24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Learning Class-based Graph Representation for Object Detection

Shuyu Miao[1] and **Rui Feng**[1,*] and **Yuejie Zhang**[1,*] and **Weiguo Fan**[2] ***Senior Member*, *IEEE***

Abstract. Object detection has achieved a tremendous advancement based on feature-based learning in the vision space, while little work has focused on reasoning in the perception space like humans. One of the greatest challenges lies in that it is difficult to build a connectivity model in the topological space for relational reasoning, since the current network is better at modeling the distribution of structured data. To settle this issue, we introduce a novel graph modeling mechanism with class-based graph representation, which contributes to modeling the high-order topology structure that maps the data distribution to make the detection models have better relational reasoning ability. In this mechanism, we propose three learning subtasks, i.e., vision-to-perception embedding, perception reasoning graph representation, and perception-to-vision modeling. The mechanism based on such subtasks effectively maintains the independence of the original detection network and the proposed mechanism-based model, thus it can be well integrated with existing detection models without additional modification. The experimental results demonstrate the feasibility and effectiveness of our proposed mechanism, and the new state-of-the-art performance can be achieved on the public challenging datasets for object detection.

1 Introduction

The strong perception reasoning ability of humans makes it easy to detect objects in an image or even a blurred image. For example, the *slender chopsticks* next to a bowl on a dining table can be easily and correctly recognized by humans, while it may only be seen as a *slender stick* independently. When seeing a *blurred vehicle* on a river, humans tend to identify it as *a boat* rather than *a car*. These cases fully demonstrate that perception reasoning plays an irreplaceable role in human visual systems. To date, neither single-stage nor two-stage remarkable detection models [29, 38, 31, 21, 4] have a good perception reasoning ability, which mainly design more effective and sophisticated models for better feature mapping to achieve the better performance in the vision space. Therefore, endowing existing detection models with better perception reasoning ability like humans sheds new light on the development of object detection in the high-order space.

A great deal of previous research on object detection has focused on effectively boosting the performance in the vision space. SNIP [31] and Trident [19] addressed the issue of multi-scale detection via the optimization for the corresponding scale. Anchor-free frameworks [32, 41, 15] were designed to alleviate the limitation of feature embedding based on anchor box (anchor-scale, anchor-ratio, anchor-object mapping, etc.). Key-point-based models [40, 5, 17] generated key points, and grouped them to determine the objects. However, such studies remain narrow in focus dealing only with making better representations of the visual features for better feature-label mapping. Especially, the relational reasoning in the high-order topological space remains to excavate, which is more in line with human cognition for visual processing.

With the increasing emergence of unstructured graph representations, more and more graph models [2, 3, 35] are raised to realize relational connectionist modeling and perception reasoning. Figure 1 shows several kinds of graph structures to encode the reasoning relations via graph nodes and edges. Figure 1(a) expresses fully-connected relations to model the unstructured graph representations by global optimization through all graph nodes and edges. The graph is initialized based on prior knowledge to fit the original distribution of data structure. The models in [2, 3] were optimized by propagating the information in the global graph at a time, to encode the global mapping relations of grid-like data. However, it is redundant to update all the graph representations at each equilibrium, because many node-to-node mutual relationships do not appear in an image. Figure 1(b) is locally responded with no prior knowledge to focus on local optimization of graph structure, so that the problem of matrix redundancy is effectively alleviated. Xu et al. [35] relied on the extracted features from the detection network to promote the attribute representation of the graph nodes and edges, and the information flow passed in the relative spatial layout of the graph. However, such method easily results in the optimization of the graph jitter due to the lack of supervision with prior knowledge, which is not conducive to simulating the raw grid-like data representation. Thus to make the detection model reasoning like humans, the crucial issues that are tightly coupled with each other can be summarized as: (1) Modeling high-order topology structure of data to realize correlation reasoning like humans; (2) Learning the optimal graph representation based on prior knowledge via the most relevant adaptive learning; and (3) Decoupling the original detection network and graph structure, and making optimization via mutual supervision.

To address the above issues, we introduce a novel class-based graph representation mechanism based on Graph Convolutional Network (GCN) [14], called ClassGCN. It aims to model the high-dimensional topology structure that maps the data distribution, and make the detection models have better relational reasoning ability, as shown in Figure 1(c). Our mechanism divides the graph representation pipeline into the following three subtasks: (1) **Vision-to-perception embedding** – ClassGCN embeds the detection features of the network in the vision space to a perception reasoning graph, and obtains the supervised embedding; (2) **Perception reasoning**

[1] School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai, China, email: symiao18@fudan.edu.cn — * Corresponding author
[2] Department of Business Analytics, Tippie College of Business, University of Iowa, Iowa City, USA

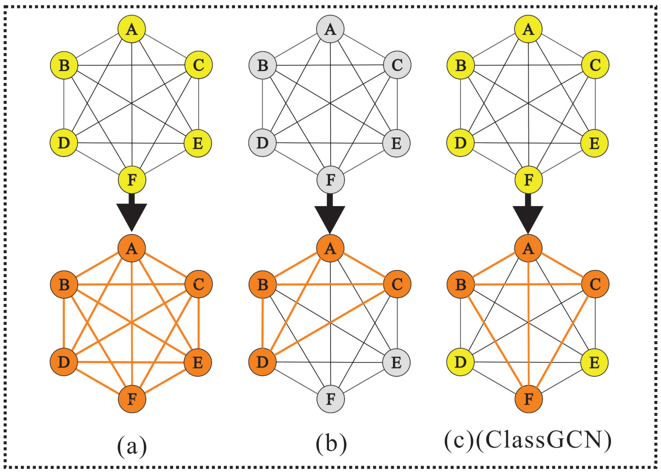

Figure 1. The comparison among different graph structures. (a) A fully connected and globally optimized graph based on prior knowledge; (b) A locally optimized graph with no prior knowledge; (c) Our ClassGCN with prior knowledge and adaptively local optimization-maximization. "A-F" are the nodes of the graph and mean the classes; the edge between two nodes denotes the class-class relationship; the node filled with yellow color represents the node with prior knowledge; the node filled with orange color indicates the relevant optimization; and the first and second rows represent the graphs without optimization and with optimization, respectively.

graph representation – the class-based graph with prior knowledge is represented by means of the locally maximized optimization learning and the supervised embedding; (3) **Perception-to-vision modeling** – the node-level output of the perception graph is encoded to supervise the optimization of detection model in the vision space. In ClassGCN, the encoded feature of detection network is adaptively employed as supervision to optimize the graph representation with local maximization based on prior knowledge, and greatly addresses the limitations of the abovementioned methods. Our method can bridge the gap between the vision hierarchy and the perception hierarchy, and makes the detection models have better reasoning ability. It has good adaptability and independence, so that it can be easily integrated into existing detection models. The experimental results on widely used *PascalVOC* [6] and *MSCOCO* [22] datasets validate that our ClassGCN can significantly improve the performance of existing detection models, and achieve the new state-of-the-art performance.

