Multi-model ensemble with rich spatial information for object detection

Jie Xu, Wei Wang, Hanyuan Wang, Jinhong Guo

PII: S0031-3203(19)30399-1
DOI: https://doi.org/10.1016/j.patcog.2019.107098
Reference: PR 107098

To appear in: *Pattern Recognition*

Received date: 8 February 2019
Revised date: 31 August 2019
Accepted date: 29 October 2019

Highlights

- Ensemble learning improves the performance of object detection and achieves the mAP of state-of-the-art detectors.
- The combination of context modeling and dilated convolution ensures the detection speed.
- The proposed multi-scale feature fusion module confers a clear improvement to the detector.
- The proposed ensemble modes demonstrate the effectiveness of ensemble learning in the field of object detection.

Multi-model ensemble with rich spatial information for object detection

Jie Xu*, Wei Wang, Hanyuan Wang, Jinhong Guo*

School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China

Abstract

Due to the development of deep learning networks and big data dimensionality, research on ensemble deep learning is receiving an increasing amount of attention. This paper takes the object detection task as the research domain and proposes an object detection framework based on ensemble deep learning. To guarantee the accuracy as well as real-time detection, the detector uses a Single Shot MultiBox Detector (SSD) as the backbone and combines ensemble learning with context modeling and multi-scale feature representation. Two modes were designed in order to achieve ensemble learning: NMS Ensembling and Feature Ensembling. In addition, to obtain contextual information, we used dilated convolution to expand the receptive field of the network. Compared with state-of-the-art detectors, our detector achieves superior performance on the PASCAL VOC set and the MS COCO set.

Keywords: Ensemble learning, Object detection, Dilated convolution, Feature fusion

*Corresponding author

Email addresses: xuj@uestc.edu.cn (Jie Xu), 201722010240@std.uestc.edu.cn (Wei Wang), 201621010319@std.uestc.edu.cn (Hanyuan Wang), guojinhong@uestc.edu.cn (Jinhong Guo)

1. Introduction

Deep learning has been widely used in recent years to solve a range of problems, such as computer vision, speech recognition, and natural language processing. As an important branch task of computer vision, some problems of object detection are solved by gradual deep learning. At the same time, ensemble learning has become popular and has been widely used to improve the performance of a single learner, especially under the promotion of competition such as ImageNet [1] and Kaggle[1]. The combination of ensemble deep learning and computer vision has become a hot and difficult point of research. In fact, these high-profile competitions also demonstrate the effectiveness and feasibility of combining ensemble learning with computer vision.

Ensemble learning is a learning paradigm that combines multiple learners to improve learner performance and can be divided into two parts: how to get multiple learners and how to combine multiple learners. For the first problem, the traditional method is implemented by algorithms such as Boosting [2], Bagging [3], or Random Forests [4]. The difference between Boosting and Bagging or Random Forests is that there is a strong dependency between the individual learners generated by the former; thus, individual learners basically need to be serially generated. However, there is no strong dependency between the individual learners generated by the latter two, so a series of individual learners can be generated in parallel. For the first problem, the neural network-based approach is related to the ensemble as represented by the Snapshot Ensemble [5] and the Fast Geometric Ensembling [6]. The difference

[1]www.kaggle.com

between this and the traditional approach is that the former takes the same amount of time to train the entire ensemble model as the training of a single traditional model.

The methods used to combine multiple learners in ensemble learning mainly include voting, averaging, and learning. For regression problems, the commonly used ensemble strategy is averaging; that is, the outputs of several weak learners are averaged to obtain the final predicted output, mainly by simple and weighted averaging. Voting is usually employed for classification problems; that is, voting on the results of the weak learner to obtain the final result as represented by majority voting, plurality voting, and weighted voting. The learning-based ensemble strategy, which is ensembled through a new learner, is more complicated. The masterpiece of the learning-based ensemble strategy is stacking [7]. When using the stacking strategy, instead of doing a simple logical processing of the primary learner's results, we add a secondary learner to the primary learners. Specifically, we take the learning result of primary learners on the training set as input and train a secondary learner to get the final result. For the test set, we first use the primary learner to predict the input data of the secondary learner and then use the secondary learner to predict the final result.

The state-of-the-art in object detection, which is a branch task of computer vision, mainly comprises two research directions: region-based [8, 9, 10] and region-free [11, 12] proposals detection. The former is mainly used to improve detection accuracy while the latter is used to improve detection efficiency.

Region-based proposals detection primarily involves a two-stage framework. The main work is R-CNN [8] along with its representative updated descendants such as Fast R-CNN [9] and Faster R-CNN [10]. R-CNN proposes a backbone network based on CNN and proposals-generated algorithms, such as Selective Search [13] and Edge Boxes [14], and becomes a typical pipeline. Fast R-CNN proposes the ROI-pooling layer based on R-CNN, which greatly alleviates the speed problem caused by R-CNN due to a large number of unnecessary computations. ROI-pooling is a single-layer Spatial Pyramid Pooling Network (SPP-Net) [15] that generates a fixed-length feature descriptor without regard to input image size. The Faster R-CNN is based on Fast R-CNN and improves the proposals-generated network. Faster R-CNN is designed as a Region Proposal Network (RPN) to generate regional proposals by sharing convoluted layers instead of Selective Search, which reduces computational overhead. However, these methods still have heavy computational costs due to the existence of feature extraction and region-recommendation generation, which will reduce the inference speed.

To solve the speed problem of the two-stage framework, significant work has begun to focus on the one-stage framework based on region-free proposals detectors. The masterpieces of the one-stage framework are YOLO [11] and SSD [12]. The proposed generation network is discarded in these methods, which improves the detection speed. However, YOLO and SSD also prove that the real-time performance of this one-stage framework is achieved at the expense of accuracy. At the same time, YOLO will produce relatively coarse features due to multiple downsampling, and YOLO and SSD are not sensitive to small objects.

In response to these problems, some methods based on context modeling and multi-scale representation have been proposed. Context modeling improves detection performance through features around the region of interest (RoI) [9] or default box [12]. Because the information around the RoI or default box may contain more important parts of the ground truth boxes, this information also helps deal with occlusions and local similarities such as in [16] and [17]. Multi-scale representation is used to obtain multi-scale features by integrating feature maps of different layers to further acquire semantic information of different spatial resolutions such as MS-CNN [16], FPN [18], HyperNet [19], and FSSD [20]. In addition, there is some work, such as ION [21] and DSSD [22], to combine the two to further improve detection performance, especially for small objects.

In recent years, ensemble learning and object detection technologies have matured and both have achieved good performance in their respective fields. However, to the best of our knowledge, little attention has been paid to the combination of ensemble learning and object detection. Most studies have focused on the combination of ensemble learning and image classification. This is mainly because object detection is a multi-task operation that needs to implement both classification and localization tasks. Object classification and object localization belong to classification and regression problems, respectively. Therefore, there is no better way to combine different models than by simply using voting, averaging, and stacking. However, from the NMS operation and feature level, this paper designs two ensemble learning modes that can combine different models well and overcome the difficulties of multi-task detection for network integration. The NMS Ensembling mode integrates the two tasks of classification and localization by integrating the inference results, whereas the Feature Ensembling mode is a multi-modal ensemble method that can integrate the features learned by different models so that the learned features of different models can complement each other and increase the robustness of the model. Hence, the proposed method can solve the problem of combining ensemble learning with object detection so that ensemble learning can efficiently and effectively improve the performance of object detection.

Based on the discussion above, in order to construct a detector with higher detection performance and without lowering the detection speed, a feasible idea is to combine a one-stage framework, context modeling, and multi-scale representation. The motivation of our work comes from this. In this paper, we adopt a new context modeling method. We apply dilated convolution, which is commonly used in the semantic segmentation domain, to object detection and build a context detection module based on the fact that dilated convolution extends the receptive field without increasing the number of computations. At the same time, we also capture fine-grained details through multi-scale representation to enhance the representation capability of the model. In addition, we also combined the idea of ensemble learning to further improve the performance of the detector. Our main contributions are as follows:

- We propose an efficient framework that combines SSD, context modeling, and multi-scale representation to improve the performance of object detection.
- We apply ensemble learning to object detection by using two novel ensemble modes to improve detector performance and then demonstrate the effectiveness

of ensemble learning in object detection.

- We conduct a series of experiments and analyses to compare the performance of different ensemble modes on the object detection model and analyze the corresponding results.
- On the PASCAL VOC object detection challenges, we obtain gratifying results: 81.1% mAP on VOC 2007 and 78.1% mAP on VOC 2012.

2. Related work

2.1. Ensemble learning

Ensemble learning has been widely used in recent years to improve the performance of single detectors. Especially in the competition between ImageNet and Kaggle, ensemble learning has developed rapidly from the ensemble of simple models, such as Random Forests to ensemble deep learning. Ensemble learning based on neural networks has been widely studied and applied in the field of machine learning. Traditional ensemble learning mainly studies how to improve the generalization performance of the learner. Its representative works are Boosting [2], Bagging [3], and Random Forests [4]. Boosting's variant, vote-boosting [23], builds individual classifiers in different weighted versions of the training data. Random Forests uses a decision tree [24] as the base learner to construct Bagging and introduces random attribute selection during the training process; a variant of Random Forests [25] constructs individual ensemble trees by using a random subspace method.

