


Gated Bi-directional CNN for Object Detection

Xingyu Zeng^{1,2}, Wanli Ouyang¹, Bin Yang², Junjie Yan²,
and Xiaogang Wang¹

¹ The Chinese University of Hong Kong, Hong Kong, China
{xyzeng,wlouyang,xgwang}@ee.cuhk.edu.hk
² Sensetime Group Limited, Sha Tin, Hong Kong
{yangbin,yanjunjie}@sensetime.com

Abstract. The visual cues from multiple support regions of different sizes and resolutions are complementary in classifying a candidate box in object detection. How to effectively integrate local and contextual visual cues from these regions has become a fundamental problem in object detection. Most existing works simply concatenated features or scores obtained from support regions. In this paper, we propose a novel gated bi-directional CNN (GBD-Net) to pass messages between features from different support regions during both feature learning and feature extraction. Such message passing can be implemented through convolution in two directions and can be conducted in various layers. Therefore, local and contextual visual patterns can validate the existence of each other by learning their nonlinear relationships and their close iterations are modeled in a much more complex way. It is also shown that message passing is not always helpful depending on individual samples. Gated functions are further introduced to control message transmission and their on-and-off is controlled by extra visual evidence from the input sample. GBD-Net is implemented under the Fast RCNN detection framework. Its effectiveness is shown through experiments on three object detection datasets, ImageNet, Pascal VOC2007 and Microsoft COCO.

1 Introduction

Object detection is one of the fundamental vision problems. It provides basic information for semantic understanding of images and videos and has attracted a lot of attentions. Detection is regarded as a problem classifying candidate boxes. Due to large variations in viewpoints, poses, occlusions, lighting conditions and background, object detection is challenging. Recently, convolutional neural networks (CNNs) have been proved to be effective for object detection [1–4] because of its power in learning features.

In object detection, a candidate box is counted as true-positive for an object category if the intersection-over-union (IOU) between the candidate box and the ground-truth box is greater than a threshold. When a candidate box cover a part of the ground-truth regions, there are some potential problems.

- Visual cues in this candidate box may not be sufficient to distinguish object categories. Take the candidate boxes in Fig. 1(a) for example, they cover parts

of bodies and have similar visual cues, but with different ground-truth class labels. It is hard to distinguish their class labels without information from larger surrounding regions of the candidate boxes.

- Classification on the candidate boxes depends on the occlusion status, which has to be inferred from larger surrounding regions. Because of occlusion, the candidate box covering a rabbit head in Fig. 1(b1) is considered as a true positive of rabbit, because of large IOU. Without occlusion, however, the candidate box covering a rabbit head in Fig. 1(b2) is **not** considered as a true positive because of small IOU.

To handle these problems, contextual regions surrounding candidate boxes are a natural help. Besides, surrounding regions also provide contextual information about background and other nearby objects to help detection. Therefore, in our deep model design and some existing works [5], information from surrounding regions are used to improve classification of a candidate box.

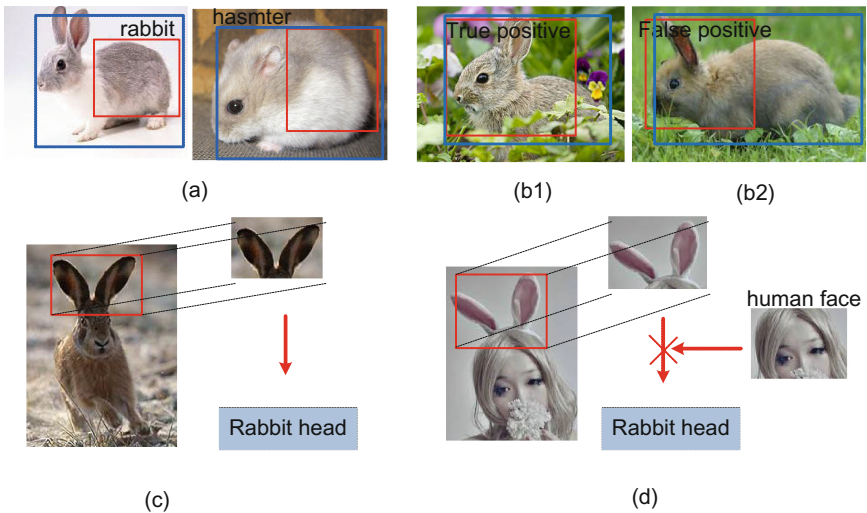


Fig. 1. Illustrate the motivation of passing messages among features from supporting regions of different resolutions, and controlling message passing according different image instances. Blue windows indicate the ground truth bounding boxes. Red windows are candidate boxes. It is hard to classify candidate boxes which cover parts of objects because of similar local visual cues in (a) and ignorance on the occlusion status in (b). Local details of rabbit ears are useful for recognizing the rabbit head in (c). The contextual human head help to find that the rabbit ear worn on human head should not be used to validate the existence of the rabbit head in (d). Best viewed in color. (Color figure online)

On the other hand, when CNN takes a large region as input, it sacrifices the ability in describing local details, which are sometimes critical in discriminating

object classes, since CNN encodes input to a fixed-length feature vector. For example, the sizes and shapes of ears are critical details in discriminating rabbits from hamsters. But they may not be identified when they are in a very small part of the CNN input. It is desirable to have a network structure that takes both surrounding regions and local part regions into consideration. Besides, it is well-known that features from different resolutions are complementary [5].

