Star Topology Convolution for Graph Representation Learning

Star Topology Convolution for Graph Representation Learning

Chong Wu[1], *Student Member, IEEE,* Zhenan Feng[1], Jiangbin Zheng[2], Houwang Zhang[2], *Student Member, IEEE,* Jiawang Cao[2], Hong Yan, *Fellow, IEEE*

Abstract—We present a novel graph convolutional method called star topology convolution (STC). This method makes graph convolution more similar to conventional convolutional neural networks (CNNs) in Euclidean feature space. Unlike most existing spectral convolutional methods, this method learns subgraphs which have a star topology rather than a fixed graph. It has fewer parameters in its convolutional filter and is inductive so that it is more flexible and can be applied to large and evolving graphs. As for CNNs in Euclidean feature space, the convolutional filter is localized and maintains a good weight sharing property. By introducing deep layers, the method can learn global features like a CNN. To validate the method, STC was compared to state-of-the-art spectral convolutional and spatial convolutional methods in a supervised learning setting on three benchmark datasets: Cora, Citeseer and Pubmed. The experimental results show that STC outperforms the other methods. STC was also applied to protein identification tasks and outperformed traditional and advanced protein identification methods.

Index Terms—Convolutional Neural Network, Graph Representation Learning, Inductive Spectral Convolution, Protein Identification, Star Topology.

1 Introduction

Convolutional neural networks (CNNs) have been successfully used to solve problems which have a Euclidean feature space [1], such as image classification [2], and machine translation [3]. However most problems, such as 3D meshes, social networks, telecommunication networks, biological networks or brain connectomes, have a non-Euclidean nature [4], which creates a challenge when introducing CNNs to solve these problems [1]. Data in the form of a graph is a typical non-Euclidean problem. There are three major problems in generalizing CNNs to graphs: (1) the numbers of directly connected neighbors for different nodes usually differ [5]; (2) the feature dimensions for different nodes may also differ; (3) the edges may have features and their feature dimensions may differ.

To solve these problems, two broad categories of methods have been proposed: (1) spatial convolutional methods; (2) spectral convolutional methods.

Spatial convolutional methods define the convolution on the spatial domain, such as the nodes and edges to provide a flexible framework [1]. For each node, the convolution is an aggregation over all the nodes located in its neighborhood [6]. The main challenge of spatial methods is how to define a convolutional kernel/aggregator which can handle neighborhoods with different sizes while at the same time maintaining the weight sharing property of a CNN [1]. Finding appropriate neighborhoods is also elusive [1] and spatial methods don't have a strict mathematical explanation.

Graph sample and aggregate (GraphSage) [6], a node-based spatial convolutional method, learns node embeddings rather than graph embeddings so that it can be applied to large graphs or evolving graphs. Graph attention network (GAT) adopts a self-attention mechanism to learn the weighting function [7]. Dual-primal graph convolutional network (DPGCN) [8] generalizes GAT by using convolutions on nodes and edges to achieve a better performance compared to GAT. Mixture model CNN (MoNet) uses a weighted average of multiple weighting functions defined over the neighborhood of a node as the spatial convolution to provide a general framework for designing spatial convolutional methods [9]. Graphs with generative adversarial nets (GraphSGAN) [10] facilitates generalization of generative adversarial nets (GANs) to graph through low-density areas by generating fake samples between subgraphs to improve the performance of semi-supervised learning on graphs.

Spectral convolutional methods define the convolution based on the graph Fourier transform [11] to transfer signals from the spatial domain to the spectral domain and do convolution on the spectral domain which can be explained mathematically and maintains the weight sharing property of CNNs. Some spectral methods have been successfully applied to the context of node classification. Spectral convolutional neural network (Spectral CNN) [12] introduces the graph Fourier transform directly and proposes spectral convolution of graph signals. It can use non-spatially localized filters, but the number of parameters of the filter

- *Chong Wu and Hong Yan are with the Department of Electrical Engineering, City University of Hong Kong, Kowloon, Hong Kong. E-mail: chongwu2-c@my.cityu.edu.hk & h.yan@cityu.edu.hk*
- *Zhenan Feng, Houwang Zhang, and Jiawang Cao are with School of Automation, China University of Geosciences, Wuhan 430074, China E-mail: fengzhenan@cug.edu.cn & zhanghw@cug.edu.cn & CJW@cug.edu.cn*
- *Jiangbin Zheng is with the School of Informatics, Xiamen University, Xiamen, 361005, China. E-mail: jiangbinzheng@stu.xmu.edu.cn*
- *[1]These authors contributed equally to the theoretical and experimental parts of this paper. [2]These authors contributed equally to the running of experiments in this paper.*

that have to be learned is large, potentially causing a severe computational load [4]. Chebyshev network (ChebyNet) [13] restricts the kernel of Spectral CNN to a polynomial expansion. Graph convolutional network (GCN) [14] is a simplification of ChebyNet to make spectral methods spectrum-free, reduce the computational cost of the eigen decomposition of the graph Laplacian matrix and is able to get spatially localized filters [1]. Graph wavelet neural network (GWNN) introduces the graph wavelet transform to replace the Fourier basis of Spectral CNN with graph wavelet basis and is able to achieve spatially and spectrally localized filters while maintaining a good computational performance.

These spectral convolutional methods are all transductive and learn the graph embedding using a fixed graph. In real applications, situations which require graph convolutional methods to be able to generate embeddings for unseen nodes or even entirely new graphs or subgraphs will be encountered [6]. Transductive methods have difficulties in facilitating generalization across graphs with the same form of features but different numbers of nodes [6]. Spectral methods have a high memory requirement, which makes application to large graphs difficult.