To solve the training time problem of traditional ensemble methods, some new ensemble approaches have been proposed. Huang et al. [5] proposed a snapshot ensemble. This method uses a cosine cyclical learning rate during training to save the model snapshot for the ensemble. Timur Garipov et al. [6] proposed Fast Geometric Ensembling based on the snapshot ensemble, which is an ensemble method using a cyclical learning rate. This is because they observed that the local optimum of modern deep neural networks are connected by very simple curves, thus, different networks can be found through relatively small steps in the weighted space. The training time of these methods is equivalent to a single common model, which reduces the training overhead of the traditional ensemble method.

In recent years, a small number of attempts have been made to combine ensemble learning with object detection. [26] designed a novel binary descriptor Boosted Local Binary (BLB) for object detection, which combines boosting with object detection. [27] introduces multi-modal deep feature learning for RGB-D object detection. However, this method requires three output branches, and for some categories, the network's detection results are not significantly improved. Furthermore, the modal-specific branches of the method separate the RGB features from the depth features and then combines the two using modal-correlation branch, resulting in an artificially created gap between the features, which is not suitable for general object detection.

In this paper, we design an NMS Ensembling mode and Feature Ensembling mode. At the same time, we compare the performance of the two ensemble modes in the field of object detection.

2.2. Context features and dilated convolution

In recent years, many works have attempted to improve detection performance through context modeling. Abhinav et al. [28] proposed a contextual priming based on a top-down feedback mechanism. Bell et al. [21] adopted the structure of spatial recurrent neural networks (RNN) to ensemble contextual information. Cai et al. [16] captured context from multiple regions using context embedding. Chu et al. [17] combined the contextual information in terms of relationships among objects and the global scene-based contextual feature to propose an ensemble object detection system. Hu et al. [29] used an attention mechanism to model relationships among objects.

In addition to the above methods, dilated convolution can also be used for context modeling. The concept of dilated convolution originates from DeepLab [30]. In DeepLab, the dilated convolution layer is also called the "astrous convolution" layer, which is used in the field of semantic segmentation. Fisher et al. [31] used dilated convolution to aggregate multi-scale context information in the field of semantic segmentation. Li et al. [32] introduced the dilated convolutional layer to increase the receptive field of the high layer in the network. Liu et al. [33] proposed RFBNet, which uses dilated convolution to extend the receptive field and obtain context information, thus improving object detection performance.

Similar to [33], our method uses dilated convolution to obtain context information. However, the difference is that our method obtains more context information through cascading and does not directly use context information for detection.

2.3. Multi-scale representation

Recently, significant work has proved the importance of multi-scale feature representation in the field of object detection. ION [21] achieved multi-scale feature fusion by connecting feature maps at different scales within the ROI region [9]. HON [34] combined high-level semantic features and low-resolution bottom features through reverse connections. HyperNet [19] aggregated hierarchical feature maps and compressed them into a unified space. To perform detection on multiple scales, SSD [12] applied a separate detector to multiple feature maps. FSSD [20] fused feature maps of different scales to the unified scale by upsampling. RON [35] used reverse connection to predict objects at different layers. FPN [18] designed an architecture with a bottom-up path, a top-down path, and multiple lateral connections, combining low-resolution but powerful semantic features with high-resolution but weakly semantic features. Cai et al. proposed MS-CNN [16], which relies on multiple scale-independent output layers to mitigate inconsistencies between the size of the object and the receptive field. [36] obtained a multi-scale image by re-sampling the original image and then used CNN to obtain the feature pyramid corresponding to each image level.

Inspired by [18], [19], and [20], we build a deep fusion module to concatenate the feature exaction network and the feature map of the context module, thereby achieving a multi-feature representation.

3. Network architecture

The overall architecture, shown in Fig. 1, consists of a feature exaction network, context module, feature fusion module, extra layers, detection convolution layers, and the ensemble module. At the ensemble module, we employ three strategies to improve object detection performance.

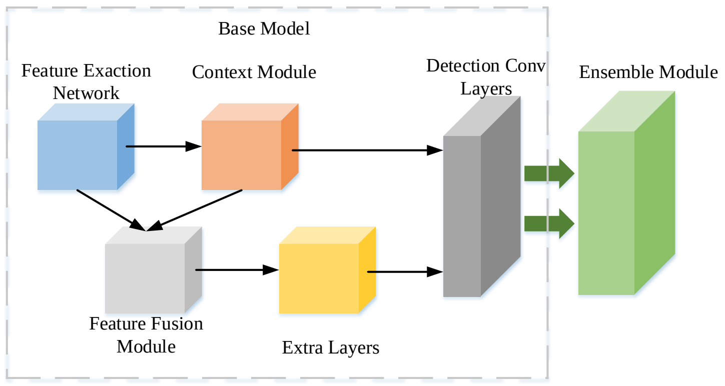
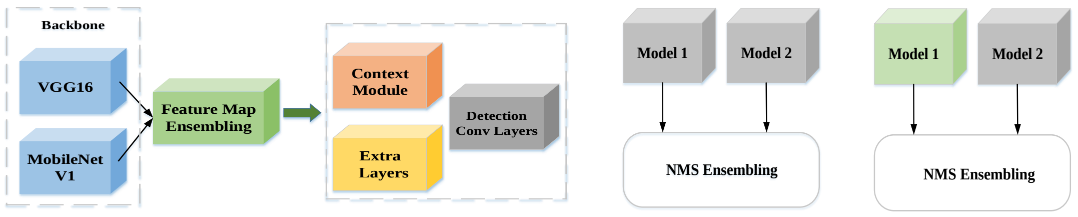


Fig 1. Overall architecture of experimental method

In this section, we first introduce three ensemble strategies (Section 3.1) and present our basic model (Section 3.2). Finally, we describe the context module that can capture more contextual information (Section 3.3) and the feature fusion module that fuses multi-scale feature maps (Section 3.4).

3.1. Ensemble of models

In this section, we introduce a practical ensemble procedure that is applied to object detection, motivated by the application of ensemble learning in image classification tasks. To be specific, first, we concatenate the inference results of two models at inference time. Suppose a single model can generate N prediction bounding boxes (N is set to be 11,620 in our work); thus, we will get 2N bounding boxes after inference. We then perform non-maximum suppression (NMS) on the 2N bounding boxes to obtain the final prediction bounding boxes. We apply NMS with Jaccard overlap of 0.45 per class and keep the top 200 detections per image, just like the operation in SSD. We call this method "NMS Ensembling". We divide our ensemble modes into three strategies based on the ensemble model as shown in Fig. 2. It should be pointed out that the first strategy does not use NMS Ensembling but rather a feature map ensembling mode.



(a) (b) (c)

Fig 2. Ensemble of modules: (a) demonstrates the ensemble of feature maps, (b) shows the ensemble of similar models, and (c) illustrates the ensemble of different models

The ensemble of feature maps: This method can only be regarded as generalized ensemble learning rather than ensemble learning in the traditional sense. As shown in Fig. 2(a), we use different feature exaction networks to extract features and then integrate the feature maps extracted by the feature exaction networks. Finally, we feed the integrated feature map to the context module and the extra feature layers for detection, respectively. Here, we choose VGG16 and MobileNet-V1 as the feature exaction network. The ensemble method is simply to choose concatenation. We choose the layer FC_7 of VGG16 with the layer dw5_5 of MobileNet-V1 and the layer 4_1 of VGG16 with the layer dw4_1 of MobileNet-V1 for the ensemble. Next, we feed the ensemble feature map to context module and extra feature layers, respectively. We chose these layers for ensemble separately because the corresponding feature maps have the same size; after integration, we added the batch normalization operation to reduce the noise caused by the different feature exaction networks. Furthermore, we also tried to add some feature maps of context module to participate in the ensemble.

The ensemble of similar models: As shown in Fig. 2(b), we refer to the snapshot ensembles [5] that use a cosine cyclical learning rate to save snapshots of the model during training at times when the learning rate achieves its minimum. However, we use a cyclical learning rate, which is introduced in FGE [6], instead of the cosine cyclical learning rate to save snapshots of the model. The cyclical learning rate $\alpha(i)$ is defined as

$$\alpha(i) = \begin{cases} (1 - 2t(i))\alpha_1 + 2t(i)\alpha_2, & 0 < t(i) \leq \frac{1}{2} \\ (2 - 2t(i))\alpha_2 + (2t(i) - 1)\alpha_1, & \frac{1}{2} < t(i) \leq 1 \end{cases} \tag{1}$$

where $i = 270, 271, \cdots, 300$ is the training epoch, $\alpha_1 = 0.0004$, $\alpha_2 = 0.000004$, $t(i) = \frac{1}{e}(mod(i+1, e) + 1)$, and e is the cycle of learning rate and set to 4. More specifically, we use the cyclical learning rate in the last 28 epochs during training time to save snapshots every 4 epochs and finally select the two models with the smallest training loss value for the ensemble.