The main contributions of our work can be summarized as:

- A novel class-based graph representation mechanism, named ClassGCN, is proposed to make the existing detection models have better reasoning ability.
- ClassGCN based on three learning subtasks, i.e., vision-to-perception embedding, perception reasoning graph representation, and perception-to-vision modeling, sheds new light on the development of object detection in high-order space.
- The related experiments fully demonstrate the advancement and effectiveness of our proposed ClassGCN, which significantly improves the performance of detection models, and achieves the new state-of-the-art performance.

2 Related Work

2.1 Graph Representation Model

A number of studies have postulated a convergence between grid-like data interaction relationships and graph representation, which are widely used in various deep learning tasks like few-shot learning [13, 7, 9], action recognition [18, 30], and so on. Graph Convolution Network [14], which was obtained by the local first order approximation of a spectral convolution, completed the semi supervised learning task through layer by layer feature mapping. Chen et al. [2] mapped features from geometric coordinate space to interaction space for reasoning through graph convolution network, which focused on aggregating features of all adjacent pixels through an attention-like mechanism. Such graphs, however, are fully connected and pass the information stream among global nodes and edges, which results in generating redundant matrices because flow transmission is just locally related in some cases. Lee et al. [1] introduced an RNN model to concentrate on the small but informative parts of the graph, in order to denoise from the rest of the graph for the classification. Veličković et al. [33] adaptively assigned various weights via attention to neighbor nodes to highlight the most relevant ones. However, it is sub-optimal to directly apply such methods to object detection, because they rely on the attention weights generated by the graph itself and ignore the supervised auxiliary of the features extracted from the base model. Correspondingly, our ClassGCN achieves the local optimization-maximization of the graph by integrating the graph-level information with the supervised embedding encoded from the detection model in the vision space.

2.2 Object Detection in the Vision Space

A considerable amount of literature has been published with better performance based on the vision space. These studies can be grouped into two categories, that is, two-stage models and one-stage models. Classical R-CNN-based models [11, 10, 29] divided the detection process into region proposal extraction and object classification and regression to detect objects from coarse to fine. In contrast, object detection was treated as a regression problem trained end-to-end in [26, 27, 28, 24]. Although more recent models [21, 38, 25] significantly boosted the performance over classical models owing to better feature embedding, they were encumbered by the design of the feature-label mapping mechanism artificially. Wu et al. [34] explored the object detection by a segmentation method, whereas how to convert the detection labels to segmentation labels limited the performance of the model. Ghiasi et al. [8] used a neural architecture search to discover a new feature pyramid architecture in a new scalable search space covering all cross-scale connections, but huge computing resources were a huge burden. What the most important is that all these models build upon the vision space modeling, which leads to the lack of critical reasoning ability like humans.

2.3 Object Detection in the Perception Space

In the field of object detection, just a few works have attempted to utilize relation networks for better feature relation representation. Hu et al. [12] proposed the relation networks to establish the relationships among objects via learnable multi-head attention. However, the relation of the graph network was fully connected, which led to a redundant graph rather than concentrating on what was the most interesting relation. Xu et al. [35] introduced a spatial-aware graph relation network to adaptively discover and infer the objects by combining key semantics and spatial relations. However, the grid-like data of graph networks were generated entirely by the inference of detection networks, thus the data imbalance might weaken the performance of the class with fewer samples and the potential relationships of the graph nodes and labels had been under utilized. In contrast to these works, our method establishes a reasoning graph network based on prior

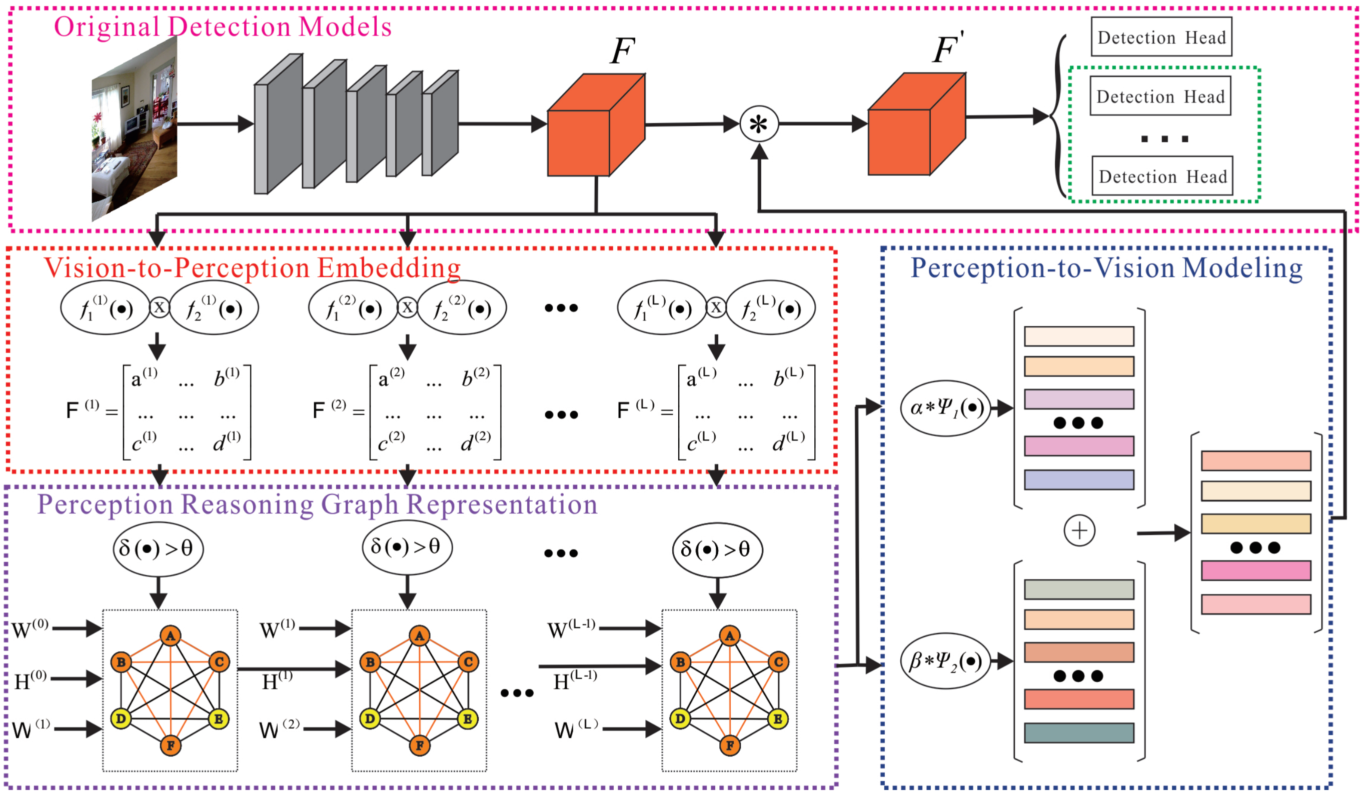
Original Detection Models
F
F'
Detection Head
Detection Head
Detection Head
Vision-to-Perception Embedding
Perception-to-Vision Modeling
Perception Reasoning Graph Representation
$\delta(\bullet)>\theta$
$\alpha*\Psi_1(\bullet)$
$\beta*\Psi_2(\bullet)$

