500 (18.5%)	500 (15.0%)	500 (2.5%)	33 (25.2%)	100 (25.1%)
# Test Nodes	1000	1000	1000	65	199
Structure & Labels	Structural proximity			Structural roles	

classified into 3 classes according to the disease type. Each node in this network is associated with a TF-IDF weighted word vector built from a dictionary consisting of 500 unique words.

2) AIR-TRAFFIC NETWORKS

For the air-traffic networks, each node represents a real airport and each edge between two nodes indicates the existence of airlines between them. Node labels of these networks are generated based on the airport activity, which is measured by the total number of landings plus takeoffs in 2016. Intuitively, the class of an airport is related to its structural roles in the corresponding network. Therefore, nodes that far away from each other may belong to the same class. In our experiments, we focus on the following two air-traffic networks:

1) *Brazilian air-traffic network* [18]: Data was collected from the National Civil Aviation Agency (ANAC) of Brazil from January to December 2016, which contains 131 nodes and 1,038 edges. The ground-truth label has 4 classes.
2) *European air-traffic network* [18]: Data was collected from the Statistical Office of the European Union (Eurostat) from January to November 2016. The network has 399 nodes and 5,995 edges. Totally, there are 4 ground-truth classes.

It is noteworthy that there are no additional node features in air-traffic networks. Therefore, we treat the identity matrix I_N as the input feature matrix X of our methods.

Finally, the statistic characteristics of all datasets are summarized in Table 1. Based on existing studies [17], [18], [34], the labels of citation networks are more relevant to structural proximity among different nodes, while the labels in air-traffic networks are more relevant to structural equivalence (or roles) of different nodes. To better understand their differences, we illustrate the Cora citation network and the European air-traffic network in Figure 2. Nodes with the same labels are shown in the same color. It can be observed that nodes with the same color are inclined to exhibit the community structure in the Cora citation network, for example, the red nodes in Figure 2(a). On the contrary, in the European air-traffic network, nodes with the same color are far apart from each other, such as the yellow nodes in Figure 2(b). In this case, one goal of our experiments is to evaluate whether the proposed methods can achieve better performance on both types of datasets.

B. BASELINE METHODS

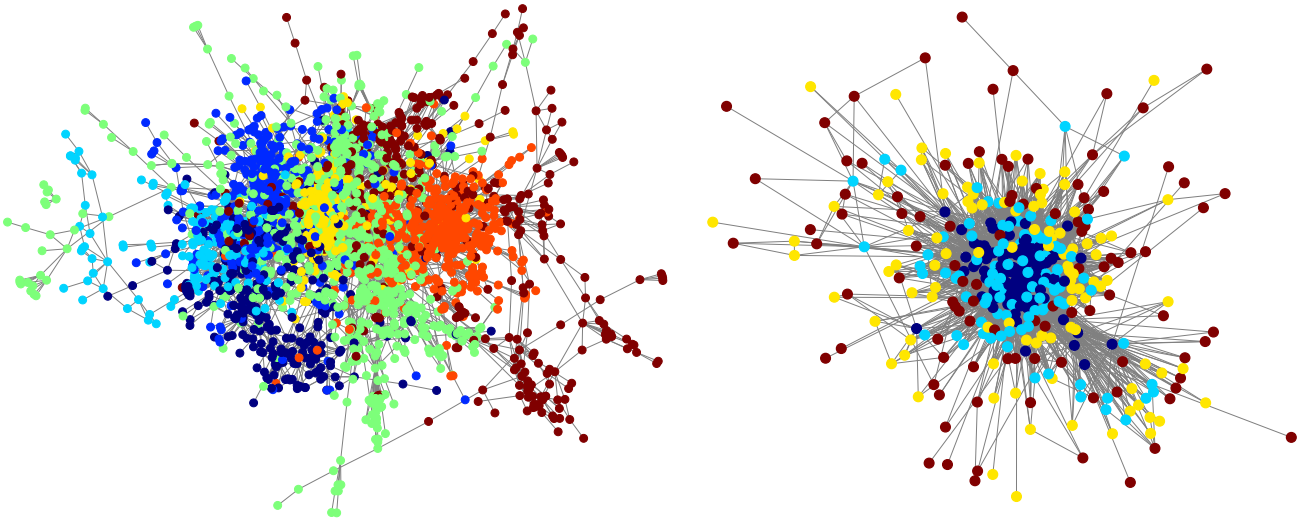
Although a large amount of GRL methods have been designed for graph learning, such as spectral clustering [49], label propagation (LP) [36], manifold regularization [50], semi-supervised embedding (SemiEmb) [51], evidence shows that most of them perform relative poor compared with the recent advanced GCN methods. As a result, we omit these results here.