One of our motivations is that features from different resolutions and support regions validate the existence of one another. For example, the existence of rabbit ears in a local region helps to strengthen the existence of a rabbit head, while the existence of the upper body of a rabbit in a larger contextual region also help to validate the existence of a rabbit head. Therefore, we propose that features with different resolutions and support regions should pass messages to each other in multiple layers in order to validate their existences jointly during both feature learning and feature extraction. This is different from the naive way of learning a separate CNN for each support region and concatenating feature vectors or scores from different support regions for classification.

Our further motivation is that care should be taken when passing messages among contextual and local regions. The messages are not always useful. Taking Fig. 1(c) as an example, the local details of the rabbit ear is helpful in recognizing the rabbit head, and therefore, its existence has a large weight in determining the existence of the rabbit head. However, when this rabbit ear is artificial and worn on a girl's head in Fig. 1(d), it should not be used as the evidence to support a rabbit head. Extra information is needed to determine whether the message from finding a contextual visual pattern, e.g. rabbit ear, should be transmitted to finding a target visual pattern, e.g. rabbit head. In Fig. 1(d), for example, the extra human-face visual cues indicates that the message of the rabbit ear should not be transmitted to strengthen the evidence of seeing the rabbit head. Taking this observation into account, we design a network that uses extra information from the input image region to adaptively control message transmission.

In this paper, we propose a gated bi-directional CNN (GBD-Net) architecture that adaptively models interactions of contextual and local visual cues during feature learning and feature extraction. Our contributions are in two-fold.

- A bi-directional network structure is proposed to pass messages among features from multiple support regions of different resolutions. With this design, local patterns pass detailed visual messages to larger patterns and large patterns passes contextual visual messages in the opposite direction. Therefore, local and contextual features cooperate with each other in improving detection accuracy. It shows that message passing can be implemented through convolution.
- We propose to control message passing with gate functions. With the designed gate functions, message from a found pattern is transmitted when it is useful in some samples, but is blocked for others.

The proposed GBD-Net is implemented under the Fast RCNN detection frameworks [6]. The effectiveness is validated through the experiments on three datasets, ImageNet [7], PASCAL VOC2007 [8] and Microsoft COCO [9].

2 Related Work

Great improvements have been achieved in object detection. They mainly come from better region proposals, detection pipeline, feature learning algorithms and CNN structures, and making better use of local and contextual visual cues.

Region proposal. Selective search [10] obtained region proposals by hierarchically grouping segmentation results. Edgeboxes [11] evaluated the number of contours enclosed by a bounding box to indicate the likelihood of an object. Deep MultiBox [12], Faster RCNN [13] and YOLO [14] obtained region proposals with the help of a convolution network. Pont-Tuest and Van Gool [15] studied statistical difference between the Pascal-VOC dataset [8] to Microsoft CoCo dataset [9] to obtain better object proposals.

Object detection pipeline. The state-of-the-art deep learning based object detection pipeline RCNN [16] extracted CNN features from the warped image regions and applied a linear svm as the classifier. By pre-training on the ImageNet classification dataset, it achieved great improvement in detection accuracy compared with previous sliding-window approaches that used handcrafted features on PASCAL-VOC and the large-scale ImageNet object detection dataset. In order to obtain a higher speed, Fast RCNN [6] shared the computational cost among candidate boxes in the same image and proposed a novel roi-pooling operation to extract feature vectors for each region proposal. Faster RCNN [13] combined the region proposal step with the region classification step by sharing the same convolution layers for both tasks.

Learning and design of CNN. A large number of works [1–4, 17, 18] aimed at designing network structures and their effectiveness was shown in the detection task. The works in [1–4, 19] proposed deeper networks. People [3, 20, 21] also investigated how to effectively train deep networks. Simonyan and Zisserman [3] learn deeper networks based on the parameters in shallow networks. Ioffe and Szegedy [20] normalized each layer inputs for each training mini-batch in order to avoid internal covariate shift. He *et al.* [21] investigated parameter initialization approaches and proposed parameterized RELU.

Our contributions focus on a novel bi-directional network structure to effectively make use of multi-scale and multi-context regions. Our design is complementary to above region proposals, pipelines, CNN layer designs, and training approaches. There are many works on using visual cues from object parts [22–24] and contextual information [22, 23]. Gidaris and Komodakis [23] adopted a multi-region CNN model and manually selected multiple image regions. Girshick *et al.* [24] and Ouyang *et al.* [22] learned the deformable parts from CNNs. In order to use the contextual information, multiple image regions surrounding the candidate box were cropped in [23] and whole-image classification scores were used in [23]. These works simply concatenated features or scores from object parts or context while we pass message among features representing local and contextual visual patterns so that they validate the existence of each other by non-linear relationship learning. As a step further, we propose to use gate functions for controlling message passing, which was not investigated in existing works.