The ensemble of different models: Here, we use the model based on VGG16, MobileNet-V1, and the model trained by Feature Ensembling mode to the ensemble as shown in Fig. 2(c). We choose two of these three models to integrate by NMS Ensembling.

A comparative analysis of the three ensemble strategies will be detailed in Section 5.1. According to the analysis, the ensemble of different models brings the best benefit to the mAP.

3.2. Base model

As is shown in Fig. 3, we adopt SSD as the backbone network and use VGG16 or MobileNet-V1 as the feature exaction network, respectively. A context module is added after the feature exaction network and a feature fusion module is applied to

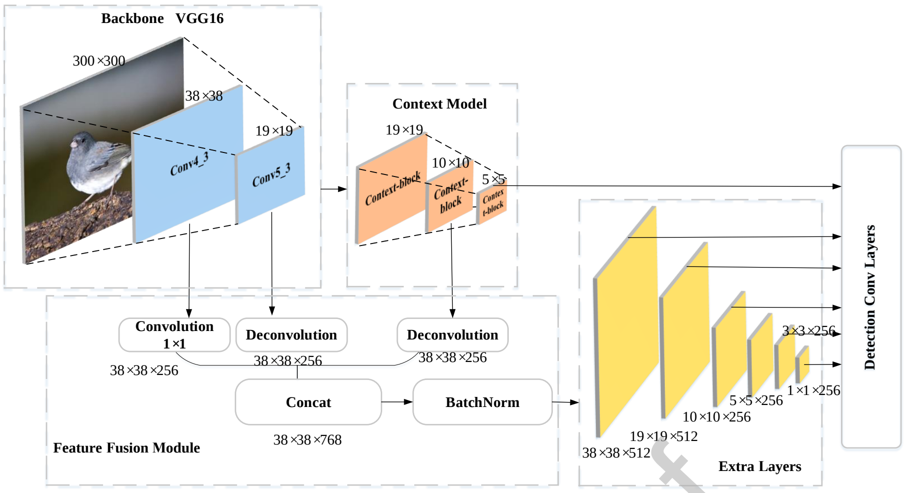


Fig 3. Framework of base model

fuse multi-scale feature maps. The generated feature map is then fed to the extra feature layers. Finally, we send feature maps of extra layers (the feature map of the third layer from the end is replaced by the output feature map of the context module) to predict the class scores and location offsets for the default bounding boxes.

3.3. Context features with dilated convolution

Dilated convolution can expand the receptive field without reducing the resolution. In other words, dilated convolution is able to capture more context features while maintaining the same number of parameters.

First, inspired by [33], we combine the dilated convolution with Inception-Resnet to design a context block as detailed in Fig. 4. The context block consists of five branches, i.e., one shortcut branch and four dilated branches. At the head of each branch a bottleneck structure (1 × 1 convolution layer) is employed to reduce the number of channels in the feature map. Then one 2-dilated convolution layer, two 3-dilated convolution layers, and one 5-dilated convolution layer are added behind the original convolutional layers in Inception, respectively, to capture more context features. The context block uses 1×3 convolution layer plus 3×1 convolution layer to take the place of the original 3×3 convolution layer and utilize two 3×3 convolution layers to replace the original 5×5 convolution layer in order to reduce parameters and increase the nonlinear layer. Furthermore, we concatenate four dilated branches and use a bottleneck structure to adjust the number of channels in the output feature map.

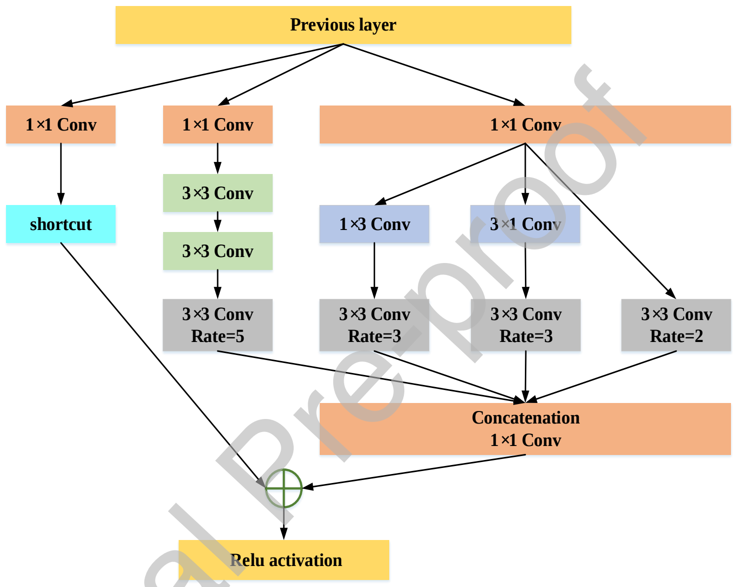


Fig 4. Detailed structure of a context block

Next, we use the three context blocks to build a context module, ensuring that all context information is completely extracted as shown in Fig. 3. In addition, the context module can easily distinguish between context and objects because the context block adds convolutional layers of different kernel sizes before the dilated convolutional layers. We send the output of the third context block of the context module to the detection layer, which improves the performance of the network because the context information extracted by the context module can assist in network detection.

3.4. Multi-scale Feature Fusion

As mentioned in Section 2, many algorithms have tried to observe and fully utilize multi-scale features. The most common method is the fusion different scale feature map, which captures more detailed information. This is because hierarchical feature maps have different characteristics in the feature extraction network. For example, the feature map comes from a shallow layer, such as layer 4, and is rich in details, whereas the feature map comes from a deep layer, such as layer 6 or more, and is rich in high-level semantics. Thus, multi-scale feature fusion is a perfect combination of the two advantages.

Our design method is similar to HyperNet and FSSD, which combine a coarse, high-level layer with a fine, low-level layer. Suppose $X_i, i \in C$ are the source feature maps we want to fuse. The feature fusion module can be described as follows:

$$X_f = \varphi_f(f_i(X_i)) \quad i \in C \tag{2}$$

where f_i means the transformation function of each source feature map before being fused together. φ_f is the feature fusion function. More details can be seen in Fig. 5. We focus on three dimensions of multi-scale fusion: the range of layers that should be fused or not (C), how to adapt feature maps of different scales to the same size (f_i), and how to fuse the selected feature maps (φ_f).

For C, we consider using the layer conv4_3, conv5_3 in the conventional SSD300 based on VGG16 and the second layer of the context module. The corresponding feature sizes are 38×38, 19×19, and 10×10. The reason why we choose the second layer of the context module for merging is because this layer has more context

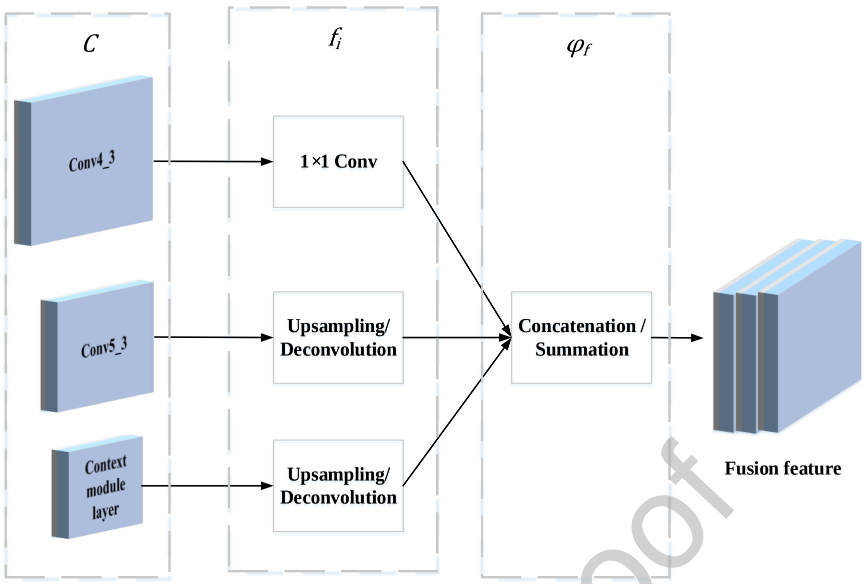

C
f_i
φ_f
Conv4_3
Conv5_3
Context module layer
1×1 Conv
Upsampling/ Deconvolution
Upsampling/ Deconvolution
Concatenation / Summation
Fusion feature

Fig 5. Architecture of multi-scale feature fusion module

information and the layers deeper than this layer have little information to merge. For f_i, we use the size of conv4_3 as the basic feature map size, which means our feature map size is 1/8 of the input size, both in width and height. As for the feature maps whose size is smaller than 38×38, we apply bilinear interpolation or deconvolution to resize the feature maps to the same size with conv4_3. In this way, all the features have the same size spatial dimension. As for φ_f, we consider two ways to fuse different feature maps together: concatenation and element-wise summation. However, according to the analysis in Section 5.3, concatenation brings better results than element-wise summation. Thus, we choose concatenation for merging the features.