The detailed descriptions about the baseline methods for comparison in our experiments are introduced as follows:

1) DeepWalk [33]: This is a classical random walk-based network embedding method, where the word2vec model is first introduced to learn node embeddings [52], [53].
2) Chebyshev [39]: The Chebyshev GCN defines graph convolutions using a k-degree polynomial of the Laplacian to approximately estimate the output of Laplacian eigendecomposition.
3) Planetoid [54]: The Planetoid GCN jointly train node representations for classification and graph context prediction, which does not depend on the graph Laplacian regularization.
4) GCN [24]: The vanilla GCN method simplifies the Chebyshev GCN via a localized first-order approximation of spectral graph convolutions, i.e., the renormalization trick.
5) GCN-3: For a fair comparison, we add an additional GCN layer to the vanilla GCN in order to aggregate the farther information.
6) Monet [41]: This is a kind of Laplacian-based approach, which designs a unified generalization of CNN architectures to graph data.
7) GAT [25]: The graph attention network incorporates the attention mechanism into each layer of the GCN framework in order to weight the information from different neighbors.
8) SGC [26]: The simplified GCN simplifies the vanilla GCN by reducing the number of non-linear activations and convolutional layers.

C. EXPERIMENTAL SETTINGS

Our experiments were implemented on a single PC with one NVIDIA RTX 2070 GPU (8Gb of RAM) and one Intel Core i7 CPU (i7-4771 @ 3.50GHz). Specially, all models in our implements were coded by Keras based TensorFlow-GPU version.



(a) The Cora citation network (b) The European air-traffic network

FIGURE 2. An illustration of the Cora citation network and the European air-traffic network with colored labels. Nodes with the same label are shown in the same color. (a) Nodes with the same label in the Cora citation network are inclined to exhibit the community structure, where densely connected nodes are more likely to have the same color. (b) Nodes with the same label in the European air-traffic network are inclined to exhibit similar structure roles, where nodes far apart may have the same color. This figure was produced using Fruchterman-Reingold force-directed algorithm [48].

Concerning the three citation networks, we use the same method to split the data sets as in [24], [25], [54]. For each network, 20 nodes per class were used for training, 500 nodes were used for validation, and another 1000 nodes were used for testing. Differently, for each air-traffic network, a quarter of all nodes were first randomly sampled for training. Then, another quarter was randomly sampled for validation. While the rest were used for testing. For comparison, all results were averaged over 50 experimental runs.

Specifically, the Glorot method [55] was used for model initialization and the cross-entropy loss was minimized using the Adam SGD optimizer [56]. During the training period, the learning rate is fixed at 0.01. Additionally, a maximum of 200 epochs were set for training and an early stopping strategy was implemented with patience of 30 epochs. In other words, the training will be interrupted automatically if the loss does not decrease in several consecutive epochs. Moreover, the dropout [57] and L_2 regularization was applied to all layers. Finally, hyperparameters of all models were tuned with grid search by validation data in following ranges: dropout rate $p \in [0.1, 0.9]$ with an interval of 0.1, the L_2 regularization term coefficient $\lambda \in \{1e-3, 5e-4, 1e-4, 5e-5\}$, the number of first hidden units $h \in \{16, 24, 32, 64\}$, $\alpha \in [0.1, 1.0]$ with an interval of 0.1. For the number of hidden units in the output layer, we set it equal to the number of classes. In Table 2, we summarize the value of all hyperparameters used by our methods.

