Mobile Networks and Applications (2021) 26:77–87
https://doi.org/10.1007/s11036-020-01723-z

RSANet: Towards Real-Time Object Detection with Residual Semantic-Guided Attention Feature Pyramid Network

Quan Zhou[1] · Jie Wang[1] · Jia Liu[1] · Shenghua Li[1] · Weihua Ou[2] · Xin Jin[3]

Accepted: 30 November 2020 / Published online: 4 January 2021

Abstract

The huge computational overhead limits the inference of convolutional neural networks on mobile devices for object detection, which plays a critical role in many real-world scenes, such as face identification, autonomous driving, and video surveillance. To solve this problem, this paper introduces a lightweight convolutional neural network, called RSANet: Towards Real-time Object Detection with Residual Semantic-guided Attention Feature Pyramid Network. Our RSANet consists of two parts: (a) Lightweight Convolutional Network (LCNet) as backbone, and (b) Residual Semantic-guided Attention Feature Pyramid Network (RSAFPN) as detection head. In the LCNet, in contrast to recent advances of lightweight networks that prefer to utilize pointwise convolution for changing the number of feature maps, we design a Constant Channel Module (CCM) to save the Memory Access Cost (MAC) and design Down Sampling Module (DSM) to save the computational cost. In the RSAFPN, meanwhile, we employ Residual Semantic-guided Attention Mechanism (RSAM) to fuse the multi-scale features from LCNet for improving detection performance efficiently. The experiment results show that, on PASCAL VOC 20007 dataset, RSANet only requires 3.24 M model size and needs only 3.54B FLOPs with a 416×416 input image. Compared to YOLO Nano, our method obtains a 6.7% improvement in accuracy and requires less computation. On MS COCO dataset, RSANet only requires 4.35 M model size and needs only 2.34B FLOPs with a 320×320 input image. Our method obtains a 1.3% improvement in accuracy compared to Pelee. The comprehensive experiment results demonstrate that our model achieves promising results in terms of available speed and accuracy trade-off.

Keywords Real-time · Object detection · Lightweight convolutinal network · Visual attention · FPN

1 Introduction

Convolutional neural networks (CNNs) [1–6] have dominated in computer vision area since AlexNet [1] popularized deep convolutional neural networks by winning the ImageNet Challenge: ILSVRC 2012 [44]. To achieve higher accuracy, it is general to design deeper and wider neural networks. To improve the representation ability of visual data, most accurate CNNs usually have hundreds even thousands of convolutional layers and feature channels, e.g., ResNet family [4, 45, 46].

Due to these advances, the recent years have witnessed remarkable progress for the task of object detection using deep CNNs. As a pionner work, RCNN [7] employsselective search [8] to find a set of region proposals, and then each proposal is entered into a CNN for classification and regression further. After that, a series of CNNs have been proposed for object detction [9–11, 28]. To our konwledge, above region-based approaches have bringed remarkable improvements on detection accuracy. However, due to the extra region proposal extraction step, above two-stage methods are computationally expensive for real-time application scenarios, such as self-driving, mobile face recognition and video surveillance. An alternative methods to try to address this problem are one-stage pipeline, where class probabilities and bounding box offsets are directly predicted based on feature maps from a full image with a forward convolutional network. Therefore, one stage methods do not need extra region proposal extraction step. Since the whole pipeline is one-stage network, one-stage detectors [17–21, 30, 35, 49] are more time-saving and more adaptive to real-time application scenarios. These methods surely bring remarkable inference speed improvements. However, these methods still have a critical limitation. They only focus on

✉ Quan Zhou
quan.zhou@njupt.edu.cn

Extended author information available on the last page of the article.

detection pipeline simplification but ignore the complexity of the network itself.

In order to overcome the challenge and adapt to the real time scenario requirements, there has been a rising interest in running efficient CNN models under strict constraints on memory and computational budgets. Many innovative architectures, such as MobileNets [12], ShuffleNet [13], NASNet-A [14], MobileNetV2 [15], have been proposed in recent years, which speed up the inference speed by decreasing the model size to save the computational cost. The factors that affect the speed of network inference include model size, computational cost and MAC. Despite achieving promising performance, above works share similar limitations: they only take model size or computational cost into account, but ignore the effects of MAC on inference time.