To solve the problems of spectral convolutional methods, inductive node-based spectral convolutional methods with a good computational performance and low memory requirements are necessary. All graphs are composed of subgraphs with a star topology, as Figures 1 & 2 show. A star topology has several advantages in its eigen decomposition, as its eigenvalues are all equal except for the first and last elements. This means that their Laplacian matrices, even of different sizes, will have some common eigenvectors, although they may be spanned to a certain degree. These properties allow the design of a flexible convolutional filter which can maintain weight sharing for subgraphs of different sizes and structures. With a flexible filter the spectral convolution over a subgraph with a star topology can be defined. Then, a star topology convolution (STC) can be introduced for graph representation learning to perform the spectral convolution on these subgraphs to obtain the spectral node embeddings. These can be aggregated to the central nodes of these subgraphs. Hence, this method can achieve localized filters in both the spectral and spatial domains. The computational complexity and memory requirements are largely reduced compared to conventional spectral methods. As this method is inductive, it can be applied to large or evolving graphs. The convolutional process of this method is very similar to that in the CNNs commonly used in image science. The contributions of this paper can be summarized as follows,

1. This is the first attempt to use star topology subgraph in graph representation learning.
2. This is an inductive node-based spectral convolutional method which has good flexibility and can be applied to large or evolving graphs.
3. Based on the good properties of a star topology, this method has a good computational performance with high memory efficiency.
4. This is a graph convolutional method that is more similar to conventional CNNs than most existing spectral convolutional methods.
5. The method provides a framework for the design of inductive spectral convolutional methods.

The rest of this paper is organized as follows. In Section 2, the background of spectral convolution for graphs is introduced and inductive spectral convolution is defined. The properties of graphs with a star topology are presented and star topology convolution with a node dropout strategy to improve the robustness of the method is introduced. The complexity of this method is discussed. In Section 3, the baselines, experimental settings, benchmarks, and experimental results are presented. In Section 4, the conclusions from the study are given.

2 METHODS

2.1 Preliminary

Given an undirected graph $G = \{\mathbb{V}, \mathbb{E}, \boldsymbol{A}\}$, where $\mathbb{V} = (V_1, V_2, \ldots, V_n)$ is a node set ($|\mathbb{V}| = n$), $\mathbb{E}$ is an edge set, and $\boldsymbol{A}$ is an adjacency matrix ($\boldsymbol{A}_{[i,j]} = \boldsymbol{A}_{[j,i]}$), its graph Laplacian matrix $\boldsymbol{L}$ can be defined as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$, where $\boldsymbol{D} = diag(\sum_{i \neq j} \boldsymbol{A}_{[i,j]})$ is a degree matrix, and $\boldsymbol{L}$ is a symmetric positive-semidefinite matrix. L has an eigen decomposition as follows,

$$\boldsymbol{L} = \boldsymbol{U}\Lambda\boldsymbol{U}^T, \tag{1}$$

where, $\boldsymbol{U} = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n)$ are the eigenvectors which are orthonormal and $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$ is the diagonal matrix of the corresponding eigenvalues which are real and non-negative and can be interpreted as the frequencies of the graph. These eigenvectors compose the basis of feature space in the spectral domain.

2.2 Spectral convolution

For a signal $\boldsymbol{F} = (\boldsymbol{F}_1, \boldsymbol{F}_2, \ldots, \boldsymbol{F}_n)$ on the nodes of graph G, its graph Fourier transform is defined as follows,

$$\hat{\boldsymbol{F}} = \boldsymbol{U}^T\boldsymbol{F}, \tag{2}$$

Given another signal $\boldsymbol{g}$, the convolution of $\boldsymbol{g}$ and $\boldsymbol{F}$ can be defined as follows,

$$\boldsymbol{F} \star \boldsymbol{g} = \boldsymbol{U}\boldsymbol{g}_\theta\boldsymbol{U}^T\boldsymbol{F}, \tag{3}$$

where, $\boldsymbol{g}_\theta$ is the Fourier transform of $\boldsymbol{g}$. Equation (3) is the spectral convolution which is similar to the convolution theorem defined in Euclidean feature space and signal $\boldsymbol{g}_\theta$ can be regarded as the convolution filter which provides a set of kernel functions.

2.3 Inductive spectral convolution

To apply spectral convolution to large or evolving graphs, we need to introduce an inductive version which can learn the local and global structural properties of each node. Inductive methods focus on obtaining inductive node embedding, rather than whole graph embedding, which provides good flexibility. Hence, the inductive spectral convolution can be defined as a node-based subgraph spectral convolution as follows,

$$\boldsymbol{F}_{V_i} \star \boldsymbol{g} = \sum_{\{V_i, V_j\} \in \mathbb{E}} \boldsymbol{W}_{V_j} \hat{\boldsymbol{F}}_{[V_j,:]}, \tag{4}$$

where, $\boldsymbol{W}$ is a flexible filter for subgraphs with different topologies. The key to this work is to find a universal $\boldsymbol{W}$ to make Equation (4) hold.

2.4 Properties of a star topology

$\boldsymbol{W}$ is related to the topology of the subgraphs. A good $\boldsymbol{W}$ should maintain the weight sharing property, at the same time, provide different kernels for different structures [15]. We need to find a common structure, lying in different subgraphs, which should be identical or, at least, symmetric to help us design filters as in a conventional CNN. We find that star topology graphs with different sizes n have an elegant symmetry in their structure and Laplacian matrix, as Figures 1 & 2 show. We also find that all graphs can be regarded as a composition of subgraphs with a star topology as Figure 2 shows. Then the eigen decomposition of an n-dim star topology (n neighboring nodes and one central node, $n \geq 1$) can use a universal formulation as follows,

$$\begin{aligned} \boldsymbol{L} &= \begin{bmatrix} n & -1 & \cdots & -1 \\ -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & 0 & \cdots & 1 \end{bmatrix} = \boldsymbol{U}\Lambda\boldsymbol{U}^T, \\ \boldsymbol{U} &= (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_{n+1}),\ \Lambda = \begin{bmatrix} n+1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \end{aligned} \tag{5}$$

where, all elements on the diagonal of Λ are 1 except the first and last elements, which means that all eigen vectors have the same weight, except the first and last ones. The largest eigenvalue corresponds to the central node and its connections. All of the value 1 eigenvalues correspond to neighboring nodes and their connections. Since the rank of $\boldsymbol{L} \in \mathbb{R}^{(n+1)*(n+1)}$ is n, the last eigenvalue is 0. The eigenvalues in Equation (5) can be obtained as follows.

By modifying Equation (5), we can have,

$$\boldsymbol{L}\boldsymbol{U} = \Lambda\boldsymbol{U}, \tag{6}$$

$$\boldsymbol{L}\boldsymbol{U} = \Lambda\mathbf{I}\boldsymbol{U}, \tag{7}$$

$$(\Lambda\mathbf{I} - \boldsymbol{L})\boldsymbol{U} = 0, \tag{8}$$

where, $\mathbf{I}$ is an identical matrix. According to the theorem of linear system of equations, Equation (8) having non-zero solutions is equivalent to $det(\Lambda\mathbf{I} - \boldsymbol{L}) = 0$. Using Gaussian elimination on $det(\Lambda\mathbf{I} - \boldsymbol{L}) = 0$, we can have,

$$(\lambda - n - 1)(\lambda - 1)^{n-1}\lambda = 0, \tag{9}$$

the solutions of Equation (9) are Λ.

