Neural Computing and Applications
https://doi.org/10.1007/s00521-022-07368-1

REVIEW

Applications of graph convolutional networks in computer vision

Pingping Cao[1] · Zeqi Zhu[1] · Ziyuan Wang[1] · Yanping Zhu[2] · Qiang Niu[1]

Received: 16 November 2021 / Accepted: 27 April 2022

Abstract

Graph Convolutional Network (GCN) which models the potential relationship between non-Euclidean spatial data has attracted researchers' attention in deep learning in recent years. It has been widely used in different computer vision tasks by modeling the latent space, topology, semantics, and other information in Euclidean spatial data and has achieved significant success. To better understand the work principles and future GCN applications in the computer vision field, this study reviewed the basic principles of GCN, summarized the difficulties and solutions using GCN in different visual tasks, and introduced in detail the methods for constructing graphs from the Euclidean spatial data in different visual tasks. At the same time, the review divided the application of GCN in basic visual tasks into image recognition, object detection, semantic segmentation, instance segmentation and object tracking. The role and performance of GCN in basic visual tasks were summarized and compared in detail for different tasks. This review emphasizes that the application of GCN in computer vision faces three challenges: computational complexity, the paradigm of constructing graphs from the Euclidean spatial data, and the interpretability of the model. Finally, this review proposes two future trends of GCN in the vision field, namely model lightweight and fusing GCN with other models to improve the performance of the visual model and meet the higher requirements of vision tasks.

Keywords Graph convolution network · Non-Euclidean space · Relational modeling · Computer vision

1 Introduction

Convolutional Neural Network (CNN) with powerful modeling capacities [1, 2] can extract and process effective data representation from Euclidean data. Therefore, it has been commonly used to process computer vision tasks and significant improvement in computer vision has been made. Despite the great success of CNNs, they are difficult to encode the intrinsic graph structures for the specific learning tasks [3], therefore the potential spatial, topological, and semantic information in the image are ignored.

To consider the correlation between the potential information in the image and the task, Gong et al. [4] proposed a ranking based learning strategy to train deep CNN. Wang et al. [5] used Recurrent Neural Networks (RNNs) to convert image labels into embedded label vectors, and the correlation between labels can be considered. On the other hand, the attention mechanism has been widely used to model latent relationships in the image. Zhu et al. [6] proposed a spatial regularization network based on the weighted attention maps to capture images' semantics and spatial relationships. Wang et al. [7] introduced a spatial transformation layer and a long short-term

P. Cao, Z. Zhu, Z. Wang and Y.Zhu have contributed equally to this work.

✉ Qiang Niu
niuq@cumt.edu.cn

Pingping Cao
cpp@cumt.edu.cn

Zeqi Zhu
12201318@cumt.edu.cn

Ziyuan Wang
12201323@cumt.edu.cn

Yanping Zhu
yz6d7@mst.edu

1 School of Computer Science and Technology, China University of Mining and Technology, Tongshan, Xuzhou 221006, Jiangsu, China

2 Department of Civil Architectural and Environmental Engineering, Missouri University of Science and Technology, 500 W. 16th Street, Rolla, MO 65409, USA

Published online: 26 May 2022

memory (LSTM) unit to capture the image label correlation.

In addition to the above mentioned structural methods, many graph methods have been used for correlation modeling. Li et al. [8] generated an image-dependent conditional label structure in the graphic lasso framework. Li et al. [9] established a tree structure graph in the label space by using a maximum spanning tree algorithm. Lee et al. [10] used knowledge graphs to describe the relationship among multiple labels. In 2019, Kuangshi proposed a multi-label image recognition method based on Graph Convolutional Neural Network (GCN) [11], which completely opened the door of computer vision for GCN.

GCN was proposed by Bruna et al. [12] in 2013. As a new convolutional network structure will make convolution usually used in the deep learning apply to graph data. Since Kipf et al. [13] first applied GCN for solving the semi-supervised classification problem in 2017, it has become a research hotspot in the deep learning. The GCN model is a neural network architecture that uses the graph structure to gather node information from the neighbor in convolution. GCN has a strong expressive ability of learning graph representation and has achieved excellent performance in different tasks and applications [3].

At present, GCN is mainly used to solve non-Euclidean spatial data problem. By encoding structural information of the non-Euclidean spatial data, the relationship between entities is modeled and the potential relationship between data is mined. With the continuous development of GCN, it has been applied to many fields, such as network analysis [14–22], recommendation system [23–26], traffic prediction [27–29], biochemistry [30–35], natural language processing [36–41] and computer vision [42–46].

Some reviews have been made on GCN recently. Zhang et al. [3] conducted a detailed review on GCN, including many existing GCN variants except basic GCN and focusing on the convolution operation defined on the graph. Jie et al. [47] introduced different computing modules in GCN in more details, such as propagation module, skip connection, and pool operation. References [48–50] are the latest reviews on GCN in recent years. Zhang et al. [48] divided GNN into four categories: cyclic GCN, convolutional GCN, graph self-encoder and spatiotemporal GCN. Wu et al. [49] mainly summarized different graph depth learning methods. Reference [50] mainly unified the network embedding model and GNN model. The above summary work primarily focuses on the GCN model. Although some work [3, 47, 49] also introduces the GCN application in computer vision, it is only briefly mentioned, not comprehensive and specific. Reference [3] introduced the application of GCN in image classification and action recognition. Reference [47] described the application of GCN in image classification, visual reasoning, and semantic segmentation with few (zero) samples. Reference [49] pointed out that GCN can be applied in the tasks of scene map generation, point cloud classification and action recognition. However, in these reviews, the in-depth discussion on the role, specific application methodology and effectiveness of GCN in basic vision tasks is still lacking. Different from the above work, in order to more comprehensively and deeply introduce the role and application prospect of GCN in the field of computer vision, this study summarizes the basic principle of GCN, and difficulties and solutions of GCN encountered in the visual tasks. Also, the role of GCN in different visual tasks, the methods for constructing graph from the Euclidean spatial data in different visual tasks, and the performance comparison with traditional CNN methods are discussed.

An overview of this article organization is shown in Fig. 1. Section 2 introduces the application background of GCN, including the working principle of GCN (Sect. 2.1) as well as the difficulties and solutions using GCN in the computer vision field (Sect. 2.2). In Sect. 3, the methods for constructing graph from Euclidean spatial data such as image or video for image recognition, object detection, semantic segmentation, instance segmentation and object tracking are analyzed in detail. Section 4 takes the constructed graph data as the input of GCN, introduces the role of GCN in different visual tasks, and compares the performance with the traditional CNN methods. Section 5 presents the challenges and future research opportunities of GCN applied in several basic visual tasks. In conclusion, we have made the following contributions:

1. We comprehensively summarize the basic principle of GCN, the difficulties and solutions of GCN encountered, the role of GCN, the methods for constructing graph from Euclidean spatial data in different visual tasks (i.e., image recognition, object detection, semantic segmentation, instance segmentation and object tracking), and the performance comparison with the traditional CNN methods.
2. We propose three challenges for the GCN application in computer vision: computational complexity, the paradigm of constructing graph from Euclidean spatial data, and the interpretability of the GCN model.
3. We also identify the future trend of GCN application in computer vision, including model lightweight and fusing GCN with other models, to improve the visual model performance and meet the higher requirements of visual tasks. This study will provide new research ideas and opportunities for the GCN application in the computer vision field.

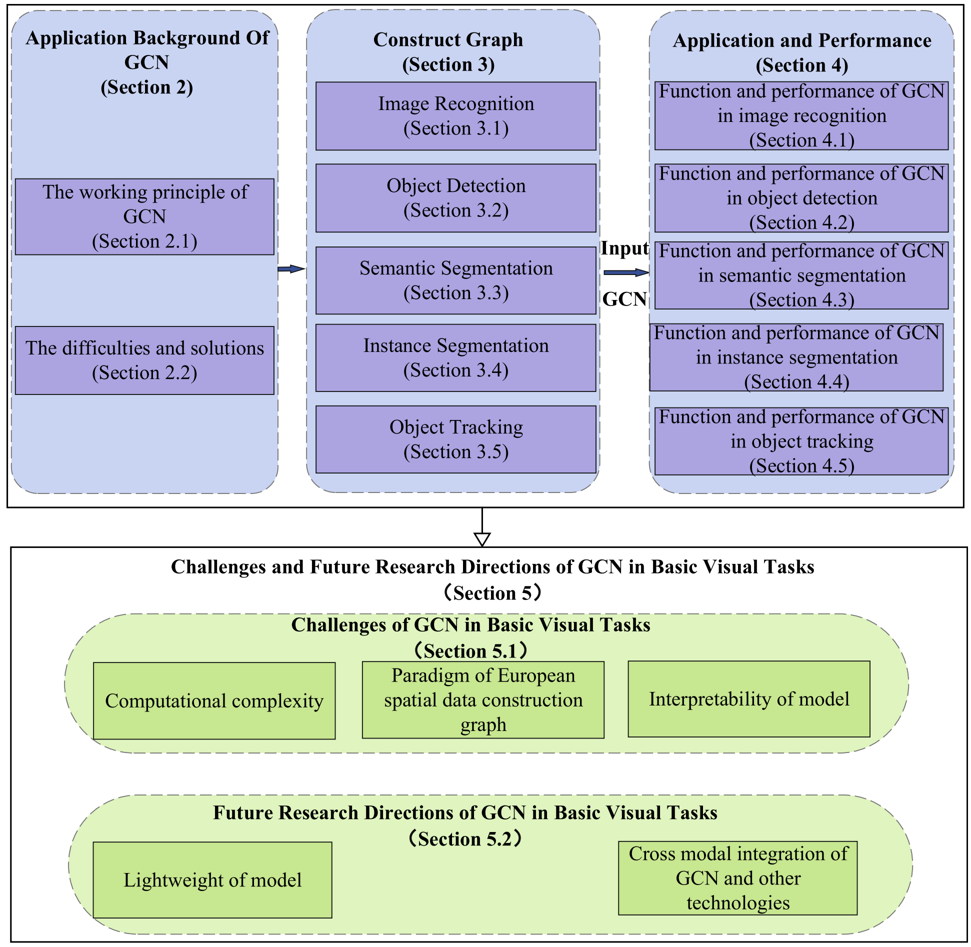


Fig. 1 Overview of this article organization

2 Application background of GCN in the field of vision

2.1 Working mechanism of GCN

This review introduces and summarizes the working principles of GCN in detail based on the graph filter. The graph filter is defined as an operation of enhancing or attenuating the frequency component strength of the graph signal. Let the graph filter be: $H \in \mathbb{R}^{N\times N}, H : \mathbb{R}^N \to \mathbb{R}^N$,and the output graph signal is y, for a given input graph signal, then:

$$y = Hx, \tag{1}$$

According to the polynomial expansion of the Laplace matrix on the graph filter, H can be expressed as: $\sum_k^K h_k L^k$, where K is the order of filter H , $h(\cdot)$ is the frequency response function of H , L is the corresponding Laplace matrix. The graph filter can be understood from both the spatial and frequency domain.