TABLE 2. The hyperparameter values used by our methods for comparison on different networks.

Parameters	Cora	Citeseer	Pubmed	Brazilian	European
p	0.6	0.6	0.6	0.5	0.5
λ	5e-4	1e-3	5e-4	5e-5	5e-4
h	24	24	24	64	64
α	0.7	0.6	0.8	1.0	1.0

TABLE 3. Classification accuracy of eight baseline methods from literature.

	Dataset		
Models	Cora	Citeseer	Pubmed
MLP	55.1	46.5	71.4
DeepWalk	67.2	43.2	65.3
Planetoid	75.7	64.7	77.2
Chebyshev	81.2	69.8	74.4
MoNet	81.7 ± 0.5	-	78.8 ± 0.4
GCN	81.5	70.3	79.0
GAT	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3
SGC	81.0 ± 0.0	71.9 ± 0.1	78.9 ± 0.0

D. PERFORMANCE COMPARISION

1) CLASSIFICATION ACCURACY

We first evaluate the performance of our methods on three citation networks. Table 3 summaries the results of classification accuracy surveyed from existing studies [25], [26]. It can be observed that the performances of both the Multi-Layer Perception (MLP) and the DeepWalk methods are worse than other GCN methods in the three citation networks. The reason is that the MLP method can only use the node feature information without taking into consideration the contribution of network structure. On the other hand, as one kind of random walk-based network embedding methods, Deepwalk can only use the structural information of a network, which ignores the contribution of node features. The other methods are different variants of GCN methods, which involve both network structure and node features into an end-to-end learning process [24]–[26]. Among them, the GCN, GAT, and SGC methods have better performance than the other three

TABLE 4. The comparison of classification results on the three citation networks.

	Dataset		
Models	Cora	Citeseer	Pubmed
GCN	81.4 ± 0.6	70.8 ± 0.7	79.0 ± 0.6
GCN-3	79.8 ± 1.1	66.1 ± 1.5	78.4 ± 0.8
GAT	82.7 ± 0.7	$\mathbf{72.1 \pm 0.9}$	-
SGC	79.5 ± 0.3	71.9 ± 0.5	75.7 ± 0.6
MCGCN-1	79.1 ± 0.5	69.4 ± 0.3	76.7 ± 0.4
MCGCN-2	81.2 ± 0.6	70.9 ± 0.7	78.5 ± 0.3
MCGCN-3	82.4 ± 0.5	71.2 ± 0.6	78.9 ± 0.5
MCGCN-Katz	$\mathbf{83.1 \pm 0.6}$	$\mathbf{71.9 \pm 0.8}$	$\mathbf{79.4 \pm 0.4}$

TABLE 5. The comparison of classification results on the two air-traffic networks.

	Dataset	
Models	Brazilian network	European network
GCN	30.8 ± 4.0	32.3 ± 4.2
GCN-3	32.4 ± 3.8	34.1 ± 5.9
GAT	36.0 ± 3.8	38.5 ± 2.7
SGC	29.2 ± 2.0	30.9 ± 3.6
MCGCN-1	30.6 ± 3.8	30.1 ± 4.3
MCGCN-2	32.2 ± 4.5	32.1 ± 3.8
MCGCN-3	32.6 ± 4.2	33.7 ± 4.0
MCGCN-Katz	$\mathbf{37.1 \pm 3.2}$	$\mathbf{39.9 \pm 2.6}$

methods. Therefore, in our experiments, we mainly compare the performance of our methods with the three GCNs.