Figure 2. An overview of our novel class-based graph representation mechanism for object detection. Our proposed method models the class-class relations in an image as a directed graph, which can be integrated into existing detection models to provide reasoning ability. "Vision-to-Perception Embedding" transforms the visual detection to perceptual reasoning; "Perception Reasoning Graph Representation" models a graph-level representation; and "Perception-to-Vision Modeling" encodes the node-level output of the graph to optimize the visual detection progressively. Our proposed ClassGCN is suitable for different detectors, whether with single or multiple detection heads.

knowledge, encodes the features of basic detection network to supervise the local optimization-maximization of the graph, and obtains the optimal graph-level representation.

3 Methodology

3.1 Overview

Inspired by human visual systems, we propose a novel class-based graph representation mechanism, ClassGCN, which is a class-class directed graph based on Graph Convolution Network (GCN) for object detection, as shown in Figure 2. ClassGCN can be easily integrated with current detection models to make them have better reasoning ability and improve their performance by a large margin. The GCN is described as $\mathcal{G} := f(\mathcal{V}, \mathcal{E}, \mathcal{X}_0, \mathcal{A})$, and defined by its nodes $\mathcal{V}$, edges $\mathcal{E}$, a feature matrix $\mathcal{X}_0$, and an adjacent matrix $\mathcal{A}$. The objective of GCN is to learn a non-linear function $f(\cdot)$ to optimize the global propagation among nodes and edges. However, such representation results in the redundant matrix and over-optimization for the grid-like data as mentioned above. To address this issue, we maximize the local response optimization of the graph by self-learning, which can be formulated as:

$$\mathcal{G} = f(\mathcal{V}, \mathcal{E}, \mathcal{X}_0, \mathcal{A}, \mathcal{F}, \mathcal{W}), \tag{1}$$

where $\mathcal{F}$ and $\mathcal{W}$ represent the supervised information for the optimization of $\mathcal{V}$ and $\mathcal{E}$, respectively.

In our proposed ClassGCN, the vision-to-perception embedding subtask encodes the feature maps from the basic detection model via cross-non-linear functions, to generate the corresponding supervised information for diverse GCN layers. The perception reasoning graph representation subtask adaptively maximizes the local optimization of the l^{th} layer $\boldsymbol{H}$ of the graph by combining the supervision information $\Upsilon \Leftarrow \{\mathcal{F}, \mathcal{W}\}$ with the node-level output of the front GCN layer and adjacent matrix, which can be formulated as the following non-linear function.

$$\boldsymbol{H}^{(l)} = \sigma(\boldsymbol{H}^{(l-1)}, \mathcal{A}, \Upsilon). \tag{2}$$

In the perception-to-vision modeling subtask, the node-level output of the last graph layer is re-encoded by the non-linear functions, which aims to supervise the detection in the vision space. As follows, we will introduce vision-to-perception embedding for embedding the visual detection to perceptual reasoning, perception reasoning graph representation for learning a graph-level representation, and perception-to-vision modeling for modeling the node-level output of the graph to optimize the visual detection in detail.

3.2 Vision-to-Perception Embedding

The vision-to-perception embedding subtask aims to explore the non-linear function $g(\cdot)$ to generate the supervised embedding $\mathcal{F}$ for graph optimization by encoding the features of the basic detection network, and bridge the representation from the vision space to the perception space. We initialize the feature map of the basic detection network as $F \in \mathbb{R}^{W\times H\times C}$, where W denotes the width of the feature map, H is the height, and C is the channel. Hence we can get $\mathcal{F} \xleftarrow{g(\cdot)} F$. There are different graph layers in the reasoning graph, and the corresponding supervised embedding needs to be generated for such layers. For the i^{th} graph layer, the function $g^{(i)}(\cdot)$ is used to encode F to obtain the corresponding $\mathcal{F}^{(i)}$, and then the following Eq. (3) is obtained.

$$\mathcal{F}^{(i)} = g^{(i)}(F), \;\; s.t.\, i \in \{1, \cdots, \mathcal{L}\}, \tag{3}$$

where $\mathcal{L}$ is the number of graph layers.

The original feature matrix is defined as $\mathcal{X}_0 \in \mathbb{R}^{N \times D}$, where N is the number of nodes and D is the number of original input features. The feature dimension of each graph layer is expressed as $\mathcal{B}$ and gained by the following Eq. (4).

$$\mathcal{B}^{(i)} = \begin{cases} D & \text{if } i = 0 \\ C + \frac{i*(C-D)}{\mathcal{L}} & \text{if } i = \{1, \cdots, \mathcal{L}-1\} \\ C & \text{if } i = \mathcal{L} \end{cases} . \tag{4}$$

The feature matrix $\mathcal{X}^{(i-1)} \in \mathbb{R}^{N \times \mathcal{B}^{(i-1)}}$ is the input feature matrix of the i^{th} graph layer, and $\mathcal{X}^{(i)} \in \mathbb{R}^{N \times \mathcal{B}^{(i)}}$ is the node-level output of the i^{th} graph layer. In our network, the supervised embedding $\mathcal{F}^{(i)} \in \mathbb{R}^{N \times \mathcal{B}^{(i-1)}}$ of the i^{th} graph layer corresponds to $\mathcal{X}^{(i-1)}$. This special progressive design can respond to the feature mapping with different scales, retain the semantics among cross-domain features, and optimize the feature correlation representation.