4. Experiment

We perform experiments on three benchmark datasets: PASCAL VOC 2007, PASCAL VOC 2012, and MS COCO. For PASCAL VOC, all models are trained on the union of VOC 2007 trainval set and VOC 2012 trainval set ("07+12"), and the results tested on PASCAL VOC 2007 and PASCAL VOC 2012 test sets, respectively. For MS COCO, we trained models on the trainval35k set (train set + val 35k set) and tested the results on the test-dev 2015 dataset. The measure of object detection accuracy is the mean Average Precision (mAP).

4.1. Experimental setup

We implement our model based on the framework of Pytorch[2] and build on SSD architecture. If not specified, the pre-training model uses the same VGG16 pre-trained on the 1000-way ImageNet classification task [1]. Our training strategy is similar to SSD, including data augmentation and hard negative mining. The scale and aspect ratios for default boxes, and loss functions (smooth L1 loss for localization and softmax loss for classification) are also consistent with the SSD. We used a warm-up strategy to gradually increase the learning rate from 10^{-6} to 4×10^{-3} during the first five epochs and then decreased this rate by a factor of 10 times at 150th, 200th, and 250th epochs for PASCAL VOC, and the 90th and 120th epochs for MS COCO; however, for the ensembles of similar models, we used the cyclical learning rate during the last 30 epochs to save snapshots. We set the weight decay to 0.0005 and the momentum to 0.9. All new layers are initialized by the MSRA method.

4.2. Result on PASCAL VOC 2007

We compare our results with the state-of-the-art detectors on the PASCAL VOC 2007 test set (see Table 1). All parameters are set as SSD except for the learning

[2]https://pytorch.org/

Table 1. Comparison of different state-of-the-art methods on **PASCAL VOC 2007**. DSSD321, RON384, SSD300, and STDN300 indicate the input image dimensions of DSSD, RON, and SSD are 321×321, 384×384 and 300×300, respectively. **C** means context module, **M** means multi-scale feature fusion module, and **E** indicates that the ensemble is used at the inference time. All models are trained with the union training set from VOC 2007 trainval and 2012 trainval, and tested on the VOC 2007 test set. Ours_1 is the ensemble of similar models. Ours_2 is the feature ensemble. Ours_3 is the ensemble of different models. * means the test on Pytorch-0.4.0 and CUDNN V7 for fair comparison.

Method	C	M	E	Backbone	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
ION[21]				VGG16	77.6	79.7	83.4	78.1	65.7	62.0	86.5	85.8	88.8	60.2	83.4	75.1	86.5	87.3	82.1	79.7	48.3	77.0	75.3	85.3	**82.4**
DSSD321[22]				Residual-101	78.6	81.9	84.9	**80.5**	68.4	53.9	85.6	86.2	**88.9**	61.1	83.5	78.7	**86.7**	88.7	86.7	79.7	51.7	78.0	80.9	87.2	79.4
HyperNet[19]				VGG16	76.3	77.4	83.3	75.0	69.1	62.4	83.1	87.4	87.4	57.1	79.8	71.4	85.1	85.1	80.0	79.1	51.2	79.1	75.7	80.9	76.5
RON384[35]				VGG16	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
MLKP[37]				VGG16	78.1	78.7	83.1	78.8	71.3	**64.4**	86.1	**88.0**	87.8	64.6	83.2	73.6	85.7	86.4	81.9	79.3	53.1	77.2	76.7	85.0	76.1
STDN300[38]				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
SSD300[12]				VGG16	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	83.97	79.4	52.3	77.9	79.5	87.6	76.8
FSSD*[20]				VGG16	79.5	82.4	85.8	77.4	73.9	58.9	87.8	87.7	87.1	63.0	86.3	76.3	85.0	88.8	87.0	80.9	56.8	77.4	80.9	87.9	79.3
RFB*[33]				VGG16	80.1	83.1	86.3	78.1	74.0	59.9	88.4	87.7	88.3	64.0	85.2	78.6	85.9	88.1	87.7	82.1	57.3	79.9	81.0	87.9	78.7
Ours	√			VGG16	79.1	82.2	86.7	77.0	72.8	56.6	88.2	87.5	88.0	64.2	82.6	77.0	85.6	87.5	85.1	81.6	55.2	79.5	79.1	88.3	78.0
Ours		√		VGG16	80.1	84.8	85.8	79	73.7	61.6	88.3	87.1	87.3	64.3	85.7	77.3	85.4	88.3	87.3	81.4	56.3	78.4	79.2	88.0	81.7
Ours	√	√		VGG16	80.5	84.7	88.4	78.4	73.7	62.5	**88.8**	87.9	88.2	66.0	86.9	75.8	86.2	88.9	87.6	81.8	57.9	79.5	79.2	**88.4**	79.2
Ours_1	√	√	√	VGG16+VGG16	80.5	83.1	87.1	79.2	72.9	61.1	**88.8**	87.8	86.4	65.9	86.8	77.3	86.5	**89.7**	87.6	82.0	57.8	80.8	80.4	88.1	80.0
Ours_2	√	√	√	VGG16+MobileNet	80.2	84.6	**87.5**	79.8	**75.1**	61.6	87.8	87.6	87.2	65.0	86.7	77.3	85.4	87.6	86.8	81.8	57.0	79.9	78.7	86.9	79.5
Ours_3	√	√	√	VGG16+MobileNet	**81.1**	84.7	**87.5**	80.3	74.9	63.0	88.1	87.8	87.5	**67.0**	**87.5**	**79.8**	86.6	89.2	87.1	**82.6**	**59.4**	**81.3**	78.7	88.2	80.4

rate. For fair comparison, we re-implement FSSD and RFB with Pytorch-0.4.0 and CUDNN V7, the same environment as that of our model, by using the released code in [20] and [33]. When only the context and fusion modules are added, our method produces a mAP of 80.5%. When adding the ensemble to our model, the performance can be improved to 81.1%, which is 3.6 points higher than SSD and 1.6 points higher than FSSD. In Table 1, the multi-scale feature fusion module uses deconvolution to resize feature maps and concatenation to combine feature maps of different scales. The input image size in our models is 300×300.

To understand the performance of our models in more detail, we use the detection analysis tool in [39]. Fig. 6 shows the cumulative fraction of detections that are correct (Cor) or false positive due to poor localization (Loc), confusion with similar

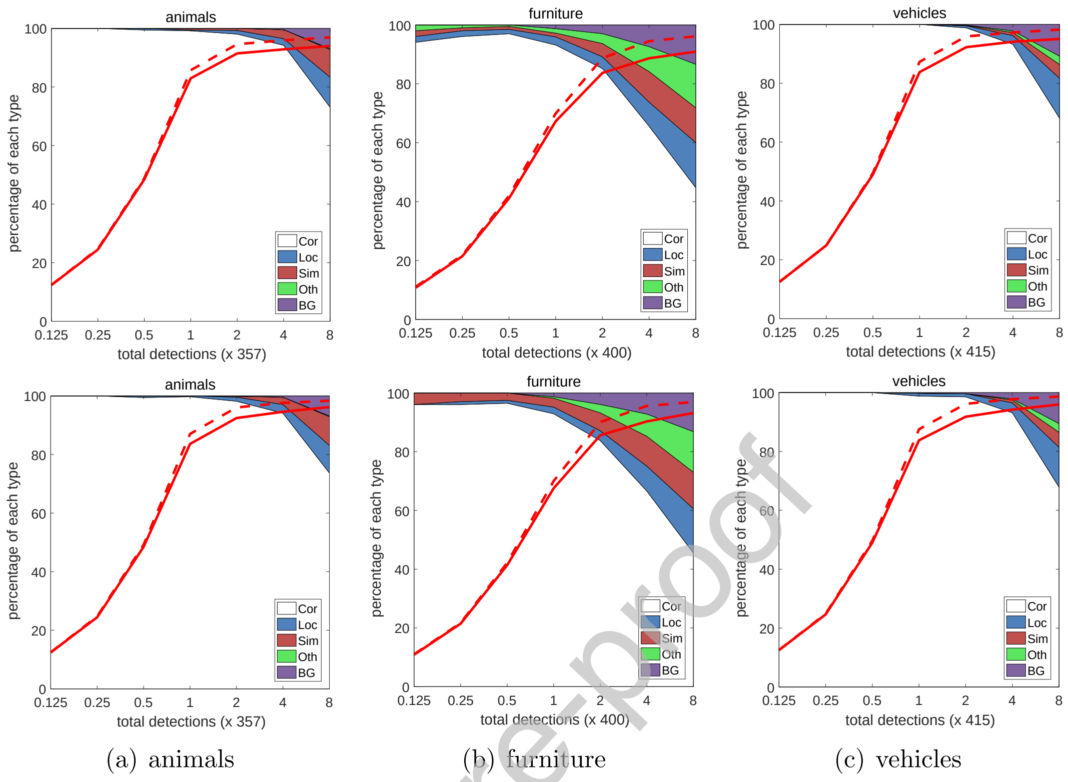
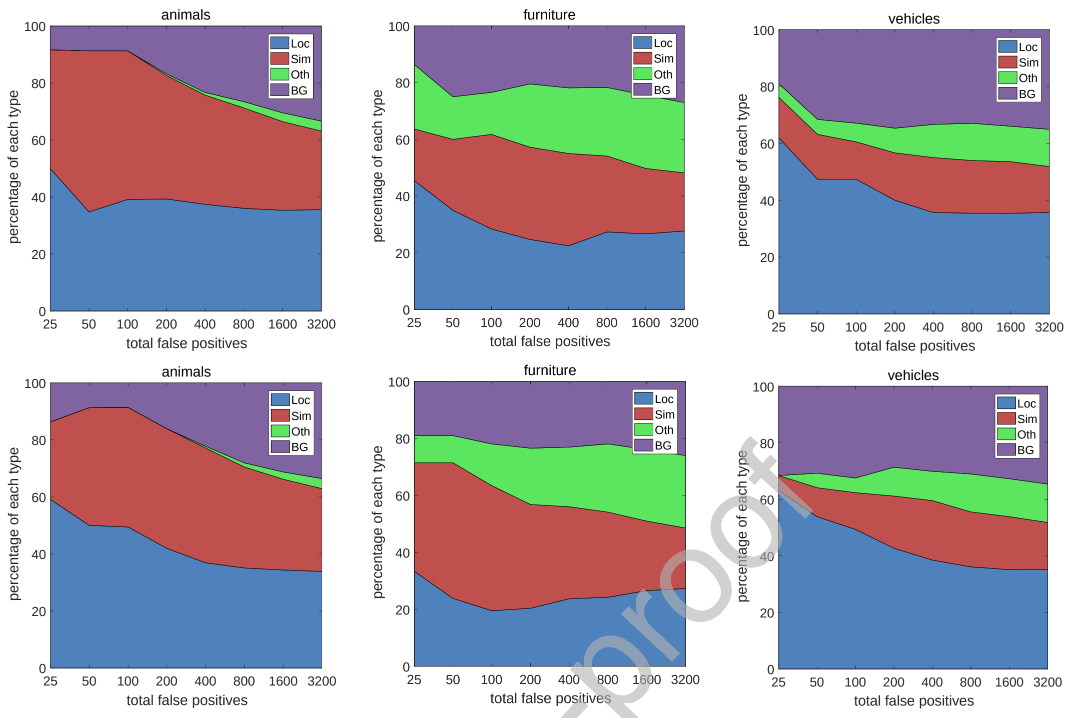