Table 4 summarized classification accuracy with the standard deviation of each method implemented following our experimental settings.[1] It can be observed that our MCGCN-Katz model has the best performance in the Cora network, which can gain 2.1% improvement compared with the GCN method. The MCGCN-3 methods can also achieve higher accuracy than the GCN-3 method. Similarly, in the Pubmed Network, the MCGCN-Katz method can also achieve the best performance. As for the Citeseer network, the performance of our models is slightly worse than the SGC and GAT methods. However, as stated in [26], SGC can reduce to a multi-class logistic regression on the network by removing the non-linear functions and a learnable kernel compared with the vanilla GCN. Consequently, the generalization ability of SGC is relatively weak. While for the GAT method, the computational overhead and memory usage much higher than our methods due to the usage of the attention mechanism [25].

Table 5 shows the classification results on the two air-traffic networks. It can be observed that our MCGCN-Katz method achieves much better performance than all baseline methods. The reason is that for the air-traffic networks, node labels are more relevant to their structural roles on the network, such as transportation hubs or mediators [17], [18]. In this case, two airports with the same label may be far apart from each other in the network. Among all these baseline methods, only the MCGCN-Katz method allows arbitrary order of information aggregation through the network. As a result, nodes with the same label are more likely to share their structural feature.

In summary, evidence has revealed that nodes far apart from each other may have similar functions in complex networks [58]. Therefore, it would be essential for GCNs to enable higher-order information aggregation without increasing the computational overhead. In this paper, the proposed multi-channel graph convolutional architecture allows information aggregation of nodes with arbitrary order of proximity in a network, where other GCN-based GRL algorithms cannot leverage in practice. Moreover, the shared weight mechanism can avoid the overfitting of the proposed MCGCN architecture by reducing the complexity of the model. In doing so, the aggregation of higher-order information together with the shared weight mechanism ensures the efficiency of the proposed MCGCN algorithm in solving the semi-supervised node classification problem on attributed networks.

[1]Codes are available at https://github.com/zhouchunpong/GCN_Keras.

2) COMPUTATIONAL OVERHEAD

We compare the training time of our methods (MCGCN-3 and MCGCN-Katz) with the state-of-the-arts GAT method on the three citation networks. Specifically, all codes run on a GPU implementation in TensorFlow with the same experimental settings and environment (see Section V-C). We measure the running time from the first epoch to the last epoch and repeat it 50 times. The average running time is used for comparison. Figure 3 shows the comparison of training time between GAT and our methods on the three citation networks. It can be observed that both MCGCN-Katz and MCGCN-3 are superior to the GAT method. In the Cora network, the computational overhead of MCGCN-Katz is just 31.5% of that of the GAT method. Moreover, the MCGCN-3 method just takes 26.8% of the GAT's overhead. Similar results can also be found in the other two networks. While for the Pubmed network, the GAT method failed to train the model because it requires more GPU memory. The reasons are threefold: First, the usage of attention mechanisms in

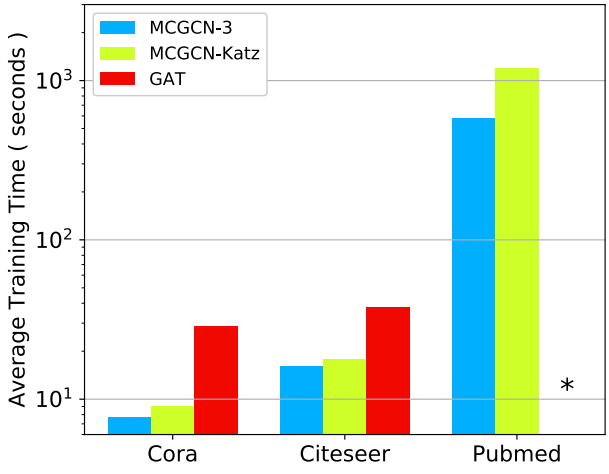


FIGURE 3. The comparison of training time between GAT and our methods on the three citation networks. The asterisk (*) indicates out-of-memory.

the GAT method enlarges the requirements of memory and computing resources. Second, to stabilize the learning process of the attention mechanism, multi-head attention will be independently implemented, which will aggravate the computational burden. Third, the output features of the first layer will be concatenated, which increases the dimension of hidden layers, as well as the number of learnable parameters. In summary, with much less computational overhead, our methods can achieve as good classification accuracy as the GAT method.