Eigenvectors $\boldsymbol{U} = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_{n+1})$ are orthonormal. We can find that different size star topology graphs have common parts in their eigenvectors for which the corresponding eigenvalue is 1. For an m-dim star topology graph G_m and an n-dim star topology graph G_n, where $m \leq n$, the eigenvectors corresponding to eigenvalues of 1 in G_m can be expressed by any $m-1$ eigenvectors with corresponding eigenvalues of 1 in G_n using the vector rotation formula as follows,

$$\boldsymbol{U}_{G_m[2:m]} = \boldsymbol{R}\boldsymbol{U}_{G_n[\boldsymbol{p}]},\ \boldsymbol{p} = randperm([2:n], m-1), \tag{10}$$

where, $\boldsymbol{R}$ is an identical rotation matrix. Hence, these eigenvectors can be regarded as equivalent and are suitable to be the universal basis for different scales of subgraphs.

Referring to Equation (3), we can find that the first row of $\boldsymbol{g}_\theta\boldsymbol{U}^T$ corresponds to the first row of the new feature matrix $\boldsymbol{g}_\theta\boldsymbol{U}^T\boldsymbol{F}$ and also to the first row of $\boldsymbol{F}$. The first row of $\boldsymbol{F}$ corresponds to the central node as $\boldsymbol{L}$ shows. Since we want to obtain the neighboring information of the central node, the first row of $\boldsymbol{F}$ can be padded to zero. The last eigenvalue is zero which means that the last eigenvector is the least important. We can manually add a artificial node to the original subgraph, then we get a new $\boldsymbol{L} \in \mathbb{R}^{(n+2)*(n+2)}$. We can pad a zero row under the last row of $\boldsymbol{F}$ as zero rows in $\boldsymbol{F}$ won't affect the non-zero rows so that any number of zero rows in $\boldsymbol{F}$ can be added if needed. Then, we can design the flexible filter for $\{1, 2, \ldots, k\}$-dim (k can be any non-zero positive integer) star topology subgraphs as follows,

$$\boldsymbol{W}(k) = \boldsymbol{U}_{[2:k+1,:]}\Theta, \tag{11}$$

$$\Theta = \begin{bmatrix} \theta_1 & 0 & \cdots & 0 \\ 0 & \theta_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \theta_k \end{bmatrix}, \tag{12}$$

where, Θ is a diagonal matrix which has k learnable parameters. The filter can provide different kernels on different star topology subgraphs and its weight sharing property is shown in Figure 3.

2.5 Star topology convolution

Using the properties of a star topology subgraph, we can get the spectral convolution defined on that star topology. The update function for the $(l+1)^{th}$ layer's neighboring information can be defined as follows,

$$\begin{aligned} h^{l+1}_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},z]} &= \sigma\left(\sum_{x=1}^{p} \boldsymbol{W}^l_{x,z}\boldsymbol{U}^T y^l_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},x]}\right), \\ &(z = 1, 2, \ldots, q),\ \{V_i, V_j\} \in \mathbb{E}, \end{aligned} \tag{13}$$

where, $y^l_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},x]}$ is the output of the l^{th} layer, σ denotes a nonlinear activation function, V_i is the central node, and N_{V_i} is the size of the neighborhood of V_i. We use the detaching method [1] to reduce the computational complexity of $h^{l+1}_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},x]}$. Differently from GWNN, we first perform spectral convolution as follows,

$$\begin{aligned} h^{l+1}_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},x]} &= \sigma\left(\boldsymbol{W}^l\boldsymbol{U}^T y^l_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},x]}\right), \\ &(x = 1, 2, \ldots, p). \end{aligned} \tag{14}$$

The information on the neighbors obtained from the spectral convolution will be aggregated to the central node using feature transformation as follows,

$$y^{l+1}_{V_i} = \sigma\left(cat\left(y^l_{[V_i,:]}, mean\left(h^{l+1}_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},:]}\right)\right)\boldsymbol{S}\right), \tag{15}$$

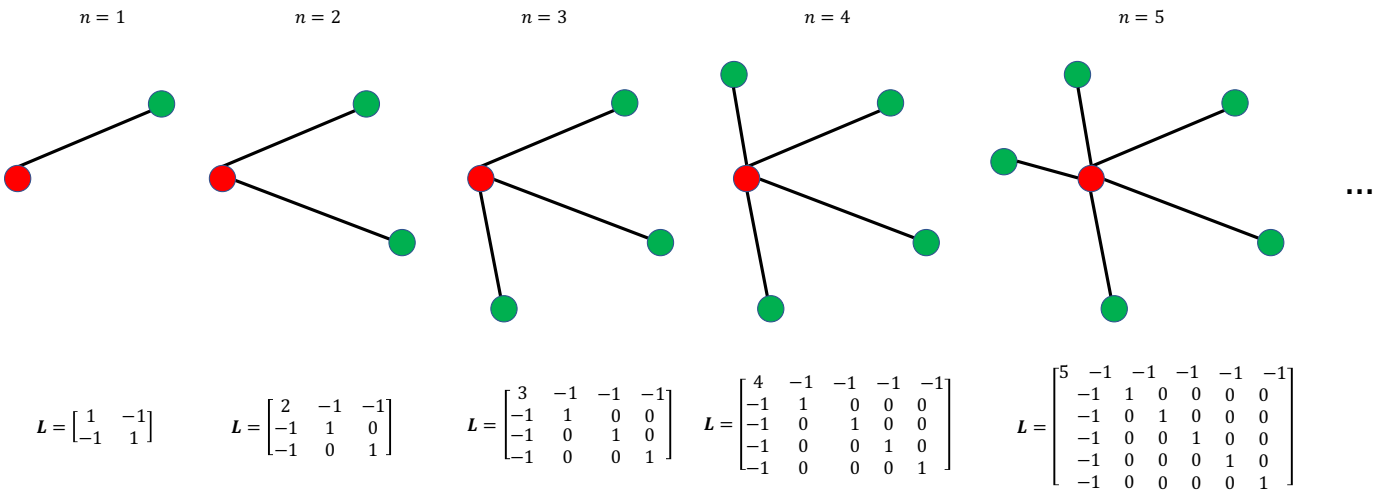

Fig. 1. The structural similarity of graphs with a star topology. The red node is the central node.