Fig 6. Visualization of the performance of our models on animals, furniture, and vehicles classes in the Pascal VOC 2007 test. **Top: without ensemble. Bottom: with ensemble.** The dashed red line reflects the change of recall with weak criteria (0.1 jaccard overlap) as the number of detections increases while the solid red line uses the strong criteria (0.5 jaccard overlap).

categories (Sim) or with others (Oth), or with background (BG). As shown in Fig. 6, our models can get a higher recall both with the strong and weak criteria as well as high-quality detection of various object categories, especially the model with the ensemble. By comparing the top and bottom rows of Fig. 6, we can observe that the recall of the ensemble model is higher than the model without the ensemble, especially in the animal categories. Comparing the two categories of furniture and vehicles in Fig. 6, we can find that the ratio of the white area to the total area in the bottom row is larger than the top row, indicating that the ensemble model can detect high-quality various object categories. Compared with other state-of-the-art detectors, our model



(a) animals (b) furniture (c) vehicles

Fig 7. Distribution of top-ranked false positive types of our models on animals, furniture, and vehicles classes in the Pascal VOC 2007 test. **Top: without ensemble. Bottom: with ensemble.**

has fewer false positive results caused by poor localization, confusion with similar categories or with others, or with background due to the context module, ensemble methods, and the multi-scale feature fusion module. The reason why is that the context module can provide more accurate localization, whereas the ensemble method is easier to distinguish the difference between categories; in addition, the multi-scale feature fusion module can learn more richness from the object's features.

Fig. 7 demonstrates that most false positive of our models are due to poor localization and confusion with background. For animals and furniture, confusion with similar categories is another cause of false positive results. Furthermore, compared with the model without the ensemble, the ensemble model has fewer false positive types caused by confusion with others because the ensemble model makes it easier to distinguish the difference between categories.

4.3. Result on PASCAL VOC 2012

Table 2. Comparison of different state-of-the-art methods on **PASCAL VOC 2012**. DSSD321, RON384, and SSD300 indicate the input image dimensions of DSSD, RON, and SSD are 321×321, 384×384 and 300×300, respectively. **C** means context module, **M** means multi-scale feature fusion module, and **E** indicates that the ensemble is used at the inference time. Ours$_1$ is the ensemble of similar models. Ours$_2$ is the feature ensemble. Ours$_3$ is the ensemble of different models. * means the test on Pytorch-0.4.0 and CUDNN V7 for fair comparison.

Method	C	M	E	Backbone	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
ION[21]				VGG16	74.7	86.9	84.5	75.2	58.2	57.7	80.5	78.3	90.4	54.4	79.9	60.5	88.4	83.0	83.0	81.2	50.7	77.3	67.6	83.5	72.3
DSSD321[22]				Residual-101	76.3	87.3	83.3	75.4	64.6	46.8	82.7	76.5	**92.9**	59.5	78.3	64.3	91.5	86.6	86.6	82.1	53.3	79.6	**75.7**	85.2	73.9
HyperNet[19]				VGG16	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.9	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7
RON384[35]				VGG16	73.0	85.4	80.6	71.9	56.3	49.8	80.6	76.8	88.2	53.6	78.1	60.4	86.4	81.5	83.8	79.4	48.6	77.4	67.7	83.4	69.5
MLKP[37]				VGG16	75.5	86.4	83.4	**78.2**	60.5	**57.9**	80.6	79.5	91.2	56.4	81.0	58.6	91.3	84.4	84.3	83.5	**56.5**	77.8	67.5	83.9	67.4
SSD300[12]				VGG16	75.8	88.1	82.9	74.4	61.9	47.6	82.7	78.8	91.5	58.1	80.0	64.1	89.4	85.7	85.5	82.6	50.2	79.8	73.6	86.6	72.1
FSSD*[20]				VGG16	75.7	88.7	82.7	74.8	62.6	51.9	83.7	79.0	90.2	58.4	80.4	61.7	87.9	84.5	85.1	83.2	49.7	80.8	70.4	86.4	72.6
RFB*[33]				VGG16	76.7	87.9	85.4	74.2	62.7	51.8	83.3	80.0	91.2	60.4	82.0	62.6	88.9	85.9	86.5	83.8	52.9	82.6	73.5	86.6	72.5
Ours	✓			VGG16	75.9	88.1	84.4	72.9	62.4	50.7	83.4	79.0	91.1	58.6	79.6	63.5	89.7	85.5	86.1	83.5	51.0	79.5	71.5	86.0	71.1
Ours		✓		VGG16	76.8	88.3	84.4	75.9	65.5	52.4	84.2	80.0	91.2	59.7	81.8	63.4	89.7	85.2	86.0	84.1	51.8	80.5	72.0	87.2	73.6
Ours	✓	✓		VGG16	77.0	88.5	85.5	74.5	63.5	51.8	84.6	80.1	91.0	60.5	82.4	64.8	**89.9**	86.2	86.6	83.8	51.1	82.7	72.2	86.7	72.9
Ours$_1$	✓	✓	✓	VGG16+VGG16	76.9	**88.8**	85.5	75.1	63.9	52.9	84.6	79.2	91.1	60.4	82.1	63.8	89.3	85.9	86.4	83.9	52.5	80.9	72.1	86.4	72.8
Ours$_2$	✓	✓	✓	VGG16+MobileNet	76.8	87.4	85.0	75.3	63.5	52.0	83.3	80.1	90.4	59.9	82.9	65.2	88.3	86.7	86.5	83.9	52.5	81.8	71.5	87.1	72.0
Ours$_3$	✓	✓	✓	VGG16+MobileNet	**78.1**	**88.8**	**86.2**	76.6	**65.6**	52.9	**84.9**	**80.9**	91.7	**62.3**	**83.1**	**67.5**	**89.9**	**87.5**	**87.1**	**84.6**	53.6	**83.6**	73.3	**87.5**	**74.1**

We used the same settings as the VOC2007 experiment and submitted the results to a public evaluation server[3] to evaluate our model on the PASCAL VOC 2012 test set. Our models are also trained on the joint training set of VOC 2007 trainval and 2012 trainval, but tested on the VOC 2012 test set. The result of comparisons between our models and some state-of-the-art networks can be seen in Table 2. Our method obtains a mAP of 78.1%, which is 1.8 points higher than DSSD. For fair comparison, we repeated FSSD and RFB in the same environment as our model

[3]Anonymous URL: http://host.robots.ox.ac.uk:8080/anonymous/GFRBJV.html

by using the released code in [20] and [33]. The input image size in our models is 300×300.