2.1.1 Spatial domain

For $y = Hx$, it can be expressed as: $y = \sum_k^K h_k L^k x$, let $x^{(k)} = L^{(k)} x = Lx^{(k-1)}$, then:

$$y = \sum_{k=0}^{K} h^k x^{(k)}. \tag{2}$$

2.1.2 Frequency domain

Since $L = V \Lambda V^T$, then:

$$
\begin{aligned}
H &= \sum_{k=0}^{K} h_k L^k = \sum_{k=0}^{K} h_k \left(V\Lambda V^T\right)^k \\
&= V\left(\sum_{k=0}^{K} h_k \Lambda^k\right) V^T \\
&= V \begin{bmatrix} \sum_{k=0}^{K} h_k \lambda_1^k & & \\ & \ddots & \\ & & \sum_{k=0}^{K} h_k \lambda_N^k \end{bmatrix} V^T.
\end{aligned} \tag{3}
$$

When the filter is used for filtering,

$$
y = Hx = V\left(\sum_{k=0}^{K} h_k \Lambda^k\right) V^T x. \tag{4}
$$

On the basis of understanding the graph filter, the convolution operation in graph signal processing is defined. Given the graph signals x_1 and x_2 on two groups of graph (G), the definition of graph convolution operation isas follows:

$$
x_1 * x_2 = \mathrm{IGFT}(\mathrm{GFT}(x_1) \odot \mathrm{GFT}(x_2)), \tag{5}
$$

where $*$ represents convolution operation, $\odot$ represents Hadamard product, $\mathrm{GFT}(\cdot)$ and $\mathrm{IGFT}(\cdot)$ represent Graph Fourier Transform and Inverse Graph Fourier transform, respectively. It is worth noting that convolution in the spatial domain is equivalent to multiplication in the frequency domain for this definition.

The derivation of formula (6) can be transformed into:

$$
\begin{aligned}
x_1 * x_2 = V((V^T x_1) \odot (V^T x_2)) &= V(\widetilde{x_1} \odot (V^T x_2)) \\
&= V(\mathrm{diag}(\widetilde{x_1})(V^T x_2)) \\
&= (V\mathrm{diag}(\widetilde{x_1})V^T)x_2,
\end{aligned} \tag{6}
$$

where $\widetilde{x}$ is the coefficient of Fourier transform. Let $H_{\widetilde{X_1}} = Vdiag(\widetilde{x_1})V^T$, $H_{\widetilde{X_1}}$ can be regarded as a graph displacement operator, and its frequency response matrix is the spectrum of x_1 , from which we can get:

$$
x_1 * x_2 = H_{\widetilde{X_1}} x_2. \tag{7}
$$

In view of the success of the convolutional neural network in computer vision, the graph convolution operation defined above can also be naturally extended to visual data.

The basic idea of GCN is to update the node representation by propagating information between nodes. The general GCN structure consists of the input layer, convolution layer, full connection layer, and output layer. Reference [51] gives an example (i.e., the node classification task): taking the word bag as the input, extracting the features on the graph through the convolution layer, then classifying the graph through the full connection layer, and finally outputting each node's label in the output layer to realize the node classification, as shown in Fig. 2.

The input of GCN is a graph structure (G) . In the convolution layer, first learning the functions $f(\cdot,\cdot)$ on graph G , and then performing downsampling and pooling operations on the graph to complete feature extraction. Where $f(\cdot,\cdot)$ takes the feature description $H^l \in \mathbb{R}^{n\times d}$ and the corresponding correlation matrix $A \in \mathbb{R}^{n\times n}$ as the input (n represents the number of nodes and d represents the dimension of node features), and updates the node to $H^{l+1} \in \mathbb{R}^{n\times d'}$. Each GCN layer can be represented by a nonlinear function:

$$
H^{l+1} = f(H^l, A). \tag{8}
$$

According to the convolution operation in [12], Eq. (8) can be expressed as:

$$
H^{(l+1)} = \sigma(\widehat{A} H^l W^l), \tag{9}
$$

In Eq. (9), $W^l \in \mathbb{R}^{d\times d'}$ is the transformation matrix to be learned, $\widehat{A} \in \mathbb{R}^{n\times n}$ is the normalized form of the correlation matrix A , and $\sigma(\cdot)$ represents the nonlinear activation function.

In the GCN structure, the full connection layer is used to integrate the local information with category discrimination in the convolution layer, and the last layer of the full connection layer is used to classify the graph. Finally, the label of each node is output in the output layer to complete the node classification.

2.2 Existing difficulties and solutions of GCN in the field of computer vision

In the existing overview of GCN [47–50], the challenges of building GCN are mainly divided into the following aspects:

1. Graph data are non-Euclidean spatial data. The graph data do not meet the translation invariance, and each point has different local structures. However, the basic operators in traditional CNN (i.e., convolution and pooling) depend on the translation invariance of data.
2. Graph data are diverse with various characteristics. Many applications in real life can be naturally represented by graph data. Various graph features bring more information for constructing GCN, but modeling of multiple features requires more complex and more delicate GCN design, which brings new challenges.
3. The scale of graph data is enormous. In the big data era, the extensive graph in the practical application may include millions or tens of millions of nodes.

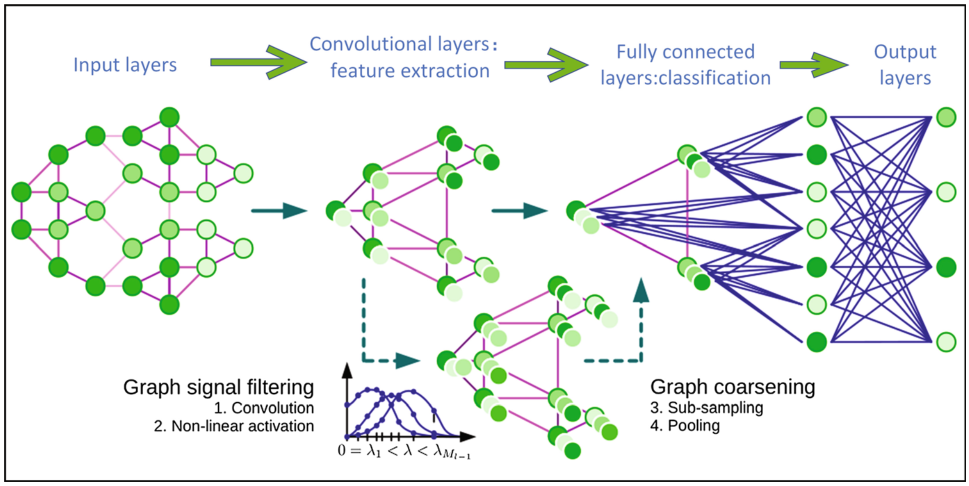


Fig. 2 Structure of GCN

Building GCN on a large-scale graph within the acceptable time and space ranges is also challenging.

With the maturity of the GCN technology, it has been gradually used in various fields and remarkable results using GCN have been achieved. However, the GCN application in computer vision still has some limitations and faces challenges different from the GCN construction. At present, the overview of GCN has not discussed this problem. This review summarizes the challenges using GCN in the computer vision field:

1. Visual data are Euclidean spatial data. Traditional visual data is image or video data basically. Image and video data are Euclidean spatial data with translation invariance, that is, the local structures of each node are the same. GCN can better show modeling ability in non-Euclidean spatial data. How to mine the non-Euclidean structure in Euclidean spatial data (shown in Fig. 3) has become a main challenge for applying GCN in the vision field. Fig. 3 (left) is our common Euclidean data, which has regular local node features, while GCN requires irregular topological features between nodes as input to infer the relationship between node features, as shown in Fig. 3 (right). How to transform the left Euclidean data into the right non-Euclidean data is an important challenge for GCN in visual application.
2. Overfitting is easy to occur. Using GCN models for different tasks is easy to produce gradient disappearance, over-smoothing, and overfitting problems. Although GCN models can achieve excellent modeling results, they are usually limited to very shallow models due to the gradient disappearance problem. Figure 4 shows an GCN application example (i.e., node classification on dataset Cora [52]) to verify the layer depth effect on the accuracy. Overfitting weakens the generalization ability of small data sets, while over-smoothing due to the increase of network depth separates the output representation from the input features, hinders the model's training and reduces the model's accuracy.

To solve the first challenge, many methods for constructing graph from Euclidean spatial data have been proposed with the deep learning technology development. The different methods for constructing graph are classified

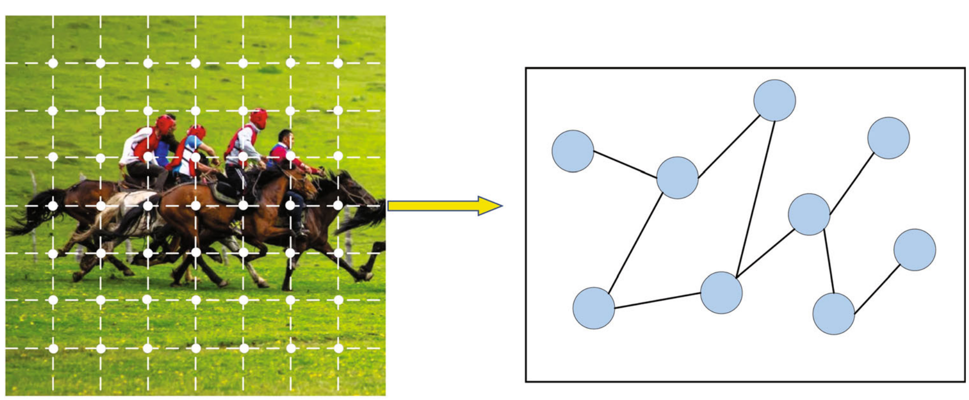


Fig. 3 The non-Euclidean structure in Euclidean spatial data

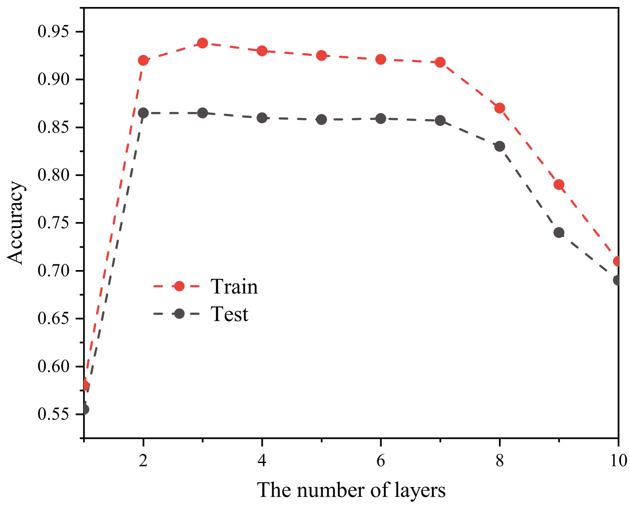


Fig. 4 Accuracy of GCN models with different number of layers for node classification

in Table 1. In the following sections, we will discuss them in detail according to specific visual tasks.

