Learning Aligned-Spatial Graph Convolutional Networks for Graph Classification

Lu Bai[1†] (✉), Yuhang Jiao[1 ‡], Lixin Cui[1§], and Edwin R. Hancock[2]

[1] Central University of Finance and Economics, Beijing, China
[2] Department of Computer Science, University of York, York, UK

Abstract. In this paper, we develop a novel Aligned-Spatial Graph Convolutional Network (ASGCN) model to learn effective features for graph classification. Our idea is to transform arbitrary-sized graphs into fixed-sized aligned grid structures, and define a new spatial graph convolution operation associated with the grid structures. We show that the proposed ASGCN model not only reduces the problems of information loss and imprecise information representation arising in existing spatially-based Graph Convolutional Network (GCN) models, but also bridges the theoretical gap between traditional Convolutional Neural Network (CNN) models and spatially-based GCN models. Moreover, the proposed ASGCN model can adaptively discriminate the importance between specified vertices during the process of spatial graph convolution, explaining the effectiveness of the proposed model. Experiments on standard graph datasets demonstrate the effectiveness of the proposed model.

Keywords: Graph Convolutional Networks · Graph Classification.

1 Introduction

Graph-based representations are powerful tools to analyze structured data that are described in terms of pairwise relationships between components [27,5]. One common challenge arising in the analysis of graph-based data is how to learn effective graph representations. Due to the recent successes of deep learning networks in machine learning, there is increasing interest to generalize deep Convolutional Neural Networks (CNN) [16] into the graph domain. These deep learning networks on graphs are the so-called Graph Convolutional Networks (GCN) [15], and have proven to be an effective way to extract highly meaningful statistical features for graph classification [9].

Generally speaking, most existing state-of-the-art GCN approaches can be divided into two main categories with GCN models based on a) spectral and b) spatial strategies. Specifically, approaches based on the spectral strategy define the convolution operation based on spectral graph theory [8,12,19]. By transforming the graph into the spectral domain through the eigenvectors of the

[†] Primary Author: bailucs@cufe.edu.cn
[‡] Co-first Author
[§] Corresponding Author: cuilixin@cufe.edu.cn

Laplacian matrix, these methods perform the filter operation by multiplying the graph by a series of filter coefficients. Unfortunately, most spectral-based approaches cannot be performed on graphs with different size numbers of vertices and Fourier bases. Thus, these approaches demand the same-sized graph structures and are usually employed for vertex classification tasks. On the other hand, approaches based on the spatial strategy are not restricted to the same-sized graph structures. These approaches generalize the graph convolution operation to the spatial structure of a graph by directly defining an operation on neighboring vertices [1,10,24]. For example, Duvenaud et al. [10] have proposed a spatially-based GCN model by defining a spatial graph convolution operation on the 1-layer neighboring vertices to simulate the traditional circular fingerprint. Atwood and Towsley [1] have proposed a spatially-based GCN model by performing spatial graph convolution operations on different layers of neighboring vertices rooted at a vertex. Although these spatially-based GCN models can be directly applied to real-world graph classification problems, they still need to further transform the multi-scale features learned from graph convolution layers into the fixed-sized representations, so that the standard classifiers can directly read the representations for classifications. One way to achieve this is to directly sum up the learned local-level vertex features from the graph convolution operation as global-level graph features through a SumPooling layer. Since it is difficult to learn rich local vertex topological information from the global features, these spatially-based GCN methods associated with SumPooling have relatively poor performance on graph classification.

To overcome the shortcoming of existing spatially-based GCN models, Zhang et al. [28] have developed a novel spatially-based Deep Graph Convolutional Neural Network (DGCNN) model to preserve more vertex information. Specifically, they propose a new SortPooling layer to transform the extracted vertex features of unordered vertices from the spatial graph convolution layers into a fixed-sized local-level vertex grid structure. This is done by sequentially preserving a specified number of vertices with prior orders. With the fixed-sized grid structures of graphs to hand, a traditional CNN model followed by a Softmax layer can be directly employed for graph classification. Although this spatially-based DGCNN model focuses more on local-level vertex features and outperforms state-of-the-art GCN models on graph classification tasks, this method tends to sort the vertex order based on each individual graph. Thus, it cannot accurately reflect the topological correspondence information between graph structures. Moreover, this model also leads to significant information loss, since some vertices associated with lower ranking may be discarded. In summary, developing effective methods to learn graph representations still remains a significant challenge.

In this paper, we propose a novel Aligned-Spatial Graph Convolutional Network (ASGCN) model for graph classification problems. One key innovation of the proposed ASGCN model is that of transitively aligning vertices between graphs. That is, given three vertices v, w and x from three different sample graphs, if v and x are aligned, and w and x are aligned, the proposed model can guarantee that v and w are also aligned. More specifically, the proposed model employs the transitive alignment procedure to transform arbitrary-sized graphs into fixed-sized aligned grid structures with consistent vertex orders, guaranteeing that the vertices on the same spatial position are also transitively aligned to each other in terms of the topological structures. The conceptual framework of the proposed ASGCN model is shown in Fig.1. Specifically, the main contributions are threefold.

First, we develop a new transitive matching method to map different arbitrary-sized graphs into fixed-sized aligned vertex grid structures. We show that the grid structures not only establish reliable vertex correspondence information between graphs, but also minimize the loss of structural information from the original graphs.

Second, we develop a novel spatially-based graph convolution network model, i.e., the ASGCN model, for graph classification. More specifically, we propose a new spatial graph convolution operation associated with the aligned vertex grid structures as well as their associated adjacency matrices, to extract multi-scale local-level vertex features. We show that the proposed convolution operation not only reduces the problems of information loss and imprecise information representation arising in existing spatially-based GCN models associated with SortPooling or SumPooling, but also theoretically relates to the classical convolution operation on standard grid structures. Thus, the proposed ASGCN model bridges the theoretical gap between traditional CNN models and spatially-based GCN models, and can adaptively discriminate the importance between specified vertices during the process of the spatial graph convolution operation. Furthermore, since our spatial graph convolution operation does not change the original spatial sequence of vertices, the proposed ASGCN model utilizes the traditional CNN to further learn graph features. In this way, we provide an end-to-end deep learning architecture that integrates the graph representation learning into both the spatial graph convolutional layer and the traditional convolution layer for graph classification.

Third, we empirically evaluate the performance of the proposed ASGCN model on graph classification tasks. Experiments on benchmarks demonstrate the effectiveness of the proposed method, when compared to state-of-the-art methods.