As shown in Figure 2, we feed F into two parallel convolution layers to get two non-linear functions $g_b^{(i)}(\cdot)$ and $g_n^{(i)}(\cdot)$ for the i^{th} graph layer to encode multi-scale high-order multimodal features. The first layer consists of the convolution with the kernel of $\{C \times 1 \times 1, \mathcal{B}^{(i-1)}\}$, and the second one consists of the convolution with the kernel of $\{C \times 1 \times 1, N\}$. Both layers are followed by batch normalization and ReLU activation to make the feature-encoding flow more robust and softer, which can be formulated as $F_b^{(i)} = g_b^{(i)}(F)$, and $F_n^{(i)} = g_n^{(i)}(F)$, where $F_b^{(i)} \in \mathbb{R}^{W \times H \times \mathcal{B}^{(i-1)}}$ and $F_n^{(i)} \in \mathbb{R}^{W \times H \times N}$ are the outputs of the two layers separately. We reshape the size of $F_b^{(i)}$ to $\mathbb{R}^{WH \times \mathcal{B}^{(i-1)}}$ and $F_n^{(i)}$ to $\mathbb{R}^{N \times WH}$ as the operation $\tau(\cdot)$, multiply them by matrix product multiplication to leverage dependency relations of feature semantics, and get the supervised signals $\mathcal{F}^{(i)} \in \mathbb{R}^{N \times \mathcal{B}^{(i-1)}}$. Finally Eq. (3) is expanded to the following Eq. (5).

$$\mathcal{F}^{(i)} = [g_n^{(i)}(F)]^{\tau} \otimes [g_b^{(i)}(F)]^{\tau}, \;\; s.t.\, i \in \{1, \cdots, \mathcal{L}\}, \tag{5}$$

where $\otimes$ means product multiplication.

The vision-to-perception embedding subtask employs the synergies of the cross-non-linear functions to simulate the multimodal representation. Furthermore, low-order visual information is transited to high-order perceptual representation via graph matrix operations. The feature F is encoded to generate the supervised information $\mathcal{F}$, which is a sparse matrix with the corresponding weights and carries the optimization relationship of the graph by the feature matrix.

3.3 Perception Reasoning Graph Representation

We generate a perception reasoning graph to achieve reasoning based on GCN, which is used for the classification task originally. Every GCN layer aims to learn the graph propagation rule to represent the grid-like data, which can be formulated as:

$$\begin{aligned} \boldsymbol{H}^{(l)} &= \sigma(\boldsymbol{H}^{(l-1)}, \widehat{\mathcal{A}}) \\ &= \sigma(\widehat{\mathcal{A}} \boldsymbol{H}^{(l-1)} W^{(l-1)}) \\ &= \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{\mathcal{A}} \tilde{D}^{-\frac{1}{2}} \boldsymbol{H}^{(l-1)} W^{(l-1)}), \end{aligned} \tag{6}$$

where $\tilde{\mathcal{A}} = \mathcal{A} + I$; $\tilde{D}$ is the diagonal node degree matrix of $\tilde{\mathcal{A}}$; I is the identity matrix; W is a weight matrix for the corresponding graph layer; and σ is an activation function (i.e., LeakyReLU in our work). Moreover, the first input node-level feature matrix is $\mathcal{X}_0$, thus $\boldsymbol{H}^{(0)} = \mathcal{X}_0$. We find that it is necessary to propagate information among nodes based on prior knowledge, which can better guide the optimization of graph network.

Our model constructs a class-class directed graph, in which the nodes denote the classes and the edges indicate the mutual synergies among the classes in the dataset space. The original feature matrix $\mathcal{X}_0$ and the representative description of the graph structure with the adjacent matrix $\mathcal{A}$ are the prior knowledge of our graph network. Two manners are adopted to encode the feature matrix. (1) We group the regions of the ground truth bounding boxes of each class in the dataset, embed the features in the corresponding regions to obtain the mean feature presentation, and encode them into the feature matrix; and (2) We encode each class by the word embedding to get the corresponding feature matrix. Given the number of the classes in the dataset is N and the feature dimension of each class is encoded as D, the features of N classes are stacked to gain the feature matrix $\mathcal{X}_0 \in \mathbb{R}^{N \times D}$. To better represent the class-class correlations for mining the value of prior knowledge in the dataset, a representative description of the graph structure in the matrix form is constructed and represented as $\mathcal{A} \in \mathbb{R}^{N \times N}$. The correlation among classes is defined via conditional coexistence class-class pairs derived from [3].

The uniform propagation in the whole graph produces a dense matrix, which is not the best choice with noise information for the graph optimization. Whereas the supervised signal Υ is used to supervise the local optimization-maximization. Υ contains two parts, that is, $\mathcal{F}$ for $\mathcal{V}$ and $\mathcal{W}$ for $\mathcal{E}$. Concretely, $\mathcal{F}$ is generated by Eq. (5), and $\mathcal{W}$ is a self-learning parameter with the size of $\mathbb{R}^{N \times N}$ experimentally. $\mathcal{F}$ and $\mathcal{W}$ are processed by sigmoid$(\cdot)$ activation function to relieve feature shuffle. The supervised signals of the i^{th} graph layer are expressed as $\mathcal{F}^{(i)} \xleftarrow{\delta(\cdot)} \mathcal{F}^{(i)}$, and $\mathcal{W}^{(i)} \xleftarrow{\delta(\cdot)} \mathcal{W}^{(i)}$. We then prune the matrix data over the threshold θ to maximize the local optimization, which can be formulated as:

$$\mathcal{F}^{(i)}_{\tilde{j}*,\tilde{k}*} = \begin{cases} \mathcal{F}^{(i)}_{\tilde{j}*,\tilde{k}*} & \text{if } \mathcal{F}^{(i)}_{\tilde{j}*,\tilde{k}*} \geq \theta \\ 0 & \text{otherwise} \end{cases} , \tag{7}$$

where $\tilde{j}* \in \{1, \cdots, N\}, \tilde{k}* \in \{1, \cdots, \; \mathcal{B}_{i-1}\}$.

$$\mathcal{W}^{(i)}_{\ddot{j}*,\ddot{k}*} = \begin{cases} \mathcal{W}^{(i)}_{\ddot{j}*,\ddot{k}*} & \text{if } \mathcal{W}^{(i)}_{\ddot{j}*,\ddot{k}*} \geq \theta \\ 0 & \text{otherwise} \end{cases} , \tag{8}$$

where $\{\ddot{j}*, \ddot{k}*\} \in \{1, \cdots, N\}$. In our experiments, θ is set as 0.5 empirically to effectively generate the sparse matrices for local optimization by thresholding setting. We optimize Eq. (6) as:

$$\begin{aligned} \boldsymbol{H}^{(l)} &= \sigma(\boldsymbol{H}^{(l-1)}, \widehat{\mathcal{A}}, \mathcal{F}^{(l)}, \mathcal{W}^{(l)}) \\ &= \sigma\big((\widehat{\mathcal{A}} \odot \mathcal{W}^{(l)})(\boldsymbol{H}^{(l-1)} \odot \mathcal{F}^{(l)}) W^{(l-1)}\big), \end{aligned} \tag{9}$$

where $\odot$ represents the Hadamard product, which is based on the pixel-level operation to maintain the optimal feature-supervision mapping distribution. By utilizing the supervised embedding encoded from the basic detection network and depending on prior knowledge, the model can better promote the graph local optimization-maximization, generate sparse matrix, and obtain the best grid-like graph representation for leveraging perception reasoning.