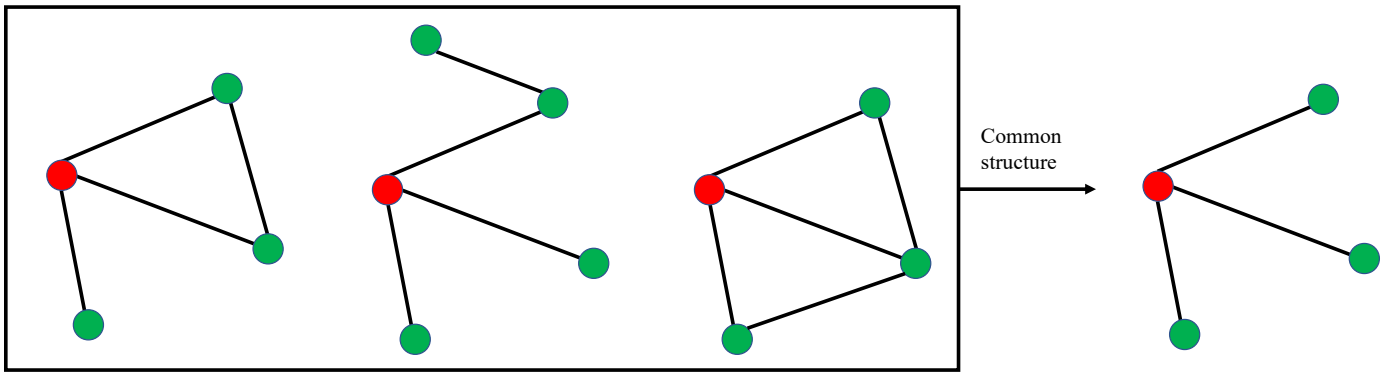

Fig. 2. Subgraphs with a star topology exist in different graphs. The red node is the central node.

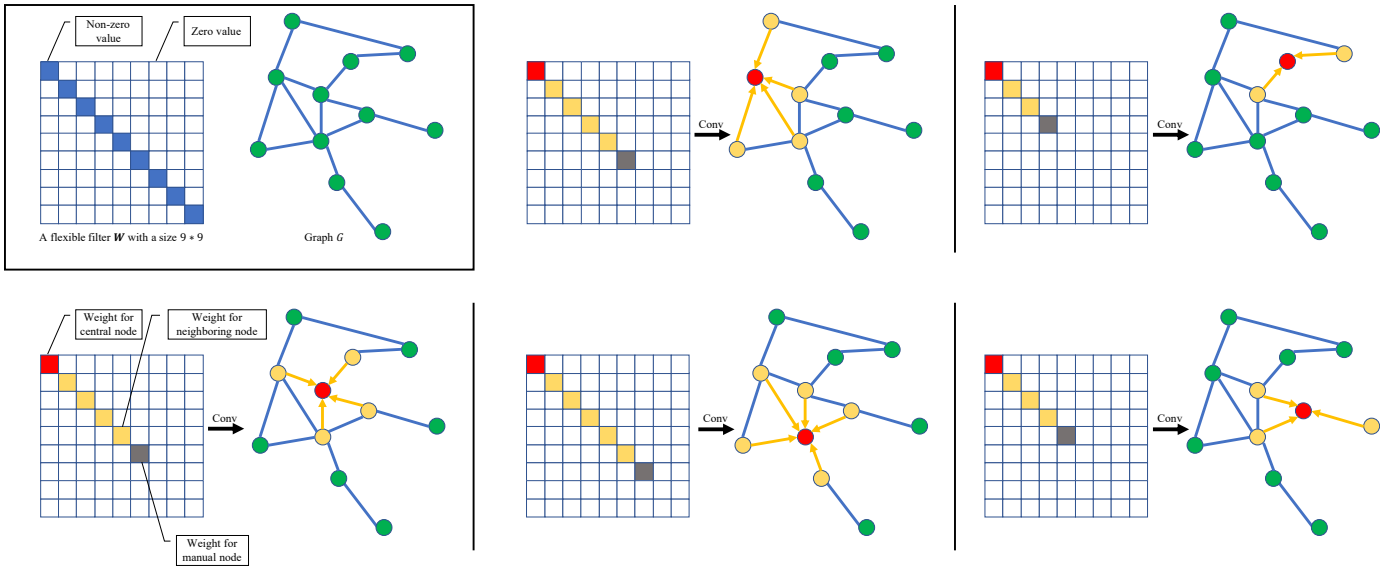

Fig. 3. A toy example showing the weight sharing property of the filter on subgraphs with different star topologies. The kernels provided by the filter differ by subgraph. The red node is the central node and the yellow nodes are neighboring nodes to be aggregated.

where, $y^l_{[V_i,:]}$ is the feature vector of the output of l^{th} layer of the central node V_i, $cat()$ denotes a concatenation function, and $\boldsymbol{S} \in \mathbb{R}^{2p*q}$ is a scaling parameter matrix for feature transformation. For the minibatch training setting, the batch normalization of h of all nodes in the minibatch is done first. Then the Equation (15) obtains the output y.