2 Related Works of Spatially-based GCN Models

In this section, we briefly review state-of-the art spatially-based GCN models in the literature. More specifically, we introduce the associated spatial graph convolution operation of the existing spatially-based Deep Graph Convolutional Neural Network (DGCNN) model [28]. To commence, consider a sample graph G with n vertices, $X = (x_1, x_2, ..., x_n) \in \mathbb{R}^{n \times c}$ is the collection of n vertex feature vectors of G in c dimensions, and $A \in \mathbb{R}^{n \times n}$ is the vertex adjacency matrix (A can be a weighted adjacency matrix). The spatial graph convolution operation of the DGCNN model takes the following form

$$Z = \mathrm{f}(\tilde{D}^{-1}\tilde{A}XW), \tag{1}$$

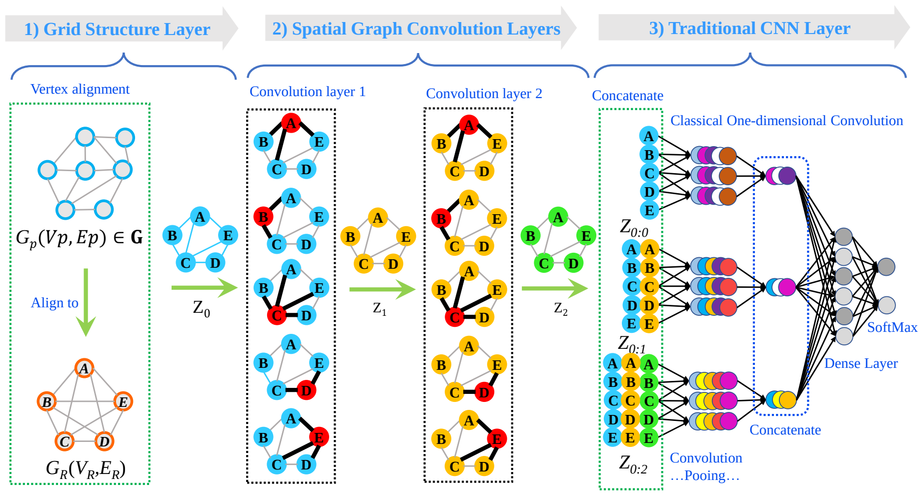


Fig. 1. The architecture of the proposed ASGCN model. An input graph $G_p(V_p, E_p) \in \mathbf{G}$ of arbitrary size is first aligned to the prototype graph $G_R(V_R, E_R)$. Then, G_p is mapped into a fixed-sized aligned vertex grid structure, where the vertex orders follow that of G_R. The grid structure of G_p is passed through multiple spatial graph convolution layers to extract multi-scale vertex features, where the vertex information is propagated between specified vertices associated with the adjacency matrix. Since the graph convolution layers preserve the original vertex orders of the input grid structure, the concatenated vertex features through the graph convolution layers form a new vertex grid structure for G_p. This vertex grid structure is then passed to a traditional CNN layer for classifications. Note that, vertex features are visualized as different colors.

where $\tilde{A} = A + I$ is the adjacency matrix of graph G with added self-loops, $\tilde{D}$ is the degree matrix of $\tilde{A}$ with $\tilde{A}_{i,i} = \sum_j \tilde{A}_{i,j}$, $W \in \mathbb{R}^{c \times c'}$ is the matrix of trainable graph convolution parameters, f is a nonlinear activation function, and $Z \in \mathbb{R}^{n \times c'}$ is the output of the convolution operation.

For the spatial graph convolution operation defined by Eq.(1), the process XW first maps the c-dimensional features of each vertex into a set of new c'-dimensional features. Here, the filter weights W are shared by all vertices. Moreover, $\tilde{A}Y$ $(Y := XW)$ propagates the feature information of each vertex to its neighboring vertices as well as the vertex itself. The i-th row $(\tilde{A}Y)_{i,:}$ represents the extracted features of the i-th vertex, and corresponds to the summation or aggregation of $Y_{i,:}$ itself and $Y_{j,:}$ from the neighboring vertices of the i-th vertex. Multiplying by the inverse of $\tilde{D}$ (i.e., $\tilde{D}^{-1}$) can be seen as the process of normalizing and assigning equal weights between the i-th vertex and each of its neighbours.

Remark: Eq.(1) indicates that the spatial graph convolution operation of the DGCNN model cannot discriminate the importance between specified vertices in the convolution operation process. This is because the required filter weights of W are shared by each vertex, i.e., the feature transformations of the vertices are all based on the same trainable function. Thus, the DGCNN model cannot directly influence the aggregation process of the vertex features. In fact, this problem also arises in other spatially-based GCN models, e.g., the Neural Graph Fingerprint Network (NGFN) model [10], the Diffusion Convolution Neural Network (DCNN) model [1], etc. Since the associated spatial graph convolution operations of these models also take the similar form with that of the DGCNN model, i.e., the trainable parameters of their spatial graph convolution operations are also shared by each vertex. This drawback influences the effectiveness of the existing spatially-based GCN models for graph classification. In this paper, we aim to propose a new spatially-based GCN model to overcome the above problems. □

3 Constructing Aligned Grid Structures for Arbitrary Graphs

Although, spatially-based GCN models are not restricted to the same graph structure, and can thus be applied for graph classification tasks. These methods still require to further transform the extracted multi-scale features from graph convolution layers into the fixed-sized characteristics, so that the standard classifiers (e.g., the traditional convolutional neural network followed by a Softmax layer) can be directly employed for classifications. In this section, we develop a new transitive matching method to map different graphs of arbitrary sizes into fixed-sized aligned grid structures. Moreover, we show that the proposed grid structure not only integrates precise structural correspondence information but also minimises the loss of structural information.

3.1 Identifying Transitive Alignment Information between Graphs

We introduce a new graph matching method to transitively align graph vertices. We first designate a family of prototype representations that encapsulate the principle characteristics over all vectorial vertex representations in a set of graphs $\mathbf{G}$. Assume there are n vertices from all graphs in $\mathbf{G}$, and their associated K-dimensional vectorial representations are $\mathbf{R}^K = \{\mathrm{R}_1^K, \mathrm{R}_2^K, \ldots, \mathrm{R}_n^K\}$. We utilize k-means [25] to locate M centroids over $\mathbf{R}^K$, by minimizing the objective function

$$\arg\min_{\Omega} \sum_{j=1}^{M} \sum_{\mathrm{R}_i^K \in c_j} \|\mathrm{R}_i^K - \mu_j^K\|^2, \tag{2}$$

$\Omega = (c_1, c_2, \ldots, c_M)$ represents M clusters, and μ_j^K is the mean of the vertex representations belonging to the j-th cluster c_j.

