Journal of Artificial Intelligence Research 79 (2024) 1091-1112 Submitted 11/2023; published 04/2024

DIGCN: A Dynamic Interaction Graph Convolutional Network Based on Learnable Proposals for Object Detection

Pingping Cao CPP@CUMT.EDU.CN
China University of Mining and Technology,
Xuzhou 221006, China

Yanping Zhu ZHUYA@MOUNTUNION.EDU
Missouri University of Science and Technology,
Rolla 65409, USA

Yuhao Jin TS22170040A31@CUMT.EDU.CN
Benkun Ruan 08213031@CUMT.EDU.CN
Qiang Niu NIUQ@CUMT.EDU.CN
China University of Mining and Technology,
Xuzhou 221006, China

Abstract

We propose a Dynamic Interaction Graph Convolutional Network (DIGCN), an image object detection method based on learnable proposals and GCN. Existing object detection methods usually work on dense candidates, resulting in redundant and near-duplicate results. Meanwhile, non-maximum suppression post-processing operations are required to eliminate negative effects, which increases the computational complexity. Although the existing sparse detector avoids cumbersome post-processing operations, it ignores the potential relationship between objects and proposals, which hinders detection accuracy improvement. Therefore, we propose a dynamic interaction GCN module in the DIGCN, which performs dynamic interaction and relational modeling on the proposal boxes and proposal features to improve the object detection accuracy. In addition, we introduce a learnable proposal method with a sparse set of learned object proposals to eliminate a huge number of hand-designed object candidates, avoiding complicated tasks such as object candidate design and many-to-one label assignment, and reducing object detection model complexity to a certain extent. DIGCN demonstrates accuracy and run-time performance on par with the well-established and highly optimized detector baselines on the challenging COCO dataset, e.g. with the ResNet-101FPN as the backbone our method attains the accuracy of 46.5 AP while processing 13 frames per second. Our work provides a new method for object detection research.

1. Introduction

Object detection is one of the most prominent problems in computer vision. High-performance object detection can help different real-world tasks such as image retrieval (Song, He, Gao, Xu, Hanjalic, & Shen, 2018; Qin, Huang, Wei, Xie, & Zhang, 2020; Öztürk, 2021), autonomous driving (Zhong, Lei, Cao, Fan, & Li, 2017), and security systems (Lien, Chen, Bai, & Lin, 2008; Joshi & Thakore, 2012). In recent years, object detection performance has been continuously improved, and outstanding achievements have been made in the most popular object detection baselines (Yuxin Wu & Girshick, 2019; Cai & Vasconcelos, 2018).

For example, Faster R-CNN (Ren, He, Girshick, & Sun, 2015) obtained 36.2AP on the MS-COCO dataset; RetinaNet (Lin, Goyal, Girshick, He, & Dollár, 2017b) increased the accuracy to 39.1; with the development of deep learning technology, CornerNet (Law & Deng, 2018) obtained 40.5 AP; in 2019, FSAF (Zhu, He, & Savvides, 2019) directly increased the AP to 42.9; in the same year, FCOS (Tian, Shen, Chen, & He, 2019) improved the MS-COCO baseline to 44.7AP. Although these object detection methods continue to refresh and improve the baseline of object detection, most of these detectors rely on dense object candidates, making the object detection work redundant and complicated. The final performance will be affected by the number of anchors, their frame size, aspect ratio, reference density, and the proposal generation algorithm. Therefore, some studies have been conducted for designing sparse detectors,such as Sparse R-CNN (Sun, Zhang, Jiang, Kong, Xu, Zhan, Tomizuka, Li, Yuan, Wang, et al., 2021) and DETR (Carion, Massa, Synnaeve, Usunier, Kirillov, & Zagoruyko, 2020a).

DETR (Carion et al., 2020a) only took 100 learning object queries as input and did not require any hand-designed post-processing operations, and directly output object prediction results. However, it is a very sophisticated end-to-end object detection framework. In DETR, each object query must interact with the context, which increases the density and reduces the convergence speed of the model. In response to this phenomenon, Sparse R-CNN (Sun et al., 2021) proposed a completely sparse object detection method, that is, a small number of initial boxes was adequate to predict all objects in an image, and the features of each box did not need to interact with all features on the entire image. Moreover, Sparse R-CNN (Sun et al., 2021) took a small number of learnable proposal boxes, proposal features, and an image as input. The object features were obtained by the dynamic interaction between the input proposal boxes and proposal features, and finally output the category and location through two specific task prediction layers. Although this method implements a pure sparse detector, some efforts can be made to improve the detection performance.

Existing object detection work (Tan, Pang, & Le, 2020; Yuan, Wan, Fu, Liu, Xu, Ji, & Ye, 2021) mainly uses hand-designed object candidates and label assignments to determine object features, while objects are generally represented by 4-d coordinates and lack detailed information such as object shape and pose. The proposal feature was introduced for this phenomenon by Sparse R-CNN (Sun et al., 2021). The proposal feature is a high-dimensional latent vector representation. Compared with the traditional 4-d coordinate rough bounding box, the proposal feature can encode richer instance features and generate a series of specific parameters for its unique object recognition head (Jia X & V., 2016; Tian, Shen, & Chen, 2020). However, Sparse R-CNN (Sun et al., 2021) only provided a simply dynamic interaction between the proposal feature and the proposal box to obtain the object feature. Although detailed information of the object is considered, the potential relationships between objects are ignored, such as semantic and topological relationships. These studies (Carion et al., 2020a; Chen, Cao, Hu, & Wang, 2020) considered the global semantic information and local positioning information of the image when extracting object features, and performed encoding-decoding operations on the image features and location information to improve the detector accuracy, but increased the input information density.

To improve the accuracy of the sparse detector, we take the problems of the above methods into account in the detector design, and propose an object detection sparse framework, i.e., Dynamic Interaction Graph Convolutional Network (DIGCN) based on learnable proposals and a GCN. DIGCN takes an image, a small group of proposals (300) and proposal features as input, and extracts image features through the backbone network. The ROI features of proposal boxes and proposal features interact with each other through the dynamic interaction GCN module. The category and location of objects are output through the predicting layers. Compared with Sparse R-CNN (Figure 1(a)), in addition to the feature interaction, DGCIN also adds the interaction of potential relationships (Figure 1(b)), which improves the accuracy of the detector by obtaining richer feature information.