Passing messages and gate functions. Message passing at the feature level is allowed in Recurrent neural network (RNN) and gate functions are used to control message passing in long short-term memory (LSTM) networks. However, both techniques have not been used to investigate feature extraction from multi-resolution and multi-context regions yet, which is fundamental in object detection. Our message passing mechanism and gate functions are specially designed under this problem setting. GBD-Net is also different from RCNN and LSTM in the sense that it does not share parameters across resolutions/contexts.

3 Gated Bi-directional CNN

We briefly introduce the fast RCNN pipeline in Sect. 3.1 and then provide an overview of our approach in Sect. 3.2. Our use of roi-pooling is discussed in Sect. 3.3. Section 3.4 focuses on the proposed bi-directional network structure and its gate functions, and Sect. 3.5 explains the details of the training scheme. The influence of different implementations is finally discussed in Sect. 3.5.

3.1 Fast RCNN Pipeline

We adopt the Fast RCNN [6] as the object detection pipeline with four steps.

1. Candidate box generation. There are multiple choices. For example, selective search [10] groups super-pixels to generate candidate boxes while Bing [25] is based on sliding window on feature maps.
2. Feature map generation. Given an input as the input of CNN, feature maps are generated.
3. Roi-pooling. Each candidate box is considered as a region-of-interest (ROI) and a pooling function is operated on the CNN feature maps generated in (2). After roi-pooling, candidate boxes of different sizes are pooled to have the same feature vector size.
4. Classification. CNN features after roi-pooling go through several convolutions, pooling and fully connected layers to predict class of candidate boxes.

3.2 Framework Overview

The overview of our approach is shown in Fig. 2. Based on the fast RCNN pipeline, our proposed model takes an image as input, uses roi-pooling operations to obtain features with different resolutions and different support regions for each candidate box, and then the gated bi-direction layer is used for passing messages among features, and final classification is made. We use the BN-net [20] as the baseline network structure, i.e. if only one support region and one branch is considered, Fig. 2 becomes a BN-net. Currently, messages are passed between features in one layer. It can be extended by adding more layers between \mathbf{f} and \mathbf{h} and also passing messages in these layers.

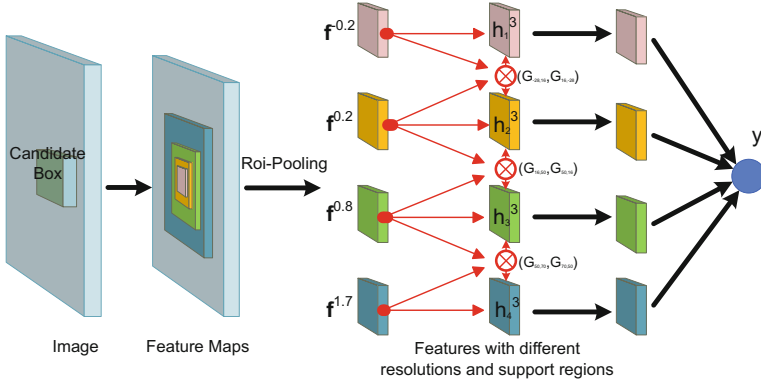


Fig. 2. Overview of our framework. The network takes an image as input and produces feature maps. The roi-pooling is done on feature maps to obtain features with different resolutions and support regions, denoted by $f^{-0.2}$, $f^{0.2}$, $f^{0.8}$ and $f^{1.7}$. Red arrows denote our gated bi-directional structure for passing messages among features. Gate functions G are defined for controlling the message passing rate. Then all features h_i^3 for $i = 1, 2, 3, 4$ go through multiple CNN layers with shared parameters to obtain the final features that are used to predict the class y . Parameters on black arrows are shared across branches, while parameters on red arrows are not shared. Best viewed in color. (Color figure online)

We use the same candidate box generation and feature map generation steps as the fast RCNN introduced in Sect. 3.1. In order to take advantage of complementary visual cues in the surrounding/inner regions, the major modifications of fast RCNN are as follows.

- In the roi-pooling step, regions with the same center location but different sizes are pooled from the same feature maps for a single candidate box. The regions with different sizes before roi-pooling have the same size after roi-pooling. In this way, the pooled features corresponds to different support regions and have different resolutions.
- Features with different resolutions optionally go through several CNN layers to extract their high-level features.
- The bi-directional structure is designed to pass messages among the roi-pooled features with different resolutions and support regions. In this way, features corresponding to different resolutions and support regions verify each other by passing messages to each other.
- Gate functions are use to control message transmission.
- After message passing, the features for different resolutions and support regions are then passed through several CNN layers for classification.

An exemplar implementation of our model is shown in Fig. 3. There are 9 inception modules in the BN-net [20]. Roi-pooling of multiple resolutions and support regions is conducted after the 6th inception module, which is inception (4d). Then the gated bi-directional network is used for passing messages

among features $\mathbf{h}_1^3\text{--}\mathbf{h}_4^3$. After message passing, $\mathbf{h}_1^3\text{--}\mathbf{h}_4^3$ go through the 7th, 8th, 9th inception modules and the average pooling layers separately and then used for classification. There is option to place ROI-pooling and GBD-Net after different layers of the BN-net. In Fig. 3, they are placed after inception (4e). In the experiment, we also tried to place them right after the input image.