Let $\mathbf{G} = \{G_1, \cdots, G_p, \cdots, G_N\}$ be the graph sample set. For each sample graph $G_p(V_p, E_p) \in \mathbf{G}$ and each vertex $v_i \in V_p$ associated with its K-dimensional vectorial representation $\mathrm{R}_{p;i}^K$, we commence by identifying a set of K-dimensional prototype representations as $\mathbf{PR}^K = \{\mu_1^K, \ldots, \mu_j^K, \ldots, \mu_M^K\}$ for the graph set $\mathbf{G}$. We align the vectorial vertex representations of each graph G_p to the family of prototype representations in $\mathbf{PR}^K$. The alignment procedure is similar to that introduced in [6] for point matching in a pattern space, and we compute a K-level affinity matrix in terms of the Euclidean distances between the two sets of points, i.e.,

$$A_p^K(i,j) = \|\mathrm{R}_{p;i}^K - \mu_j^K\|_2. \tag{3}$$

where A_p^K is a $|V_p| \times M$ matrix, and each element $A_p^K(i,j)$ represents the distance between the vectrial representation $\mathrm{R}_{p;i}^K$ of $v \in V_p$ and the j-th prototype representation $\mu_j^K \in \mathbf{PR}^K$. If $A_p^K(i,j)$ is the smallest element in row i, we say that the vertex v_i is aligned to the j-th prototype representation. Note that for each graph there may be two or more vertices aligned to the same prototype representation. We record the correspondence information using the K-level correspondence matrix $C_p^K \in \{0,1\}^{|V_p| \times M}$

$$C_p^K(i,j) = \begin{cases} 1 \text{ if } A_p^K(i,j) \text{ is the smallest in row } i \\ 0 \text{ otherwise.} \end{cases} \tag{4}$$

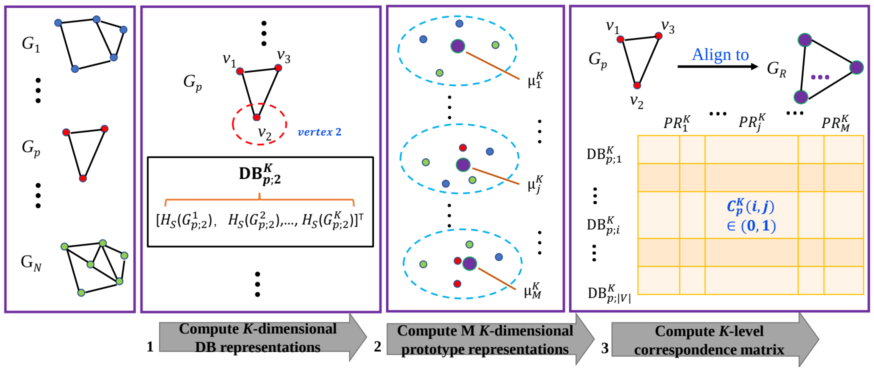
For each pair of graphs $G_p \in \mathbf{G}$ and $G_q \in \mathbf{G}$, if their vertices v_p and v_q are aligned to the same prototype representation μ_j^K, we say that v_p and v_q are also aligned. Thus, we identify the transitive correspondence information between all graphs in $\mathbf{G}$, by aligning their vertices to a common set of prototype representations.

Remark: The alignment process is equivalent to assigning the vectorial representation $\mathrm{R}_{p;i}^K$ of each vertex $v_i \in V_p$ to the mean μ_j^K of the cluster c_j. Thus, the proposed alignment procedure can be seen as an optimization process that gradually minimizes the inner-vertex-cluster sum of squares over the vertices of all graphs through k-means, and can establish reliable vertex correspondence information over all graphs. □

3.2 Aligned Grid Structures of Graphs

We employ the transitive correspondence information to map arbitrary-sized graphs into fixed-sized aligned grid structures. Assume $G_p(V_p, E_p, \tilde{A}_p)$ is a sample graph from the graph set $\mathbf{G}$, with V_p representing the vertex set, E_p representing the edge set, and $\tilde{A}_p$ representing the vertex adjacency matrix with added self-loops (i.e., $\tilde{A} = A + I$, where A is the original adjacency matrix with no self-loops and I is the identity matrix). Let $X_p \in \mathbb{R}^{n \times c}$ be the collection of n ($n = |V_p|$) vertex feature vectors of G_p in c dimensions. Note that, the row of X_p follows the same vertex order of $\tilde{A}_p$. If G_p are vertex attributed graphs, X_p can be the one-hot encoding matrix of the vertex labels. For un-attributed graphs, we use the vertex degree as the vertex label.

For each graph G_p, we utilize the proposed transitive vertex matching method to compute the K-level vertex correspondence matrix C_p^K that records the



G_1
G_p
G_N
G_p
v_1
v_3
v_2
vertex 2
$\mathbf{DB}_{p;2}^K$
$[H_S(G_{p;2}^1), H_S(G_{p;2}^2), \ldots, H_S(G_{p;2}^K)]^T$
μ_1^K
μ_j^K
μ_M^K
Align to
G_R
PR_1^K
PR_j^K
PR_M^K
$DB_{p;1}^K$
$DB_{p;i}^K$
$DB_{p;|V|}^K$
$C_p^K(i,j) \in (0,1)$
1 Compute K-dimensional DB representations
2 Compute M K-dimensional prototype representations
3 Compute K-level correspondence matrix

Fig. 2. The procedure of computing the correspondence matrix. Given a set of graphs, for each graph G_p: (1) we compute the K-dimensional depth-based (DB) representation $\mathrm{DB}_{p;v}^K$ rooted at each vertex (e.g., vertex 2) as the K-dimensional vectorial vertex representation, where each element $H_s(G_{p;2}^K)$ represents the Shannon entropy of the K-layer expansion subgraph rooted at vertex v_2 of G_p [2]; (2) we identify a family of K-dimensional prototype representations $\mathbf{PR}^K = \{\mu_1^K, \ldots, \mu_j^K, \ldots, \mu_M^K\}$ using k-means on the K-dimensional DB representations of all graphs; (3) we align the K-dimensional DB representations to the K-dimensional prototype representations and compute a K-level correspondence matrix C_p^K.

correspondence information between the K-dimensional vectorial vertex representations of G_p and the K-dimensional prototype representations in $\mathbf{PR}^K = \{\mu_1^K, \ldots, \mu_j^K, \ldots, \mu_M^K\}$. With C_p^K to hand, we compute the K-level aligned vertex feature matrix for G_p as

$$\bar{X}_p^K = (C_p^K)^T X_p, \tag{5}$$

where $\bar{X}_p^K \in \mathbb{R}^{M\times c}$ and each row of $\bar{X}_p^K$ represents the feature of a corresponding aligned vertex. Moreover, we also compute the associated K-level aligned vertex adjacency matrix for G_p as

$$\bar{A}_p^K = (C_p^K)^T (\tilde{A}_p)(C_p^K), \tag{6}$$

where $\bar{A}_p^K \in \mathbb{R}^{M\times M}$. Both $\bar{X}_p^K$ and $\bar{A}_p^K$ are indexed by the corresponding prototypes in $\mathbf{PR}^K$. Since $\bar{X}_p^K$ and $\bar{A}_p^K$ are computed from the original vertex feature matrix X_p and the original adjacency matrix $\tilde{A}_p$, respectively, by mapping the original feature and adjacency information of each vertex $v_p \in V_p$ to that of the new aligned vertices, $\bar{X}_p^K$ and $\bar{A}_p^K$ encapsulate the original feature and structural information of G_p. Note that, according to Eq. 4 each prototype may be aligned by more than one vertex from V_p,thus $\bar{A}_p^K$ may be a weighted adjacency matrix.