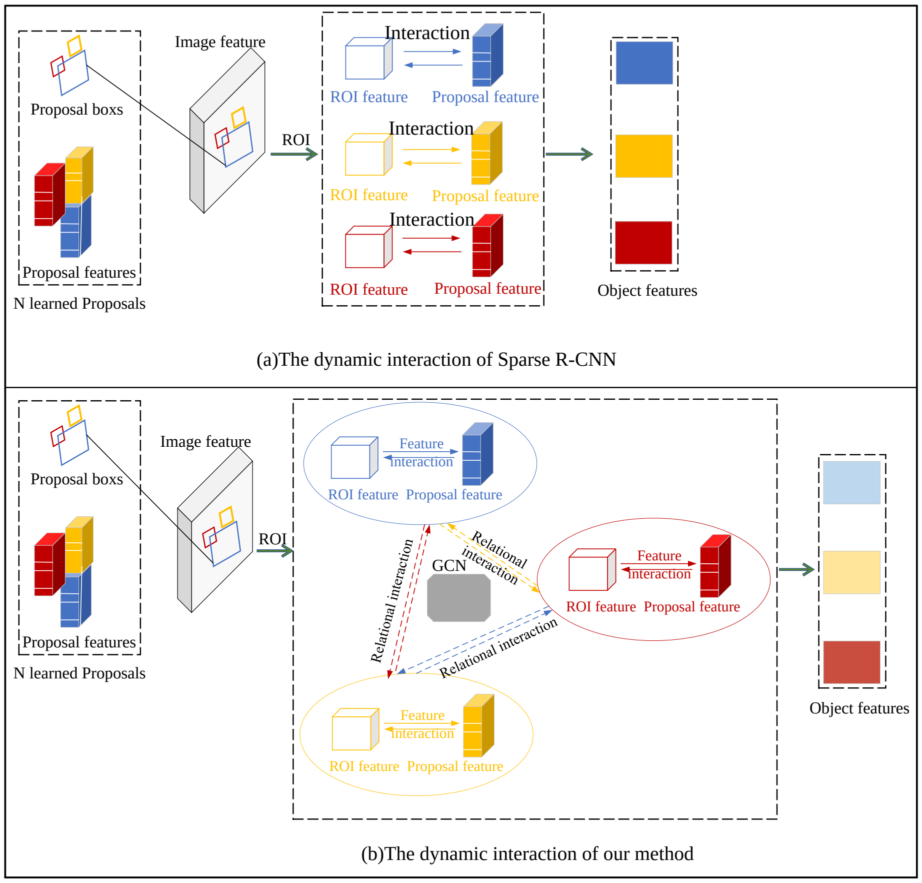


Figure 1: Comparison of the principle of feature interaction in sparse detectors.

In Figure 1, (a) represents the dynamic interaction process between ROI features and proposal features in Sparse R-CNN;(b) represents the dynamic interaction process between ROI features and proposal features in our DIGCN. In DIGCN,we have added relational interaction between features, and this relational interaction is implemented through the GCN module. Specifically,we build the object features obtained from the interaction between ROI features and proposal features as graph nodes, and dynamically generate the adjacency matrix between nodes (we will describe the construction of graph nodes and adjacency matrix in detail in Section 3.2). Through the GCN layer, we model the spatial and semantic information of object features to obtain the final object feature, which is used for subsequent object detection and improves the accuracy of the detector.

Although there are already detectors using sparse learnable proposals (Sun et al., 2021) and GCN technology (Bruna, Zaremba, Szlam, & LeCun, 2013; Zhao, Ge, & Yu, 2021), compared with these methods, our method has its unique novelty. In reference (Sun et al., 2021), it only considered the interaction at the feature level of the proposal, which largely omitted spatial and semantic information about object features in the inspected image, thus reducing the detection performance of the detector. However, we can remedy this defect by using GCN to model the relationship between the proposals. In (Zhao et al., 2021), GCN is used to model the topological relationship between multi-scale features. The focus is still on the multi-scale features of images, while the GCN in our method focuses on the relationship between proposals. Although (Bruna et al., 2013) uses GCN for box association, it is essentially different from our method. Reference (Bruna et al., 2013) constructs the graph structure on the dataset, with the word embedding of the label as the node. The detector models the relationship between the label and the region proposal. The whole process relies heavily on the label information of the data, and dense candidate boxes are used in this work. However, in our method, we directly model the relationship between proposal boxes and proposal features. We do not rely on the labels of data sets. We can use sparse proposals to achieve high-performance object detection. At present, no work has ever used ideas similar to our methods, and the novelty of our methods is particularly prominent.In general, the main contributions of this article are as follows:

- A novel sparse object detection framework: DIGCN is proposed, which uses dynamic interaction and GCN to obtain object features in the image to be detected for prediction of object position and category.
- A dynamic interaction GCN module is proposed, which includes a dynamic instance interaction operation and a GCN for relationship modeling. In the dynamic interaction GCN module, the input proposal boxes and proposal features are dynamically interacted to obtain preliminary object features, and then GCN modeling is used to obtain more prominent object features.
- A graph structure is constructed and the object features obtained from the interaction of dynamic instances are nodes. The customized parameters generated by the proposal features form an adjacency matrix, which maps the relationship between the object features.
- On the challenging public object detection data set COCO2017, our method was tested for accuracy, running time, and training convergence performance, which is comparable to a well-established detector baseline. e.g., with the ResNet-101FPN as the backbone our method attains the accuracy of 46.5 AP while processing 13 frames per second.

2. Previous Work

2.1 The Conventional Method

At present, popular detectors are mainly based on a general object detection framework i.e., deep Convolutional Neural Networks (CNNs) (Szegedy, Toshev, & Erhan, 2013; Ioffe &

Szegedy, 2015; Krizhevsky, Sutskever, & Hinton, 2012), which greatly improves the object detection model performance (Everingham, Van Gool, Williams, Winn, & Zisserman, 2010; Lin, Maire, Belongie, Hays, Perona, Ramanan, Dollár, & Zitnick, 2014). In the general object detection work based on CNNs, there are two commonly used methods. One is to directly predict the category and location of the anchor box by densely covering the spatial location, scale and aspect ratio in a single shot, such as OverFeat (Mathieu, LeCun, Fergus, Eigen, Sermanet, & Zhang, 2013), YOLO9000 (Redmon & Farhadi, 2017), SSD (Liu, Anguelov, Erhan, Szegedy, Reed, Fu, & Berg, 2016) and RetinaNet (Lin et al., 2017b). OverFeat (Mathieu et al., 2013) is an integrated framework that can simultaneously implement multi-scale and sliding windows in the same convolutional network. The learning process for location identification is to predict the object boundary, and continuously accumulate bounding boxes to improve the detection confidence. YOLO9000 (Redmon & Farhadi, 2017) realizes real-time object detection by using multi-scale training and joint training on the object detection and classification based on YOLO, which has a high advancement level. SSD (Liu et al., 2016) is a simple object detection method that eliminates the proposal generation and subsequent pixel or feature resampling stages and encapsulates all calculations in a single network. It discretizes the output space of the bounding box into a set of default boxes, and generates scores for the existence of each object category in each default box. Also, SSD combines feature maps of different scales for prediction, which can adapt to objects of varying sizes. RetinaNet (Lin et al., 2017b) eliminates extreme foreground-background imbalances by reshaping standard cross-entropy loss, and trains a sparse example set to detect objects accurately and quickly in images or videos.

The other object detection work based on CNNs is an anchorless method that replaces manual anchor boxes with reference points (Law & Deng, 2018; Tian et al., 2019; Huang, Yang, Deng, & Yu, 2015; Zhou, Wang, & Krähenbühl, 2019; Kong, Sun, Liu, Jiang, Li, & Shi, 2020). CornerNet (Law & Deng, 2018) used a CNN to detect the upper left corner and the lower right corner of the object bounding box as a pair of key points for the object, eliminating the effort of manually designing the anchor. The introduction of the corner pooling layer in CornerNet can help the network locate corners more accurately and improve the detection accuracy of the detector. FCOS (Tian et al., 2019) was a detector without anchor and proposal box, solving object detection by a pixel-by-pixel prediction. It completely avoided all the hyperparameters related to the anchor box and made the detection performance more stable. Huang et al. (Huang et al., 2015) introduced DenseBox in their work, which can directly predict the bounding box and object classification confidence through all positions of the object and scales of the image. Zhou et al. (Zhou et al., 2019) modeled each object as a center point of its bounding box, and the key point estimation was used to find the center point and to regress other object attributes. FoveaBox (Kong et al., 2020) directly learned the existence possibility. The bounding box coordinates of the object can be determined by predicting category-sensitive semantic maps for the object existing possibility and generating category-agnostic bounding boxes for each position that may contain the object, which avoided anchor references. Compared with detectors based on bounding boxes, detectors without anchors are simpler, faster, and more accurate.