Figure 4 shows the comparison of a two-layer conventional CNN and a two-layer STC. The formulation of the two-layer STC is as follows,

$$\begin{aligned} &1^{st}\ layer: y^1_{V_i} = \\ &\quad \textbf{ReLU}\left(cat\left(\boldsymbol{F}_{[V_i,:]}, mean\left(h^1_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},:]}\right)\right)\boldsymbol{S}_1\right), \\ &2^{nd}\ layer: y^2_{V_i} = \\ &\quad \textbf{ReLU}\left(cat\left(y^1_{[V_i,:]}, mean\left(h^2_{[\{V_j\}\in\mathbb{R}^{N_{V_i}},:]}\right)\right)\boldsymbol{S}_2\right), \end{aligned} \tag{16}$$

where, $\boldsymbol{S}_1 \in \mathbb{R}^{2p*q}$ and $\boldsymbol{S}_2 \in \mathbb{R}^{2q*d}$ are two scaling parameter matrices for feature transformation, p, q, and d are the dimensions of the embeddings. STC divides the convolution process of a conventional CNN into two cascading steps: (1) spatial search; (2) spectral convolution. Spatial search is used to find spatially localized areas. Hence, our method can directly achieve spatially localized filters. Like the conventional CNN, our method can also aggregate global information through the introduction of deep layers. After the spatial search, our method conducts a spectral convolution on the subgraphs obtained. In a conventional CNN, the shape of different areas selected by a window function/kernel is usually the same. Take a 3*3 kernel for example, the different areas selected are all square and have the same size (9 pixels). In the graph, star topology subgraphs also have symmetry in their spatial structures which allows STC to do a similar convolution process as in a conventional CNN. If a new filter is used to replace the $\boldsymbol{W}$ here, a new inductive spectral convolutional method can be designed. Hence, our method provides a framework to design inductive spectral convolutional methods.

2.6 Node dropout

Since the size of the filter is fixed, a simple way to confirm the filter size is to just assign it as $N + 2$, where N is the maximum number of connections of a node in the graph. Inspired by the dropout strategy [16], we propose a node based dropout strategy, called node dropout, to improve the robustness of our method. For a node V_i which has N_{V_i} directly connected neighbors in its neighboring information aggregation process, when $N_{V_i} > k$, only k neighbors will be randomly selected in each training iteration as follows,

$$\begin{aligned} &y^l_{[:,x]} = y^l_{[\boldsymbol{p},x]}, \\ &\quad \boldsymbol{p} = randperm(\{V_j\} \in \mathbb{R}^{N_{V_i}}, k),\ \{V_i, V_j\} \in \mathbb{E}, \end{aligned} \tag{17}$$

where, $\boldsymbol{p}$ is the mask, the size of which is determined by the size of the filter. For nodes with $N_{V_i} \leq k$, all neighboring information will be selected, and these nodes can be considered as nodes that do not have enough information. Hence, it is necessary to take advantage of all their information. Through controlling the size of the filter, our method can achieve a good robustness. The effect of the size of filter and how to select it will be discussed in the ablation study in subsection 3.3.

2.7 Complexity analysis

Since we use the detaching method proposed by [1] to further reduce the complexity, the computational complexity of STC depends on the filter size and the size of scaling matrix. The computational complexity of the $(l+1)^{th}$ layer is $p * q$ + k, where $p * q$ is the size of scaling matrix and k is the size of the filter. k is much smaller than $p * q$, and, in our experiment, k is set to ≤ 30 and $p * q \geq 9 * 9$. Compared to other spectral convolutional methods like Spectral CNN [12] (with a complexity of $p * q * N$, where N is the number of nodes) and GWNN [1] (with a complexity of $p * q + N$), the complexity of our method is much smaller and is able to be used in large or evolving graphs. For the memory requirement, our method is an inductive node-based convolutional method and it learns node embeddings instead of a whole graph embedding, hence, it doesn't need to store information on the whole graph in each layer and just needs to allocate the memory for the filter and the information on the central node and its neighbors.

3 Experiments and Results

3.1 Experiment settings

To validate our method, we compared STC with several state-of-the-art spectral convolutional methods: Spectral CNN [12], GCN [14], Spectral CNN detaching and GWNN [1], and spatial convolutional methods: GraphSage [6] and MoNet [9] in a supervised learning setting on three benchmark datasets: Cora [17], Citeseer [17], and Pubmed [17]. All methods used their default settings in all experiments in this paper. The depth of all methods is kept the same: a two-layer encoder with a fully connected layer as the decoder is used for all experiments in this paper. The embedding dimensions of the two layers of the encoder for all methods are kept the same as shown in Table 1. The optimizer for STC is stochastic gradient descent (SGD) and the initial learning rate is set to 0.7 for all experiments in this paper. The number of epochs of all methods is kept the same: 200.

We introduce STC in an essential protein identification task and compare STC with several state-of-the-art essential protein identification methods: deep neural network based method (DNN) [18], support vector machine (SVM) [18], Adaboost [18], decision tree [18], random forest [18], multiobjective optimization based method (IMAMOBH) [19], degree centrality (DC) [20], betweeness centrality (BC) [21], closeness centrality [22], eigenvector centrality [23], edge clustering coefficient centrality (NC) [24], and local average connectivity-based method (LAC) [25]. Essential protein identification can be seen as a binary node classification task [19]. The PPI network dataset of Saccharomyces cerevisiae (Yeast) was downloaded from the DIP database [26] updated to Oct.10, 2010. There are 4541 proteins and 22091 interactions, without self and repeated interactions, in this dataset. We selected Yeast because its PPI data and gene essentiality data are the most complete and reliable among various species. The essential protein dataset is selected from MIPS [27], SGD [28], DEG [29], and SGDP [30], [31]. There are 1285 essential proteins in this dataset, 1036 of which are in the PPI network. We take these 1036 proteins as essential proteins while the other 3505 proteins are