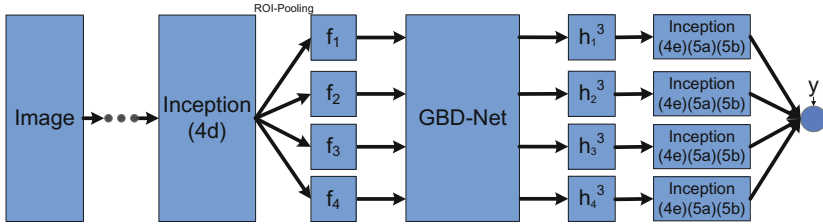


Fig. 3. Exemplar implementation of our model. The gated bi-directional network, dedicated as GBD-Net, is placed between Inception (4d) and Inception (4e). Inception (4e), (5a) and (5b) are shared among all branches.

3.3 Roi-Pooling of Features with Different Resolutions and Support Regions

We use the roi-pooling layer designed in [6] to obtain features with different resolutions and support regions. Given a candidate box $\mathbf{b}^o = [x^o, y^o, w^o, h^o]$ with center location (x^o, y^o) width w^o and height h^o , its padded bounding box is denoted by \mathbf{b}^p . \mathbf{b}^p is obtained by enlarging the original box \mathbf{b}^o along both x and y directions in scale p as follows:

$$\mathbf{b}^p = [x^o, y^o, (1+p)w^o, (1+p)h^o]. \quad (1)$$

In RCNN [16], p is 0.2 by default and the input to CNN is obtained by warping all the pixels in the enlarged bounding box \mathbf{b}^p to a fixed size $w \times h$, where $w = h = 224$ for the BN-net [20]. In fast RCNN [6], warping is done on feature maps instead of pixels. For a box \mathbf{b}^o , its corresponding feature box \mathbf{b}^f on the feature maps is calculated and roi-pooling uses max pooling to convert the features in \mathbf{b}^f to feature maps with a fixed size.

In our implementation, a set of padded bounding boxes $\{\mathbf{b}^p\}$ with different $p = -0.2, 0.2, 0.8, 1.7$ are generated for each candidate box \mathbf{b}^o . These boxes are warped into the same size by roi-pooling on the CNN features. The CNN features of these padded boxes have different resolutions and support regions. In the roi-pooling step, regions corresponding to $\mathbf{b}^{-0.2}$, $\mathbf{b}^{0.2}$, $\mathbf{b}^{0.8}$ and $\mathbf{b}^{1.7}$ are warped into features $\mathbf{f}^{-0.2}$, $\mathbf{f}^{0.2}$, $\mathbf{f}^{0.8}$ and $\mathbf{f}^{1.7}$ respectively. Figure 4 illustrates this procedure.

Since features $\mathbf{f}^{-0.2}$, $\mathbf{f}^{0.2}$, $\mathbf{f}^{0.8}$ and $\mathbf{f}^{1.7}$ after roi-pooling are in the same size, the context scale value p determines both the amount of padded context and also the resolution of the features. A larger p value means a lower resolution for the original box but more contextual information around the original box, while a small p means a higher resolution for the original box but less context.

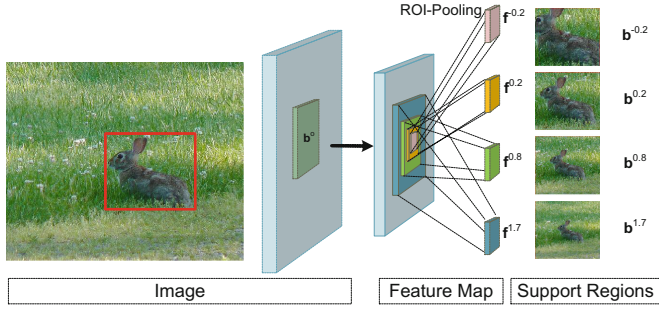


Fig. 4. Illustration of using roi-pooling to obtain CNN features with different resolutions and support regions. The red rectangle in the left image is a candidate box. The right four image patches show the supporting regions for $\{b^p\}$. Best viewed in color. (Color figure online)

3.4 Gated Bi-directional Network Structure

Bi-direction Structure. Figure 5 shows the architecture of our proposed bi-directional network. It takes features $f^{-0.2}, f^{0.2}, f^{0.8}$ and $f^{1.7}$ as input and outputs features h_1^3, h_2^3, h_3^3 and h_4^3 for a single candidate box. In order to have features $\{h_i^3\}$ with different resolutions and support regions cooperate with each other, this new structure builds two directional connections among them. One directional connection starts from features with the smallest region size and ends at features with the largest region size. The other is the opposite.