2.2 Sparse Method

General sparse object detection methods can avoid the design of dense candidate objects, but the accuracy is significantly lower than conventional dense candidate-based detectors. The work by Najibi et al. (Najibi, Rastegari, & Davis, 2016) was representative of the first batch of sparse detection algorithms. In their work, the object detection problem model found a path in a fixed grid to a box tightly surrounding the object. Although this method achieves the sparse detection purpose, the performance of this proposal box, which is manually designed based on prior knowledge, needs to be improved in object detection tasks. Afterwards, Sun et al. (Sun et al., 2021) proposed a sparse R-CNN, applying a learnable proposal and Deformable-DETR (Zhu, Su, Lu, Li, Wang, & Dai, 2020) to limit the sampling points of each object query to its reference point surrounding, which improved the sparse detector performance. Although the sparse R-CNN (Sun et al., 2021) has achieved good results in sparse object detection, it ignores other key information that helps improve the detector's accuracy.

A learnable proposal (Sun et al., 2021) refers to a small set of proposal boxes defined in the object image. It is an initial guess of the possible area of the object in the image, usually represented by 4-d coordinates: (x, y, h, w), which respectively represent the center coordinates (x, y), height and width of the proposal box. During the training process, the parameters of the proposal box can be continuously updated through backpropagation to find the optimal area (that is, the area closest to the object). Due to the learnability of the proposal box, the influence of the initial parameters of the proposal box on the entire detection framework is not significant.

The 4-d proposal box can only roughly represent the object's position information, and cannot describe detailed information such as the posture and shape of the object. To provide the detector with more detailed object information, a high-dimensional vector of the proposal feature is introduced to represent the potential features of the object. The proposal features can be used to encode rich instance features and are one of the key technologies to improve detector accuracy.

2.3 GCN-Based Method

The basic idea of GCN is to update node representation by propagating information between nodes (Kipf & Welling, 2016), showing a strong modeling ability for the relationship between non-Euclidean spatial data. Therefore, the original GCN has been widely used in 3D object detection tasks (Chai, Sun, Ngiam, Wang, Caine, Vasudevan, Zhang, & Anguelov, 2021; Wang, Wang, Zhang, Lan, & Li, 2021). After mastering in-depth graph knowledge, GCN has been successfully applied for conventional object detection by constructing graphs of object images or videos in many studies (Li, Miao, & Feng, 2020; Du, Shi, & Huang, 2019). In the object detection process, Li et al. (Li et al., 2020) used GCN for feature fusion of images with different resolutions, dynamically transferring knowledge through learnable weights between all nodes and introducing semantic information to guide the fusion process. The optimal feature fusion strategy was learned for the detector to improve the object detection performance. Du et al. (Du et al., 2019) proposed a Relational Proposal Graph Network (RepGN), which modeled semantics and space as boundaries. In the object detector, GCN is used to model relationships and context constraints, and act in regional feature extraction and bounding box regression classification to improve the object detection performance. However, the above GCN-based object detector relies on dense object candidates. To effectively eliminate manually designed candidates and components and to improve the sparse detectors' accuracy, we propose a sparse object detection method-DIGCN based on learnable proposals and GCN. Our method applies learnable proposals and inspired by GNN's interaction with features in (Zhou, Wang, Qi, Ling, & Shen, 2020; Zhou, Li, Li, Feng, Li, & Shao, 2021), we designed a GCN dynamic interaction module for proposal box association and potential relationship modeling, which improves the performance of the detector. Our method provides a new research idea for the application of GCN in object detection tasks.

3. Method

Dynamic interaction graph convolutional network (DIGCN) is an end-to-end object detection framework based on learnable proposals and dynamic interaction GCN. It consists of a backbone network, a dynamic interaction GCN module and two prediction layers for output location and classification. DIGCN has a total of three inputs, an image, a set of proposal boxes and proposal features. The framework of DIGCN is shown in Figure 2.We will describe each component in detail in this section.

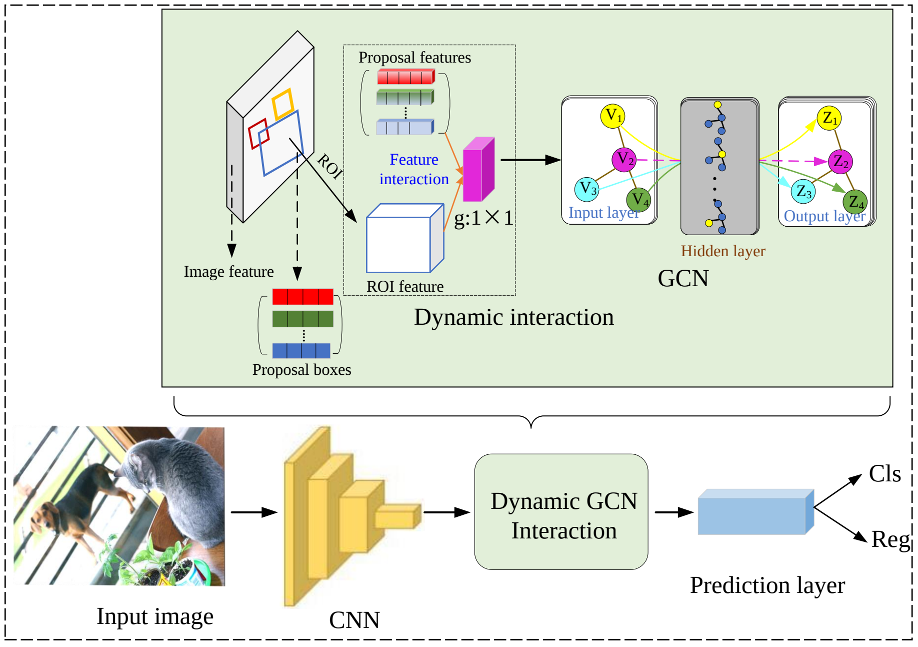


Figure 2: An overview of DIGCN pipeline.

3.1 Backbone

We use the Feature Pyramid Network (Lin, Dollár, Girshick, He, Hariharan, & Belongie, 2017a) based on ResNet(He, Zhang, Ren, & Sun, 2016) architecture as the backbone of

DIGCN to extract image features from the input image and input them to the GCN dynamic interaction module. Like Sparse R-CNN (Sun et al., 2021), our DIGCN also has the potential to obtain higher performance from more complex backbone network designs (such as stacked encoder layers (Carion, Massa, Synnaeve, Usunier, Kirillov, & Zagoruyko, 2020b) and deformable revolution networks (Dai, Qi, Xiong, Li, Zhang, Hu, & Wei, 2017)). To confirm this claim,we use a more complex stacked encoder layers network instead of the ResNet architecture as the backbone of DIGCN, with the other modules remaining unchanged, and achieve a mAP of 46.7 on the standard object detection dataset COCO2017. However, to show the simplicity and effectiveness of DIGCN, we will keep the settings consistent with Faster R-CNN (Ren et al., 2015; Lin et al., 2017a) and Sparse R-CNN (Sun et al., 2021). At present, some advanced object detection work based on deep learning (Ren et al., 2015; Lin et al., 2017b; Sun et al., 2021; Carion et al., 2020a) mainly uses the network with ResNet structure as the backbone. In order to compare with existing methods more intuitively and fairly, we choose the Feature Pyramid Network based on the ResNet structure as the backbone of DIGCN.