Aiming at the second challenge, Yu et al. [61] proposed a DropEdge technology to alleviate the over-smoothing and overfitting problems. The core of this technology is to randomly remove a certain number of edges from the input graph in each training period, just like data enhancer and message passing reducer. The results show that the DropEdge either reduces the convergence speed of over-smoothing or minimizes the information loss caused by over-smoothing. In addition, the latest work [62] proposed a variant GCN $\prod$. The GCN $\prod$ includes two simple and effective techniques: initial residual and identity mapping, which significantly alleviates the over-smoothing problem. To solve the gradient disappearance problem of deep GCN, Li et al. [63] embedded residual connection, dense connection, and expansion convolution into a GCN architecture, successfully constructed the deep GCN with 56 layers, and achieved remarkable results in the point cloud semantic segmentation task.

3 Graph construction in basic visual tasks

The GCN application in the computer vision field has certain limitations due to the challenge of the GCN application in Euclidean spatial data. These limitations result in that the GCN application mainly focuses on small sample learning, zero sample learning, point cloud modeling and scene graph. With the development of GCN technology, the related challenges of Euclidean spatial data encountered by GCN in the vision field have been solved. GCN can be used to deal with more visual tasks. By referring to the relevant work of existing GCN in the basic vision field (as shown in Table 2), we will make a comprehensive and in-depth summary of the application of GCN in five basic tasks: image recognition, object detection, semantic segmentation, instance segmentation and object tracking.

To apply GCN in the visual tasks, the image data needs to be constructed into a graph first, and the Euclidean spatial data needs to be transformed into non-Euclidean spatial data. Then, the data relationship is described or the node feature information is updated by GCN to complete the visual task.

3.1 Conversion of image or video data to graph structure in image recognition

In the existing GCN based image recognition, the probability and region attention methods are mainly used for realizing the graph construction of an image.

Graph construction based on probability method: the mathematical representation of the graph is: $G = (V, E)$, where V represents node and E represents the edge between nodes. In the probability method based graph construction, the label is generally used as the graph node by using the word embedding, and then the correlation matrix between nodes is calculated by the probability statistics method.

The word embedding is a distributed representation based on a neural network, also known as the Word Vector. The core of the word embedding is the context representation and relationship modeling between the context and target word. The word embedding methods can be roughly divided into word frequency-based embedding and prediction-based embedding. At present, the more popular word frequency-based embedding methods include Count Vector, Term Frequency−Inverse Document Frequency (TF_IDF) vector and Co-Occurrence Vector. However, the prediction-based embedding method-word2vec is commonly used and effective in the practical application. The word2vec is a combination of Continuous bag of words and

Table 1 Classification of graphic construction methods for Euclidean Spatial Data

Graph construction methods	Related studies
Probability method	CGL [8], Li [9], ML-GCN [11], KSSNet [53]
Region attention	SRN [6], Wang [7], ADD-GCN [54], 2S-DGCN [43]
Prior knowledge	Akata [55], ZSL [56], Romera-Paredes [57], Frome [58], Li [59], DGP [60]

Table 2 Application of GCN in basic visual tasks

Visual tasks	GCN based model
Image recognition	ML-GCN [11], ADD-GCN [54], 2S-DGCN [43], KSSNet [53], Sequential-GCN [64]
Object detection	Chai [65], DG-FPN [66], Edge-Gated-GCN [67], SAT-GCN [68], SIN [69], RepGN [70]
Semantic segmentation	SPG [71], 3DGNN [72], RGCNN [73], SketchGCN [74], Graph-FCN [75], GMNet [45]
Instance segmentation	BCNet [76], Curve-GCN [77], DACN [78], Split-GCN [79], Pixel2mesh [80]
Object tracking	SiamGAT [81], Graph Networks for MOT [82], Neural-Solver [83], EDA-GNN [84]

Skip-gram algorithm models (CBOW). About the word embedding, Shi et al. [85] made a detailed introduction for readers' interest.

In the probability-based graph construction method, the correlation matrix between nodes is calculated by the probability statistics. To build the correlation matrix, the label co-occurrence possibilities of the training data are computed first to obtain the correlation matrix $M \in \mathbb{R}^{C \times C}$, where C is the number of categories, and M_{ij} represents the number of times L_i and L_j appearing at the same time. Then, the conditional probability matrix is calculated according to the tag co-occurrence matrix:

$$P_i = M_i / N_i, \tag{10}$$

where N_i represents the number of appearing times of label L_i in the training set. The calculated probability matrix is the correlation matrix of the constructed graph, and the subsequent improvement of the probability matrix is derived on this basis.

Graph construction based on region attention: in the image recognition, the regional attention based graph construction steps are as follows: the input feature map is converted into an output map with fixed size, and then the feature map X' is converted into the corresponding attention area map through conversion modules such as Spatial Transformer (ST) [86] or Semantic Attention Module (SAM) [54]. Finally, the corresponding graph structure is established with the region attention content as node V , as shown in Fig. 5.

3.2 Conversion of image or video data to graph structure in object detection

In the object detection task, in addition to the graph construction methods based on probability and regional attention mentioned in the image recognition task, the graph construction method based on super- pixel is also used in this visual task.

Graph construction based on super pixel: the input frame is modeled as a group of super pixels by using the super-pixel segmentation algorithm; then, the super-pixel level spatial-temporal graph $G_{t,t+1,t+2} = (V_{t,t+1,t+2}, \varepsilon_{t,t+1,t+2})$ composed of three frame pairs (F_t, F_{t+1}), (F_t, F_{t+2}) and F_{t+2}; finally, the edges of the original graph are generated by measuring the Euclidean distance between nodes. Fig. 6a shows each node connected to its nearest neighbor K in the same frame or across multiple frames according to the spatial coordinate distance between super-pixel pairs.

The coordinates of each super-pixel are calculated by averaging coordinates of all pixels involved in the same super-pixel. By directly measuring the Euclidean distance, the spatial distance of super-pixel $\left(V_i^t, V_j^t\right)$ in the same frame can be calculated. To calculate the spatial coordinates between the super pixels belonging to different frames, such as $\left(V_i^t, V_j^{t+1}\right)$, it is necessary to map the spatial coordinates of the super pixels d from F_t to F_{t+1} frames, as shown in Fig. 6b, which can be obtained by the operation of Eq. (11):

$$coord_i^{t \to t+1}(x,y) = coord_i^t(x,y) + \Delta coord_i^{t,t+1}(x,y), \tag{11}$$

where $\Delta coord_i^{t,t+1}(x,y)$ represents the coordinate displacement inferred from the optical flow [87] information in the frame pair, and $coord_i^{t \to t+1}(x,y)$ represents the coordinates of the super-pixel V_i^t mapped from frame F_t to frame F_{t+1}.

3.3 Conversion of image or video data to graph structure in semantic segmentation

In addition to the graph construction methods used in image recognition and object detection, the image pixels of are also used as nodes for the graph construction in the semantic segmentation task. The image can be represented as a graph model $G(N, E)$, where N represents the graph node, and E represents the edge of the graph. The node construction is shown in Fig. 7. The dissimilarity between nodes can be regarded as the graph edge, and the edge is affected by the adjacent matrix in the graph model.

In addition, there are other methods for constructing graphs in the semantic segmentation task [45, 71–73]. Landrieu et al. [71] took edge features as nodes, and