4.4. Result on MS COCO

Table 3. Comparison of different state-of-the-art methods on **MS COCO test-dev 2015 dataset**. DSSD321, RON384, SSD300, and STDN300 indicate the input image dimensions of DSSD, RON, and SSD are 321×321, 384×384 and 300×300, respectively. * indicates that the models use the NMS Ensembling.

Method	Train data	Backbone	Avg. Precision, IoU			Avg. Precision, Area		
			0.5:0.95	0.5	0.75	S	M	L
ION[21]	train	VGG16	23.6	43.2	23.6	6.4	24.1	38.3
DSSD321[22]	trainval35k	Residual-101	28.0	46.1	29.2	7.4	28.1	47.6
RON384[35]	trainval	VGG16	25.4	46.5	25.0	-	-	-
MLKP[37]	trainval35k	VGG16	26.9	48.4	26.9	8.6	29.2	41.1
STDN300[38]	trainval	DenseNet-169	28.0	45.6	29.4	7.9	29.7	45.1
SSD300[12]	trainval35k	VGG16	25.1	43.1	25.8	6.6	25.9	41.4
FSSD[20]	trainval35k	VGG16	27.1	47.7	27.8	8.7	29.2	42.2
RFB[33]	trainval35k	VGG16	30.3	49.3	31.8	11.8	31.9	45.9
Ours	trainval35k	VGG16	30.7	49.8	32.2	12.4	34.4	46.2
Ours*	trainval35k	VGG16	**31.4**	**51.0**	**32.7**	**12.6**	**34.9**	**48.7**

To further validate our model, in addition to the PASCAL VOC, we also test our model on the MS COCO dataset. The result can be seen in Table 3. We train our model on trainval135 set and test on test-dev2015 set. Because test-dev2017 and test-dev2015 contain the same image, the results obtained from them are comparable. Our model can improve the SSD over 6.3% at $IOU = [0.5 : 0.05 : 0.95]$ and is superior to other competing methods. When adopting NMS Ensembling, our model outperforms state-of-the-art methods MLKP, STDN, and RFB by 4.5%, 3.4% and 1.1%, respectively. In particular, this demonstrates that our models improve competing methods in detecting small or medium-sized objects.

5. Ablation analysis

To study the impact of the ensemble module, context module, and multi-scale feature fusion module, we implemented some contrast ablation experiments. We use

SSD as the baseline because our improvements are implemented on SSD. As shown in Table 4, our method improved the baseline from 77.5% to 79.1% when adding the context module. The multi-scale feature fusion module and ensemble further improve the baseline mAP to 80.1% and 81.1%, respectively. Note that all models are trained on the joint training sets of VOC 2007 and 2012 trainval, and tested on the VOC 2007 test set. For a fair comparison, all parameters and image sizes are set to the same.

Table 4. Ablation analysis on PASCAL VOC 2007 test set.

Module	Our model				Baseline(SSD300)
Context module	✓		✓	✓	
Multi-scale feature fusion module		✓	✓	✓	
Ensemble				✓	
mAP(%)	79.1	80.1	80.5	81.1	77.5

5.1. Analysis for ensemble

5.1.1. Do ensemble strategies help?

We propose three ensemble strategies. Except for Feature Ensembling, the other two ensemble strategies involve two models, so we use NMS Ensembling to ensemble the two models. The summary results of our experiments are given in Table 5. When we use different models for the ensemble, our detectors reached a competitive result (81.1% mAP), which is 0.6% higher than without the ensemble method. However, when we use similar models for the ensemble or the feature ensemble method, the results are not very competitive, especially the feature ensemble method. From the results, we can summarize the key findings: (1) the ensemble of different models is helpful for object detection because different models can extract a wider range of features and semantic information, and this effect is more obvious when the difference between the ensemble models is larger; (2) the impact of the ensemble of similar models on object detection task is not obvious. This is because the ensemble models come from the same calculation graph and the differences between the models are small; thus, the extracted features and semantic information are easy to replicate; and (3) the features ensemble is less effective for the object detection task because different backbone extraction features introduce more noise information after the ensemble is implemented. (4)To further illustrate the efficiency of our ensemble model, we compare our ensemble results with [40](81.1%), which is the ensemble Faster RCNN with ResNet-101 and ResNet-152, on the premise of a fair comparison by ensuring the number of ensemble models and the dataset are consistent. The results show that our ensemble model can achieve state-of-the-art performance through two one-stage models. In further experiments, we found that adding ResNet-50 and VGG16 to the ensemble produced results 0.14% higher than the result of [40].

Table 5. Detection performance with different ensemble strategies.

Ensemble method	Ensemble of similar models	Feature ensemble	Ensemble of different models
mAP(%)	80.5	80.2	81.1

5.1.2. *Which ensemble strategy is better?*

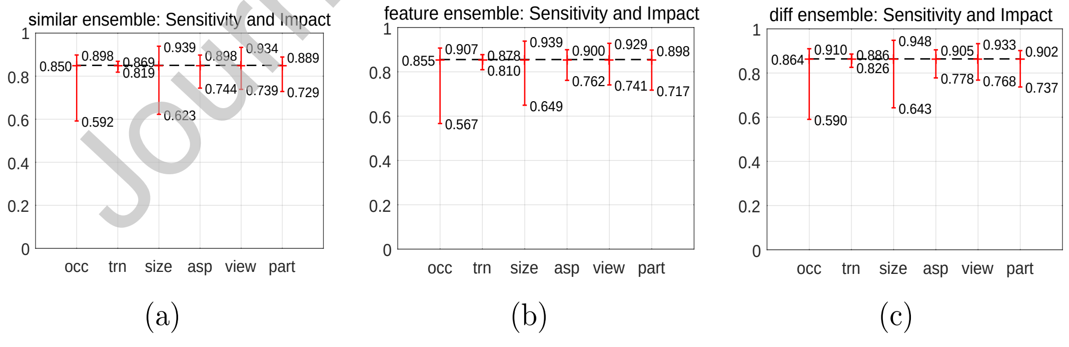


Fig 8. Summary of sensitivity and impact of object characteristics on VOC2007 test set using [39]. We show the performance of three ensemble strategies within each characteristic (occlusion, truncation, bounding box area, aspect ratio, viewpoint, and part-visibility). (a) Ensemble of similar models. (b) Feature ensemble. (c) Ensemble of different models. Both ends of the solid line are the highest performing (top) and lowest performing (bottom) subsets, respectively, and the difference between max and min indicates sensitivity; the dashed line is the overall performance and the difference between max and overall indicates the impact.

From the perspective of the mAP metric, the effect of the ensemble on different models is positive for object detection tasks. However, if we analyze these three ensemble strategies for specific tasks, we will find that each method has its own advantages and disadvantages. As shown in Fig. 8, among the three ensemble strategies, the overall performance of the ensemble of different models is highest, feature ensemble is second, and ensemble of similar models is the worst. The detector with the ensemble of different models is more sensitive to occlusion than the detector with the ensemble of similar models; however, the impact of occlusion for the latter is smaller than the former. Overall, the detector with the ensemble of similar models is more robust than the other two detectors to truncation and the detector with feature ensemble is more robust than the other two detectors to different object sizes. The detector with the ensemble of different models is robust to aspect ratios, viewpoints, and part-visibility. In conclusion, the detector with the ensemble of different models is better for object detection tasks.

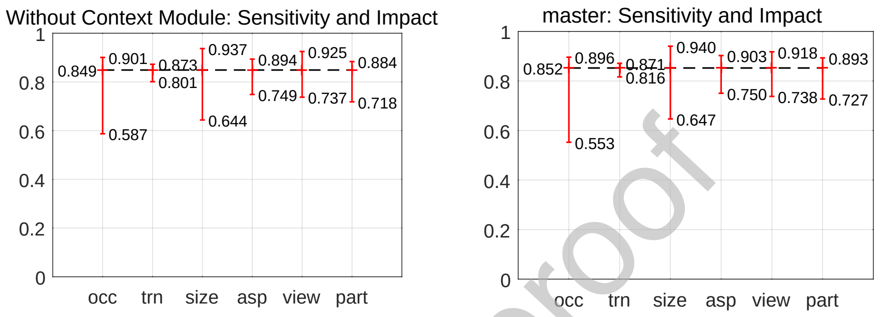
5.2. Analysis for context module

5.2.1. Does context module help?

As shown in Table 6, when the context module is added, the detector outperforms the detector without the context module. Fig. 9 reveals that the overall performance of the detector with the context module exceeds the detector without the context module. Although the detector with the context module is more sensitive than the detector without the context module to an occluded object, the impact of occlusion on overall performance is smaller compared with the latter. The detector with the context module is more robust to truncation and viewpoint, whereas the detector without the context module is more robust to aspect ratio. Therefore, our model improves the performance of detection by embedding contextual information and our context module is helpful for the overall performance.

Table 6. Detection performance with and without feature fusion module.