3.2 Dynamic Interaction GCN Module

The dynamic interaction GCN module is the core of our method, which is mainly divided into two stages: dynamic convolutional layer interaction and dynamic GCN interaction. This module takes a set of proposal features, proposal boxes and images as inputs. Then the module achieves dynamic interaction between proposal boxes and proposal features through a convolutional layer and a single-layer GCN.

In the dynamic interaction GCN module, for the given paired proposal boxes and proposal features (f_{pro}), during the dynamic convolutional layer interaction stage, we first extract the ROI features (f_{roi}) of each proposal box corresponding to the image through RoIAlign operation. Then, the obtained ROI features are interacted with the proposal features (f_{pro}) through a $1{\times}1$ convolutional layer to filter out ineffective bins, the interaction process can be seen in Figure 3(a). The parameter of the $1{\times}1$ convolutional layer is the dynamic parameter matrix, which is generated by the filter-generate network in the dynamic filtering network (Jia X & V., 2016). We called the features obtained from the interaction of convolutional layers preliminary object features (obj_f_{pre}).

In the dynamic graph convolution interaction stage, we first use the object features obj_f_{pre} as nodes and the dynamic parameter matrix generated by the proposal features as the adjacency matrix to construct the graph structure of the object features (see Section 3.2.1). Then, a single-layer GCN is used to update node features, thereby achieving dynamic interaction between obj_f_{pre} and f_{pro}. At this point, our dynamic interaction graph convolution module can complete the dynamic interaction between proposal boxes and proposal features. The final object features (obj_f_{fin}) obtained from the interaction are used for object localization and classification.

3.2.1 Constructing the graph structure of object features

In our design, we use the object features (obj_f_{pre}) as nodes V and generate adjacency matrix A based on the proposal features for object features , as shown in Figure 3(b).

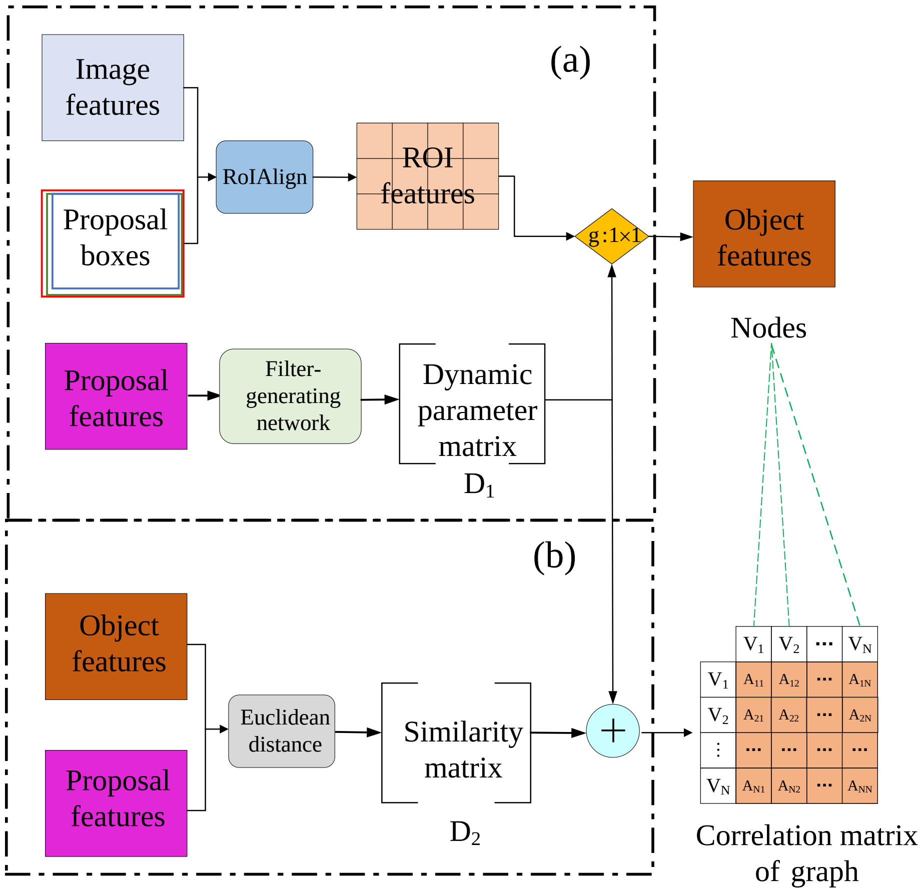


Figure 3: Graph construction method based on object features.

Specifically, the construction principle of adjacency matrix A is as follows: Firstly, we use the proposal features to generate a dynamic parameter matrix $D1$ through the filter-generate network. The filter-generate network (Jia X & V., 2016) can be implemented using any differentiable architecture. As the input of our task is an image, we chose a convolutional network to obtain the dynamic parameter matrix. Then we use Euclidean distance to calculate the similarity between the proposal features and obj_f_{pre} obtain a similarity matrix $D2$. Finally, by fusing the two matrices, we can obtain the adjacency matrix of the graph. Since the proposal features in the dynamic interaction graph convolution module participate in both the dynamic convolution layer interaction and GCN interaction stages, the fusion of the two matrices is necessary. In addition, the graph is constructed based on sparse proposals, avoiding the complex distribution caused by dense connections, thereby reducing noise edges, and saving the operation of noise filtering.

In the figure, "$g : 1 \times 1$" represents the convolution operation of 1×1 and "$\oplus$" represents matrix addition operations. "A" represents the adjacency matrix of the graph and V represents the nodes.

3.2.2 Dynamic GCN

We designed a dynamic GCN to replace the static GCN to realize dynamic interaction between proposal boxes and proposal features in the dynamic interaction GCN module. The working principle of dynamic GCN is as follows:

First, the dynamic GCN uses obj_f_{pre} as the node V and the adjacency matrix A as the initial matrix to update the node. The updated node H is as follows:

$$H = \delta(AVW), \tag{1}$$

Among them, W is learnt during training, and $\delta(\cdot)$ represents the activation function. In this experiment, the activation function is $ReLU$.

Then, the updated node H is input into the dynamic convolutional layer. In this layer, the adjacency matrix A_d is dynamically generated:

$$A_d = \delta\left(W_A H^{'}\right), \tag{2}$$

where $\delta(\cdot)$ is the activation function *sigmoid*; W_A is the weight obtained by training the convolutional layer and can be used for calculating the dynamic adjacency matrix. $H^{'}$ is synthesized by connecting H and its global representation H_g (generated by global average pooling and a convolutional layer), thus, the definition of $H^{'}$ is as follows:

$$H^{'} = \left[(h_1; h_g), (h_2; h_g), \ldots, (h_c; h_g)\right]. \tag{3}$$

Finally, the updated node information is obtained through the dynamic GCN module as equation (4).

$$Z = LReLU(A_d H W_d). \tag{4}$$

The final object feature obj_f_{fin} can be obtained by linear mapping the node information output from the dynamic GCN module.