3.4 Perception-to-Vision Modeling

The node-level output of the final graph layer is $\boldsymbol{X} \in \mathbb{R}^{N \times C}$. We map the node-level output in the perception space to the detection in the vision space by feature modeling, and supervise the detection optimization progressively. $\boldsymbol{X}$ is encoded by two manners, ψ_1 and ψ_2, in order to model the global and local optimal feature representation separately. The encoded $\boldsymbol{X}$ utilizes two learnable weights, α and β, to adaptively learn the best way to integrate the global and local representation, which is obtained by:

$$\boldsymbol{X} = \alpha\psi_1(\boldsymbol{X}) \oplus \beta\psi_2(\boldsymbol{X}). \tag{10}$$

In the specific implementation, ψ_1 is achieved by an average pooling function to embed the global representation, and ψ_2 by a max pooling function in the first dimension to model the local representation. The global and local representation are then aggregated by using two learnable parameters to maximize the most favorable supervised information. α and β are initialized as $\mathbb{1}$, and adaptively assigned the best weights gradually with end-to-end training.

The encoded supervision from the graph-level representation is mapped to the detection in the vision space to facilitate optimization. Each channel of the feature map often corresponds to various interested targets, and the dimensions of the channel and $\boldsymbol{X}$ are generated with the same size as C. The supervised embedding as the attention weight is mapped to the channel to focus on the most valuable channel. Finally, the node-level output of the perception reasoning graph is encoded as the supervision, which can be utilized as attention in the channel dimension of the feature map to optimize the detection in the vision space.

4 Experiments

4.1 Dataset and Evaluation Metric

We carry out the experiments on two widely used benchmark datasets for object detection, *PascalVOC* [6] and *MSCOCO* [22].

- ***PascalVOC*** It contains 20 classes of objects in daily life, which consists of two versions, that is, *PascalVOC*2007 (including 5k training images and 12k annotated objects) and *PascalVOC*2012 (including 11k training images and 27k annotated objects). We train the ClassGCN on the union set of *PascalVOC*2007 and *PascalVOC*2012 `trainval`, and test on the *PascalVOC*2007 `test` set.
- ***MSCOCO*** It is a large-scale dataset including 80 classes. We use the *MSCOCO* `trainval35k` set (including 80k training images and 35k validation images) for training, and the *MSCOCO* `test-dev` set for testing.

The same and popular evaluation metrics in all experiments are adopted to measure the performance of the models on both datasets.

- **Evaluation on *PascalVOC*** The mean Average Precision (mAP) is adopted as the evaluation metric to evaluate the model performance. When the Intersection over Union (IoU) between the predicted box and the ground truth box is more than 0.5, the predicted box is labeled as "*positive*", otherwise as "*negative*".
- **Evaluation on *MSCOCO*** The Average Precision (AP) is adopted as the evaluation metric to evaluate the model performance via the official *COCO API*[3].

[3] https://github.com/cocodataset/cocoapi

4.2 Implementation Details

To demonstrate the effectiveness of our proposed ClassGCN, we integrate it into the classical detection models SSD [24], RFBNet [23], and RetinaNet [21]. More specifically, ClassGCN is embedded between the basic backbone for feature extraction and the detection heads of these models. To be fair, we adopt the same strategy, such as training datasets, loss functions, matching strategy, training objective, scales and aspect ratios for default boxes, hard negative mining, data augmentation, non-maximum suppression step, and other basic settings, as the baseline models. Our network is implemented based on *Pytorch*[4]. The hyperparameter θ is set as 0.5 empirically in all experiments. SGD is applied to optimize the training models on *NVIDIA* 1080*Ti*. More importantly, we adopt the consistent parameter setting, train and test the model on the consistent datasets with the consistent detection backbone.

4.3 Overall Performance

We evaluate the performance of our proposed ClassGCN on *PascalVOC* and *MSCOCO* with the consistent experimental setting and single-scale training strategy. The related experimental results are shown in Tables 1 and 2. The number in the "Method", like 300 or 512, means the network with the corresponding input scale.

Results on *PascalVOC*. Table 1 presents the summary statistics for the performance of ClassGCN on *PascalVOC*. We verify the performance of ClassGCN on SSD and RFBNet. As shown in Table 1, *ClassGCN* (1), (2), (3), and (4) mean the experiments are conducted by integrating our proposed ClassGCN into the SSD model, but (1) with no prior knowledge, (2) with global optimization, and (3) and (4) with full ClassGCN; *ClassGCN* (5) and (6) mean the experiments are conducted by integrating ClassGCN into the RFBNet model. Closer inspection of the table shows that for the SSD network, ClassGCN significantly improves the mAP by 3.2% from 77.5% to 80.7% in the 300x300 input scale, and increases the mAP by 2.6% from 79.5% to 82.1% in the 512x512 input scale; for RFBNet, the mAP is improved by 1.7% in both the 300x300 and 512x512 input scales. These results suggest that the proposed ClassGCN is not only effective for different models, but also for multi-scale inputs. More importantly, the mAP is just improved by 1.9% for SSD based on ClassGCN with no prior knowledge, and 2.2% based on ClassGCN with global optimization in the 300x300 input scale, which reveals the rationality of our proposed ClassGCN based on prior knowledge with adaptively local optimization-maximization. What is striking about the figures in this table is that ClassGCN can adapt to the object-level detection task, and significantly elevate the detection performance of diverse categories, like *boat*, *bottle*, *bus*, and so on. It can be observed that ClassGCN achieves the new state-of-the-art performance on *PascalVOC*.

Results on *MSCOCO*. Table 2 illustrates the summary statistics for the performance of ClassGCN on *MSCOCO*. We test the performance of ClassGCN on SSD (ClassGCN300 and ClassGCN512) and RetinaNet (ClassGCN500 and ClassGCN800). It can be seen from Table 2 that for the SSD model with the 300x300 input scale, we can obtain 2.0%, 1.8%, and 2.1% improvement of AP, AP$_{50}$, and AP$_{75}$ separately, which shows the robustness of ClassGCN; for the SSD model with the 512x512 input scale, these metrics are raised by 1.9%, 2.0%, and 1.7%, respectively. It is worth noting that for the state-of-the-art RetinaNet model, ClassGCN is equally efficient. We can notice that ClassGCN outperforms the other detection models, and promotes the metrics of AP, AP$_{50}$, and AP$_{75}$ by 2.0%, 1.9%,

[4] https://pytorch.org

Table 1. The detection results on *PascalVOC*2007 `test`. The bold fonts indicate the best performance, and '–' means that no relevant data is provided in the original work. (1), (2), (3), and (4) mean the experiments are conducted by integrating our proposed ClassGCN into SSD [24], but (1) with no prior knowledge, (2) with global optimization, and (3) and (4) with full ClassGCN; (5) and (6) mean the experiments are conducted by integrating ClassGCN into RFBNet [23]. Note that all these models are trained on *PascalVOC*2007 `trainval` and *PascalVOC*2012 `trainval`.