For a single candidate box $b^o, h_i^0 = f^{p_i}$ represents features with context pad value p_i . The forward propagation for the proposed bi-directional structure can be summarized as follows:

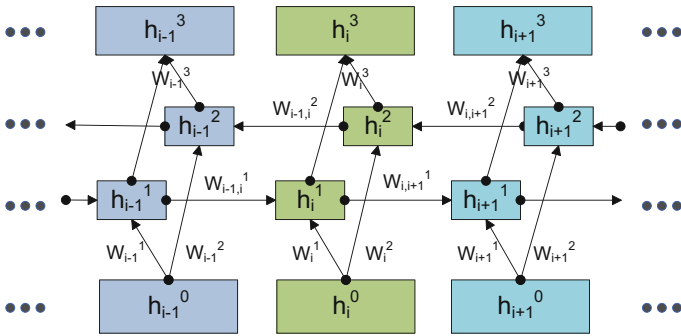


Fig. 5. Details of our bi-directional structure. The input of this structure is the features $\{h_i^0\}$ of multiple resolutions and contextual regions. Then bi-directional connections among these features are used for passing messages across resolutions/contexts. The output h_i^3 are updated features for different resolutions/contexts after message passing.

$$\mathbf{h}_i^1 = \sigma(\mathbf{h}_i^0 \otimes \mathbf{w}_i^1 + \mathbf{b}_i^{0,1}) + \sigma(\mathbf{h}_{i-1}^1 \otimes \mathbf{w}_{i-1,i}^1 + \mathbf{b}_i^1) \quad \text{--high res. to low pass} \quad (2)$$

$$\mathbf{h}_i^2 = \sigma(\mathbf{h}_i^0 \otimes \mathbf{w}_i^2 + \mathbf{b}_i^{0,2}) + \sigma(\mathbf{h}_{i+1}^2 \otimes \mathbf{w}_{i,i+1}^2 + \mathbf{b}_i^2) \quad \text{--low res. to high pass} \quad (3)$$

$$\mathbf{h}_i^3 = \sigma(\text{cat}(\mathbf{h}_i^1, \mathbf{h}_i^2) \otimes \mathbf{w}_i^3 + \mathbf{b}_i^3) \quad \text{--message integration} \quad (4)$$

- Since there are totally four different resolutions/contexts, $i = 1, 2, 3, 4$.
- \mathbf{h}_i^1 represents the updated features after receiving message from \mathbf{h}_{i-1}^1 with a higher resolution and a smaller support region. It is assumed that $\mathbf{h}_0^1 = 0$, since \mathbf{h}_1^1 has the smallest support region and receives no message.
- \mathbf{h}_i^2 represents the updated features after receiving message from \mathbf{h}_{i+1}^2 with a lower resolution and a larger support region. It is assumed that $\mathbf{h}_5^2 = 0$, since \mathbf{h}_4^2 has the largest support region and receives no message.
- $\text{cat}()$ concatenates CNN features maps along the channel direction.
- The features \mathbf{h}_i^1 and \mathbf{h}_i^2 after message passing are integrated into \mathbf{h}_i^3 using the convolutional filters \mathbf{w}_i^3 .
- \otimes represent the convolution operation. The biases and filters of convolutional layers are respectively denoted by \mathbf{b}_*^* and \mathbf{w}_*^* .
- Element-wise RELU is used as the non-linear function $\sigma(\cdot)$.

From the equations above, the features in \mathbf{h}_i^1 receive the messages from the high-resolution/small-context features and the features \mathbf{h}_i^2 receive messages from the low-resolution/large-context features. Then \mathbf{h}_i^3 collects messages from both directions to have a better representation of the i th resolution/context. For example, the visual pattern of a rabbit ear is obtained from features with a higher resolution and a smaller support region, and its existence (high responses in these features) can be used for validating the existence of a rabbit head, which corresponds to features with a lower resolution and a larger support region. This corresponds to message passing from high resolution to low resolution in (2). Similarly, the existence of the rabbit head at the low resolution also helps to validate the existence of the rabbit ear at the high resolution by using (3). $\mathbf{w}_{i-1,i}^1$ and $\mathbf{w}_{i,i+1}^2$ are learned to control how strong the existence of a feature with one resolution/context influences the existence of a feature with another resolution/context. Even after bi-directional message passing, $\{\mathbf{h}_i^3\}$ are complementary and will be jointly used for classification in later layers.

Our bi-directional structure is different from the bi-direction recurrent neural network (RNN). RNN aims to capture dynamic temporal/spatial behavior with a directed cycle. It is assumed that parameters are shared among directed connections. Since our inputs differ in both resolutions and contextual regions, convolutions layers connecting them should learn different relationships at different resolution/context levels. Therefore, the convolutional parameters for message passing are not shared in our bi-directional structure.

Gate Functions for Message Passing. Instead of passing messages in the same way for all the candidate boxes, gate functions are introduced to adapt message passing for individual candidate boxes. Gate functions are also implemented through convolution. The design of gate filters consider the following aspects.

- \mathbf{h}_i^k has multiple feature channels. A different gate filter is learned for each channel.
- The message passing rates should be controlled by the responses to particular visual patterns which are captured by gate filters.
- The message passing rates can be determined by visual cues from nearby regions, e.g. in Fig. 1, a girl’s face indicates that the rabbit ear is artificial and should not pass message to the rabbit head. Therefore, the size of gate filters should not be 1×1 and 3×3 is used in our implementation.

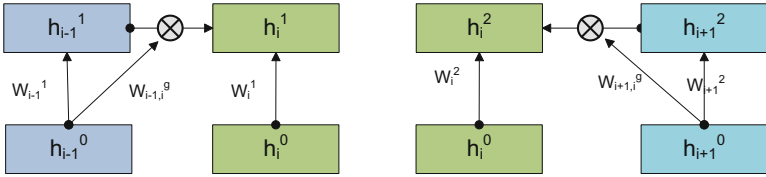


Fig. 6. Illustration of the bi-directional structure with gate functions. Here \otimes represents the gate function.