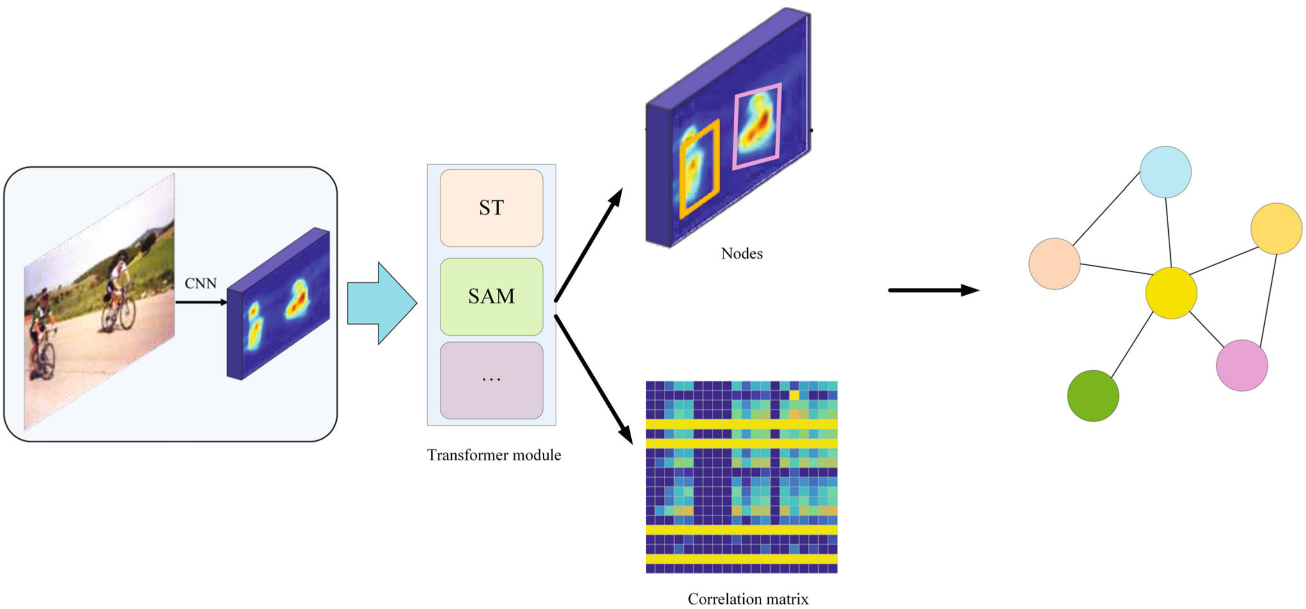


Fig. 5 Graph construction method based on region attention

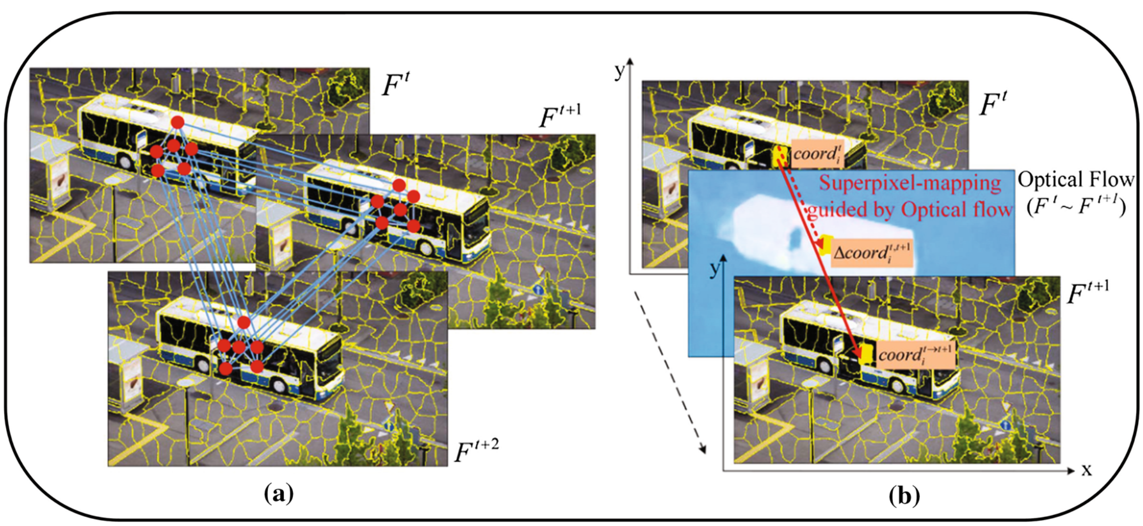
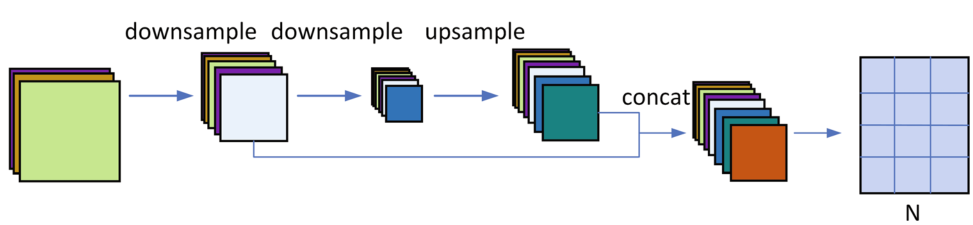


Fig. 6 Super pixel-based graph construction

Fig. 7 Construction process of graph nodes



encoded the contextual relationship between object parts in 3D point clouds. A k-nearest neighbor graph on top of 3D point cloud was built by Qi et al. [72] Each node in the graph represented a set of points and was related to a hidden representation vector initialized with an appearance feature extracted by CNN from 2D images. Te et al. [73] directly regarded the characteristics of points in the point cloud as signals on the graph. Michieli et al. [45] modeled the relationship between the detected object and the object as the node of the graph and the edge of the graph. The application of GCN in semantic segmentation task is realized through the above construction graph methods.

3.4 Conversion of image or video data to graph structure in instance segmentation

In the existing work of applying GCN to realize instance segmentation, the conversion methods from image or video data to graph structure mainly include the following two categories:

(1) Each pixel on the feature graph is taken as the graph nodes, and the adjacency matrix between nodes is obtained through feature similarity. Generally, the feature similarity is obtained by Eq. (12).

$$Sim(x_i, x_j) = \phi(x_i)^T \phi'(x_j), \tag{12}$$

where x_i and x_j represent the object proposal feature, which are d-dimensional vectors, and ϕ and ϕ' represent two different transformations of the original features. The graph structure for instance segmentation can be constructed by graph nodes and adjacency matrix.

(2) The shape of the target object in the image or video is represented by N control points, the initial graph structure is formed by connecting the control points, and the constructed graph structure is input into GCN to predict the shape and position of the target object in the image or video.

3.5 Conversion of image or video data to graph structure in object tracking

Reference [81] constructed a graph for object tracking. Each grid of each frame feature map was a node, and the similarity between node features was an edge; References [82, 83] took the object and detection as nodes, and constructed the graph edge by using the relationship between the object and detection. Reference [84] took the object feature as the node and calculated the appearance affinity and motion affinity of the object in the video as the graph edge. After constructing a graph for object tracking task, graph network plays a significant role in object tracking and greatly improves the model performance.

4 Application and performance of GCN in basic visual tasks

4.1 Function and performance of GCN in image recognition

Image recognition is the most basic task in the computer vision field. It plays a vital role in object detection [88, 89], human attribute recognition [90], medical image recognition [91] and recommendation system [92, 93]. Although CNN has achieved good results in image recognition, it ignores the spatial and semantic relationship between image labels when modeling labels. To better model the relationship between image labels, GCN is introduced into the image recognition task. GCN plays two main roles in the image recognition task: (1) a classifier and (2) propagating and updating image information.

4.1.1 GCN as a classifier

As shown in Fig. 8, GCN is used to generate a classifier for the input image features to realize image recognition in the image recognition task. Specifically, GCN learns interdependent object classifiers from the input graph, expressed as: $W = \{w_i\}_{i=1}^{C}$, where C represents the number of categories. The learned classifier is applied for image representation, and the prediction score is as follows:

$$\hat{y} = Wx. \tag{13}$$

Let the truth image labels be $y \in \mathbb{R}^C$ and $y^i = \{0, 1\}$ indicating whether label i exists in the image. The traditional multi-label classification loss is used for training the image recognition model, and the loss is as follows:

$$L = \sum_{c=1}^{C} y^c \log(\sigma(\hat{y}^c)) + (1 - y^c) \log(1 - \sigma(\hat{y}^c)), \tag{14}$$

where $\sigma(\cdot)$ is the sigmoid activation function.

4.1.2 Image information updating based on GCN

As described in the basic idea of GCN, it updates node information and propagates information between nodes. In the image recognition, the GCN output aggregates and updates the image information. The output features include semantic and spatial information, which provides more feature information for image recognition and improves image recognition accuracy. The workflow of GCN-based image information updating in the image recognition is shown in Fig. 9.

According to the input image, the graph can be constructed. Each node of the graph contains image information. The node information is propagated after inputting the

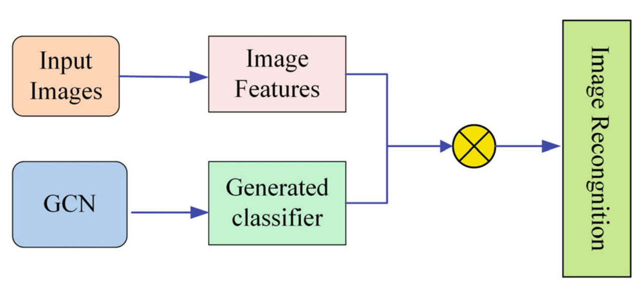


Fig. 8 Image recognition method based on GCN classifier

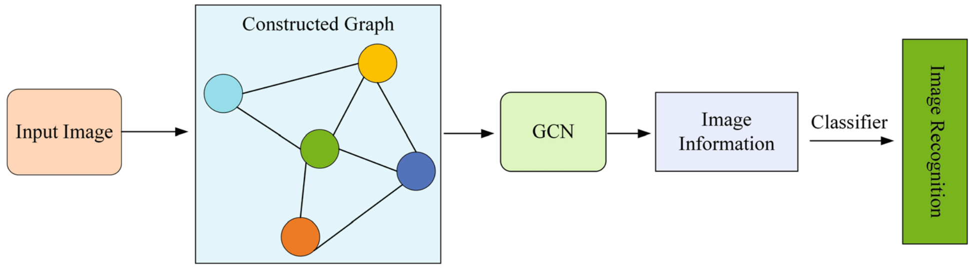


Fig. 9 GCN-based image information update in image recognition

graph into GCN. For the input node V , the GCN layer is updated as follows:

$$Z = f(\mathrm{AVW}), \tag{15}$$

where Z represents the updated node; A is the correlation matrix of the input graph; W is the state update weight learned in GCN; $f(\cdot)$ is the activation function, and it is generally *LeakyReLU* function in image recognition.

Because each vector output by GCN is aligned with its specific class and contains rich relationship information with other vectors, category score can be predicted by putting each vector into a binary classifier to achieve the image recognition task.

In the image recognition, whether GCN is used for a classifier or updating image information, the introduction of GCN can improve the performance of image recognition and achieve better results. As shown in Table 3, the traditional CNN methods continuously improve the baseline of COCO2014. After introducing GCN, the baseline is continuously refreshed to the SOTA. The average of Overall/Per-Class Precision (OP/CP),Overall/Per-Class recall (OR/CR), Overall/Per-Class F1-score (OF1/CF1) and the mean Average Precision (mAP) in the table is the evaluation index of model performance. The higher the index value, the better the model performance.

"↑ " in the table indicates the increased inaccuracy. It can be seen from the table that the image recognition methods based on CNN and GCN have improved the recognition baseline of COCO2014, where DecoupleNet [95] has the highest accuracy based on CNN, while the accuracy based on GCN is higher than DecoupleNet [95], and 2S-DGCN [43] reaches the SOTA.

4.2 Function and performance of GCN in object detection

Due to the rapid deep learning development, great breakthroughs have been made in object detection. To further improve the object detection performance, GCN is introduced into the object detection.

At present, studies on the object detection using GCN are lacking. The current GCN application in this aspect is mainly divided into 2D, 3D and salient object detection. Since few existing salient object detection studies using GCN are published, we only introduce the GCN application in 2D and 3D object detection in detail.