Context module	Yes	No
mAP(%)	80.5	80.1



(a) (b)

Fig 9. Summary of sensitivity and impact of object characteristics on VOC2007 test set using [39]. We show the performance of the context module within each characteristic (occlusion, truncation, bounding box area, aspect ratio, viewpoint, and part-visibility): (a) without context module and, (b) with context module. Both ends of the solid line are the highest performing (top) and lowest performing (bottom) subsets, respectively, and the difference between max and min indicates sensitivity; the dashed line is the overall performance and the difference between max and overall indicates the impact.

5.2.2. Speed

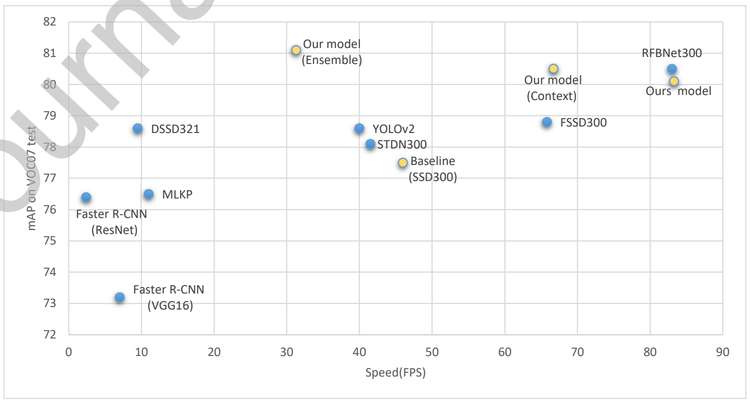


Fig 10. Accuracy and speed on PASCAL VOC2007. Speeds are measured on TitanX GPU except for FSSD, MLKP, and our models, which are measured on Nvidia 1080Ti. For FSSD300, we use the result published in their own papers.

As shown in Fig. 10, because we added a complex context module, our detection consumes about 25% extra time. However, compared with DSSD and STDN, our methods are still much faster. The reason why is the use of dilated convolution. It is precisely because dilated convolution has the feature of increasing the receptive field without increasing the computational cost that allows us to add complex context modules without unduly reducing the detector speed. Obviously, the dilated convolution makes our detectors faster than most object detectors and with competitive precision. Our model has a certain speed drop after adding the ensemble module. However, compared with the SSD, our model sacrifices a certain speed, which brings a great improvement in accuracy. This also proves that ensemble learning improves accuracy at the expense of speed.

5.3. Analysis for multi-scale feature fusion module

To illustrate the effectiveness of the multi-scale feature fusion module, we design a series of experiment and analyze how the feature fusion module affects the final performance. All results in this section are tested using the PASCAL VOC 2007 test set.

5.3.1. Does multi-scale feature fusion module help?

An important property of our method is fusing feature maps with different scales. To better understand the importance of multi-scale feature fusion, we choose the layer conv4_3, conv5_3 in the conventional SSD300 based on VGG16 to fuse with the second layer of the context module in order to ascertain whether the multi-scale feature fusion module really helps. Table 7 reports the results of our experiments,

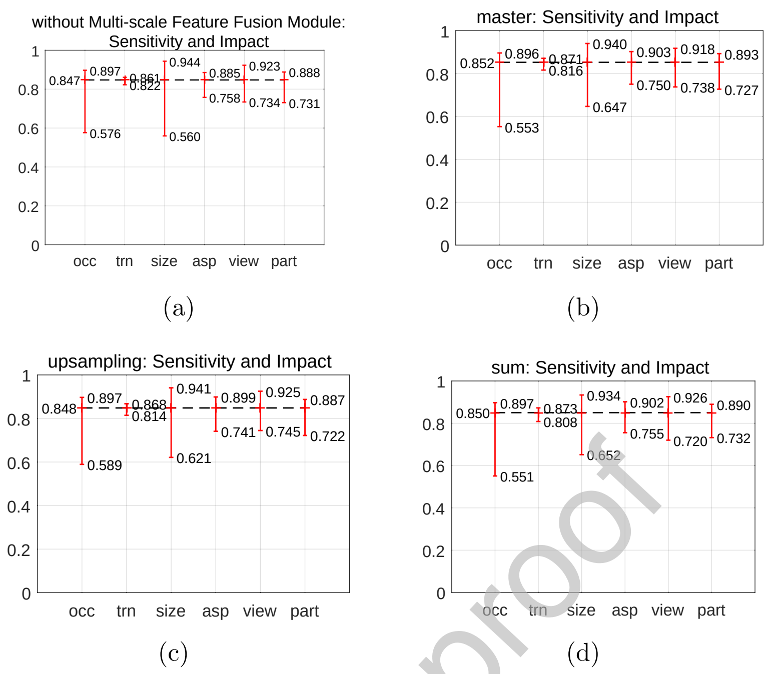


Fig 11. Summary of sensitivity and impact of object characteristics on the VOC2007 test set using [39]. We show feature fusion module performance within each characteristic (occlusion, truncation, bounding box area, aspect ratio, viewpoint, and part-visibility): (a) without the fusion module; (b) with the feature module with deconvolution and concatenation; (c) the feature module with upsampling and concatenation; and (d) the feature module with upsampling and element-wise summation. Both ends of the solid line are the highest performing (top) and lowest performing (bottom) subsets, respectively, and the difference between max and min indicates sensitivity; the dashed line is the overall performance and the difference between max and overall indicates the impact.

indicating that model performance was significantly better with than without the fusion module and that even the worst fusion modules can increase mAP by 0.8%. In addition, we can draw the following conclusion from Fig. 11: the overall performance of a detector with a fusion module is significantly higher than one without a fusion module and the former is more robust to object size than the latter. This proves that the fusion module is helpful for improving the performance of the detector.

Table 7. Detection performance with different feature fusion modules. The first row is a model without the fusion module. f_i is a method for adapting feature maps of different scales to the same size. φ_f is a way for fusing the selected feature maps.

fusion	f_i		φ_f		mAP(%)
	deconvolution	upsampling	concatenation	element-wise summation	
✓					79.1
✓	✓		✓		80.5
✓		✓	✓		80.4
✓		✓		✓	79.9

5.3.2. *Which method of adjusting the feature map is better?*

We use deconvolution and upsampling to adjust the feature map size and compare the performance of both. Table 7 illustrates that the fusion module with deconvolution has similar competitiveness to upsampling. Furthermore, we can draw the following conclusions by comparing Figs. 11(b) and 11(c). (1) The overall performance of the detector with the deconvolution fusion module is higher than with the upsampling fusion module. (2) The detector with the deconvolution fusion module and the detector with the upsampling fusion module are more sensitive to occlusion and different object size; however, the impacts of occlusion and object size are smaller for the detector with the deconvolution fusion module. (3) The detector with the deconvolution fusion module is more robust than the detector with the upsampling fusion module to occlusion, object size, and viewpoint. Therefore, deconvolution is better than upsampling and more suitable for multi-scale feature fusion modules.

5.3.3. *Which fusion method is better?*

We fuse feature maps of different scales via concatenation and element-wise summation, respectively. Table 7 demonstrates that the fusion module with concatenation is more competitive than with element-wise summation. Based on the comparison of Figs. 11(c) and 11(d), we have summarized the following conclusions. The detector with element-wise summation is more robust than the detector with concatenation to occlusion, object size, and aspect ratios. Moreover, the overall performance of the detector with element-wise summation is higher than the detector with concatenation to occlusion. Therefore, for specific tasks, such as occlusion, object size, and aspect ratios, element-wise summation is more suitable for the multi-scale feature fusion module than concatenation even if the fusion module with concatenation has higher mAP.

6. Conclusion

In this paper, we proposed a faster and powerful object detector. The detector combines ensemble learning and deep learning for object detection. For ensemble learning, we proposed two ensemble modes and analyzed their characteristics, respectively. For deep learning, we constructed a new convolutional neural network framework based on context features and multi-scale feature fusion. Our network is built on SSD and trained end-to-end by optimizing a multi-task loss. A series of experiments on PASCAL VOC and MS COCO datasets demonstrated that our detector can improve the performance of conventional SSD and outperforms several state-of-the-art object detectors in terms of accuracy and efficiency. Our studies illustrate that ensemble learning can further improve the performance of object detection. Therefore, how to apply ensemble learning to the field of object detection will be a problem worth studying. We hope that this paper not only promotes the research of object detectors but also facilitates future research activities in the combination of ensemble learning and object detection.

Although we use MobileNet-V1 for integration and use dilated convolution to control the amount of computation, which guarantees a good compromise between efficiency and accuracy, the computational overhead and complexity of the network is still unsatisfactory. Therefore, future work will mainly have the following two directions:

- Explore more efficient networks for the ensemble to reduce network complexity without compromising accuracy.
- Feature Ensembling is more robust to object size, which is beneficial for detecting small objects. In the future, we will explore more reasonable ensemble models and features that reduce the impact of noise caused by Feature Ensembling and improve the accuracy of detection models based on Feature Ensembling.

Acknowledgment

This work was supported by National Key Research and Development Program [Grant No.2016YFB0800105]; Sichuan Province Scientific and Technological Support Project [Grant Nos.2018GZ0255, 2019YFG0191].

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255. doi:10.1109/CVPRW.2009.5206848.