E. PARAMETER SENSITIVITY

In this part, we will study how the different settings of hyperparameter α can affect the performance of the semi-supervised nodes classification task. In the proposed architecture, α is a decay parameter determining the relative importance of information channels with higher-order proximity. As α increases, features from higher-order channels will have a relatively high weight. However, to keep the Katz-index converging, the value of α cannot increase infinitely. Figure 4 shows the sensitivity analysis of the MCGCN-Katz method on the Cora and Citesser networks as α increases. It can be observed that the best choice of α is different on different networks. The best performance on the Cora and Citesser networks is achieved when $\alpha = 0.7$ and $\alpha = 0.6$, respectively. In the future, the value of α can also be learned automatically during the training stage.

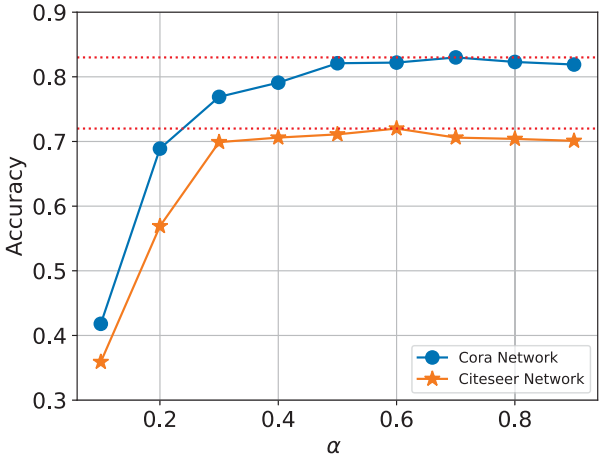


FIGURE 4. The sensitivity analysis of the MCGCN-Katz method on the Cora and Citesser networks as the hyperparameters α increases.

VI. CONCLUSION

In this paper, we have proposed a multi-channel graph convolutional architecture to tackle the semi-supervised node classification problem on attributed networks in an end-to-end manner. Different from existing graph neural networks, the proposed architecture allows higher-order information aggregation through the network by increasing the number of input channels rather than the depth of a GCN (i.e., the number of layers). Further, based on the notion of Katz index, we have also introduced a variation of the architecture that supports arbitrary order of information aggregation without aggravating the computational overhead. Moreover, we have introduced a shared weight mechanism to access the relative importance of different attributes. To evaluate the performance of the proposed GCN architecture, we have carried out experiments on several benchmark networks, including 3 citation networks and 2 air-traffic networks. Through comparing with the state-of-the-art GRL methods, we have shown that our framework can achieve better performance in terms of both node classification and computation efficiency. The proposed multi-channel architecture offers new insight into investigating the function of long-range week connections in complex networks.

In the future, the proposed MCGCN architecture can be extended into following two directions. First, in this paper, we only considered the information aggregation of nodes with higher-order proximity. In the field of complex networks, evidence has shown that higher-order connectivity patterns play essential roles in understanding fundamental functions of complex systems [58]. Therefore, it is worthwhile to construct input channels of the proposed MCGCN based on other higher-order structures, such as motifs, graphlets, Weisfeiler-Lehman isomorphism, and shortest path length. Second, an appropriate attention mechanism can be involved to automatically assess the relative importance of information channels. In doing so, we can quantify which type of higher-order structures is more relevant to the machine learning task on networks. The relationship between higher-order structures and network functions can further help improve the interoperability of the GRL architecture.

REFERENCES

[1] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2224–2232.

[2] G. Durand, N. Belacel, and F. LaPlante, "Graph theory based model for learning path recommendation," *Inf. Sci.*, vol. 251, pp. 10–21, Dec. 2013.

[3] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "KGAT: Knowledge graph attention network for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 950–958.