4.2.1 2D object detection

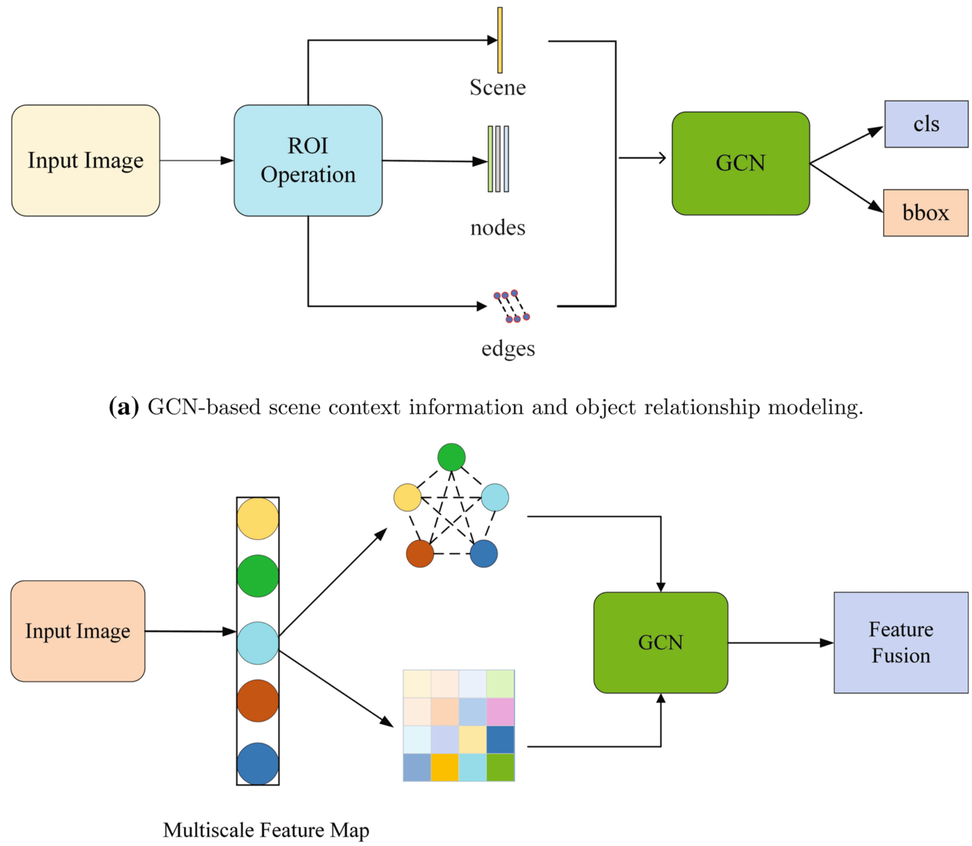
In the existing 2D object detection work, GCN is used for modeling the scene context information and object relationships in a single image (as shown in Fig. 10a or for fusing input image features (as shown in Fig. 10b to improve the object detection performance.

In Fig. 10a, the input image constructs a graph $G = (V, E, s)$ containing nodes, edges and scenes through a series of ROI (region of interest) operations. Taking scene and node, edge and node as the input of GCN, respectively,

Table 3 Comparison results with traditional CNN methods on public dataset COCO 2014

Method		All						
Baseline		mAP 47.4	CP 59.0	CR 57.0	CF1 58.0	OP 60.2	OR 62.1	OF1 61.1
CNN	SRN [5]	62.0↑	65.2↑	55.8↑	58.5↑	75.5↑	71.5↑	73.4↑
	ResNet-101 [94]	79.7↑	82.7↑	67.4↑	74.3↑	86.4↑	71.8↑	78.4↑
	DecoupleNet [95]	82.2↑	83.1↑	71.6↑	76.3↑	84.7↑	74.8↑	79.5↑
GCN	ML-GCN [11]	83.0↑	85.1↑	72.0↑	78.0↑	85.8↑	75.4↑	80.3↑
	KSSNet [53]	83.7↑	84.6↑	73.2↑	77.2↑	87.8↑	76.2↑	81.5↑
	ADD-GCN [54]	84.9↑	85.1↑	75.1↑	79.8↑	85.7↑	78.3↑	81.8↑
	2S-DGCN [43]	85.6↑	84.9↑	75.7↑	80.0↑	86.8↑	78.0↑	82.2↑

Fig. 10 2D object detection based on GCN



(a) GCN-based scene context information and object relationship modeling.

(b) Feature fusion based on GCN.

the object category and position in the image are recognized to realize object detection through the transmission and aggregation of scene context information and the relationship between object pairs.

In Fig. 10b, the relationship between multi-scale feature map and pixel matrix is modeled by GCN to find the optimal fusion strategy for feature fusion. In GCN, the propagation between feature nodes is shown in Eq. (16):

$$C_n^{s+1} = \delta(\widehat{A}(C_n^s W^S)), \tag{16}$$

where $C_N^s = \{C_1^s, C_2^s, \cdots, C_n^s\}$ represents the characteristic node in the s-th fusion stage; $\widehat{A}$ is the normalized form of adjacency matrix A ; W^S is the parameter matrix; $\delta(\cdot)$ is the nonlinear activation function.

In multi-scale feature fusion, GCN learns the optimal feature fusion strategy for object detection. The fused features are used for object detection, which helps to improve the object detection model performance.

4.2.2 3D object detection

The GCN application in 3D object detection is processing and modeling of point cloud data. Point cloud can be regarded as an undirected graph. GCN is used to maintain the points' spatial relationship and extract the local area information of the point cloud.

In the 3D object detection, the asymmetric edge function combining shape structure and domain information is used to extract the edge feature $e'_{i,j,m}$ of vertices:

$$e'_{i,j,m} = \mathrm{ReLU}\left(\theta_m \cdot \left(x_{i,j} - x_i\right) + \phi_m \cdot x_i\right), \tag{17}$$

where θ_m and ϕ_m are learnable parameters; $m = 1, \ldots M$ is the index of M-dimensional features; $x_{i,j}$ represents the feature of vertex $v_{i,j} \in N_i$.

Then, the linear projection is used to generate query Q and key K in one-dimensional feature space, and matrix multiplication and activation function are used to generate attention graph, which can be interpreted as a correlation matrix (as shown in Fig. 11).

In Fig. 11, $Q_i = \left[Q_{i,1}, \ldots, Q_{i,k}\right]^T$, $K_i = \left[K_{i,1}, \ldots, Q_{i,k}\right]^T$, and:

$$Q_{i,j} = \sum_{m=1}^{M} \alpha_m e'_{i,j,m}, K_{i,j} = \sum_{m=1}^{M} \beta_m e'_{i,j,m}, \tag{18}$$

$$V_i = Q_i \times K_i^T, \tag{19}$$

$$A_i = V_i \times E'_i. \tag{20}$$

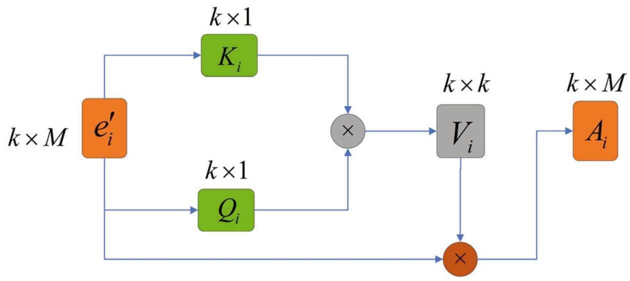


Fig. 11 Generation of attention graph

Where $\vec{\alpha} = [\alpha_1, \ldots, \alpha_m]^T$ and $\vec{\beta} = [\beta_1, \ldots, \beta_m]^T$ are learnable parameters, and $E'_i = [e'_{i,j,m}] \in R^{k \times M}$ is the matrix representation of K vertices with M-dimensional features.

Then, aggregate the point cloud features:

$$x'_{i,m} = \max_{j=1,\ldots,k} S_{i,j,m}, \tag{21}$$

where $S_{i,j} = \sigma_{i,j} \cdot A_{i,j}$.

Finally, GCN uses the aggregated features as an output for subsequent object detection.

Table 4 compares the object detection methods based on GCN and CNN in terms of model category, context information, and suggestion relationship. GCN provides an end-to-end detection model for object detection, which reduces the loss of model and engineering complexity to a certain extent.

4.3 Function and performance of GCN in semantic segmentation

Regardless of 3D point cloud data or image grid data, GCN is mainly used for propagating graph feature representation and category classification in semantic segmentation.

4.3.1 Propagating feature representation of graph

The core of GCN is graph convolution. To define a meaningful translation operator in the vertex domain, reference [73] conducted starts with the filtering of the graph signal in the spectral domain, and then deployed Chebyshev approximation to reduce computing complexity. According to the spectral filtering of the graph signal, the spectral filtering of the graph signal x through g_θ is as follows:

$$y = g_\theta(L)x = g_\theta\left(U \Lambda U^T\right)x = U g_\theta(\Lambda) U^T x, \tag{22}$$

where U is the eigenvector set of the Laplace matrix.

Defferrard et al. [51] used truncated Chebyshev polynomials to approximate the spectral filtering. The K-localized filtering operation is described as follows:

$$y = g_\theta(L)x = \sum_{k=0}^{K-1} \theta_k T_k(L)x, \tag{23}$$

In Eq. (23), θ_k is the K-th Chebyshev coefficient; $T_k(L)$ is the K-order Chebyshev polynomial, which is calculated by $T_k(L) = 2LT_{k-1}(L) - T_{k-2}(L)$ loop, and $T_0(L) = 1, T_1(L) = L$.

After the graph convolution, the graph's feature representation is propagated by the weight matrix, deviation and activation function. The graph node information is updated, as follows:

$$y = \mathrm{ReLU}(g_\theta(L)xW + b), \tag{24}$$

where W is the weight matrix of the trained network; b is the deviation; the activation function is *ReLU* .

4.3.2 Using GCN for category classification

In the image semantic segmentation, GCN is used to classify the graph model nodes. Semantic segmentation can be treated as a node classification problem, as shown in Fig. 12.