In order to construct the fixed-sized aligned grid structure for each graph $G_p \in \mathbf{G}$, we need to sort the vertices to determine their spatial orders. Since the vertices of each graph are all aligned to the same prototype representations, we sort the vertices of each graph by reordering the prototype representations. To this end, we construct a prototype graph $G_{\mathrm{R}}(V_{\mathrm{R}}, E_{\mathrm{R}})$ that captures the pairwise similarity between the K-dimensional prototype representations in $\mathbf{PR}^K$, with each vertex $v_j \in V_{\mathrm{R}}$ representing the prototype representation $\mu_j^K \in \mathbf{PR}^K$ and each edge $(v_j, v_k) \in E_{\mathrm{R}}$ representing the similarity between $\mu_j^K \in \mathbf{PR}^K$ and $\mu_k^K \in \mathbf{PR}^K$. The similarity between two vertices of G_{R} is computed as

$$s(\mu_j^K, \mu_k^K) = \exp(-\frac{\|\mu_j^K - \mu_k^K\|_2}{K}). \tag{7}$$

The degree of each prototype representation μ_j^K is $D_R(\mu_j^K) = \sum_{k=1}^{M} s(\mu_j^K, \mu_k^K)$. We propose to sort the K-dimensional prototype representations in $\mathbf{PR}^K$ according to their degree $D_R(\mu_j^K)$. Then, we rearrange $\bar{X}_p^K$ and $\bar{A}_p^K$ accordingly.

To construct reliable grid structures for graphs, in this work we employ the depth-based (DB) representations as the vectorial vertex representations to compute the required K-level vertex correspondence matrix C_p^K. The DB representation of each vertex is defined by measuring the entropies on a family of k-layer expansion subgraphs rooted at the vertex [3], where the parameter k varies from 1 to K. It is shown that such a K-dimensional DB representation encapsulates rich entropy content flow from each local vertex to the global graph structure, as a function of depth. The process of computing the correspondence matrix C_p^K associated with depth-based representations is shown in Fig.3. When we vary the number of layers K from 1 to L (i.e., $K \leq L$), we compute the final **aligned vertex grid structure** for each graph $G_p \in \mathbf{G}$ as

$$\bar{X}_p = \sum_{K=1}^{L} \frac{\bar{X}_p^K}{L}, \tag{8}$$

and the associated **aligned grid vertex adjacency matrix** as

$$\bar{A}_p = \sum_{K=1}^{L} \frac{\bar{A}_p^K}{L}, \tag{9}$$

where $\bar{X}_p \in \mathbb{R}^{M \times c}$, $\bar{A}_p \in \mathbb{R}^{M \times M}$, the i-th row of $\bar{X}_p$ corresponds to the feature vector of the i-th aligned grid vertex, and the i-row and j-column element of $\bar{A}_p$ corresponds to the adjacency information between the i-th and j-th aligned grid vertices.

Remark: Eq.(8) and Eq.(9) indicate that they can transform the original graph $G_p \in \mathbf{G}$ with arbitrary number of vertices $|V_p|$ into a new aligned grid graph structure with the same number of vertices, where $\bar{X}_p$ is the corresponding aligned grid vertex feature matrix and $\bar{A}_p$ is the corresponding aligned grid vertex adjacency matrix. Since both $\widehat{X}_p$ and $\bar{A}_p$ are mapped from the original graph G_p, they not only reflect reliable structure correspondence information between G_p and the remaining graphs in graph set $\mathbf{G}$ but also encapsulate more original feature and structural information of G_p. □

4 The Aligned-Spatial Graph Convolutional Network Model

In this section, we propose a new spatially-based GCN model, namely the Aligned-Spatial Graph Convolutional Network (ASGCN) model. The core stage of a spatially-based GCN model is the associated graph convolution operation that extracts multi-scale features for each vertex based on the original features of its neighboring vertices as well as itself. As we have stated, most existing spatially-based GCN models perform the convolution operation by first applying a trainable parameter matrix to map the original feature of each vertex in c dimensions to that in c' dimensions, and then averaging the vertex features of specified vertices [1,10,24,28]. Since the trainable parameter matrix is shared by all vertices, these models cannot discriminate the importance of different vertices and have inferior ability to aggregate vertex features. To overcome the shortcoming, in this section we first propose a new spatial graph convolution operation associated with aligned grid structures of graphs. Unlike existing methods, the trainable parameters of the proposed convolution operation can directly influence the aggregation of the aligned grid vertex features, thus the proposed convolution operation can discriminate the importance between specified aligned grid vertices. Finally, we introduce the architecture of the ASGCN model associated with the proposed convolution operation.

4.1 The Proposed Spatial Graph Convolution Operation

In this subsection, we propose a new spatial graph convolution operation to further extract multi-scale features of graphs, by propagating features between aligned grid vertices. Specifically, given a sample graph $G(V,E)$ with its aligned vertex grid structure $\bar{X} \in \mathbb{R}^{M\times c}$ and the associated aligned grid vertex adjacency matrix $\bar{A} \in \mathbb{R}^{M\times M}$, the proposed spatial graph convolution operation takes the following form

$$Z^h = \mathrm{Relu}(\bar{D}^{-1}\bar{A}\sum_{j=1}^{c}(\bar{X} \odot W^h)_{:,j}), \tag{10}$$

where Relu is the rectified linear units function (i.e., a nonlinear activation function), $W^h \in \mathbb{R}^{M\times c}$ is the trainable graph convolution parameter matrix of the h-th convolution filter with the filter size $M \times 1$ and the channel number c, $\odot$ represents the element-wise Hadamard product, $\bar{D}$ is the degree matrix of $\bar{A}$, and $Z^h \in \mathbb{R}^{M\times 1}$ is the output activation matrix. Note that, since the aligned grid vertex adjacency matrix $\bar{A}$ is computed based on the original vertex adjacency matrix with added self-loop information, the degree matrix also encapsulates the self-loop information from $\bar{A}$.

An instance of the proposed spatial graph convolution operation defined by Eq.(10) is shown in Fig.3. Specifically, the proposed convolution operation consists of four steps. **In the first step**, the procedure $\sum_{j=1}^{c}(\bar{X} \odot W^h)_{:,j}$ commences by computing the element-wise Hadamard product between $\bar{X}$ and W^h,