We design gate functions by convolution layers with the sigmoid non-linearity to make the message passing rate in the range of (0,1). With gate functions, message passing in (2) and (3) for the bi-directional structure is changed:

$$\mathbf{h}_i^1 = \sigma(\mathbf{h}_i^0 \otimes \mathbf{w}_i^1 + \mathbf{b}_i^{0,1}) + G(\mathbf{h}_{i-1}^0, \mathbf{w}_{i-1,i}^g, \mathbf{b}_{i-1,i}^g) \bullet \sigma(\mathbf{h}_{i-1}^1 \otimes \mathbf{w}_{i-1,i}^1 + \mathbf{b}_i^1), \quad (5)$$

$$\mathbf{h}_i^2 = \sigma(\mathbf{h}_i^0 \otimes \mathbf{w}_i^2 + \mathbf{b}_i^{0,2}) + G(\mathbf{h}_{i+1}^0, \mathbf{w}_{i+1,i}^g, \mathbf{b}_{i+1,i}^g) \bullet \sigma(\mathbf{h}_{i+1}^2 \otimes \mathbf{w}_{i,i+1}^2 + \mathbf{b}_i^2), \quad (6)$$

$$G(\mathbf{x}, \mathbf{w}, \mathbf{b}) = \text{sigm}(\mathbf{x} \otimes \mathbf{w} + \mathbf{b}), \quad (7)$$

where $\text{sigm}(\mathbf{x}) = 1/[1 + \exp(-\mathbf{x})]$ is the element-wise sigmoid function and \bullet denotes element-wise product. G is the gate function to control message message passing. It contains learnable convolutional parameters $\mathbf{w}_*^g, \mathbf{b}$ and uses features from the co-located regions to determine the rates of message passing. When $G(\mathbf{x}, \mathbf{w}, \mathbf{b})$ is 0, the message is not passed. The formulation for obtaining \mathbf{h}_i^3 is unchanged. Figure 6 illustrates the bi-directional structure with gate functions.

Discussion. Our GBD-Net builds upon the features of different resolutions and contexts. Its placement is independent of the place of roi-pooling. In an extreme implementation, roi-pooling can be directly applied on raw pixels to obtain features of multiple resolutions and contexts, and in the meanwhile GBD-Net can be placed in the last convolution layer for message passing. In this implementation, fast RCNN is reduced to RCNN where multiple regions surrounding a candidate box are cropped from raw pixels instead of feature maps.

3.5 Implementation Details, Training Scheme, and Loss Function

For the state-of-the-art fast RCNN object detection framework, CNN is first pre-trained with the ImageNet image classification data, and then utilized as the initial point for fine-tuning the CNN to learn both object confidence scores s and bounding-box regression offsets t for each candidate box. Our proposed framework also follows this strategy and randomly initialize the filters in the gated bi-direction structure while the other layers are initialized from the pre-trained CNN. The final prediction on classification and bounding box regression is based on the representations \mathbf{h}_i^3 in Eq. (4). For a training sample with class label y and ground-truth bounding box offsets $\mathbf{v} = [v_1, v_2, v_3, v_4]$, the loss function of our framework is a summation of the cross-entropy loss for classification and the smoothed L_1 loss for bounding box regression as follows:

$$L(y, t_y, \mathbf{v}, t_v) = L_{cls}(y, t_y) + \lambda[y \geq 1]L_{loc}(\mathbf{v}, \mathbf{t}_v), \quad (8)$$

$$L_{cls}(y, t_y) = - \sum_c \delta(y, c) \log t_c, \quad (9)$$

$$L_{loc}(\mathbf{v}, \mathbf{t}_v) = \sum_{i=1}^4 \text{smooth}_{L_1}(v_i - t_{v,i}), \quad (10)$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| \leq 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}, \quad (11)$$

where the predicted classification probability for class c is denoted by t_c , and the predicted offset is denoted by $\mathbf{t}_v = [t_{v,1}, t_{v,2}, t_{v,3}, t_{v,4}]$, $\delta(y, c) = 1$ if $y = c$ and $\delta(y, c) = 0$ otherwise. $\lambda = 1$ in our implementation. Parameters in the networks are learned by back-propagation.

4 Experimental Results

4.1 Implementation Details

Our proposed framework is implemented based on the fast RCNN pipeline using the BN-net as the basic network structure. The exemplar implementation in Sect. 3.2 and Fig. 3 is used in the experimental results if not specified. The gated bi-directional structure is added after the 6th inception module (4d) of BN-net. In the GBD-Net, layers belonging to the BN-net are initialized by the baseline BN-net pre-trained on the ImageNet 1000-class classification and localization dataset. The parameters in GBD-Net as shown in Fig. 5, which are not present in the pre-trained BN-net, are randomly initialized when finetuning on the detection task. In our implementation of GBD-Net, the feature maps \mathbf{h}_i^n for $n = 1, 2, 3$ in (2)–(4) have the same width, height and number of channels as the input \mathbf{h}_i^0 for $i = 1, 2, 3, 4$.