Method	Backbone	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	persn	plant	sheep	sofa	train	tv
Fast R-CNN [10]	VGG-16	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster R-CNN [29]	VGG-16	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster R-CNN [29]	ResNet-101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	89.8	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72.0
R-FCN [4]	ResNet-101	80.5	79.9	87.2	81.5	72.0	69.8	86.8	88.5	89.8	67.0	88.1	74.5	**89.8**	**90.6**	79.9	81.2	53.7	81.8	81.5	85.9	79.9
RON384++ [16]	VGG-16	77.6	86.0	82.5	76.9	69.1	59.2	86.2	85.5	87.2	59.9	81.4	73.3	85.9	86.8	82.2	79.6	52.4	78.2	76.0	86.2	78.0
DES300 [37]	VGG-16	79.7	83.5	86.0	78.1	74.8	53.4	87.9	87.3	88.6	64.0	83.8	77.2	85.9	88.6	87.5	80.8	57.3	80.2	80.4	88.5	79.5
RefineDet320 [36]	VGG-16	80.0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
STDN300 [39]	DenseNet-169	78.1	81.1	86.9	76.4	69.2	52.4	87.7	84.2	88.3	60.2	81.3	77.6	86.6	88.9	87.8	76.8	51.8	78.4	81.3	87.5	77.8
DES512 [37]	VGG-16	81.7	**87.7**	86.7	**85.2**	76.3	60.6	88.7	89.0	88.0	67.0	86.9	78.0	87.2	87.9	87.4	84.4	59.2	86.1	79.2	88.1	80.5
RefineDet512 [36]	VGG-16	81.8	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
STDN513 [39]	DenseNet-169	80.9	86.1	**89.3**	79.5	74.3	61.9	88.5	88.3	89.4	67.4	86.5	79.5	86.4	89.2	88.5	79.3	53.0	77.9	81.4	86.6	**85.5**
SSD300 [24]	VGG-16	77.5	79.5	83.9	76.0	69.6	50.5	87.0	85.7	88.1	60.3	81.5	77.0	86.1	87.5	84.0	79.4	52.3	77.9	79.5	87.6	76.8
SSD512 [24]	VGG-16	79.5	84.8	85.1	81.5	73.0	57.8	87.8	88.3	87.4	63.5	85.4	73.2	86.2	86.7	83.9	82.5	55.6	81.7	79.0	86.6	80.0
RFBNet300 [23]	VGG-16	80.5	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
RFBNet512 [23]	VGG-16	82.2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ClassGCN300 (1)	VGG-16	79.4	82.8	86.0	77.5	72.5	53.7	88.8	87.9	87.8	62.2	63.5	79.5	85.9	88.4	87.1	81.0	55.2	81.5	78.1	88.4	79.4
ClassGCN300 (2)	VGG-16	79.7	83.4	87.9	77.9	72.7	55.3	87.4	87.7	88.1	63.1	83.2	76.7	87.1	88.0	88.2	82.1	57.3	80.8	80.1	88.1	79.0
ClassGCN300 (3)	VGG-16	80.7	85.2	88.3	79.0	74.9	56.6	87.2	88.1	89.3	65.4	84.6	78.3	87.6	88.9	88.7	82.6	58.0	81.5	80.8	87.5	81.7
ClassGCN512 (4)	VGG-16	82.1	87.6	88.5	82.1	77.2	66.8	88.8	89.3	88.2	65.8	88.5	76.8	87.1	89.3	87.9	84.3	58.7	83.9	79.7	88.5	82.3
ClassGCN300 (5)	VGG-16	82.2	87.1	89.1	82.4	76.3	63.9	89.1	88.8	88.9	67.6	87.5	78.9	86.7	89.0	89.0	84.7	60.4	80.1	**81.9**	88.8	84.2
ClassGCN512 (6)	VGG-16	**83.9**	86.4	89.1	84.4	**79.6**	**70.2**	**90.1**	**89.9**	**89.9**	**68.9**	**89.2**	**80.7**	88.2	**90.6**	**89.6**	**86.3**	**62.1**	**86.5**	80.3	**89.9**	85.0

Table 2. The detection results on *MSCOCO* `test-dev` set. (1) and (2) mean the experiments are conducted by integrating our proposed ClassGCN into SSD [24]; (3) and (4) mean the experiments are conducted by integrating ClassGCN into RetinaNet [24].

Method	Data	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN [29]	trainval	VGG-16	21.9	42.7	-	-	-	-
R-FCN [4]	trainval	ResNet-101	29.9	51.9	-	10.8	32.8	45.0
CoupleNet [42]	trainval	ResNet-101	34.4	54.8	37.2	13.4	38.1	50.8
Faster R-CNN w FPN [20]	trainval35k	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
YOLOv3 [28]	trainval35k	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9
RefineDet512 [36]	trainval35k	ResNet-101	36.4	57.5	39.5	16.6	39.9	51.4
FSAF [41]	trainval35k	ResNet-101	–	54.0	33.6	17.8	35.4	46.5
CornerNet511 [17]	trainval35k	Hourglass-104	40.5	56.5	43.1	19.4	42.7	53.9
SSD300 [24]	trainval35k	VGG-16	25.1	43.1	25.8	6.6	25.9	41.4
SSD512 [24]	trainval35k	VGG-16	28.8	48.5	30.3	10.9	31.8	43.5
RetinaNet500 [21]	trainval35k	ResNet-101	34.4	53.1	36.8	14.7	38.5	49.1
RetinaNet800 [21]	trainval35k	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
ClassGCN300 (1)	trainval35k	VGG-16	27.1	44.9	27.9	8.1	27.6	44.2
ClassGCN512 (2)	trainval35k	VGG-16	30.7	50.5	32.0	13.8	34.6	47.1
ClassGCN500 (3)	trainval35k	ResNet-101	36.4	55.0	39.0	16.7	40.2	51.1
ClassGCN800 (4)	trainval35k	ResNet-101-FPN	**41.4**	**61.4**	**44.8**	**23.8**	**44.9**	**52.5**

2.2% with the 500x500 input scale and 2.3%, 2.3%, 2.5% with the 800x800 input scale.

Comparison with Stated-of-the-art Models. We can summarize that no matter the 300x300, 500x500, 512x512, or 800x800 input scales, the performances of the baseline models all surpass the original detection performance, which exhibits the good adaptability of our method. From Tables 1, 2 and Figure 4, it is interesting to note that our proposed ClassGCN outperforms other excellent models like [36, 39, 23, 24, 37, 29, 4] on *Pascal*. Meanwhile, compared with the most advanced models [17, 42, 41, 36, 4] on *MSCOCO*, the RetinaNet with ClassGCN can achieve better performance.