In Fig. 12, the node feature is first propagated and updated for the established graph in GCN. The updated feature can be expressed as:

$$Z = soft\max\left(\hat{A}\mathrm{ReLU}\left(\hat{A}XW^{(0)}\right)W^{(1)}\right), \tag{25}$$

Table 4 Comparison of models based on GCN and CNN in object detection

Method		Model	Context information	Proposals relationship	Loss	Complexity
CNN	R-CNN [96]	Two-stage	×	×	High	High
	Faster R-CNN [97]	Two-stage	×	×	High	High
	FPN [98]	Two-stage	×	×	High	High
	Yes-net [99]	One-stage	√	×	Higher	Higher
	RetinaNet [100]	One-stage	×	×	Higher	Higher
GCN	DG-FPN [66]	End-to-end	√	×	Lower	Low
	S-AT GCN [68]		√	×	Lower	Low
	SIN [69]		√	√	Low	Low
	RepGN [70]		√	√	Low	Low

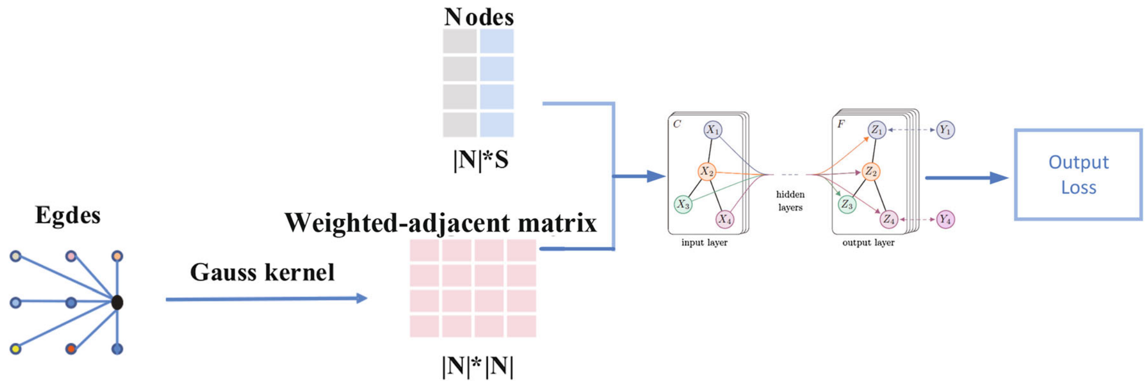


Fig. 12 Node classification of graph model by GCN

where $\widehat{A} = \widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}$, and X are input features; $W^{(0)}$ is the weight matrix from input layer to hidden layer; $W^{(1)}$ is the weight matrix from hidden layer to output layer; $soft\max(\cdot)$ and $\mathrm{Re}LU(\cdot)$ are activation functions.

For the node classification, the loss function is as follows:

$$L = -\sum_{l \in y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}, \tag{26}$$

where y_L is the known node category, and Y_{lf} is the classification of nodes by GCN.

GCN was first introduced in [75] as a new semantic segmentation method and became popular in the computer vision field. The Graph-FCN model is introduced and compared with the basic semantic segmentation FCN-16s model, which can intuitively reflect the GCN contribution to the model performance, as shown in Table 5.

4.4 Function and performance of GCN in instance segmentation

GCN is mainly used to refine the target contour and predict the vertex position of the mesh in the instance segmentation task. The refinement of target contour and the prediction of mesh vertices are very important steps in the instance segmentation task. However, in the instance segmentation task, whether GCN is used to refine the target contour or predict the mesh vertices, its essence is to update the node features. GCN takes the constructed graph as the input. On the $l-th$ layer, the graph propagation of node can be expressed as:

$$F_i^{l+1} = w_0^l F_i^l + \sum_{V_j \in N(V_i)} w_1^l F_j^l, \tag{27}$$

where $N(V_i)$ represents the node connected to V_i in the graph, w_0^l and w_1^l are the weight matrixes of the $l-th$ layer.

When GCN is used to refine the target contour, the control points on the shape of the target object in the image or video are propagated and exchanged by Eq. (27), so as to gradually realize the movement of the control points to the target object boundary, and to achieve the purpose of thinning the target contour.

When GCN is used for the prediction of mesh vertices, the characteristics of mesh vertices are updated by Eq. (27). Finally, GCN outputs the updated results of all mesh vertices to realize the prediction of mesh vertices.

The refinement of target object contour and accurate prediction of mesh vertices are realized through GCN, which shows obvious advantages in improving the instance segmentation performance. The comparison results with other general instance segmentation methods are shown in Table 6.

It can be seen from table 6 that on Cityscapes, the segmentation accuracy of the GCN based method for eight categories in the dataset is higher than that of the general instance segmentation method, which proves the effectiveness of GCN in the instance segmentation.

Table 5 Segmentation results of Graph-FCN and FCN-16s on VOC

Method	mIOU[1]	ACC[2]	f.w.IU[3]
FCN-16s	64.57	90.67	84.19
Graph-FCN	68.91	91.98	85.68

[1] "mIOU", "ACC" and "f.w.IU" in the table are the evaluation metrics of the semantic segmentation model performance.

[1] "mIOU" represents the mean intersection over union of semantic segmentation.

[2] "ACC" means the accuracy of semantic segmentation.

[3] "f.w.IU" means the abbreviation of frequency weighted intersection over union of semantic segmentation

4.5 Function and performance of GCN in object tracking

The introduction of GCN in object tracking task improves the object tracking performance to a certain extent. In the existing GCN based object tracking methods, GCN is used to update the graph's node and edge information and infer the relationship between objects.

4.5.1 Updating graph node and edge information

Generally, GCN takes edge weights and node features as inputs in the object tracking and updates the features through matrix multiplication in the graph. Taking graph $G = (V, E)$ constructed in object tracking as the input of GCN, where V and E represent the set of nodes and the set of edges, respectively, the relevant scores e_{ij} of nodes i and j for each $(i,j) \in E$ can be expressed as follows:

$$e_{ij} = f\left(h^i, h^j\right), \tag{28}$$

where h^i and h^j represent the eigenvectors of node i and node j , respectively. In GCN, the update of edges is as follows:

$$e_{ij} = f\left(h^i, h^j\right) = \left(Wh^i\right)^T \left(Wh^j\right), \tag{29}$$

where W can be regarded as the initial weight of the edge.

Nodes update is to aggregate historical features into new nodes. The update of nodes in GCN is as follows:

$$v_i = \sum_{j \in V} a_{ij} W_v h^j, \tag{30}$$

where W_v can be regarded as the initial weight of nodes, and a_{ij} can be expressed as the relationship between nodes, which is the normalized form of e_{ij} :

$$a_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{k \in V} \exp\left(e_{ik}\right)}. \tag{31}$$

Finally, the new edges and nodes are used to update the object and detection as well as allocate state information to achieve object tracking.

4.5.2 Inferring the relationship between objects

Generally, graph networks are composed of nodes, edges and global variables. The iterative updating of nodes, edges and global variables is a reasoning process. Letting V and E represent the set of nodes and the set of edges, respectively, each node is represented by a feature. The representation of V and E is as follows:

$$V = \{v_p^s, v_q^r\}, \tag{32}$$

$$E = \{e_k\}, \tag{33}$$

where $p = 1, \ldots, |O_t|$, $q = 1, \ldots, |D_t|$, $k = 1, \ldots, K$. v_p^s represents the p-th object; v_q^r represents the q-th detection; e_k represents the k-th edge or relationship between an object and a detection; K represents the total number of object detection pairs.

The edge and node are updated through GCN. With the object node, detection node, edge, and global variable u as inputs, the update of the edge is as follows:

$$e'_k = \Phi^e\left(v_k^s, v_k^r, e_k, u\right) = NN_\Phi\left(\left[v_k^s, v_k^r, e_k, u\right]\right). \tag{34}$$

where $NN_\Phi(\cdot)$ is a neural network framework, and u represents the global variables encoding all objects and detection as well as allocating state information.

Taking the object node, detection node, updated edge and global variable as inputs, the update of nodes in GCN is as follows:

$$\widetilde{v}_k^r = \Phi^e\left(v_k^s, v_k^r, e'_k, u\right) = NN_\Phi\left(\left[v_k^s, v_k^r, e'_k, u\right]\right). \tag{35}$$

The global variable are updated as follows:

$$\overline{V} = \frac{1}{2K}\left(\sum_{k=1}^{K} v_k^s + \sum_{k=1}^{K} \widetilde{v}_k^r\right), \overline{E} = \frac{1}{K}\sum_{k=1}^{K} e'_k. \tag{36}$$

Finally, the relationship between objects is inferred according to the updated nodes, edges and global variable.

The existing object tracking studies include online, near online and offline methods. The GCN based object tracking method is compared with the traditional method. The comparison results on the public data set MOT17 are shown in Table 7.

It can be seen from Table 7 that the offline method's accuracy is the highest. Regardless of the offline method, near online method or online method, the GCN based object tracking model shows better performance than the traditional object tracking methods.

Table 6 Comparison results of GCN based method and general method on Cityscapes for instance segmentation

Method		Bicycle	Bus	Person	Train	Truck	Motorcycle	Car	Rider	Mean
General	DeepMask [101]	47.19	69.82	47.93	62.20	63.15	47.47	61.64	52.20	56.45
	SharpMask [102]	52.08	73.02	53.63	64.06	65.49	51.92	65.17	56.32	60.21
	Polygon-RNN [103]	52.13	69.53	63.94	53.74	68.03	52.07	71.17	60.58	61.40
GCN	Curve-GCN [77]	64.75	81.71	72.53	65.87	79.14	62.00	80.16	70.57	72.09
	DACN [78]	64.58	82.60	72.93	61.25	80.51	63.85	80.31	71.29	72.17

5 Challenges and future opportunities

Although the GCN application in computer vision has made achievements in the academia and industry, GCN needs CNN as a basic framework to realize the modeling of image and video data, which increases the scale of visual task model, improves the computational complexity of model and increases the training time of model to a certain extent. Moreover, the current methods only compress the model at a shallow level in attempt to reduce the model scale of the seen model. In theory, the research on this problem needs great efforts from research community and a further breakthrough may be needed.

Having good GCN performance in visual tasks largely depends on the graph construction of data. Thus, improving the model performance by introducing GCN is significantly and directly related to the graph structure constructed from the Euclidean spatial data. However, to date, there is no general graph construction paradigm, which can build the most relevant graph for the original Euclidean spatial data according to the specific task. Also, no unified standard is established for evaluating feasibility of the final constructed graph. Careful analysis and multiple training for the specific task are needed to achieve satisfied results, which is time-consuming. Therefore, accurately building the most matching graph structure for the Euclidean spatial data is urgently needed through extensive researches.