[2] R. E. Schapire, Y. Freund, Boosting: Foundations and Algorithms, The MIT Press, 2012. doi:10.1108/03684921311295547.

[3] L. Breiman, Bagging predictors, Machine Learning 24 (1996) 123–140. doi:10.1007/BF00058655.

[4] L. Breiman, Random forests, Machine Learning 45 (2001) 5–32. doi:10.1023/A:1010933404324.

[5] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, K. Q. Weinberger, Snapshot ensembles: Train 1, get m for free, in: International Conference on Learning Representations, 2017. arXiv:1704.00109.

[6] T. Garipov, P. Izmailov, D. Podoprikhin, D. Vetrov, A. G. Wilson, Loss surfaces, mode connectivity, and fast ensembling of dnns, in: Neural Information Processing Systems (NIPS), 2018. arXiv:1802.10026.

[7] N. Rooney, D. Patterson, A weighted combination of stacking and dynamic integration, Pattern Recognition 40 (2007) 1385–1388. doi:10.1016/j.patcog.2006.10.008.

[8] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587. doi:10.1109/CVPR.2014.81.

[9] G. Ross, Fast r-cnn, in: 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1440–1448. doi:10.1109/ICCV.2015.169.

[10] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (2016) 1137–1149. doi:10.1109/TPAMI.2016.2577031.

[11] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788. doi:10.1109/CVPR.2016.91.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, Ssd: Single shot multibox detector, in: European Conference on Computer Vision (ECCV), 2016, pp. 21–37. doi:10.1007/978-3-319-46448-0_2.

[13] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, A. W. M. Smeulders, Selective search for object recognition, International Journal of Computer Vision 104 (2013) 154–171. doi:10.1007/s11263-013-0620-5.

[14] C. L. Zitnick, P. Dollr, Edge boxes: Locating object proposals from edges, in: European Conference on Computer Vision (ECCV), 2014, pp. 391–405. doi:10.1007/978-3-319-10602-1_26.

[15] K. He, X. Zhang, S. Ren, J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, in: European Conference on Computer Vision (ECCV), 2014, pp. 346–361. doi:10.1007/978-3-319-10578-9_23.

[16] Z. Cai, Q. Fan, R. S. Feris, N. Vasconcelos, A unified multi-scale deep convolutional neural network for fast object detection, in: European Conference on Computer Vision (ECCV), 2016, pp. 354–370. doi:10.1007/978-3-319-46493-0_22.

[17] W. Chu, D. Cai, Deep feature based contextual model for object detection, Neurocomputing 275 (2018) 1035–1042. doi:10.1016/j.neucom.2017.09.048.

[18] T.-Y. Lin, P. Dollr, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 936–944. doi:10.1109/CVPR.2017.106.

[19] T. Kong, A. Yao, Y. Chen, F. Sun, Hypernet: Towards accurate region proposal generation and joint object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 845–853. doi:10.1109/CVPR.2016.98.

[20] Z. Li, F. Zhou, Fssd: Feature fusion single shot multibox detector (2017). arXiv:1712.00960.

[21] S. Bell, C. L. Zitnick, K. Bala, R. Girshick, Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2874–2883. doi:10.1109/CVPR.2016.314.

[22] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, A. C. Berg, Dssd : Deconvolutional single shot detector (2017). arXiv:1701.06659.

[23] M. Sabzevari, G. Martnez-Muoz, A. Surez, Vote-boosting ensembles, Pattern Recognition 83 (2018) 119–133. doi:10.1016/j.patcog.2018.05.022.

[24] H. KuanSok, M. P.-L. Ooi, Y. C. Kuang, S. Demidenko, Multivariate alternating decision trees, Pattern Recognition 50 (2016) 195–209. doi:10.1016/j.patcog.2015.08.014.

[25] Z. Yu, D. W. J. You, H.-S. W. S. Wu, J. Zhang, G. Han, Progressive subspace ensemble learning, Pattern Recognition 60 (2016) 692–705. doi:10.1016/j.patcog.2016.06.017.

[26] H. Ren, Z.-N. Li, Object detection using boosted local binaries, Pattern Recognition 60 (2016) 793–801. doi:10.1016/j.patcog.2016.07.010.

[27] X. Xu, Y. Li, G. Wu, J. Luo, Multi-modal deep feature learning for rgb-d object detection, Pattern Recognition 72 (2017) 300–313. doi:10.1016/j.patcog.2017.07.026.

[28] A. Shrivastava, A. Gupta, Contextual priming and feedback for faster r-cnn, in: European Conference on Computer Vision (ECCV), 2016, pp. 330–348. doi:10.1007/978-3-319-46448-0_20.

[29] H. Hu, J. Gu, Z. Zhang, J. Dai, Y. Wei, Relation networks for object detection, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 3588–3597. doi:10.1109/CVPR.2018.00378.

[30] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE Transactions on Pattern Analysis and Machine Intelligence 40 (2018) 834–848. doi:10.1109/TPAMI.2017.2699184.

[31] V. K. Fisher Yu, Multi-scale context aggregation by dilated convolutions, in: International Conference on Learning Representations, 2016. arXiv:1511.07122.

[32] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, J. Sun, Detnet: Design backbone for object detection, in: European Conference on Computer Vision (ECCV), 2018, pp. 339–354. doi:10.1007/978-3-030-01240-3_21.

[33] S. Liu, D. Huang, Y. Wang, Receptive field block net for accurate and fast object detection, in: European Conference on Computer Vision (ECCV), 2018, pp. 404–419. doi:10.1007/978-3-030-01252-6_24.

[34] W. Juan, T. Xiaoming, X. Mai, D. Yiping, L. Jianhua, Hierarchical objectness network for region proposal generation and object detection, Pattern Recognition 83 (2018) 260–272. doi:10.1016/j.patcog.2018.05.009.

[35] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, Y. Chen, Ron: Reverse connection with objectness prior networks for object detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5244–5252. doi:10.1109/CVPR.2017.557.

[36] E. Ohn-Bar, M. M. Trivedi, Multi-scale volumes for deep object detection and localization, Pattern Recognition 61 (2017) 557–572. doi:10.1016/j.patcog.2016.06.002.

[37] H. Wang, Q. Wang, M. Gao, P. Li, W. Zuo, Multi-scale location-aware kernel representation for object detection, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 1248–1257. doi:10.1109/CVPR.2018.00136.

[38] P. Zhou, B. Ni, C. Geng, J. Hu, Y. Xu, Scale-transferrable object detection,

in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 528–537. doi:10.1109/CVPR.2018.00062.

[39] D. Hoiem, Y. Chodpathumwan, Q. Dai, Diagnosing error in object detectors, in: European Conference on Computer Vision (ECCV), 2012, pp. 340–353. doi:10.1007/978-3-642-33712-3_25.

[40] B. Cheng, Y. Wei, H. Shi, R. Feris, J. Xiong, T. Huang, Revisiting rcnn: On awakening the classification power of faster rcnn, in: European Conference on Computer Vision (ECCV), 2018, pp. 473–490. doi:10.1007/978-3-030-01267-0_28.

Jie Xu received the Bachelor's degree in automation from Chong Qing University, China, in 2003, the Master's. degree in information and automation engineering from University of Besancon, France, in 2004, and the Ph.D. degree in automatic system from National Institute of Applied Sciences (INSA-Toulouse), France, in 2008.He is an Associate Professor of the School of Information and Communication Engineering in University of Electronic Science and Technology of China. He has authored or coauthored nearly 40 publications in journals and conferences. His research interests include the pattern recognition, deep learning, object detection, and network and information security.

Wei Wang received the B.S. degree in communication engineering from Jiangsu University, Zhenjiang, China, in 2017. He is now pursuing the M.S. degree in electronics and communication engineering at University of Electronic Science and Technology of China, Chengdu, China. His current research interests include pattern recognition, object detection, computer vision and deep learning.

Hanyuan Wang received the B.S. degree in communication engineering from Yunnan University, Kunming, China. She is currently working toward the M.S.degree in communication and information system from University of Electronic Science and Technology of China, Chengdu, China. Her current research interests include object detection and pattern recognition, computer vision and deep learning.

Jinhong Guo received the Bachelor's degree in electronic engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2010, and the Ph.D. degree in biomedical engineering from Nanyang Technological University, Singapore, in 2014. He is currently a Full Professor with the School of

Information and Communication Engineering, University of Electronic Science and Technology of China. After his doctoral studies, he was a Postdoctoral Fellow in the Pillar of Engineering Design with MIT-SUTD Singapore from 2014 to 2015. He then worked as a Visiting Professor with the School of Mechanical Engineering, University of Michigan, Ann Arbor, MI, USA, from January 2016 to July 2016. His current research focuses on electrochemical sensor and lab-on-a-chip devices for point of care test toward clinical use. He has authored or coauthored more than 70 publications in top journals, such as the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, the IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS, Analytical Chemistry, Biosensor and Bioelectronics, etc. He was the recipient of the China Sichuan Thousand Talents Plan for Scholars Award (2015) and Chengdu Expert in Science and Technology Award (2015).