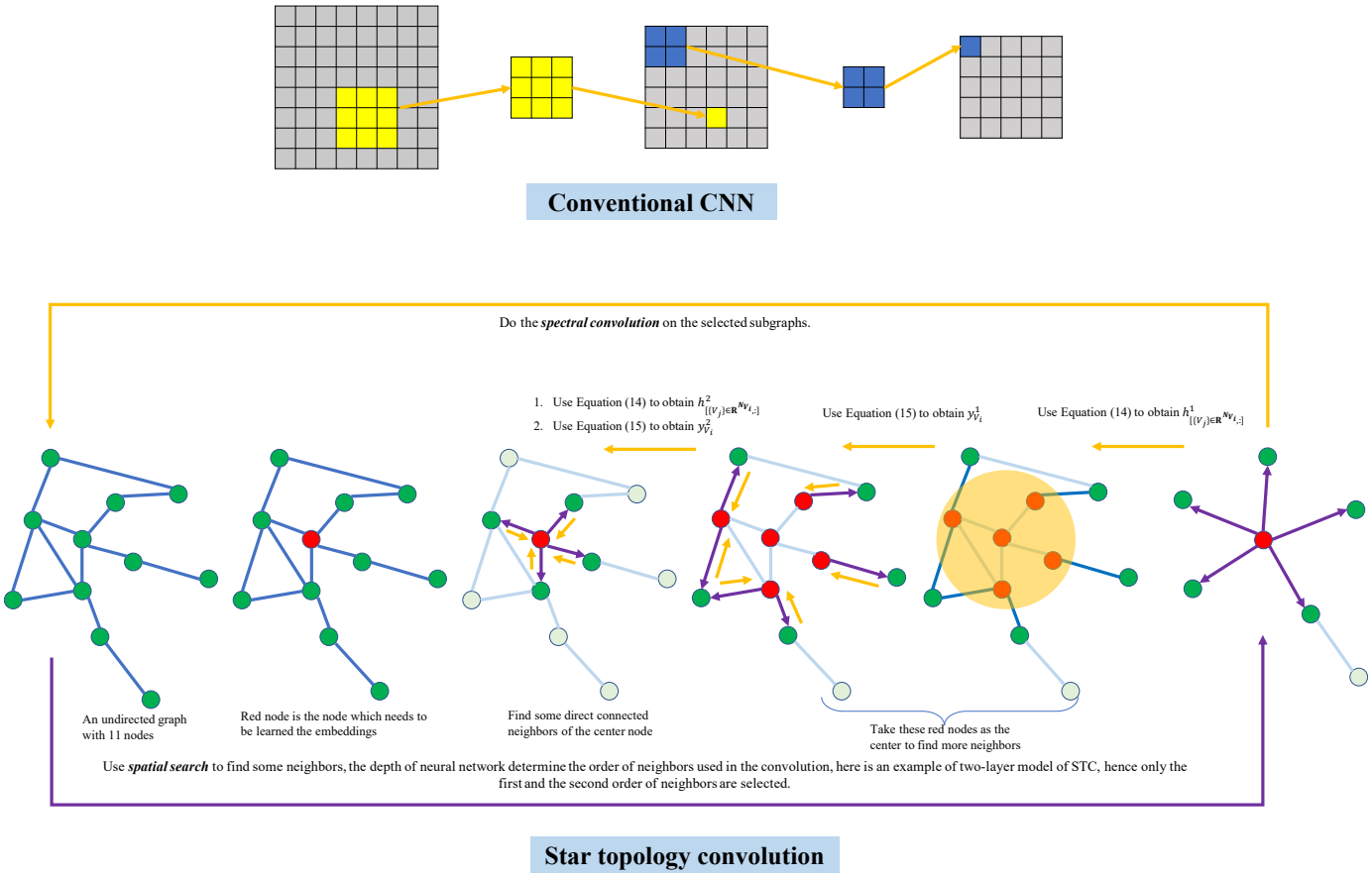

Fig. 4. Comparison of a conventional CNN and a star topology convolution (STC).

considered as non-essential ones. The gene expression data used as the feature matrix was from GEO [32], and sampled 36 time points during successive Yeast metabolic cycles. Table 1 shows the statistics of four datasets. The minibatch size for the four datasets used by STC: 100 (Cora), 100 (Citeseer), 512 (Pubmed), and 200 (Protein). The division of training/validation/test set is according to [33]. The Potein dataset including PPI network, labels, and features used in this paper is available in supplemental material.

Our method has a hyper-parameter: the size of the filter. In the comparison with baseline methods, we set the size of the filter of the first layer as: 12 for Cora, 9 for Citeseer, 12 for Pubmed, and 30 for Protein. And the size of the filter of the second layer is set to: 8 for Cora, 9 for Citeseer, 27 for Pubmed, and 30 for Protein. To validate the effect of this hyper-parameter on the performance of STC, we introduced an ablation study.

All experiments except the ablation study were conducted on a personal computer with Ubuntu OS 16.04.5 LTS, Intel(R) Xeon(R) E5-2603 v4 1.70GHz 12-Core CPU, 64 GB RAM, CUDA version 10.0.130, torch version 1.5.1, and 4 NVIDIA GeForce RTX 2080Ti GPUs. The ablation study was run on a personal computer with Windows OS 10 Home, AMD Ryzen 5 3600 3.59 Ghz 6-Core CPU, 16 GB RAM, CUDA version 10.2, torch version 1.5.0, and one NVIDIA GeForce GTX 1660 Super GPU.

3.2 Performance analysis

Table 2 shows the accuracy obtained by all graph convolutional methods on the three most commonly used benchmark datasets. Our method achieves the highest accuracy on Cora and Pubmed. For Citeseer, the performance of our method is just inferior to GCN. Table 3 shows the comparison of our method with some state-of-the-art essential protein identification methods. Our method significantly outperforms these methods, particularly the advanced essential protein identification methods like DNN, LAC, and IMAMOBH. GraphSage has an accuracy only slightly lower than our method in this dataset.

3.3 Ablation study of hyper-parameter

To validate the effect of the size of the filter on the performance of STC, we set different sizes of the filter in the first layer and kept the size of the filter in the second layer to that used in the prediction model and tested on the Cora dataset. Figure 5 shows the accuracy obtained by our prediction model with different filter sizes in its first layer. Accuracy improved with increasing filter size to a peak value and then reduced. When the filter size is too small, there is insufficient information to train a good model. When the filter size is too large, redundant information will be introduced to the trained model, causing overfitting and leading to a degradation in performance by the model. Figure 6 shows the average batch time obtained by our prediction model

TABLE 1
The statistics of four benchmark datasets.

	Citeseer	Cora	Pubmed	Protein
Nodes	3327	2708	19717	4541
Edges	4732	5429	44338	22091
Classes	6	7	3	2
Features	3703	1433	500	36
Embeddings (Output of the input layer)	128	128	128	9
Embeddings (Output of the hidden layer)	128	128	128	9
Traing set	1827	1208	18217	3041
Test set	1000	1000	1000	1000
Validation set	500	500	500	500

TABLE 2
Validation results of all methods on three citation datasets.

Methods	Citeseer	Cora	Pubmed
Ours	0.7280	**0.8760**	**0.8840**
GraphSage	0.7020	0.8500	0.8140
MoNet	0.6910	0.7930	0.8220
Spectral CNN	0.6840	0.8180	0.7920
GCN	**0.7410**	0.8640	0.8610
Spectral CNN detaching	0.6320	0.7780	0.7860
GWNN	0.7020	0.7480	0.8640

TABLE 3
Comparison of the accuracy of identification of essential proteins between our and state-of-the-art methods.