4.4 Ablation Study

To verify the reasonability and reliability of our proposed ClassGCN, we perform the related ablation studies. We train the models on the union set of the *PascalVOC*2007 and *PascalVOC*2012 `trainval` set, and test them on the *PascalVOC*2007 `test` set. All ablation studies are performed on the SSD model in the 300x300 input scale with the same experimental settings.

Experiments on Different Settings of The Hyper-parameter θ. The setting of the hyper-parameter θ determines the scope of local optimization. The experimental results for ClassGCN with different settings of θ are shown in Table 3. It can be viewed that setting θ to a fixed value 0.1 is the worst choice, because this is largely equivalent to the effect of global optimization. Fixing θ as 0.5 can get the best performance, which is beneficial in fitting the most primitive data distribution and guiding the optimization of the graph. It is apparent that setting θ as 0.9 leads to the network failing to converge, and the

Table 3. The experimental results for our ClassGCN with different settings of the hyper-parameter θ.

θ	0.1	0.3	0.5	0.7	0.9
mAP	79.9	80.2	80.7	80.0	N/A

Table 4. The experimental results for ClassGCN with different feature-modeling styles.

Operation	[ψ_1]	[ψ_2]	[ψ_1, ψ_2]
mAP	80.2	79.9	80.7

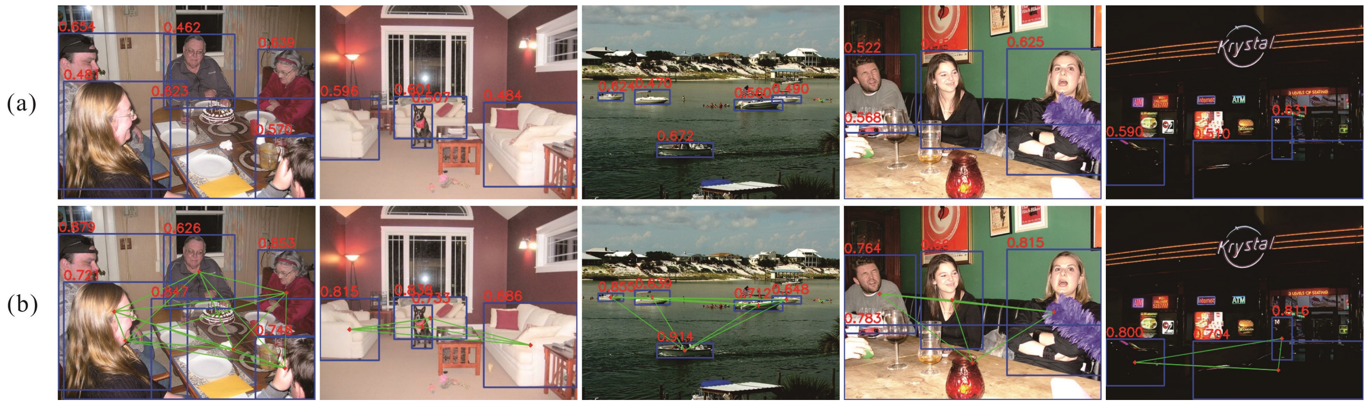

Figure 3. Some examples of learned class-based graph representation from our ClassGCN. The red numbers are the predicted classification scores; the blue boxes are the predicted bounding boxes; and the green graph structures are the graph relations with various weights. Row (a) is the detection visualization of the base model [24], and Row (b) by means of our ClassGCN. Our method outperforms baseline with better classification scores due to the help of graph relations.

main reason is that the embedding matrix with a lot of zero values is too sparse to cause weight imbalance.

Experiments on Feature-modeling from Perception to Vision. The node-level output of the graph is fed into two functions ψ_1 and ψ_2, to obtain the most compete encoding information. As shown in Table 4, it can be found that encoding features with a combination of two functions increases the model performance than using each separate one by 0.5% and 0.8%, respectively. ψ_1 can fully embed the global representation of the graph, and ψ_2 can model the most effective local representation. Two encoding styles, along with two self-learning weights, can obtain the optimal supervised embedding, which is adopted by our proposed method.

Table 5. The experimental results for ClassGCN with different numbers of graph layer. 'M' represents million.

Graph layer	SSD	1	2	3	4	5
Parameters	26.3M	0.3M	0.7M	1.1M	1.5M	1.9M
mAP	77.5	79.7	80.7	80.8	80.4	80.1

Experiments on Different Numbers of Graph Layer. To determine the most appropriate layer setting, we explore the comparison with different numbers of graph layer, as shown in Table 5. Keeping other parameters consistent, only the layers of the graph are changed. We can intuitively find that the performance of the graph reasoning network is not positively related to the number of layer. Graph model can obtain almost the best similar performance when the graph layer is 2 or 3. Finally, we choose two layers as the graph structure to better balance the trade-off between the performance and parameter. The model without and with ClassGCN achieves the speed of 53 and 51 frames per second (FPS), respectively. ClassGCN can greatly improve the performance of the original detection model at little cost.

4.5 Visualization and Discussion

To reveal the efficiency of our method intuitively, we visualize some representative examples, as shown in Figure 3. The most important relevant finding is that the detection models based on the vision space obtain better performance at the aid of class-class relations by the graph representation in the perception space. This can provide some supports for the conceptual premise that using graph representation to model the topological structure of high-order unstructured data is of great significance for low-order visual object detection. In addition, we find that the performance of ClassGCN is not so significant

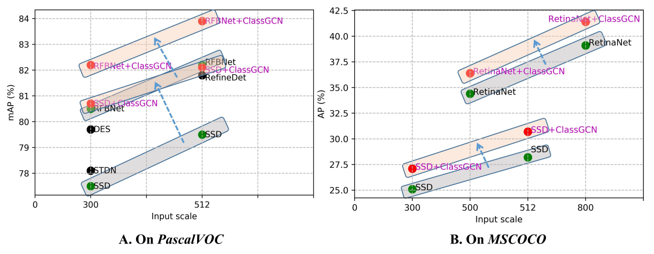

Figure 4. The comparison results between the stated-of-the-art detection models and our proposed ClassGCN in different input scales. "- - →" denotes the performance improvement by ClassGCN. It is clear that the performance of the baseline models can be greatly improved by integrating ClassGCN.

when there is only one object in the image. Because our method aims to build the topological space relationships among different objects, which cannot effectively model the mutual support connectivity based on one object.

5 Conclusion and Future Work

In this paper, considering that most of the existing object detection models are only modeled in the vision space and little work has focused on reasoning in the perception space like humans, a novel class-based graph representation mechanism ClassGCN is proposed to make the detection models adaptively learn to model the high-order topology structure that maps the data distribution and have better reasoning ability in the perception space. Our proposed ClassGCN is adaptable, simple, and highly effective, which can be easily integrated into the existing models without any extra modification to such models. Extensive experiments show that our method can greatly boost the model performance, while keeping the model highly efficient. In the future, we will explore to design the independent perceptual reasoning detection framework without building on the existing detection models.