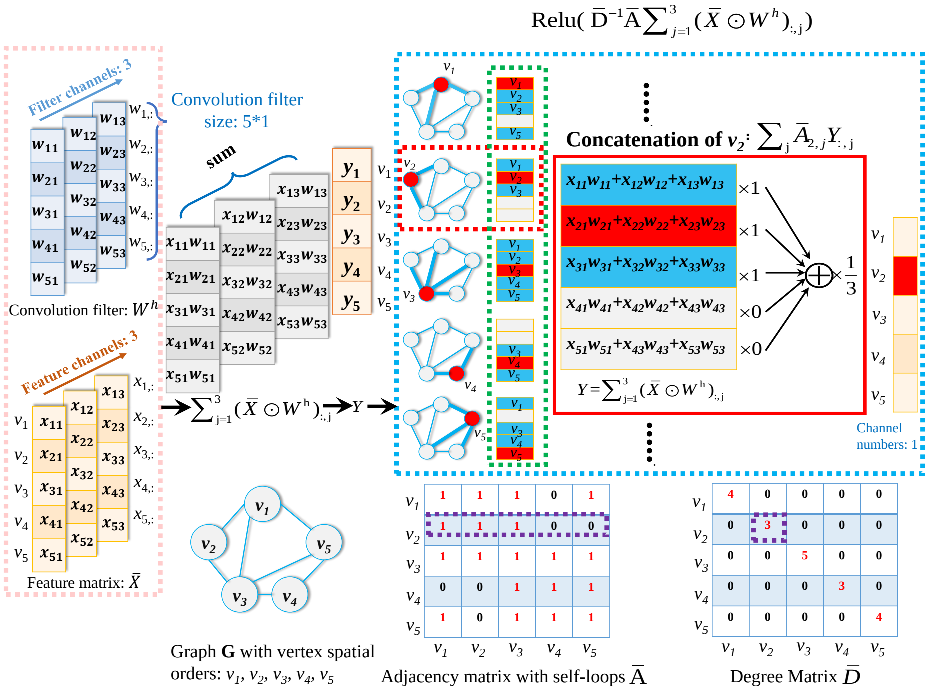

Relu($\bar{D}^{-1}\bar{A}\sum_{j=1}^{3}(\bar{X}\odot W^h)_{:,j}$)
Filter channels: 3
Convolution filter size: 5*1
sum
Convolution filter: W^h
Feature channels: 3
Feature matrix: $\bar{X}$
Concatenation of v_2: $\sum_j \bar{A}_{2,j}Y_{:,j}$
$Y=\sum_{j=1}^{3}(\bar{X}\odot W^h)_{:,j}$
Channel numbers: 1
Graph **G** with vertex spatial orders: v_1, v_2, v_3, v_4, v_5
Adjacency matrix with self-loops $\bar{A}$
Degree Matrix $\bar{D}$

Fig. 3. An Instance of the Proposed Spatial Graph Convolution Operation.

and then summing the channels of $\bar{X} \odot W^h$ (i.e., summing the columns of $\bar{X} \odot W^h$). Fig.3 exhibits this process. Assume $\bar{X}$ is the collection of 5 aligned grid vertex feature vectors in the 3 dimensions (i.e., 3 feature channels), W^h is the h-th convolution filter with the filter size 5×1 and the channel number 3. The resulting $\sum_{j=1}^{3}(\bar{X} \odot W^h)_{:,j}$ first assigns the feature vector $x_{i,:}$ of each i-th aligned grid vertex a different weighted vector $w_{i,:}$, and then sums the channels of each weighted feature vector. Thus, for the first step, $\sum_{j=1}^{c}(\bar{X} \odot W^h)_{:,j}$ can be seen as a new weighted aligned vertex grid structure with 1 vertex feature channel. **The second step** $\bar{A}Y$, where $Y := \sum_{j=1}^{c}(\bar{X} \odot W^h)_{:,j}$, propagates the weighted feature information between each aligned grid vertex as well as its neighboring aligned grid vertices. Specifically, each i-th row $(\bar{A}Y)_{i,:}$ of $\bar{A}Y$ equals to $\sum_j \bar{A}_{i,j}Y_{:,j}$, and can be seen as the aggregated feature vector of the i-th aligned grid vertex by summing its original weighted feature vector as well as all the original weighted feature vectors of the j-th aligned grid vertex that is adjacent to it. Note that, since the first step has assigned each i-th aligned grid vertex a different weighted vector $w_{i,:}$, this aggregation procedure is similar to performing a standard fixed-sized convolution filter on a standard grid structure, where the filter first assigns different weighted vectors to the features of each grid element as well as its neighboring grid elements and then aggregates (i.e., sum) the weighted features as the new feature for each grid element. This indicates that the trainable parameter matrix W^h of the proposed convolution operation can directly influence the aggregation process of the vertex features, i.e., it can adaptively discriminate the importance between specified vertices. Fig.3 exhibits this propagation process. For the 2-nd aligned grid vertex v_2 (marked by the red broken-line frame), the 1-st and 3-rd aligned grid vertices v_1 and v_3 are adjacent to it. The process of computing $\sum_j \bar{A}_{2,j} Y_{:,j}$ (marked by the red real-line frame) aggregates the weighted feature vectors of aligned grid vertex v_2 as well as its neighboring aligned grid vertices v_1 and v_3 as the new feature vector of v_2. The vertices participating in this aggregation process are indicated by the 2-nd row of $\bar{A}$ (marked by the purple broken-line frame on $\bar{A}$) that encapsulates the aligned grid vertex adjacent information. **The third step** normalizes each i-th row of $\bar{A}Y$ by multiplying $\bar{D}_{i,i}^{-1}$, where $\bar{D}_{i,i}$ is the i-th diagonal element of the degree matrix $\bar{D}$. This process can guarantee a fixed feature scale after the proposed convolution operation. Specifically, Fig.3 exhibits this normalization process. The aggregated feature of the 2-nd aligned grid vertex (marked by the red real-line frame) is multiplied by 3^{-1}, where 3 is the 3-rd diagonal element of $\bar{D}$ (marked by the purple broken-line frame on $\bar{D}$). **The last step** employs the Relu activation function and outputs the result.

Note that, since the proposed spatial graph convolution operation only extracts new features for the aligned grid vertex and does not change the vertex orders, the output Z^h is still an aligned vertex grid structure with the same vertex order of $\bar{X}$.

4.2 The Architecture of the Proposed ASGCN Model

In this subsection, we introduce the architecture of the proposed ASGCN Model. Fig.1 has shown the overview of the ASGCN model. Specifically, the architecture is composed of three sequential stages, i.e., 1) the grid structure construction and input layer, 2) the spatial graph convolution layer, and 3) the traditional CNN and Softmax layers.

The Grid Structure Construction and Input Layer: For the proposed ASGCN model, we commence by employing the transitive matching method defined earlier to map each graph $G \in \mathbf{G}$ of arbitrary sizes into the fixed-sized aligned grid structure, including the aligned vertex grid structure $\bar{X}$ and the associated aligned grid vertex adjacency matrix $\bar{A}$. We then input the grid structures to the proposed ASGCN model.

The Spatial Graph Convolutional Layer: For each graph G, to extract multi-scale features of the aligned grid vertices, we stack multiple graph convolution layers associated with the proposed spatial graph convolution operation defined by Eq.(10) as

$$Z_t^h = \mathrm{Relu}(\bar{D}^{-1}\bar{A} \sum_{j=1}^{H_{t-1}} (Z_{t-1} \odot W_t^h)_{:,j}), \tag{11}$$

where Z_0 is the input aligned vertex grid structure $\bar{X}$, H_{t-1} is the number of convolution filters in graph convolution layer $t-1$, $Z_{t-1} \in \mathbb{R}^{M \times H_{t-1}}$ is the concatenated output of all H_{t-1} convolution filters in layer $t-1$, Z_t^h is the output of the h-th convolution filter in layer t, and $W_t^h \in \mathbb{R}^{M \times H_{t-1}}$ is the trainable parameter matrix of the h-th convolution filter in layer t with the filter size $M \times 1$ and the channel number H_{t-1}.