In addition to the above problems, the model interpretability strongly limits the GCN application in the computer vision field. To successfully implement the GCN-based methods in visual tasks, the interpretability is an essential and prominent connection. Also, mathematical analyses for the convergence as well as over-smoothing and overfitting of deep GCN need to be established. Therefore, the current GCN application in the computer vision field is mainly based on the experience accumulated in deep learning.

With the development of deep learning and artificial intelligence technology, model lightweight is an inevitable developing trend. By improving the model architecture and optimization algorithm, the model size and the computational complexity of the model is reduced to complete the visual task efficiently and accurately. Therefore, a lightweight visual task processing model based on GCN is a new research direction.

The supervised method can achieve better performance in processing visual tasks, but in practical applications, obtaining many labeled data is challenging, and data with few or no labels are easily encountered. Many unlabeled data promote the development of semi-supervised and unsupervised learning related models. Because the multi-model fusion has achieved good results in many large competitions (such as the Otto product classification challenge on Kaggle), it is also a promising research direction to explore how the GCN model combines with semi-supervised or unsupervised learning models to better realize visual tasks.

6 Summary and conclusion

Graph convolution network (GCN), as a convolution network, has been successfully applied in many fields. In recent years, the GCN application in visual tasks has attracted more and more attention. First, this review introduces and summarizes the working mechanism of GCN, as well as the challenges and solving strategies in the vision field. Secondly, the specific GCN applications in the vision field are summarized based on different tasks. Then, the constructing graph methods in different visual tasks are discussed in detail. Finally, the role of GCN in image recognition, object detection, semantic segmentation, instance segmentation and object tracking is deeply analyzed, and compared with the commonly used CNN methods, which highlights the GCN performance in visual tasks. At the end of this review, we put forward the challenges encountered by GCN in the vision field and point out the future development directions of GCN in the vision field.

Through the above discussion, compared to the traditional CNN methods, the visual task processing methods based on GCN take the semantic, spatial and temporal

Table 7 Comparison results of GCN based method and traditional object tracking method on MOT17

Method		Style	Reasoning ability	MOTA↑
General	EDMT17 [104]	Offline	Weak	50.0
	eHAF17 [105]	Offline		51.8
	EAMTT [106]	Online		42.6
	MTDF [107]	Online		49.6
	MHTbLSTM [108]	Near-Online		47.5
GCN	Neural-Solver [83]	Offline	Stronger	58.8
	EDA-GNN [84]	Online	Strong	45.5
	Graph Networks for MOT [82]	Near-Online	Strong	50.2

relations of the object in the image into account in the visual task, which promotes the development of computer vision to a certain extent.

Appendix: Open-source code for real application of GCN in the field of vision

ML-GCN: http://github.com/Megvii-Nanjing/ML-GCN

ADD-GCN: http://github.com/Yejin0111/ADD-GCN

KSSNet: http://github.com/mathkey/mssnet

Sequential-GCN: http://github.com/razvancaramalau/Sequential-GCN-for-Active-Learning

S-AT GCN: http://github.com/Link2Link/FE-GCN

SIN: http://github.com/choasup/SIN

SPG: http://github.com/loicland/superpoint_graph

3DGNN: http://github.com/yanx27/3DGNN_pytorch

RGCNN: http://github.com/tegusi/RGCNN

Graph-FCN: http://github.com/muntam/FCN-graph

GMNet: http://github.com/LTTM/GMNet

BCNet: http://github.com/lkeab/BCNet

Curve-GCN:http://github.com/fidler-lab/curve-gcn

DACN: http://github.com/moliqingcha/DACN

Pixel2mesh: http://github.com/nywang16/Pixel2Mesh

SiamGAT: http://github.com/ohhhyeahhh/SiamGAT

Graph Networks for MOT: http://github.com/yinizhizhu/GNMOT

Neural-Solver: http://github.com/selflein/GraphNN-Multi-Object-Tracking

EDA-GNN: http://github.com/peizhaoli05/EDA-GNN

GCN: http://github.com/tkipf/gcn

Acknowledgements This work is supported by the National Natural Science Foundation of China under Grant No. 51674255, the Postgraduate Research and Practice Innovation Program of Jiangsu Province No.KYCX22_2564 and the Graduate Innovation Program of China University of Mining and Technology No. 2022WLKXJ114.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
2. Zhou FY, Jin LP, Dong J (2017) Review of convolutional neural network. Chin J Comput
3. Zhang S, Tong H, Xu J, Maciejewski R (2019) Graph convolutional networks: a comprehensive review. Comput Social Netw 6(1):1–23
4. Gong Y, Jia Y, Leung T, Toshev A, Ioffe S (2013) Deep convolutional ranking for multilabel image annotation. arXiv preprint arXiv:1312.4894
5. Wang J, Yang Y, Mao J, Huang Z, Huang C, Xu W (2016) Cnn-rnn: a unified framework for multi-label image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2285–2294
6. Zhu F, Li H, Ouyang W, Yu N, Wang X (2017) Learning spatial regularization with image-level supervisions for multi-label image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5513–5522
7. Wang Z, Chen T, Li G, Xu R, Lin L (2017) Multi-label image recognition by recurrently discovering attentional regions. In: Proceedings of the IEEE international conference on computer vision, pp 464–472
8. Li Q, Qiao M, Bian W, Tao D (2016) Conditional graphical lasso for multi-label image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2977–2986
9. Li X, Zhao F, Guo Y (2014) Multi-label image classification with a probabilistic label enhancement model. In: UAI 1:1–10
10. Lee C-W, Fang W, Yeh C-K, Wang Y-CF (2018) Multi-label zero-shot learning with structured knowledge graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1576–1585
11. Chen Z-M, Wei X-S, Wang P, Guo Y (2019) Multi-label image recognition with graph convolutional networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5177–5186
12. Bruna J, Zaremba W, Szlam A, LeCun Y (2013) Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203
13. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907
14. Zhang M, Chen Y (2018) Link prediction based on graph neural networks. Adv Neural Inf Process Syst 31:5165–5175
15. Simonovsky M, Komodakis N (2017) Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3693–3702
16. Xu K, Hu W, Leskovec J, Jegelka S (2018) How powerful are graph neural networks? arXiv preprint arXiv:1810.00826
17. Ying R, You J, Morris C, Ren X, Hamilton WL, Leskovec J (2018) Hierarchical graph representation learning with differentiable pooling. arXiv preprint arXiv:1806.08804
18. Zhang M, Cui Z, Neumann M, Chen Y (2018) An end-to-end deep learning architecture for graph classification. In: Thirty-Second AAAI conference on artificial intelligence
19. Hamilton WL, Ying R, Leskovec J (2017) Inductive representation learning on large graphs. In: Proceedings of the 31st international conference on neural information processing systems, pp 1025–1035
20. Pei H, Wei B, Chang KC-C, Lei Y, Yang B (2020) Geom-gcn: Geometric graph convolutional networks. arXiv preprint arXiv:2002.05287
21. Veličković P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. arXiv preprint arXiv:1710.10903
22. Xu B, Shen H, Cao Q, Qiu Y, Cheng X (2019) Graph wavelet neural network. arXiv preprint arXiv:1904.07785
23. Wang H, Zhang F, Zhang M, Leskovec J, Zhao M, Li W, Wang Z (2019) Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining, pp 968–977