[4] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[5] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," 2017, *arXiv:1709.05584*. [Online]. Available: http://arxiv.org/abs/1709.05584

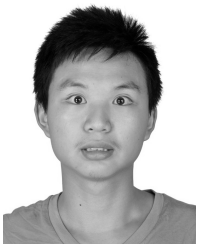
[6] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," 2018, *arXiv:1812.08434*. [Online]. Available: http://arxiv.org/abs/1812.08434

[7] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowl.-Based Syst.*, vol. 151, pp. 78–94, Jul. 2018.

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

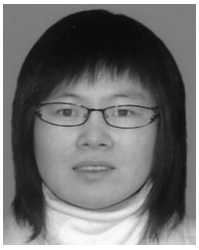
[9] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, Sep. 2018.

[10] Y. Bo, J. Liu, and J. Feng, "On the spectral characterization and scalable mining of network communities," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 2, pp. 326–337, Feb. 2012.
[11] B. Shi, J. Zhong, Q. Bao, H. Qiu, and J. Liu, "EpiRep: Learning node representations through epidemic dynamics on networks," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. (WI)*, 2019, pp. 486–492.
[12] L. Jian, J. Li, and H. Liu, "Toward online node classification on streaming networks," *Data Mining Knowl. Discovery*, vol. 32, no. 1, pp. 231–257, Jan. 2018.
[13] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, 2007.
[14] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1105–1114.
[15] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 203–209.
[16] K. Tu, P. Cui, X. Wang, P. S. Yu, and W. Zhu, "Deep recursive network embedding with regular equivalence," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2357–2366.
[17] B. Shi, C. Zhou, H. Qiu, X. Xu, and J. Liu, "Unifying structural proximity and equivalence for network embedding," *IEEE Access*, vol. 7, pp. 106124–106138, 2019.
[18] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, "Struc2vec: Learning node representations from structural identity," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 385–394.
[19] C. Donnat, M. Zitnik, D. Hallac, and J. Leskovec, "Learning structural node embeddings via diffusion wavelets," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1320–1329.
[20] F. Scarselli, M. Gori, A. Chung Tsoi, M. Hagenbuchner, and G. Monfardini, "Computational capabilities of graph neural networks," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 81–102, Jan. 2009.
[21] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," 2018, *arXiv:1812.04202*. [Online]. Available: http://arxiv.org/abs/1812.04202
[22] F. Scarselli, M. Gori, A. Chung Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.
[23] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*. [Online]. Available: http://arxiv.org/abs/1312.6203
[24] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907
[25] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*. [Online]. Available: http://arxiv.org/abs/1710.10903
[26] F. Wu, T. Zhang, A. H. de Souza, Jr., C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," 2019, *arXiv:1902.07153*. [Online]. Available: http://arxiv.org/abs/1902.07153
[27] H. Dai, Z. Kozareva, B. Dai, A. Smola, and L. Song, "Learning steady-states of iterative algorithms over graphs," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 1106–1114.
[28] K. Henderson, B. Gallagher, T. Eliassi-Rad, H. Tong, S. Basu, L. Akoglu, D. Koutra, C. Faloutsos, and L. Li, "RolX: Structural role extraction & mining in large graphs," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 1231–1239.
[29] S. Gilpin, T. Eliassi-Rad, and I. Davidson, "Guided learning for role discovery (GLRD): Framework, algorithms, and applications," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining KDD*, 2013, pp. 113–121.
[30] R. A. Rossi and N. K. Ahmed, "Role discovery in networks," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 4, pp. 1112–1131, Apr. 2015.
[31] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed large-scale natural graph factorization," in *Proc. 22nd Int. Conf. World Wide Web (WWW)*, 2013, pp. 37–48.
[32] S. Cao, W. Lu, and Q. Xu, "GraRep: Learning graph representations with global structural information," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2015, pp. 891–900.
[33] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2014, pp. 701–710.
[34] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 855–864.
[35] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1067–1077.
[36] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 912–919.
[37] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, pp. 321–328.
[38] J. Liang, P. Jacobs, J. Sun, and S. Parthasarathy, "Semi-supervised embedding in attributed networks with outliers," in *Proc. SIAM Int. Conf. Data Mining*, 2018, pp. 153–161.
[39] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.
[40] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.
[41] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5115–5124.
[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[43] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" 2018, *arXiv:1810.00826*. [Online]. Available: http://arxiv.org/abs/1810.00826
[44] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.
[45] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," 2015, *arXiv:1511.07289*. [Online]. Available: http://arxiv.org/abs/1511.07289
[46] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, p. 93, Sep. 2008.
[47] G. Namata, B. London, L. Getoor, B. Huang, and U. Edu, "Query-driven active surveying for collective classification," in *Proc. 10th Int. Workshop Mining Learn. Graphs*, vol. 8, 2012, pp. 1–8.
[48] T. M. J. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement," *Softw., Pract. Exper.*, vol. 21, no. 11, pp. 1129–1164, Nov. 1991.
[49] L. Tang and H. Liu, *Leveraging Social Media Networks for Classification*. Norwell, MA, USA: Kluwer, 2011.
[50] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Nov. 2006.
[51] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, "Deep learning via semi-supervised embedding," in *Neural Networks: Tricks of the Trade*. Cham, Switzerland: Springer, 2012, pp. 639–655.
[52] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 2013, pp. 3111–3119.
[53] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: http://arxiv.org/abs/1301.3781
[54] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," 2016, *arXiv:1603.08861*. [Online]. Available: http://arxiv.org/abs/1603.08861
[55] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[57] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[58] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Science*, vol. 353, no. 6295, pp. 163–166, Jul. 2016.