3.3 Loss Function

In DIGCN, the prediction layer mainly relies on the loss function to realize the prediction of object category and location. Like the loss function used in (Carion et al., 2020a; Stewart, Andriluka, & Ng, 2016; Yang, Wang, Clark, Hu, Wang, Markham, & Trigoni, 2019), we adopted a set-based one-to-one matching loss for classification and box coordinate prediction. According to (Carion et al., 2020a), when i represents the elements of the truth label set, c_i represents the truth label of the object class, and $\sigma(i)$ represents the prediction index, the probability of the object class c_i can be expressed as: $\widehat{p}_{\sigma(i)}(c_i)$,then the loss between the predicted category of the target and the truth label is equation (5):

$$L_{cls} = \widehat{p}_{\widehat{\sigma}(i)}(c_i). \tag{5}$$

When b_i and $\widehat{b}_{\sigma(i)}$ are used to represent the set of truth boxes and prediction boxes, box losses can be represented as:

$$L_{box}(b_i, \widehat{b}_{\widehat{\sigma}(i)}) = \| b_i - \widehat{b}_{\widehat{\sigma}(i)} \| \left(1 + \left(b_i, \widehat{b}_{\widehat{\sigma}(i)}\right)\right), \tag{6}$$

Then the final loss can be calculated by equation (7):

$$L\left(y_i, \widehat{y}\right) = \lambda_{cls} \cdot \widehat{p}_{\widehat{\sigma}(i)}\left(c_i\right) + \lambda_{L_1} \parallel b_i - \widehat{b}_{\widehat{\sigma}(i)} \parallel \left(1 + \lambda_{giou}\left(b_i, \widehat{b}_{\widehat{\sigma}(i)}\right)\right), \tag{7}$$

In the equation, λ_{cls} , λ_{L1} and λ_{giou} are the coefficients of the object category loss, box loss, and generalized IOU loss, respectively. Compared with other detectors, using set-based one-to-one matching loss avoids the problem of many-to-one label allocation, thereby reducing computational costs. Compared with other end-to-end detectors based on set prediction, the linear combination of L_1 loss and generalized IOU loss is used to directly predict boxes of different sizes, bypassing the preliminary prediction steps of other detectors for boxes.

4. Experiment

In this section, we first introduce the evaluation metrics, data sets and implementation details of the detector. Then, the test results of our method on the COCO2017 data set are given and compared with other methods. Finally, the advantage of our method is proved through the ablation research and result visualization.

4.1 Evaluation Metrics

According to previous object detection work (Ren et al., 2015; Sun et al., 2021; Carion et al., 2020a), we use Average Precision (AP) and Frame Per Second (FPS) as evaluation metrics. For a fair comparison, we also compared AP_{50}, AP_{75}, AP_s, AP_m and AP_l. AP_{50} and AP_{75} represent the accuracy of taking the positive result when the IOU threshold is greater than 0.5 and 0.75, respectively. The larger the IOU threshold, the more difficult the measurement. AP_s, AP_m and AP_l represent detection AP for small objects, medium objects, and large objects, respectively.

4.2 Dataset

We use our method to perform object detection performance experiment on the COCO2017 (Lin et al., 2014). COCO2017 contains about 118k training images and 5k validation images. Each image is labeled with bounding box information. On average, each image contains 7 instances. In the training set, each image can have 63 instances at most. These instances have different sizes, which brings great challenges to object detection.

4.3 Implementation Details

Our DIGCN is implemented by Pytorch, and ResNet-101 (Szegedy et al., 2013) is the backbone network we mainly use to extract image features. In order to improve the training speed of DIGCN, we use the pre-training weights on ImageNet (Deng, Dong, Socher, Li, Li, & Fei-Fei, 2009) to initialize the backbone network. The AdamW optimizer is used during training, and the weight decays to 0.0001. The minimum training batch for each GPU is 2 images. Our main detection result is trained using 4 GPUs and the batch size is set to 8. The whole training process includes 36 epochs, and the initial learning rate is 2.5×10^{-5} which is reduced by 10 times at the epochs 27 and 33. According to the previous work (Sun et al., 2021; Carion et al., 2020a) λ_{cls} , λ_{L1} and λ_{giou} are set to 2, 5 and 2. The input image of the network is enhanced with random horizontal scaling, and the image is adjusted to be at least 480 pixels to 800 pixels on the shortest side, and up to 1333 pixels on the longest side (Ren et al., 2015). In addition, in our training, the number of proposal boxes, proposal features, and iterations are 300, 300, and 6, respectively. Except for the initial proposal box, the gradients in the iteration process are limited into the proposal box to ensure the stability of training.

4.4 Experimental Results

Our method has been applied to the challenging object detection dataset COCO2017 using 4 GPUs of NVIDIA GeForce RTX 2080Ti with a batch size of 8. The trained model is applied to the validation dataset. Since the detector performance is related to the training hardware, we reproduce one of the most advanced sparse detector Sparse R-CNN (Sun et al., 2021) on the same hardware device as our method to make a fair comparison. During the training process, all parameters remain the same. At the same time, we also compare with other detectors that including the latest detector based on GCN method (Chen, Li, Huang, Zhang, & Ma, 2021; Zhao et al., 2021), and the results are shown in Table 1.

Table 1: Comparison of the detection results of Sparse GCN and other detectors on COCO2017.

Method		EP	GPU/num	Acc						Sp
				AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	FPS
RetinaNet-R50		36	v100/8	38.7	58.0	41.5	23.3	42.3	50.3	24
RetinaNet-R101		36	v100/8	40.4	60.2	43.2	24.0	44.3	52.2	18
Faster R-CNN-R50		36	v100/8	40.2	61.0	43.8	24.2	43.5	52.0	26
Faster R-CNN-R101		36	v100/8	42.0	62.5	45.9	25.2	45.6	54.6	20
Cascade R-CNN-R50		36	v100/8	44.3	62.2	48.0	26.6	47.7	57.7	19
DETR-R50		500	v100/8	42.0	62.4	44.2	20.5	45.8	61.1	**28**
DETR-R101		500	v100/8	43.5	63.8	46.4	21.9	48.0	61.8	20
DETR-DC5-R50		500	v100/8	43.3	63.1	45.9	22.5	47.3	61.1	12
DETR-DC5-R101		500	v100/8	44.9	64.7	47.7	23.7	49.5	**62.3**	10
Deformable DETR-R50		50	v100/8	43.8	62.6	47.7	26.4	47.1	58.0	19
DN-DETR-DC5-R101		50	A100/8	47.3	67.5	50.8	28.6	51.5	65.0	-
DINO-5scale		36	A100/8	51.2	69.0	55.8	35.0	54.3	65.3	10
YOLOv7-E6E		-	cloudGPU	**56.8**	**74.4**	**62.1**	**40.8**	**62.1**	**70.6**	-
Sparse R-CNN-R50*		36	2080Ti/4	44.7	63.5	48.4	27.8	47.6	59.1	17
Sparse R-CNN-R101*		36	2080Ti/4	45.6	64.6	49.5	29.0	48.2	61.4	14
GCN-Based	Relation R-CNN-R101	20	-	36.2	56.9	39.3	19.5	41.2	49.1	-
	GraphFPN-R101	50	2080Ti/4	46.3	65.1	50.1	29.5	49.1	61.5	10
	DIGCN-R50(ours)	36	2080Ti/4	45.2	64.1	49.5	28.7	47.8	59.4	16
	DIGCN-R101(ours)	36	2080Ti/4	46.5	65.7	50.6	30.0	49.5	61.0	13

In Table 1, "Ep" is the abbreviation of "Epoch", which represents the number of training cycles; "Acc" and "Sp" respectively represent the accuracy and speed of model training; "V100", "A100" and "2080Ti" represent the NVIDIA Tesla V100, "NVIDIA A100 Tensor Core" and NVIDIA GeForce RTX 2080Ti GPU types, "/8" and "/4" represent the number of GPU is 8 and 4, respectively; "*" means the result of reproducing and training this model on our hardware device. Black bold represents the best result among all methods, while red represents the best result based on GCN methods.