24. Wang H, Zhang F, Wang J, Zhao M, Li W, Xie X, Guo M (2019) Exploring high-order user preference on the knowledge graph for recommender systems. ACM Trans Inf Syst (TOIS) 37(3):1–26
25. Fan W, Ma Y, Li Q, He Y, Zhao E, Tang J, Yin D (2019) Graph neural networks for social recommendation. In: The World Wide Web conference, pp. 417–426
26. Ying R, He R, Chen K, Eksombatchai P, Hamilton WL, Leskovec J (2018) Graph convolutional neural networks for web-scale recommender systems. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, pp 974–983
27. Cui Z, Henrickson K, Ke R, Wang Y (2019) Traffic graph convolutional recurrent neural network: a deep learning framework for network-scale traffic learning and forecasting. IEEE Trans Intell Transp Syst 21(11):4883–4894
28. Zhang J, Shi X, Xie J, Ma H, King I, Yeung D-Y (2018) Gaan: Gated attention networks for learning on large and spatiotemporal graphs. arXiv preprint arXiv:1803.07294
29. Yu B, Yin H, Zhu Z (2017) Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875
30. Duvenaud D, Maclaurin D, Aguilera-Iparraguirre J, Gómez-Bombarelli R, Hirzel T, Aspuru-Guzik A, Adams RP (2015)Convolutional networks on graphs for learning molecular fingerprints. arXiv preprint arXiv:1509.09292
31. Kearnes S, McCloskey K, Berndl M, Pande V, Riley P (2016) Molecular graph convolutions: moving beyond fingerprints. J Comput Aided Mol Des 30(8):595–608
32. Zitnik M, Agrawal M, Leskovec J (2018) Modeling polypharmacy side effects with graph convolutional networks. Bioinformatics 34(13):457–466
33. Fout AM (2017) Protein interface prediction using graph convolutional networks. Ph.D. thesis, Colorado State University
34. De Cao N, Kipf T (2018) Molgan: an implicit generative model for small molecular graphs. arXiv preprint arXiv:1805.11973
35. You J, Liu B, Ying R, Pande V, Leskovec J (2018) Graph convolutional policy network for goal-directed molecular graph generation. arXiv preprint arXiv:1806.02473
36. Liu X, Luo Z, Huang H (2018) Jointly multiple events extraction via attention-based graph information aggregation. arXiv preprint arXiv:1809.09078
37. Nguyen TH, Grishman R (2018) Graph convolutional networks with argument-aware pooling for event detection. In: Thirty-second AAAI conference on artificial intelligence
38. Beck D, Haffari G, Cohn T (2018) Graph-to-sequence learning using gated graph neural networks. arXiv preprint arXiv:1806.09835
39. Peng H, Li J, He Y, Liu Y, Bao M, Wang L, Song Y, Yang Q (2018) Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In: Proceedings of the 2018 world wide web conference, pp 1063–1072
40. Yao L, Mao C, Luo Y (2019) Graph convolutional networks for text classification. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 7370–7377
41. Song L, Wang Z, Yu M, Zhang Y, Florian R, Gildea D (2018) Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. arXiv preprint arXiv:1809.02040
42. Gupta A, Narayan S, Khan S, Khan FS, Shao L, van de Weijer J (2021) Generative multi-label zero-shot learning. arXiv preprint arXiv:2101.11606
43. Cao P, Chen P, Niu Q (2021) Multi-label image recognition with two-stream dynamic graph convolution networks. Image Vis Comput 113:104238
44. Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM (2019) Dynamic graph cnn for learning on point clouds. Acm Trans Graph (tog) 38(5):1–12
45. Michieli U, Borsato E, Rossi L, Zanuttigh P (2020) Gmnet: Graph matching network for large scale part semantic segmentation in the wild. In: European conference on computer vision, pp 397–414. Springer
46. Hu R, Rohrbach A, Darrell T, Saenko K (2019) Language-conditioned graph networks for relational reasoning. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10294–10303
47. Zhou J, Cui G, Hu S, Zhang Z, Yang C, Liu Z, Wang L, Li C, Sun M (2020) Graph neural networks: a review of methods and applications. AI Open 1:57–81
48. Zhang Z, Cui P, Zhu W (2020) Deep learning on graphs: a survey. IEEE Trans Knowl Data Eng
49. Wu Z, Pan S, Chen F, Long G, Zhang C, Philip SY (2020) A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learn Syst 32(1):4–24
50. Chami I, Abu-El-Haija S, Perozzi B, Ré C, Murphy K (2020) Machine learning on graphs: a model and comprehensive taxonomy
51. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. Adv Neural Inf Process Syst 29:3844–3852
52. Sen P, Namata G, Bilgic M, Getoor L, Galligher B, Eliassi-Rad T (2008) Collective classification in network data 29:93–93
53. Wang Y, He D, Li F, Long X, Zhou Z, Ma J, Wen S (2020) Multi-label classification with label graph superimposing. Proc AAAI Conf Artif Intell 34:12265–12272
54. Ye J, He J, Peng X, Wu W, Qiao Y (2020) Attention-driven dynamic graph convolutional network for multi-label image recognition. In: European conference on computer vision, pp 649–665. Springer
55. Akata Z, Reed S, Walter D, Lee H, Schiele B (2015) Evaluation of output embeddings for fine-grained image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2927–2936
56. Long Y, Liu L, Shao L, Shen F, Ding G, Han J (2017) From zero-shot learning to conventional supervised classification: Unseen visual data synthesis. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1627–1636
57. Romera-Paredes B, Torr P (2015) An embarrassingly simple approach to zero-shot learning. In: International conference on machine learning, pp 2152–2161. PMLR
58. Frome A, Corrado G, Shlens J, Bengio S, Dean J, Ranzato M, Mikolov T (2013) Devise: a deep visual-semantic embedding model
59. Li Y, Wang D, Hu H, Lin Y, Zhuang Y (2017) Zero-shot recognition using dual visual-semantic mapping paths. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3279–3287
60. Kampffmeyer M, Chen Y, Liang X, Wang H, Zhang Y, Xing EP (2019) Rethinking knowledge graph propagation for zero-shot learning. In: Proceedings of the IEEE/cvf conference on computer vision and pattern recognition, pp 11487–11496
61. Rong Y, Huang W, Xu T, Huang J (2019) Dropedge: towards deep graph convolutional networks on node classification. arXiv preprint arXiv:1907.10903
62. Chen M, Wei Z, Huang Z, Ding B, Li Y (2020) Simple and deep graph convolutional networks. In: International conference on machine learning, pp 1725–1735. PMLR
63. Li G, Muller M, Thabet A, Ghanem B (2019) Deepgcns: Can gcns go as deep as cnns? In: Proceedings of the IEEE/CVF international conference on computer vision, pp 9267–9276

64. Caramalau R, Bhattarai B, Kim T-K (2021) Sequential graph convolutional network for active learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9583–9592
65. Chai Y, Sun P, Ngiam J, Wang W, Caine B, Vasudevan V, Zhang X, Anguelov D (2021) To the point: Efficient 3d object detection in the range image with graph convolution kernels. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 16000–16009
66. Li H, Miao S, Feng R (2020) Dg-fpn: Learning dynamic feature fusion based on graph convolution network for object detection. In: 2020 IEEE international conference on multimedia and expo (ICME), pp 1–6. IEEE
67. Xu M, Fu P, Liu B, Li J (2021) Multi-stream attention-aware graph convolution network for video salient object detection. IEEE Trans Image Process 30:4183–4197
68. Zhang W, Lin Z, Cheng J, Ma C, Deng X, Wang H (2020) Sta-gcn: two-stream graph convolutional network with spatial-temporal attention for hand gesture recognition. Vis Comput 36(10):2433–2444
69. Liu Y, Wang R, Shan S, Chen X (2018) Structure inference net: Object detection using scene-level context and instance-level relationships. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6985–6994
70. Du X, Shi X, Huang R (2019) Repgn: Object detection with relational proposal graph network. arXiv preprint arXiv:1904.08959
71. Landrieu L, Simonovsky M (2018) Large-scale point cloud semantic segmentation with superpoint graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4558–4567
72. Qi X, Liao R, Jia J, Fidler S, Urtasun R (2017) 3d graph neural networks for rgbd semantic segmentation. In: Proceedings of the IEEE international conference on computer vision, pp 5199–5208
73. Te G, Hu W, Zheng A, Guo Z (2018) Rgcnn: Regularized graph cnn for point cloud segmentation. In: Proceedings of the 26th ACM international conference on multimedia, pp 746–754
74. Yang L, Zhuang J, Fu H, Zhou K, Zheng Y (2020) Sketchgcn: Semantic sketch segmentation with graph convolutional networks. arXiv e-prints, 2003
75. Lu Y, Chen Y, Zhao D, Chen J (2019) Graph-fcn for image semantic segmentation. In: International symposium on neural networks, pp 97–105. Springer
76. Ke L, Tai Y-W, Tang C-K (2021) Deep occlusion-aware instance segmentation with overlapping bilayers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 4019–4028
77. Ling H, Gao J, Kar A, Chen W, Fidler S (2019) Fast interactive object annotation with curve-gcn. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5257–5266
78. Dong Z, Zhang R, Shao X (2019) Automatic annotation and segmentation of object instances with deep active curve network, vol 7, pp 147501–147512. IEEE
79. Kim N, Kang B, Cho Y (2021) Split gcn: Effective interactive annotation for segmentation of disconnected instance
80. Wang N, Zhang Y, Li Z, Fu Y, Liu W, Jiang Y-G (2018) Pixel2mesh: Generating 3d mesh models from single rgb images. In: Proceedings of the European conference on computer vision (ECCV), pp 52–67
81. Guo D, Shao Y, Cui Y, Wang Z, Zhang L, Shen C (2021) Graph attention tracking. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9543–9552
82. Li J, Gao X, Jiang T (2020) Graph networks for multiple object tracking. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 719–728
83. Brasó G, Leal-Taixé L (2020) Learning a neural solver for multiple object tracking. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 6247–6257
84. Jiang X, Li P, Li Y, Zhen X (2019) Graph neural based end-to-end data association framework for online multiple-object tracking. arXiv preprint arXiv:1907.05315
85. Shi Y, Zheng Y, Guo K, Zhu L, Qu Y (2018) Intrinsic or extrinsic evaluation: an overview of word embedding evaluation. In: 2018 IEEE international conference on data mining workshops (ICDMW), pp 1255–1262. IEEE
86. Jaderberg M, Simonyan K, Zisserman A et al (2015) Spatial transformer networks. Adv Neural Inf Process Syst 28:2017–2025
87. Yang G, Ramanan D (2019) Volumetric correspondence networks for optical flow. Adv Neural Inf Process Syst 32:794–805
88. Kang K, Ouyang W, Li H, Wang X (2016) Object detection from video tubelets with convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 817–825
89. Kang K, Li H, Yan J, Zeng X, Yang B, Xiao T, Zhang C, Wang Z, Wang R, Wang X et al (2017) T-cnn: Tubelets with convolutional neural networks for object detection from videos. IEEE Trans Circuits Syst Video Technol 28(10):2896–2907
90. Li Y, Huang C, Loy CC, Tang X (2016) Human attribute recognition by deep hierarchical contexts. In: European conference on computer vision, pp 684–700. Springer
91. Ge Z, Mahapatra D, Sedai S, Garnavi R, Chakravorty R (2018) Chest x-rays classification: a multi-label and fine-grained problem. arXiv preprint arXiv:1807.07247
92. Jain H, Prabhu Y, Varma M (2016) Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 935–944
93. Yang X, Li Y, Luo J (2015) Pinterest board recommendation for twitter users. In: Proceedings of the 23rd ACM international conference on multimedia, pp 963–966
94. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
95. Liu L, Guo S, Huang W, Scott MR (2019) Decoupling category-wise independence and relevance with self-attention for multi-label image classification. In: ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 1682–1686. IEEE
96. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 580–587
97. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. Adv Neural Inf Process Syst 28:91–99
98. Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2117–2125
99. Ma L, Kan X, Xiao Q, Liu W, Sun P (2017) Yes-net: an effective detector based on global information. arXiv preprint arXiv:1706.09180
100. Lin T-Y, Goyal P, Girshick R, He K, Dollár P (2017) Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision, pp 2980–2988

101. O Pinheiro PO, Collobert R, Dollár P (2015) Learning to segment object candidates. Adv Neural Inf Process Syst 28
102. Pinheiro PO, Lin T-Y, Collobert R, Dollár P (2016) Learning to refine object segments. In: European conference on computer vision, pp 75–91. Springer
103. Castrejon L, Kundu K, Urtasun R, Fidler S (2017) Annotating object instances with a polygon-rnn. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5230–5238
104. Chen J, Sheng H, Zhang Y, Xiong Z (2017) Enhancing detection model for multiple hypothesis tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp 18–27
105. Sheng H, Zhang Y, Chen J, Xiong Z, Zhang J (2018) Heterogeneous association graph fusion for target association in multiple object tracking. IEEE Trans Circuits Syst Video Technol 29(11):3269–3280
106. Sanchez-Matilla R, Poiesi F, Cavallaro A (2016) Online multi-target tracking with strong and weak detections. In: european conference on computer vision, pp 84–99. Springer
107. Fu Z, Angelini F, Chambers J, Naqvi SM (2019) Multi-level cooperative fusion of gm-phd filters for online multiple human tracking. IEEE Trans Multimedia 21(9):2277–2291
108. Kim C, Li F, Rehg JM (2018) Multi-object tracking with neural gating using bilinear lstm. In: Proceedings of the European conference on computer vision (ECCV), pp 200–215