ACKNOWLEDGEMENTS

This work was supported in part by National Natural Science Foundation of China (No. 61976057, No. 61572140), Military Key Research Foundation Project (No. AWS15J005), National Key Research and Development Program (No. 17YFC08037020), and Shanghai Science and Technology Development Fund (No. 7511101702, No. 19DZ2205700).

REFERENCES

[1] John Boaz Lee, Ryan Rossi, and Xiangnan Kong, 'Graph classification using structural attention', *The 24th ACM SIGKDD International Conference*, 1666–1674, (2018).

[2] Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Shuicheng Yan, and Yannis Kalantidis, 'Graph-based global reasoning networks', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 433–442, (2019).

[3] Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo, 'Multi-label image recognition with graph convolutional networks', in *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2019).

[4] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun, 'R-FCN: object detection via region-based fully convolutional networks', *In Conference and Workshop on Neural Information Processing Systems (NeurIPS)*, 379–387, (2016).

[5] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian, 'Centernet: Keypoint triplets for object detection', *In arxiv, abs/1904.08189*, (2019).

[6] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman, 'The pascal visual object classes (voc) challenge', *In International Journal of Computer Vision (IJCV)*, (2), 303–338, (2010).

[7] Victor Garcia and Joan Bruna, 'Few-Shot Learning with Graph Neural Networks', *The International Conference on Learning Representations (ICLR)*, (2018).

[8] Golnaz Ghiasi, Tsung-Yi Lin, Ruoming Pang, and Quoc V. Le, 'Nas-fpn: Learning scalable feature pyramid architecture for object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2019).

[9] Spyros Gidaris and Nikos Komodakis, 'Generating classification weights with gnn denoising autoencoders for few-shot learning', in *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21–30, (2019).

[10] Ross B. Girshick, 'Fast R-CNN', *In IEEE International Conference on Computer Vision (ICCV)*, 1440–1448, (2015).

[11] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, 'Rich feature hierarchies for accurate object detection and semantic segmentation', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 580–587, (2014).

[12] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei, 'Relation networks for object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3588–3597, (2018).

[13] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D. Yoo, 'Edge-labeling graph neural network for few-shot learning', in *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11–20, (2019).

[14] Thomas N. Kipf and Max Welling, 'Semi-supervised classification with graph convolutional networks', *International Conference on Learning Representations (ICLR)*, (2017).

[15] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, and Jianbo Shi, 'Foveabox: Beyond anchor-based object detector.', *arXiv: Computer Vision and Pattern Recognition*, (2019).

[16] Tao Kong, Fuchun Sun, Anbang Yao, Huaping Liu, Ming Lu, and Yurong Chen, 'Ron: Reverse connection with objectness prior networks for object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5244–5252, (2017).

[17] Hei Law and Jia Deng, 'Cornernet: Detecting objects as paired keypoints', *In European Conference on Computer Vision (ECCV)*, 765–781, (2018).

[18] Chaolong Li, Zhen Cui, Wenming Zheng, Chunyan Xu, and Jian Yang, 'Spatio-temporal graph convolution for skeleton based action recognition', *In AAAI Conference on Artificial Intelligence(AAAI)*, 3482–3489, (2018).

[19] Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang, 'Scale-aware trident networks for object detection', *In IEEE International Conference on Computer Vision (ICCV)*, (2019).

[20] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie, 'Feature pyramid networks for object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2017).

[21] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár, 'Focal loss for dense object detection', *In IEEE International Conference on Computer Vision (ICCV)*, 2999–3007, (2017).

[22] Tsungyi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C Lawrence Zitnick, 'Microsoft coco: Common objects in context', *In European Conference on Computer Vision (ECCV)*, 740–755, (2014).

[23] Songtao Liu, Di Huang, and Yunhong Wang, 'Receptive field block net for accurate and fast object detection', 404–419, (2018).

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg, 'SSD: single shot multibox detector', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 21–37, (2015).

[25] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin, 'Libra R-CNN: towards balanced learning for object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 821–830, (2019).

[26] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi, 'You only look once: Unified, real-time object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788, (2016).

[27] Joseph Redmon and Ali Farhadi, 'YOLO9000: better, faster, stronger', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6517–6525, (2017).

[28] Joseph Redmon and Ali Farhadi, 'Yolov3: An incremental improvement', *In arXiv, abs/1804.02767*, (2018).

[29] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun, 'Faster R-CNN: towards real-time object detection with region proposal networks', *In Conference and Workshop on Neural Information Processing Systems (NIPS)*, 1–10, (2015).

[30] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu, 'Skeleton-based action recognition with directed graph neural networks', in *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[31] B. Singh and L. S. Davis, 'An analysis of scale invariance in object detection - snip', in *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3578–3587, (2018).

[32] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He, 'FCOS: fully convolutional one-stage object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2019).

[33] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio, 'Graph Attention Networks', *International Conference on Learning Representations (ICLR)*, (2018).

[34] Yuxiang Wu, Zehua Cheng, Zhenghua Xu, and Weiyang Wang, 'Segmentation is all you need', *In arxiv, abs/1904.13300*, (2019).

[35] Hang Xu, Chenhan Jiang, Xiaodan Liang, and Zhenguo Li, 'Spatial-aware graph relation network for large-scale object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 9298–9307, (2019).

[36] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z. Li, 'Single-shot refinement neural network for object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4203–4212, (2018).

[37] Zhishuai Zhang, Siyuan Qiao, Cihang Xie, Wei Shen, Bo Wang, and Alan L. Yuille, 'Single-shot object detection with enriched semantics', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2018).

[38] Qijie Zhao, Tao Sheng, Yongtao Wang, Zhi Tang, Ying Chen, Ling Cai, and Haibin Ling, 'M2det: A single-shot object detector based on multi-level feature pyramid network', *In AAAI Conference on Artificial Intelligence(AAAI)*, 9259–9266, (2019).

[39] Peng Zhou, Bingbing Ni, Cong Geng, Jianguo Hu, and Yi Xu, 'Scale-transferrable object detection', 528–537, (2018).

[40] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krähenbühl, 'Bottom-up object detection by grouping extreme and center points', *In arxiv, abs/1901.08043*, (2019).

[41] Chenchen Zhu, Yihui He, and Marios Savvides, 'Feature selective anchor-free module for single-shot object detection', *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 840–849, (2019).

[42] Yousong Zhu, Chaoyang Zhao, Jinqiao Wang, xu zhao, Yi Wu, and Hanqing Lu, 'Couplenet: Coupling global structure with local parts for object detection', *In IEEE International Conference on Computer Vision (ICCV)*, 4146–4154, (2017).