It can be seen from the comparison results in Table 1 that our method has higher accuracy than either the most advanced sparse detector Sparse R-CNN (Sun et al., 2021) or the latest detector based on GCN method (Chen et al., 2021; Zhao et al., 2021). Although our method appears disadvantages in detection accuracy compared to recent work (Li, Zhang, Liu, Guo, Ni, & Zhang, 2022; Zhang, Li, Liu, Zhang, Su, Zhu, Ni, & Shum, 2022; Wang, Bochkovskiy, & Liao, 2023), it still has advantages in computational cost. DN-DETR-DC5 (Li et al., 2022), DINO-5scale (Zhang et al., 2022) and YOLOv7 (Wang et al., 2023) reported FLOPs of 839.8G, 275.4G, and 843.2G in their work, respectively. However, our method has only 24.1G FLOPs, indicating that our method has a lower computational cost. In addition, DN-DETR-DC5-R101 (Li et al., 2022) and DINO-5scale (Zhang et al., 2022) still rely on dense query sequences or anchors for object detection, which do not belong to the same category of detectors as our method.

GraphFPN (Zhao et al., 2021), which is also based on the GCN method, uses "Contextual Graph Layers" to achieve feature interaction at the same scale, with the same goal as dynamic GCN interaction in our method, but their structures are different. The contextual graph layers use a graph self-attention network to update graph nodes. From the feature update formula mentioned in GraphFPN (Zhao et al., 2021), this graph self-attention network is composed of two GCNs based on attention mechanisms. Our dynamic GCN interaction only requires a single GCN to achieve feature interaction, nor does it require the addition of additional attention mechanisms. Although our method has only 0.2AP gain compared with GraphFPN (Zhao et al., 2021), GraphFPN (Zhao et al., 2021) is implemented by relying on dense candidate frames, and FPS is significantly reduced. The comparison with GraphFPN (Zhao et al., 2021) further shows that our DIGCN can still achieve detection performance comparable to the dense candidate box method when using sparse proposal boxes. In addition, we compared all APs of DIGCN with Sparse R-CNN (Sun et al., 2021) to illustrate the convergence rate of our model. The comparison results are shown in Figure 4.

In Figure 4, our detector starts to converge at the epoch 33, and Sparse RCNN starts to converge at the epoch 34. It can be seen from Figure 4 that our method has the same convergence speed as the most advanced sparse detector.

4.5 Ablation Researches

In this section, we conduct ablation studies on each component of the dynamic interaction GCN module in the DIGCN, and explore the impact of the dynamic interaction, GCN module, two-layer GCN, three-layer GCN, dynamic GCN, and graph construction ways on the detector performance, and we reported the FLOPs, parameter numbers and running times of each ablation experiment to demonstrate the trade-off between the effectiveness

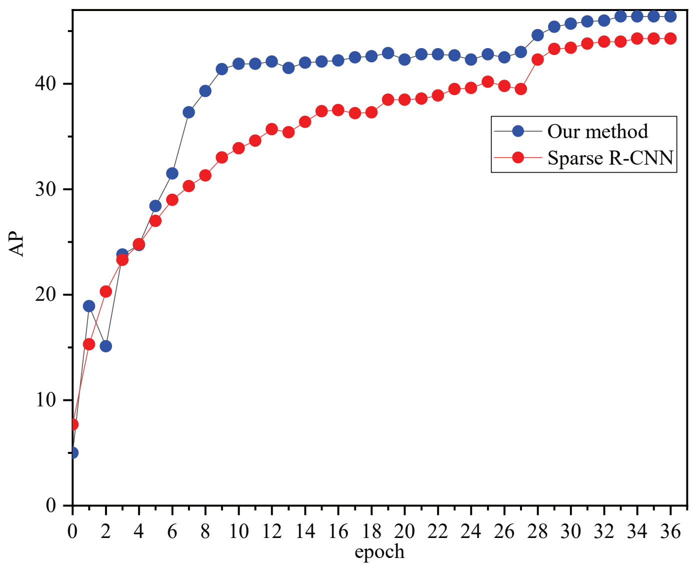


Figure 4: AP comparison of our DIGCN and Sparse RCNN.

and efficiency of our method. All ablation studies are based on ResNet 101 as the backbone network, 300 proposals and 3x training schedule based on 4 GPUs of NVIDIA GeForce RTX 2080Ti with a batch size of 8, unless otherwise specified.

4.5.1 Dynamic interaction

In order to verify the contribution of the dynamic interaction in the DIGCN, we cancel the dynamic interaction operation in the model, and simply use the ROI feature and the proposal feature as the input of the GCN module, and the other modules remain unchanged. The detection AP drops from 46.5 to 45.5, the detection results and the corresponding FLOPs, parameter numbers, running times are shown in Table 2.

Table 2: Comparison of DIGCN full model and DIGCN without dynamic interactive operation.

Model	Dynamic interaction	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	$FLOPs$	$Params$	$Times$
DIGCN	Yes	46.5	65.7	50.6	30.0	49.5	61.0	24.1G	77.85M	52.5h
DIGCN	No	45.5	64.7	49.4	29.3	48.3	60.1	23.7G	77.83M	52h

It can be seen from Table 2 that the APs of the full DIGCN model is higher than that of the DIGCN model without dynamic interaction. This is because each ROI feature in the dynamic interaction interacts with its corresponding proposal feature, thus eliminating the impact of invalid bins, avoiding the impact of invalid information on object features, improving the accuracy of object detection, and verifying the effectiveness of our dynamic interaction in improving the accuracy of object detection. At the same time the total DIGCN model and the DIGCN without dynamic interactive operation have approximately the same running times and parameter numbers. Although the FLOPs, parameter numbers and running times of the full model is slightly increased compared with that without dynamic interaction, it is wise to use the full model with a higher detection accuracy based on the AP gain in Table 2.

4.5.2 GCN module

In order to verify the role of GCN in our method, we designed an experiment to remove GCN modules: leave the other modules unchanged, remove the GCN module, and directly use the proposal features after dynamic interaction for object classification and localization. After removing GCN from DIGCN, train the model and record the average value of 5 times of training compared with our whole model. The detection results and the corresponding FLOPs, parameter numbers , running times are shown in Table 3.

Table 3: Comparison of DIGCN with GCN removed and DIGCN full model.

Model	GCN	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	$FLOPs$	$Params$	$Times$
DIGCN	Yes	46.5	65.7	50.6	30.0	49.5	61.0	24.1G	77.85M	52.5h
DIGCN	No	44.6	63.2	47.9	27.6	47.0	59.1	22.6G	77.83M	49.7h

As can be seen from Table 3, with the removal of GCN module in DIGCN, the AP of the model decreases from 46.5 to 44.6, and the corresponding AP_{50}, AP_{75}, AP_s, AP_m and AP_l decrease. This is because compared with DIGCN without GCN, DIGCN full model adds interaction modeling of semantic and topological features between proposal boxes in dynamic interaction, and has richer object features. This shows that GCN module plays an important role in improving the detection accuracy of DIGCN.

Although the removal of GCN module can reduce FLOPs, parameter numbers and running times, it also reduces the AP of the detector, and the reduction of accuracy is unbalanced with the improvement of efficiency.

At the same time, we also try to use an Attention Mechanism (AM) to replace GCN for relationship modeling, and other modules remain unchanged. Input the proposal features and proposal boxes into the attention mechanism, and first calculate the similarity score of each proposal through dot product calculation; Then, using normalization operation, the attention score is converted into attention weight; Next, the attention weight is multiplied by the corresponding eigenvalues vector to obtain the weighted eigenvalues, resulting in a more focused feature representation; Finally, the weighted features will be used for object classification and localization. The training results and the corresponding FLOPs, parameter numbers, running times are shown in Table 4.

Table 4: Comparison results between DIGCN and Attention Mechanism.