The Traditional CNN Layer: After each t-th spatial graph convolution layer, we horizontally concatenate the output Z_t associated with the outputs of the previous 1 to $t-1$ spatial graph convolutional layers as well as the original input Z_0 as $Z_{0:t}$, i.e., $Z_{0:t} = [Z_0, Z_1, \ldots, Z_t]$ and $Z_{0:t} \in \mathbb{R}^{M \times (c + \sum_{z=1}^{t} H_t)}$. As a result, for the concatenated output $Z_{0:t}$, each of its row can be seen as the new multi-scale features for the corresponding aligned grid vertex. Since $Z_{0:t}$ is still an aligned vertex grid structure, one can directly utilize the traditional CNN on the grid structure. Specifically, Fig.1 exhibits the architecture of the traditional CNN layers associated with each $Z_{0:t}$. Here, each concatenated vertex grid structure $Z_{0:t}$ is seen as a $M \times 1$ (in Fig.1 $M = 5$) vertex grid structure and each vertex is represented by a $(c + \sum_{z=1}^{t} H_t)$-dimensional feature, i.e., the channel of each grid vertex is $c + \sum_{z=1}^{t} H_t$. Then, we add a one-dimension convolution layer. The convolution operation can be performed by sliding a fixed-sized filter of size $k \times 1$ (in Fig.1 $k = 3$) over the spatially neighboring vertices. After this, several AvgPooling layers and remaining one-dimensional convolutional layers can be added to learn the local patterns on the aligned grid vertex sequence. Finally, when we vary t from 0 to T (in Fig.1 $T = 2$), we will obtain $T + 1$ extracted pattern representations. We concatenate the extracted patterns of each $Z_{0:t}$ and add a fully-connected layer followed by a Softmax layer.

4.3 Discussions of the Proposed ASGCN

Comparing to existing state-of-the-art spatially-based GCN models, the proposed ASGCN model has a number of advantages.

First, in order to transform the extracted multi-scale features from the graph convolution layers into fixed-sized representations, both the Neural Graph Fingerprint Network (NGFN) model [10] and the Diffusion Convolution Neural Network (DCNN) model [1] sum up the extracted local-level vertex features as global-level graph features through a SumPooling layer. Although the fixed-sized features can be directly read by a classifier for classifications, it is difficult to capture local topological information residing on the local vertices through the global-level graph features. By contrast, the proposed ASGCN model focuses more on extracting local-level aligned grid vertex features through the proposed spatial graph convolution operation on the aligned grid structures of graphs. Thus, the proposed ASGCN model can encapsulate richer local structural information than the NGFN and DCNN models associated with SumPooling.

Second, similar to the proposed ASGCN model, both the PATCHY-SAN based Graph Convolution Neural Network (PSGCNN) model [17] and the Deep Graph Convolution Neural Network (DGCNN) model [28] also need to form fixed-sized vertex grid structures for arbitrary-sized graphs. To achieve this, these models rearrange the vertex order of each graph structure, and preserve a specified number of vertices with higher ranks. Although, unify the number of vertices for different graphs, the discarded vertices may lead to significant information loss. By contrast, the associated aligned grid structures of the proposed ASGCN model can encapsulate all the original vertex features from the original graphs, thus the proposed ASGCN model constrains the shortcoming of information loss arising in the PSGCNN and DGCNN models. On the other hand, both the PSGCNN and DGCNN models tend to sort the vertices of each graph based on the local structural descriptor, ignoring consistent vertex correspondence information between different graphs. By contrast, the associated aligned grid structure of the proposed ASGCN model is constructed through a transitive vertex alignment procedure. As a result, only the proposed ASGCN model can encapsulate the structural correspondence information between any pair of graph structures, i.e., the vertices on the same spatial position are also transitively aligned to each other.

Finally, as we have stated in Sec.4.1, the spatial graph convolution operation of the proposed ASGCN model is similar to performing standard fixed-sized convolution filters on standard grid structures. To further reveal this property, we explain the convolution process one step further associated with Fig.3. For the sample graph G shown in Fig.3, assume it has 5 vertices following the fixed spatial vertex orders (positions) v_1, v_2, v_3, v_4 and v_5, $\bar{X}$ is the collection of its vertex feature vectors with 3 feature channels, and W^h is the h-th convolution filter with the filter size 5×1 and the channel number 3. Specifically, the procedure marked by the blue broken-line frame of Fig.3 indicates that performing the proposed spatial graph convolution operation on the aligned vertex grid structure $\bar{X}$ can be seen as respectively performing the same 5×1-sized convolution filter W^h on five 5×1-sized local-level neighborhood vertex grid structures included in the green broken-line frame. Here, each neighborhood vertex grid structure only encapsulates the original feature vectors of a root vertex as well as its adjacent vertices from G, and all the vertices follow their original vertex spatial positions in G. For the non-adjacent vertices, we assign dummy vertices (marked by the grey block) on the corresponding spatial positions of the neighborhood vertex grid structures, i.e., the elements of their feature vectors are all 0. Since the five neighborhood vertex grid structures are arranged by the spatial orders of their root vertices from G, the vertically concatenation of these neighborhood vertex grid structures can be seen as a 25×1-sized global-level grid structure $\bar{\mathbf{X}}_G$ of G. We observe that the process of the proposed spatial convolution operation on $\bar{X}$ is equivalent to sliding the 5×1 fixed-sized convolution filter W^h over $\bar{\mathbf{X}}_G$ with 5-stride, i.e., this process is equivalent to sliding a standard classical convolution filter on standard grid structures. As a result, the spatial graph convolution operation of the proposed ASGCN model is theoretically related to the classical convolution operation on standard grid structures, bridging the theoretical gap between the traditional CNN models and the spatially-based GCN models. Furthermore, since the convolution filter W^h of the proposed ASGCN model is related to the classical convolution operation, it assigns each vertex a different weighted parameter. Thus, the proposed ASGCN model can adaptively discriminate the importance between specified vertices during the convolution operation. By contrast, as stated in Sec.2, the existing spatial graph convolution operation of the DGCNN model only maps each vertex feature vector in c dimensions to that in c' dimensions, and all the vertices share the same trainable parameters. As a result, the DGCNN model has less ability to discriminate the importance of different vertices during the convolution operation.

5 Experiments

In this section, we compare the performance of the proposed ASGCN model to both state-of-the-art graph kernels and deep learning methods on graph classification problems based on seven standard graph datasets. These datasets are abstracted from bioinformatics and social networks. Detailed statistics of these datasets are shown in Table.1.