Methods	Protein
Ours	**0.7789**
GraphSage	0.7740
MoNet	0.7280
Spectral CNN	0.6381
GCN	0.7310
Spectral CNN detaching	0.6476
GWNN	0.7517
DNN	0.7507
IMAMOBH	0.7555
Decision Tree	0.6405
Adaboost	0.6748
Random Forest	0.6650
SVM	0.7033
BC	0.7139
CC	0.7151
DC	0.7336
EC	0.7194
LAC	0.7563
NC	0.7469

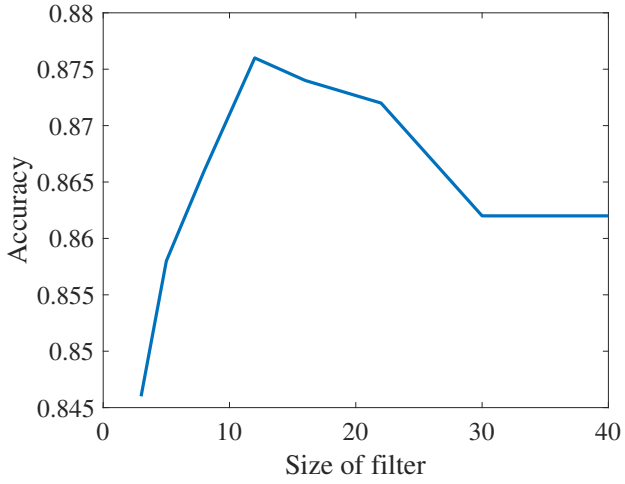

Fig. 5. Accuracy at different filter sizes on Cora.

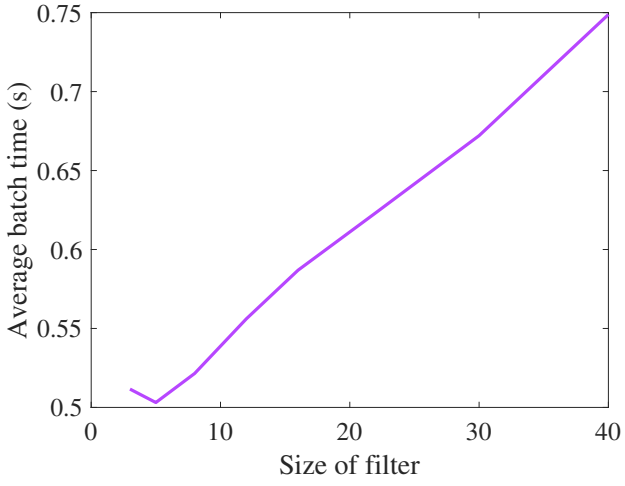

Fig. 6. Average batch time with different filter size on Cora.

with different filter sizes in its first layer. Running time is linear with respect to filter size.

4 Conclusion

In this paper, we presented a novel graph convolutional method, called star topology convolution (STC), which is an inductive spectral convolutional method. It learns node embedding on subgraphs within a star topology. In validation experiments, our method outperformed state-of-the-art graph convolutional methods. In the identification of essential proteins, our method also outperformed state-of-the-art essential protein identification methods.

Acknowledgments

This work is supported by the Hong Kong Research Grants Council (Project 11200818) and City University of Hong Kong (Project 9610460).

REFERENCES

[1] B. Xu, H. Shen, Q. Cao, Y. Qiu, and X. Cheng, "Graph wavelet neural network," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=H1ewdiR5tQ

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[3] J. Gehring, M. Auli, D. Grangier, and Y. N. Dauphin, "A convolutional encoder model for neural machine translation," *CoRR*, vol. abs/1611.02344, 2016. [Online]. Available: http://arxiv.org/abs/1611.02344

[4] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[5] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[6] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 1024–1034. [Online]. Available: http://papers.nips.cc/paper/6703-inductive-representation-learning-on-large-graphs.pdf

[7] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[8] F. Monti, O. Shchur, A. Bojchevski, O. Litany, S. Günnemann, and M. M. Bronstein, "Dual-primal graph convolutional networks," *CoRR*, vol. abs/1806.00770, 2018. [Online]. Available: http://arxiv.org/abs/1806.00770

[9] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5425–5434.

[10] M. Ding, J. Tang, and J. Zhang, "Semi-supervised learning on graphs with generative adversarial nets," *CoRR*, vol. abs/1809.00130, 2018. [Online]. Available: http://arxiv.org/abs/1809.00130

[11] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129 – 150, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1063520310000552

[12] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *CoRR*, vol. abs/1312.6203, 2014.

[13] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 3844–3852. [Online]. Available: http://papers.nips.cc/paper/6081-convolutional-neural-networks-on-graphs-with-fast-localized-spectral-filtering.pdf

[14] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl&noteId=SJU4ayYgl

[15] P. de Haan, T. Cohen, and M. Welling, "Natural graph networks," vol. abs/2007.08349, 07 2020. [Online]. Available: https://arxiv.org/abs/2007.08349v1

[16] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint*, vol. arXiv, 07 2012.

[17] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data articles," *AI Magazine*, vol. 29, pp. 93–106, 09 2008.

[18] M. Zeng, M. Li, Z. Fei, F. Wu, Y. Li, and Y. Pan, "A deep learning framework for identifying essential proteins based on protein-protein interaction network and gene expression data," in *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2018, pp. 583–588.

[19] C. Wu, H. Zhang, L. Zhang, and H. Zheng, "Identification of essential proteins using a novel multi-objective optimization method," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 1329–1333.

[20] H. Jeong, S. Mason, A.-L. Barabasi, and Z. Oltvai, "Lethality and centrality in protein networks," *Nature*, vol. 411, pp. 41–2, 06 2001.

[21] M. Joy, A. Brock, D. Ingber, and S. Huang, "High-betweenness proteins in the yeast protein interaction network," *Journal of biomedicine & biotechnology*, vol. 2005, pp. 96–103, 07 2005.

[22] S. Wuchty and P. Stadler, "Centers of complex networks," *Journal of theoretical biology*, vol. 223, pp. 45–53, 08 2003.

[23] P. Bonacich, "Power and centrality: A family of measures," *American Journal of Sociology*, vol. 92, pp. 1170–1182, 03 1987.