Model	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	$FLOPs$	$Params$	$Times$
DIGCN	46.5	65.7	50.6	30.0	49.5	61.0	24.1G	77.85M	52.5h
AM	44.2	63.6	47.6	27.7	47.3	58.3	24.3G	77.93M	53.6h

The results in Table 4 further illustrate the effectiveness and rationality of our relationship modeling using GCN.

During the training process, we believe that the number of GCN layers also affects the training results. To verify this assumption, we design an experiment using single-layer GCN, two-layer GCN and three-layer GCN for model training. The two-layer GCN is designed on the basis of the single-layer GCN. The output of the single-layer GCN is taken as the input feature of the two-layer GCN. The input feature is linearly mapped to obtain the corresponding dynamic parameter matrix. The dynamic parameter matrix is fused with the adjacency matrix of the single-layer GCN to obtain the adjacency matrix of the two-layer GCN. Similarly, we have added a three-layer GCN on the basis of the two-layer GCN. Train the DIGCN model of single-layer GCN, two-layer GCN and three-layer GCN respectively, the training results and the corresponding FLOPs, parameter numbers, running times are shown in Table 5.

Table 5: Comparison results of GCN with different layers.

Model	GCN layers	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	$FLOPs$	$Params$	$Times$
DIGCN	1	46.5	65.7	50.6	30.0	49.5	61.0	24.1G	77.85M	52.5h
DIGCN	2	45.9	65.0	49.9	29.7	49.1	60.3	24.8G	77.85M	54h
DIGCN	3	45.1	64.4	49.1	29.1	48.5	59.3	26.1G	77.85M	55.4h

It can be seen from the comparison results in Table 5 that the proposed detector uses single-layer GCN to obtain higher APs than two-layer GCN and three-layer GCN. This is because our input is sparse proposal features and the corresponding sparse proposal boxes. After a single-layer GCN, we can obtain rich spatial and semantic information. Using two-layer GCN and three-layer GCN for sparse input features is prone to over-smoothing and over-fitting, which reduces the accuracy of the model. In addition, with the increase of GCN layers, the accuracy and efficiency of the detector are decreasing. It can be seen that multi-layer GCN is not suitable for our DIGCN.

Moreover, to verify the role of GCN in our method, we also use dynamic GCN instead of static GCN in the dynamic interaction GCN module, other modules remain unchanged for training. The comparison results with our method on the COCO2017 dataset are shown in Table 6.

Table 6: Comparison results of using dynamic GCN and static GCN.

Model	GCN	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
DIGCN	Static	46.5	65.7	50.6	30.0	49.5	61.0
DIGCN	Dynamic	44.6	64.0	48.3	27.4	48.0	59.1

It can be seen from the comparison results in Table 6 that in DIGCN, static GCN can better improve the detector accuracy. We think that this is because the initial adjacency matrix in the dynamic interaction is a dynamic parameter matrix generated by the proposal features through dynamic algorithm (Jia X & V., 2016). The dynamic adjacency matrix has been matched for the proposal features. When the dynamic GCN is used again to generate the dynamic matrix, some spatial and semantic information is lost, so the accuracy of the detection model is reduced.

4.5.3 Ways of graph construction

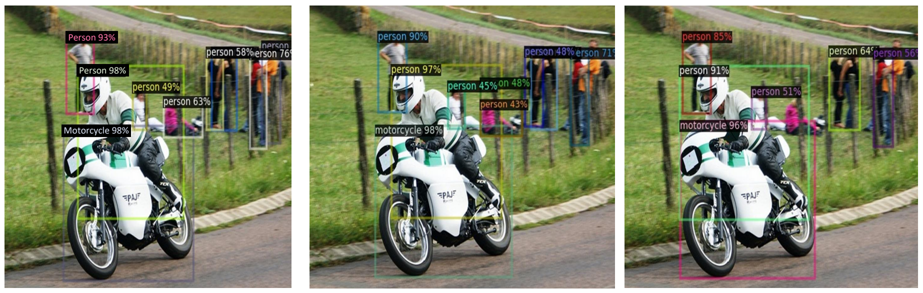
The graph construction of object features in this article includes the construction of nodes and adjacency matrices. It can be seen from Section 3.2.1 that the adjacency matrix is obtained by adding two parameter matrices. In order to prove the validity and rationality of the adjacency matrix construction, we combine the two parameter matrices by adding (Add), multiplying (Mul) and concatenating (Concat) methods, and the impact on the detector accuracy is shown in Table 7.

Table 7: Comparison results of different adjacency matrix construction methods on COCO2017.

Method	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Add	46.5	65.7	50.6	30.0	49.5	61.0
Mul	46.2	65.3	50.4	30.4	49.1	60.7
Concat	46.2	65.2	50.3	28.7	49.6	60.2

It can be seen from Table 7 that adding the parameter matrix to construct the adjacency matrix is an effective and reasonable construction way.

In order to compare the construction methods of the adjacency matrix more clearly, we visualize the adjacency matrices constructed by the three methods, and further verify the rationality of the graph construction by visualizing the detection results using different matrices. The matrix visualization and detection results are shown in Figure 5.



(a) The correlation matrix constructed by "Add" (b) The correlation matrix constructed by "Mul" (c) The correlation matrix constructed by "Concat"

Figure 5: Visualization detection results for different composition methods.

Figure 5 (a), (b) and (c) are the visualization results and detection results of the correlation matrices constructed by "Add", "Mul" and "Concat", respectively. It can be seen from the visualization results in Figure 5 that constructing the adjacency matrix in an additive manner is the most correct choice.

4.6 Visualization of Results

In order to more intuitively reflect the effectiveness of our method, we use our detector and the most advanced sparse detector Sparse R-CNN (Sun et al., 2021) to detect objects in the image, and visualize the detection results. The visual comparison results are shown in Figure 6.

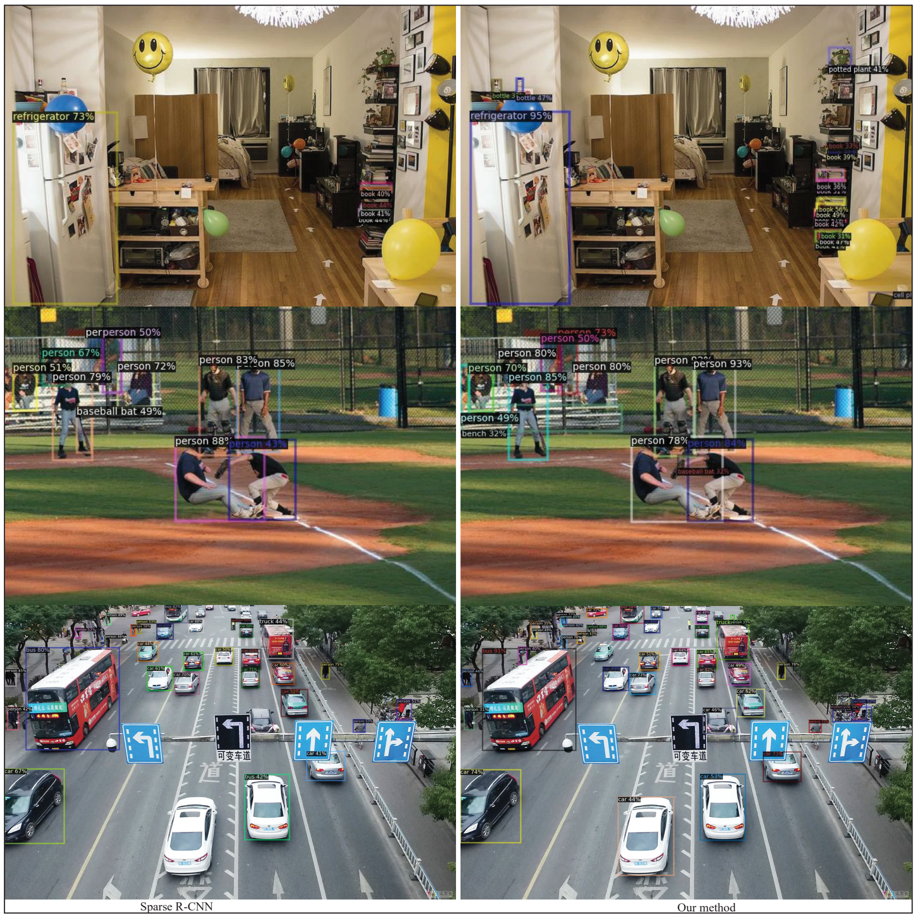


Figure 6: Comparison of visual detection results between our detector and Sparse R-CNN.