We evaluate our method on three public datasets, ImageNet object detection dataset [7], Pascal VOC 2007 dataset [8] and Microsoft COCO object detection dataset [9]. Since the ImageNet object detection task contains a sufficiently large number of images and object categories to reach a conclusion, evaluations

on component analysis of our training method are conducted on this dataset. This dataset has 200 object categories and consists of three subsets. i.e., train, validation and test data. In order to have a fair comparison with other methods, we follow the same setting in [16] and split the whole validation subset into two sub-folders, val1 and val2. The network finetuning step uses training samples from train and val1 subsets and evaluation is done on the val2 subset. Because the input for fast RCNN is an image from which both positive and negative samples are sampled, we discard images with no ground-truth boxes in the val1. Considering that lots of images in the train subset do not annotate all object instances, we reduce the number of images from this subset in the batch. Both the learning rate and weight decay are fixed to 0.0005 during training for all experiments below. We use batch-based stochastic gradient descent to learn the network and the batch size is 192. The overhead time at inference due to gated connections is less than 40 %.

4.2 Overall Performance

ILSVRC2014 Object Detection Dataset. We compare our framework with several other state-of-art approaches [4, 16, 19, 20, 22, 26]. The mean average precision for these approaches are shown in Table 1. Our work is trained using the provided data of ImageNet. Compared with the published results and recent results in the provided data track on ImageNet 2015 challenge, our single model result ranks No. 2, lower than the ResNet [19] which uses a much deeper network structure. In the future work, we may integrate GBD-Net with ResNet.

The BN-net on Fast RCNN implemented by us is our baseline, which is denoted by BN+FRCN. From the table, it can be seen that BN-net with our GBD-Net has 5.1 % absolute mAP improvement compared with BN-net. We also report the performance of feature combination method as opposed to gated connections, which is denoted by BN+FC+FRCN. It uses the same four region features as GBD-net by simple concatenation and obtains 47.3 % mAP, while ours is 51.4 %.

Table 1. Object detection mAP (%) on ImageNet val2 for state-of-the-art approaches with single model (sgl) and averaged model (avg).

Approach	RCNN [16]	Berkeley [16]	GoogleNet [4]	BN+ FRCN	BN+FC +FRCN	DeepID- Net [22]	Superpixel [26]	ResNet [19]	Ours
val2 (sgl)	31.0	33.4	38.5	46.3	47.3	48.2	42.8	60.5	51.4
val2 (avg)	n/a	n/a	40.9	n/a	n/a	50.7	45.4	63.6	n/a

PASCAL VOC2007 Dataset. Contains 20 object categories. Following the most commonly used approach in [16], we finetune the network with the 07+12 trainval set and evaluate the performance on the test set. Our GBD-net obtains 77.2 % mAP while the baseline BN+FRCN is only 73.1 %.

Microsoft COCO Object Detection Dataset. We use MCG [27] for region proposal and report both the overall AP and AP⁵⁰ on the closed-test data. The baseline BN+FRCN implemented by us obtains 24.4% AP and 39.3% AP⁵⁰, which is comparable with Faster RCNN (24.2% AP) on COCO detection leaderboard. With our proposal gated bi-directional structure, the network is improved by 2.6% AP points and reaches 27.0% AP and 45.8% AP⁵⁰, which further proves the effectiveness of our model.

4.3 Component-Wise Investigation

Investigation on Using Roi-Pooling for Different Layers. The placement of roi-pooling is independent of the placement of the GBD-Net. Experimental results on placing the roi-pooling after the image pixels and after the 6th inception module are reported in this section. If the roi-pooling is placed after the 6th inception module (4d) for generating features of multiple resolutions, the model is faster in both training and testing stages. If the roi-pooling is placed after the image pixels for generating features of multiple resolutions, the model is slower because the computation in CNN layers up to the 6th inception module cannot be shared. Compared with the GBD-Net placing roi-pooling after the 6th inception module with mAP 48.9%, the GBD-Net placing the roi-pooling after the pixel values with mAP 51.4% has better detection accuracy. This is because the features for GBD-Net are more diverse and more complementary to each other when roi-pooling is placed after pixel values.

Investigation on Gate Functions. Gate functions are introduced to control message passing for individual candidate boxes. Without gate functions, it is hard to train the network with message passing layers in our implementation. It is because nonlinearity increases significantly by message passing layers and gradients explode or vanish, just like it is hard to train RNN without LSTM (gating). In order to verify it, we tried different initializations. The network with message passing layers but without gate functions has 42.3% mAP if those message passing layers are randomly initialized. However, if those layers are initialized from a well-trained GBD-net, the network without gate functions reaches 48.2% mAP. Both two results also show the effectiveness of gate functions.

Investigation on Using Different Feature Region Sizes. The goal of our proposed gated bi-directional structure is to pass messages among features with different resolutions and contexts. In order to investigate the influence from different settings of resolutions and contexts, we conduct a series of experiments. In these experiments, features of a particular padding value p is added one by one. The experimental results for these settings are shown in Table 2. When single padding value is used, it can be seen that simply enlarging the support region of CNN by increasing the padding value p from 0.2 to 1.7 does harm to detection performance because it loses resolution and is influenced by background clutter. On the other hand, integrating features with multiple resolutions and contexts

using our GBD-Net substantially improves the detection performance as the number of resolutions/contexts increases. Therefore, with the GBD-Net, features with different resolutions and contexts help to validate the existence of each other in learning features and improve detection accuracy.