Table 1. Information of the Graph Datasets

Datasets	MUTAG	PROTEINS	D&D	ENZYMES	IMDB-B	IMDB-M	RED-B
Max # vertices	28	620	5748	126	136	89	3783
Mean # vertices	17.93	39.06	284.30	32.63	19.77	13.00	429.61
# graphs	188	1113	1178	600	1000	1500	2000
# vertex labels	7	61	82	3	−	−	−
# classes	2	2	2	6	2	3	2
Description	Bioinformatics	Bioinformatics	Bioinformatics	Bioinformatics	Social	Social	Social

Experimental Setup: We compare the performance of the proposed ASGCN model on graph classification problems with a) six alternative state-of-the-art graph kernels and b) seven alternative state-of-the-art deep learning methods for graphs. Specifically, the graph kernels include 1) the Jensen-Tsallis q-difference kernel (JTQK) with $q = 2$ [4], 2) the Weisfeiler-Lehman subtree kernel (WLSK) [21], 3) the shortest path graph kernel (SPGK) [7], 4) the shortest path kernel based on core variants (CORE SP) [18], 5) the random walk graph kernel (RWGK) [14], and 6) the graphlet count kernel (GK) [20]. The deep learning methods include 1) the deep graph convolutional neural network (DGCNN) [28], 2) the PATCHY-SAN based convolutional neural network for graphs (PSGCNN) [17], 3) the diffusion convolutional neural network (DCNN) [1], 4) the deep graphlet kernel (DGK) [26], 5) the graph capsule convolutional neural network (GCCNN) [23], 6) the anonymous walk embeddings based on feature driven (AWE) [13], and 7) the edge-conditioned convolutional network (ECC) [22].

For the evaluation, we employ the same network structure for the proposed ASGCN model on all graph datasets. Specifically, we set the number of the prototype representations as $M = 64$, the number of the proposed spatial graph

Table 2. Classification Accuracy (In % ± Standard Error) for Comparisons.

Datasets	MUTAG	PROTEINS	D&D	ENZYMES	IMDB-B	IMDB-M	RED-B
ASGCN	**89.70** ± 0.85	**76.50** ± 0.59	**80.40** ± 0.95	50.61 ± 1.01	**73.86** ± 0.92	50.86 ± .85	90.60 ± 0.24
JTQK	85.50 ± 0.55	72.86 ± 0.41	79.89 ± 0.32	**56.41** ± 0.42	72.45 ± 0.81	50.33 ± 0.49	77.60 ± 0.35
WLSK	82.88 ± 0.57	73.52 ± 0.43	79.78 ± 0.36	52.75 ± 0.44	71.88 ± 0.77	49.50 ± 0.49	76.56 ± 0.30
SPGK	83.38 ± 0.81	75.10 ± 0.50	78.45 ± 0.26	29.00 ± 0.48	71.26 ± 1.04	**51.33** ± 0.57	84.20 ± 0.70
CORE SP	88.29 ± 1.55	–	77.30 ± 0.80	41.20 ± 1.21	72.62 ± 0.59	49.43 ± 0.42	**90.84** ± 0.14
GK	81.66 ± 2.11	71.67 ± 0.55	78.45 ± 0.26	24.87 ± 0.22	65.87 ± 0.98	45.42 ± 0.87	77.34 ± 0.18
RWGK	80.77 ± 0.72	74.20 ± 0.40	71.70 ± 0.47	22.37 ± 0.35	67.94 ± 0.77	46.72 ± 0.30	72.73 ± 0.39
Datasets	MUTAG	PROTEINS	D&D	ENZYMES	IMDB-B	IMDB-M	RED-B
ASGCN	**89.70** ± 0.85	**76.50** ± 0.59	**80.40** ± 0.95	50.61 ± 1.01	**73.86** ± 0.92	50.86 ± .85	**90.60** ± 0.24
DGCNN	85.83 ± 1.66	75.54 ± 0.94	79.37 ± 0.94	51.00 ± 7.29	70.03 ± 0.86	47.83 ± 0.85	76.02 ± 1.73
PSGCNN	88.95 ± 4.37	75.00 ± 2.51	76.27 ± 2.64	–	71.00 ± 2.29	45.23 ± 2.84	86.30 ± 1.58
DCNN	66.98	61.29 ± 1.60	58.09 ± 0.53	42.44 ± 1.76	49.06 ± 1.37	33.49 ± 1.42	–
GCCNN	–	76.40 ± 4.71	77.62 ± 4.99	**61.83** ± 5.39	71.69 ± 3.40	48.50 ± 4.10	87.61 ± 2.51
DGK	82.66 ± 1.45	71.68 ± 0.50	78.50 ± 0.22	53.40 ± .90	66.96 ± 0.56	44.55 ± 0.52	78.30 ± 0.30
AWE	87.87 ± 9.76	–	71.51 ± 4.02	35.77 ± 5.93	73.13 ± 3.28	**51.58** ± 4.66	82.97 ± 2.86
ECC	76.11	–	72.54	45.67	–	–	–

convolution layers as 5, and the number of the spatial graph convolutions in each layer as 32. Based on Fig.1 and Sec.4.2, we will get 6 concatenated outputs after the graph convolution layers, we utilize a traditional CNN layer with the architecture as C32-P2-C32-P2-C32-F128 to further learn the extracted patterns, where Ck denotes a traditional convolutional layer with k channels, Pk denotes a classical AvgPooling layer of size and stride k, and FCk denotes a fully-connected layer consisting of k hidden units. The filter size and stride of each Ck are all 5 and 1. With the six sets of extracted patterns after the CNN layers to hand, we concatenate and input them into a new fully-connected layer followed by a Softmax layer with a dropout rate of 0.5. We use the rectified linear units (ReLU) in either the graph convolution or the traditional convolution layer. The learning rate of the proposed model is 0.00005 for all datasets. The only hyperparameter we optimized is the number of epochs and the batch size for the mini-batch gradient decent algorithm. To optimize the proposed ASGCN model, we use the Stochastic Gradient Descent with the Adam updating rules. Finally, note that, our model needs to construct the prototype representations to identify the transitive vertex alignment information over all graphs. In this evaluation we propose to compute the prototype representations from both the training and testing graphs. Thus, our model is an instance of transductive learning [11], where all graphs are used to compute the prototype representations but the class labels of the testing graphs are not used during the training process. For our model, we perform 10-fold cross-validation to compute the classification accuracies, with nine folds for training and one fold for testing. For each dataset, we repeat the experiment 10 times and report the average classification accuracies and standard errors in Table.2.

For the alternative graph kernels, we follow the parameter setting from their original papers. We perform 10-fold cross-validation using the LIBSVM implementation of C-Support Vector Machines (C-SVM) and we compute the classification accuracies. We perform cross-validation on the training data to select the optimal parameters for each kernel and fold. We repeat the experiment 10 times for each kernel and dataset and we report the average classification accuracies and standard errors in Table.2. Note that for some kernels we directly report the best results from the original corresponding papers, since the evaluation of these kernels followed the same setting of ours. For the alternative deep learning methods, we report the best results for the PSGCNN, DCNN, DGK models from their original papers, since these methods followed the same setting of the proposed model. For the AWE model, we report the classification accuracies of the feature-driven AWE, since the author have stated that this kind of AWE model can achieve competitive performance on label dataset. Finally, note that the PSGCNN and ECC models can leverage additional edge features, most of the graph datasets and the alternative methods do not leverage edge features. Thus, we do not report the results associated with edge features in the evaluation. The classification accuracies and standard errors for each deep learning method are also shown in Table.2.