The detected image in Figure 6 contains small objects, overlapping objects, occlusion, etc., which increase the detection difficulty. From the comparison results of the two methods, it can be seen that the detection results of our method have a higher confidence score, and our detector can detect objects that Sparse R-CNN cannot detect, and has stronger detection abilities.

5. Conclusions

We propose DIGCN, a sparse detector based on learnable proposal boxes and GCN. A dynamic interaction GCN module is proposed for obtaining object feature, and finally the classification and location of the object can be directly output through two simple prediction layers to achieve object detection. In the process of designing the detector, we also build a specific graph structure for the dynamic interaction GCN module, which provides our detector with more detailed object information, establishing a foundation for improving the detector accuracy. Our detector on the COCO2017 dataset proves comparable detection accuracy, detection speed and modeling convergence accuracy with existing the state-of-the-art sparse detectors. Our work provides a new research idea for the application of GCN in object detection tasks.

Acknowledgments

This work is supported by 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.

References

Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Cai, Z., & Vasconcelos, N. (2018). Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6154–6162.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020a). End-to-end object detection with transformers. In *European Conference on Computer Vision*, pp. 213–229. Springer.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020b). End-to-end object detection with transformers., 213–229.

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.

Chen, S., Li, Z., Huang, F., Zhang, C., & Ma, H. (2021). Object detection using dual graph network. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 3280–3287. IEEE.

Chen, Y., Cao, Y., Hu, H., & Wang, L. (2020). Memory enhanced global-local aggregation for video object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10337–10346.

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 764–773.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee.

Du, X., Shi, X., & Huang, R. (2019). Repgn: Object detection with relational proposal graph network. *arXiv preprint arXiv:1904.08959*.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, *88*(2), 303–338.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition., 770–778.

Huang, L., Yang, Y., Deng, Y., & Yu, Y. (2015). Densebox: Unifying landmark localization with end to end object detection. *arXiv preprint arXiv:1509.04874*.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR.

Jia X, De Brabandere B, T., & V., G. L. (2016). Dynamic filter networks. In *Advances in neural information processing systems*, pp. 667–675.

Joshi, K. A., & Thakore, D. G. (2012). A survey on moving object detection and tracking in video surveillance system. *International Journal of Soft Computing and Engineering*, *2*(3), 44–48.

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kong, T., Sun, F., Liu, H., Jiang, Y., Li, L., & Shi, J. (2020). Foveabox: Beyound anchor-based object detection. *IEEE Transactions on Image Processing*, *29*, 7389–7398.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*, 1097–1105.

Law, H., & Deng, J. (2018). Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 734–750.

Li, F., Zhang, H., Liu, S., Guo, J., Ni, L. M., & Zhang, L. (2022). Dn-detr: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13619–13627.

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.

Lien, C.-H., Chen, P.-T., Bai, Y.-W., & Lin, M.-B. (2008). Monitoring system with moving object detection based on msn messenger. In *2008 IEEE Instrumentation and Measurement Technology Conference*, pp. 229–234. IEEE.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017a). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017b). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision*, pp. 21–37. Springer.

Mathieu, M., LeCun, Y., Fergus, R., Eigen, D., Sermanet, P., & Zhang, X. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks..

Najibi, M., Rastegari, M., & Davis, L. S. (2016). G-cnn: an iterative grid based object detector. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2369–2377.

Öztürk, Ş. (2021). Convolutional neural network based dictionary learning to create hash codes for content-based image retrieval. *Procedia Computer Science*, *183*, 624–629.

Qin, Q., Huang, L., Wei, Z., Xie, K., & Zhang, W. (2020). Unsupervised deep multi-similarity hashing with semantic structure for image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*.

Redmon, J., & Farhadi, A. (2017). Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, *28*, 91–99.

Song, J., He, T., Gao, L., Xu, X., Hanjalic, A., & Shen, H. T. (2018). Binary generative adversarial networks for image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

Stewart, R., Andriluka, M., & Ng, A. Y. (2016). End-to-end people detection in crowded scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2325–2333.

Sun, P., Zhang, R., Jiang, Y., Kong, T., Xu, C., Zhan, W., Tomizuka, M., Li, L., Yuan, Z., Wang, C., et al. (2021). Sparse r-cnn: End-to-end object detection with learnable proposals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14454–14463.

Szegedy, C., Toshev, A., & Erhan, D. (2013). Deep neural networks for object detection..

Tan, M., Pang, R., & Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10781–10790.

Tian, Z., Shen, C., & Chen, H. (2020). Conditional convolutions for instance segmentation. In *Computer Vision-ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pp. 282–298. Springer.

Tian, Z., Shen, C., Chen, H., & He, T. (2019). Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9627–9636.

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2023). Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7464–7475.

Wang, L., Wang, C., Zhang, X., Lan, T., & Li, J. (2021). S-at gcn: Spatial-attention graph convolution network based feature enhancement for 3d object detection. *arXiv preprint arXiv:2103.08439*.

Yang, B., Wang, J., Clark, R., Hu, Q., Wang, S., Markham, A., & Trigoni, N. (2019). Learning object bounding boxes for 3d instance segmentation on point clouds. *Advances in neural information processing systems*, *32*.

Yuan, T., Wan, F., Fu, M., Liu, J., Xu, S., Ji, X., & Ye, Q. (2021). Multiple instance active learning for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5330–5339.

Yuxin Wu, Alexander Kirillov, F. M. W.-Y. L., & Girshick, R. (2019). Detectron2.. Detectron2 https://github.com/facebookresearch/detectron2.

Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L. M., & Shum, H.-Y. (2022). Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*.

Zhao, G., Ge, W., & Yu, Y. (2021). Graphfpn: Graph feature pyramid network for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2763–2772.

Zhong, Z., Lei, M., Cao, D., Fan, J., & Li, S. (2017). Class-specific object proposals re-ranking for object detection in automatic driving. *Neurocomputing*, *242*, 187–194.

Zhou, T., Li, L., Li, X., Feng, C.-M., Li, J., & Shao, L. (2021). Group-wise learning for weakly supervised semantic segmentation. *IEEE Transactions on Image Processing*, *31*, 799–811.

Zhou, T., Wang, W., Qi, S., Ling, H., & Shen, J. (2020). Cascaded human-object interaction recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4263–4272.

Zhou, X., Wang, D., & Krähenbühl, P. (2019). Objects as points. *arXiv preprint arXiv:1904.07850*.

Zhu, C., He, Y., & Savvides, M. (2019). Feature selective anchor-free module for single-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 840–849.

Zhu, X., Su, W., Lu, L., Li, B., Wang, X., & Dai, J. (2020). Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*.