CHUNPENG ZHOU received the B.Eng. degree in information security from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2017. He is currently pursuing the master's degree with the School of Cyberspace, Hangzhou Dianzi University, China. His research interests include complex networks, data mining, machine learning, and network representation/embedding.

BENYUN SHI received the B.Sc. degree in mathematics from Hohai University, Nanjing, China, in 2003, and the M.Phil. and Ph.D. degrees in computer science from Hong Kong Baptist University, Hong Kong, in 2008 and 2012, respectively. He worked as a Full Professor with Hangzhou Dianzi University, China, and a Research Assistant Professor with Hong Kong Baptist University. He is currently a Professor with the School of Computer Science and Technology, Nanjing Tech University, China. His research interests are in the areas of data-driven modeling and analytics, machine learning, complex systems/networks, multi-agent autonomy-oriented computing, computational epidemiology, and health informatics.



HONGJUN QIU received the B.Sc. degree in computer science and technology from Beijing Forestry University, Beijing, China, in 2003, and the Ph.D. degree in computer application from the Beijing University of Technology, Beijing, in 2010. She is currently a Lecturer with the School of Cyberspace, Hangzhou Dianzi University, China. Her research interests include complex systems/networks, autonomy-oriented computing, and health informatics.

JIMING LIU received the B.Sc. degree in physics from East China Normal University, the M.A. degree from Concordia University, and the M.Eng. and Ph.D. degrees in electrical engineering (robotics) from McGill University (with Centre for Intelligent Machines and funded by Hydro-Quebec). He is currently the Chair Professor of computer science and an Associate Vice-President (Research) with Hong Kong Baptist University (HKBU), where he directs the Centre for Health Informatics and the Joint Research Laboratory for Intelligent Disease Surveillance and Control, a Partnership between the Computer Science Department, and the National Institute of Parasitic Diseases (NIPD) in Chinese Center for Diseases Control and Prevention (China CDC). His research interests are in the areas of data-driven modeling, machine learning, Web intelligence, complex networks, multi-agent autonomy-oriented computing, healthcare informatics, and computational epidemiology. He was named IEEE Fellow, in 2011, for original contributions to Multi-Agent Autonomy-Oriented Computing (AOC) and Web Intelligence (WI).