Experimental Results and Discussions: Table.2 indicates that the proposed ASGCN model can significantly outperform either the remaining graph kernel methods or the remaining deep learning methods for graph classification. Specifically, compared with the alternative graph kernel methods, only the accuracies on the ENZYMES, IMDB-M and RED-B datasets are not the best for the proposed model. However, the proposed model is still competitive on the IMDB-M and RED-B datasets. On the other hand, compared with the alternative deep learning methods, only the accuracies on the ENZYMES and IMDB-M datasets are not the best for the proposed model. But the proposed model is still competitive on the IMDB-M dataset.

Overall, the reasons for the effectiveness are fourfold. First, the C-SVM classifier associated with graph kernels are instances of shallow learning methods [29]. By contrast, the proposed model can provide an end-to-end deep learning architecture, and thus better learn graph characteristics. Second, as we have discussed earlier, most deep learning based graph convolution methods cannot integrate the correspondence information between graphs into the learning architecture. Especially, the PSGCNN and DGCNN models need to reorder the vertices and some vertices may be discarded, leading to information loss. By contrast, the associated aligned vertex grid structures can preserve more information of each original graph, reducing the problem of information loss. Third, unlike the proposed model, the DCNN model needs to sum up the extracted local-level vertex features as global-level graph features. By contrast, the proposed model can learn richer multi-scale local-level vertex features. The experiments demonstrate the effectiveness of the proposed model. Finally, as instances of spatially-based GCN models, the trainable parameters of the DGCNN and CNN models are shared for each vertex. Thus, these models cannot directly influence the aggregation process of the vertex features. By contrast, the required graph convolution operation of the proposed model is theoretically related to the classical convolution operation on standard grid structures and can adaptively discriminate the importance between specified vertices.

6 Conclusions

In this paper, we have developed a new spatially-based GCN model, namely the Aligned-Spatial Graph Convolutional Network (ASGCN) model, to learn effective features for graph classification. This model is based on transforming the arbitrary-sized graphs into fixed-sized aligned grid structures, and performing a new developed spatial graph convolution operation on the grid structures. Unlike most existing spatially-based GCN models, the proposed ASGCN model can adaptively discriminate the importance between specified vertices during the process of the spatial graph convolution operation, explaining the effectiveness of the proposed model. Experiments on standard graph datasets demonstrate the effectiveness of the proposed model.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant no. 61602535 and 61503422), the Open Projects Program of National Laboratory of Pattern Recognition (NLPR), the program for innovation research in Central University of Finance and Economics, and the Youth Talent Development Support Program by Central University of Finance and Economics, No. QYP1908. Primary Author: Lu Bai, Email: bailucs@cufe.edu.cn. Corresponding Author: Lixin Cui, Email: cuilixin@cufe.edu.cn. The data/code will be available at `https://github.com/baiuoy/ASGCN_ECML-PKDD2019`.

References

1. Atwood, J., Towsley, D.: Diffusion-convolutional neural networks. In: Proceedings of NIPS. pp. 1993–2001 (2016)
2. Bai, L., Cui, L., Rossi, L., Xu, L., Hancock, E.: Local-global nested graph kernels using nested complexity traces. Pattern Recognition Letters (To appear in 2019)
3. Bai, L., Hancock, E.R.: Depth-based complexity traces of graphs. Pattern Recognition **47**(3), 1172–1186 (2014)
4. Bai, L., Rossi, L., Bunke, H., Hancock, E.R.: Attributed graph kernels using the jensen-tsallis q-differences. In: Proceedings of ECML-PKDD. pp. 99–114 (2014)
5. Bai, L., Rossi, L., Cui, L., Cheng, J., Wang, Y., Hancock, E.R.: A quantum-inspired similarity measure for the analysis of complete weighted graphs. IEEE Transactions on Cybernetics (To appear in 2019)
6. Bai, L., Rossi, L., Zhang, Z., Hancock, E.R.: An aligned subtree kernel for weighted graphs. In: Proceedings of ICML
7. Borgwardt, K.M., Kriegel, H.P.: Shortest-path kernels on graphs. In: Proceedings of the IEEE International Conference on Data Mining. pp. 74–81 (2005)

8. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. CoRR **abs/1312.6203** (2013)
9. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Proceedings of NIPS. pp. 3837–3845 (2016)
10. Duvenaud, D.K., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P.: Convolutional networks on graphs for learning molecular fingerprints. In: Proceedings of NIPS. pp. 2224–2232 (2015)
11. Gammerman, A., Azoury, K.S., Vapnik, V.: Learning by transduction. In: Proceedings of UAI. pp. 148–155 (1998)
12. Henaff, M., Bruna, J., LeCun, Y.: Deep convolutional networks on graph-structured data. CoRR **abs/1506.05163** (2015), `http://arxiv.org/abs/1506.05163`
13. Ivanov, S., Burnaev, E.: Anonymous walk embeddings. In: Proceedings of ICML. pp. 2191–2200 (2018)
14. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: Proceedings of ICML. pp. 321–328 (2003)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. CoRR **abs/1609.02907** (2016), `http://arxiv.org/abs/1609.02907`
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
17. Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. In: Proceedings of ICML. pp. 2014–2023 (2016)
18. Nikolentzos, G., Meladianos, P., Limnios, S., Vazirgiannis, M.: A degeneracy framework for graph similarity. In: Proceedings of IJCAI. pp. 2595–2601 (2018)
19. Rippel, O., Snoek, J., Adams, R.P.: Spectral representations for convolutional neural networks. In: Proceddings of NIPS. pp. 2449–2457 (2015)
20. Shervashidze, N., Vishwanathan, S., T. Petri, K.M., Borgwardt, K.M.: Efficient graphlet kernels for large graph comparison. Journal of Machine Learning Research **5**, 488–495 (2009)
21. Shervashidze, N., Schweitzer, P., van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-lehman graph kernels. Journal of Machine Learning Research **1**, 1–48 (2010)
22. Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of CVPR. pp. 29–38 (2017)
23. Verma, S., Zhang, Z.: Graph capsule convolutional neural networks. CoRR **abs/1805.08090** (2018), `http://arxiv.org/abs/1805.08090`
24. Vialatte, J., Gripon, V., Mercier, G.: Generalizing the convolution operator to extend cnns to irregular domains. CoRR **abs/1606.01166** (2016), `http://arxiv.org/abs/1606.01166`
25. Witten, I.H., Frank, E., Hall, M.A.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann (2011)
26. Yanardag, P., Vishwanathan, S.V.N.: Deep graph kernels. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015. pp. 1365–1374 (2015)
27. Zambon, D., Alippi, C., Livi, L.: Concept drift and anomaly detection in graph streams. IEEE Trans. Neural Netw. Learning Syst. **29**(11), 5592–5605 (2018)
28. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: Proceedings of AAAI (2018)
29. Zhang, S., Liu, C., Yao, K., Gong, Y.: Deep neural support vector machines for speech recognition. In: Proceedings of ICASSP. pp. 4275–4279 (2015)