Table 2. Detection mAP (%) for features with different padding values p for our GBD-Net using BN-net as the baseline. Different p s leads to different resolutions and contexts.

Padding value p	Single resolution				Multiple resolutions			
	-0.2	0.2	0.8	1.7	-0.2, 0.2	0.2 + 1.7	-0.2 + 0.2 + 1.7	-0.2 + 0.2 + 0.8 + 1.7
mAP	46.3	46.3	46.0	45.2	47.4	47.0	48.0	48.9

Investigation on Combination with Multi-region. This section investigates experimental results when combining our gated bi-directional structure with the multi-region approach. We adopt the simple straightforward method and average the detection scores of the two approaches. The baseline BN model has mAP 46.3%. With our GBD-Net the mAP is 48.9%. The multi-region approach based on BN-net has mAP 47.3%. The performance of combining our GBD-Net with multi-region BN is 51.2%, which has 2.3% mAP improvement compared with the GBD-Net and 3.9% mAP improvement compared with the multi-region BN-net. This experiment shows that the improvement brought by our GBD-Net is complementary to the multi-region approach in [23].

5 Conclusion

In this paper, we propose a gated bi-directional CNN (GBD-Net) for object detection. In this CNN, features of different resolutions and support regions pass messages to each other to validate their existence through the bi-directional structure. And the gate function is used for controlling the message passing rate among these features. Our GBD-Net is a general layer design which can be used for any network architecture and placed after any convolutional layer for utilizing the relationship among features of different resolutions and support regions. The effectiveness of the proposed approach is validated on three object detection datasets, ImageNet, Pascal VOC2007 and Microsoft COCO.

Acknowledgment. This work is supported by SenseTime Group Limited and the General Research Fund sponsored by the Research Grants Council of Hong Kong (Project Nos. CUHK14206114, CUHK14205615, CUHK417011, and CUHK14207814). Both Xingyu Zeng and Wanli Ouyang are corresponding authors.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
2. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: integrated recognition, localization and detection using convolutional networks. arXiv preprint [arXiv:1312.6229](https://arxiv.org/abs/1312.6229) (2013)
3. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)
4. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR (2015)
5. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Learning hierarchical features for scene labeling. PAMI **35**(8), 1915–1929 (2013)
6. Girshick, R.: Fast R-CNN. In: CVPR (2015)
7. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge. IJCV **115**, 211–252 (2014)
8. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. IJCV **88**(2), 303–338 (2010)
9. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Heidelberg (2014). doi:[10.1007/978-3-319-10602-1_48](https://doi.org/10.1007/978-3-319-10602-1_48)
10. Uijlings, J.R., van de Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. IJCV **104**(2), 154–171 (2013)
11. Zitnick, C.L., Dollár, P.: Edge boxes: locating object proposals from edges. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 391–405. Springer, Heidelberg (2014). doi:[10.1007/978-3-319-10602-1_26](https://doi.org/10.1007/978-3-319-10602-1_26)
12. Szegedy, C., Reed, S., Erhan, D., Anguelov, D.: Scalable, high-quality object detection. arXiv preprint [arXiv:1412.1441](https://arxiv.org/abs/1412.1441) (2014)
13. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: NIPS (2015)
14. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. arXiv preprint [arXiv:1506.02640](https://arxiv.org/abs/1506.02640) (2015)
15. Pont-Tuset, J., Van Gool, L.: Boosting object proposals: from Pascal to COCO. In: ICCV (2015)
16. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
17. Zagoruyko, S., Lerer, A., Lin, T.Y., Pinheiro, P.O., Gross, S., Chintala, S., Dollár, P.: A multipath network for object detection. arXiv preprint [arXiv:1604.02135](https://arxiv.org/abs/1604.02135) (2016)
18. Ren, S., He, K., Girshick, R., Zhang, X., Sun, J.: Object detection networks on convolutional feature maps. arXiv preprint [arXiv:1504.06066](https://arxiv.org/abs/1504.06066) (2015)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. arXiv preprint [arXiv:1512.03385](https://arxiv.org/abs/1512.03385) (2015)
20. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: NIPS (2015)
21. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: ICCV (2015)

22. Ouyang, W., Wang, X., Zeng, X., Qiu, S., Luo, P., Tian, Y., Li, H., Yang, S., Wang, Z., Loy, C.C., et al.: DeepID-Net: deformable deep convolutional neural networks for object detection. In: CVPR (2015)
23. Gidaris, S., Komodakis, N.: Object detection via a multi-region and semantic segmentation-aware CNN model. In: ICCV (2015)
24. Girshick, R., Iandola, F., Darrell, T., Malik, J.: Deformable part models are convolutional neural networks. In: CVPR (2015)
25. Cheng, M.M., Zhang, Z., Lin, W.Y., Torr, P.: BING: binarized normed gradients for objectness estimation at 300fps. In: CVPR (2014)
26. Yan, J., Yu, Y., Zhu, X., Lei, Z., Li, S.Z.: Object detection by labeling superpixels. In: CVPR (2015)
27. Arbeláez, P., Pont-Tuset, J., Barron, J.T., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: CVPR (2014)