[24] J. Wang, M. Li, H. Wang, and Y. Pan, "Identification of essential proteins based on edge clustering coefficient," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 4, pp. 1070–1080, 2012.

[25] M. Li, J. Wang, X. Chen, H. Wang, and Y. Pan, "A local average connectivity-based method for identifying essential proteins from the network level," *Computational Biology and Chemistry*, vol. 35, no. 3, pp. 143 – 150, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1476927111000296

[26] I. Xenarios, L. Salwínski, X. J. Duan, P. Higney, S.-M. Kim, and D. Eisenberg, "DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions," *Nucleic Acids Research*, vol. 30, no. 1, pp. 303–305, 01 2002. [Online]. Available: https://doi.org/10.1093/nar/30.1.303

[27] H. W. Mewes, D. Frishman, K. F. X. Mayer, M. Münsterkötter, O. Noubibou, P. Pagel, T. Rattei, M. Oesterheld, A. Ruepp, and V. Stümpflen, "MIPS: analysis and annotation of proteins from whole genomes in 2005," *Nucleic Acids Research*, vol. 34, no. suppl_1, pp. D169–D172, 01 2006. [Online]. Available: https://doi.org/10.1093/nar/gkj148

[28] J. M. Cherry, C. Adler, C. Ball, S. A. Chervitz, S. S. Dwight, E. T. Hester, Y. Jia, G. Juvik, T. Roe, M. Schroeder, S. Weng, and D. Botstein, " SGD: Saccharomyces Genome Database ," *Nucleic Acids Research*, vol. 26, no. 1, pp. 73–79, 01 1998. [Online]. Available: https://doi.org/10.1093/nar/26.1.73

[29] R. Zhang and Y. Lin, "DEG 5.0, a database of essential genes in both prokaryotes and eukaryotes," *Nucleic Acids Research*, vol. 37, no. suppl_1, pp. D455–D458, 10 2008. [Online]. Available: https://doi.org/10.1093/nar/gkn858

[30] A. M. Deutschbauer, R. M. Williams, A. M. Chu, and R. W. Davis, "Parallel phenotypic analysis of sporulation and postgermination growth in saccharomyces cerevisiae," *Proceedings of the National Academy of Sciences*, vol. 99, no. 24, pp. 15 530–15 535, 2002. [Online]. Available: https://www.pnas.org/content/99/24/15530

[31] G. Giaever, A. M. Chu, L. Ni, C. Connelly, L. Riles, S. Véronneau, S. Dow, A. Lucau-Danila, K. Anderson, B. André, A. P. Arkin, A. Astromoff, M. El-Bakkoury, R. Bangham, R. Benito, S. Brachat, S. Campanaro, M. Curtiss, K. Davis, A. Deutschbauer, K.-D. Entian, P. Flaherty, F. Foury, D. J. Garfinkel, M. Gerstein, D. Gotte, U. Güldener, J. H. Hegemann, S. Hempel, Z. Herman, D. F. Jaramillo, D. E. Kelly, S. L. Kelly, P. Kötter, D. LaBonte, D. C. Lamb, N. Lan, H. Liang, H. Liao, L. Liu, C. Luo, M. Lussier, R. Mao, P. Menard, S. L. Ooi, J. L. Revuelta, C. J. Roberts, M. Rose, P. Ross-Macdonald, B. Scherens, G. Schimmack, B. Shafer, D. D. Shoemaker, S. Sookhai-Mahadeo, R. K. Storms, J. N. Strathern, G. Valle, M. Voet, G. Volckaert, C.-y. Wang, T. R. Ward, J. Wilhelmy, E. A. Winzeler, Y. Yang, G. Yen, E. Youngman, K. Yu, H. Bussey, J. D. Boeke, M. Snyder, P. Philippsen, R. W. Davis, and M. Johnston, "Functional profiling of the saccharomyces cerevisiae genome," *Nature*, vol. 418, no. 6896, p. 387—391, July 2002. [Online]. Available: https://doi.org/10.1038/nature00935

[32] R. Edgar, M. Domrachev, and A. E. Lash, "Gene Expression Omnibus: NCBI gene expression and hybridization array data repository," *Nucleic Acids Research*, vol. 30, no. 1, pp. 207–210, 01 2002. [Online]. Available: https://doi.org/10.1093/nar/30.1.207

[33] Y. Rong, W. Huang, T. Xu, and J. Huang, "DropEdge: Towards deep graph convolutional networks on node classification," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=Hkx1qkrKPr

Chong Wu (S'19) was born in Zhejiang, China. He received the B.E. degree in automation from the School of Automation, China University of Geosciences, Wuhan, China, in 2018. He is currently pursuing the Ph.D. degree in electrical engineering with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong.

His current research interests include graph representation learning, image/video processing, and artificial intelligence.

Zhenan Feng received the B.E. degree in automation in 2018 from the School of Automation, China University of Geosciences, Wuhan, China, where she is currently working toward the master's degree in control science and engineering.

Her current research interests include graph representation learning, image/video processing, motors and controls, design and optimization of electrical systems, and artificial intelligence.

Jiangbin Zheng was born in Taizhou, Zhejiang, China, in 1996. He is currently pursuing the master's degree with Xiamen University, specializing in intelligent science and technology.

His current research interests include cross-modal sign language translation, machine translation and graph neural network.

Houwang Zhang (S'19) received the B.E. degree in industrial design in 2018 from the School of Mechanical Engineering and Electronic Information, China University of Geosciences, Wuhan, China. Since then, he is currently working toward the master's degree in control science and engineering with the School of Automation, China University of Geosciences, Wuhan, China.

His current research interests include graph representation learning, image/video processing, and bioinformatics.

Jiawang Cao received the B.E. degree in automation from the School of Automation, China University of Geosciences, Wuhan, China, in 2019. He is currently pursuing the master's degree with Fudan University.

His current research interests include computer vision and deep learning.

Hong Yan received the Ph.D. degree from Yale University. He was a Professor of imaging science with the University of Sydney. He is currently the Chair Professor of computer engineering with the City University of Hong Kong. He was elected an IAPR Fellow for contributions to document image analysis and an IEEE Fellow for contributions to image recognition techniques and applications. He received the 2016 Norbert Wiener Award from the IEEE SMC Society for contributions to image and biomolecular pattern recognition techniques.

His current research interests include bioinformatics, image processing, and pattern recognition.