Rather than these methods, this paper employs CCM for minimizing the MAC with limited computational cost. Besides, we adopt RSAM to obtain more powerful feature representation. More specifically, we design a novel real time network called RSANet, adopting an efficient RSAFPN architecture to achieve the trade-off of accuracy and efficiency. As shown in Fig. 1, our RSANet consists of two parts: (a) Lightweight Convolutional Network (LCNet) as backbone, and (b) Residual Semantic-guided Attention Feature Pyramid Network (RSAFPN) as detection head. Motivated from ShuffleNetv2 [16], the backbone network is mainly composed of CCM, where the number of feature map channels remains constant across the block all the way, which aims at minimizing the MAC. In contrast to standard convolution with stride 2, our DSM combines depthwise convolution with stride 2 and max pooling for saving computational cost. To our knowledge, low-level features contain local detail information for localization, and high-level features contain global semantic information for classification. Most object detection works [28, 50–53] adopt FPN [28] framework to combine high-level features with low-level features using addition or concatenation operation. In this way, a feature pyramid that has rich semantics at all levels is constructed from a single input image scale. In our framework RSAFPN, we introduce RSAM based FPN, where strong semantic information from high-level features are used to guide the low-level features for accurate classification. Meanwhile, RSAFPN employs Lightweight Convolution Unit (LCCU) that utilizes depthwise convolution to capture context information with small computational overhead. In summary, our contributions are summarized as follows:

- We propose a novel CCM, minimizing the MAC, which has been often ignored yet is a significant factor to speed.
- Based on CCM and DSM, we put forward a backbone network named LCNet to extract features.
- We design the RSAFPN head with LCCU for saving computational cost. And we propose RSAM embedded into RSAFPN to achieves detection accuracy improvement with only small additional computational budgets. Therefore, our whole network composed of LCNet and RSAPFN is lightweight.
- The experiment results show that RSANet performs promisingly for object detection in terms of available trade-off between speed and accuracy. RSANet only has 3.23M model size and needs only 3.52B FLOPs with input size 416×416. On the other hand, RSANet achieves a mean average precision (mAP) of 75.5% on PASCAL VOC 2007 [32]. RSANet only has 4.34M model size and needs only 3.89B FLOPs with input size 416×416. On the other hand, RSANet achieves a mmAP of 24.8% on MS COCO [33].

The remainder of this paper is organized as follows. After a brief introduction of related work in Section 2,

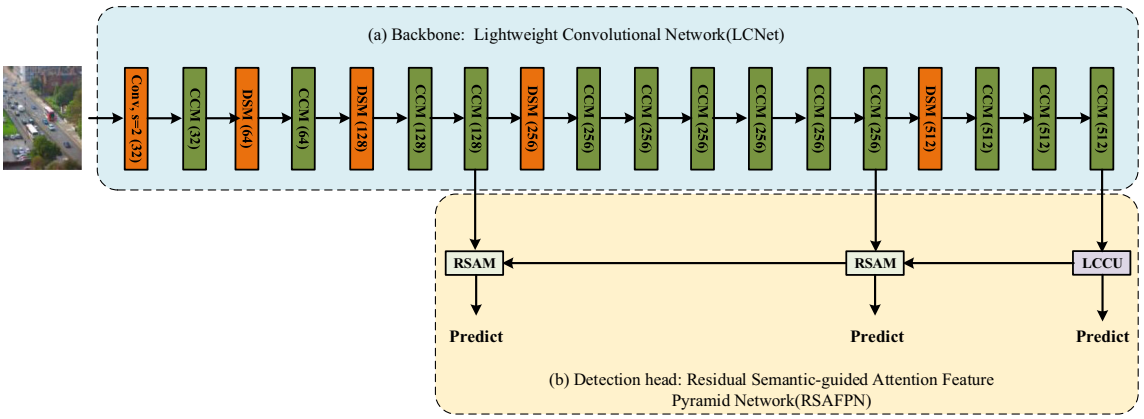

Fig. 1 The overall real-time object detection architecture of the proposed RSANet. The entire network is composed by two parts: LCNet composed of CCM and DSM for extracting features efficiently, and RSAFPN head composed of LCCU and RSAM for improving detection performance with only small additional computational budgets. (Best viewed in color)

we elaborate on the details of our RSANet in Section 3. Experimental results are given in Section 4, and Section 5 provides conclusion remarks and future work.

2 Related work

Currently, deep learning based object detection pipelines can mainly divided into two categories: (1) two-stage detectors, such as Region-based CNN (R-CNN) [7] and its variants [9–11, 47], and (2)one-stage detectors, such as You Only Look Once (YOLO) [17] and its variants [18, 19, 49]. Two-stage detectors first use a proposal generator to produce a sparse set of proposals and extract features from each proposal, followed by region classifiers which predict the category of the proposed region. One-stage detectors, on the contrary, directly make categorical prediction of objects on each location of the feature maps without the cascaded region classification step. Two-stage detectors commonly achieve better detection performance and report state-of-the-art results on public benchmarks, while one-stage detectors are more efficient and thus suitable to detect objects in limited computational resources. In this paper, we follow one-stage scheme for real-time object detection, where we employ the visual attention for detection accuracy improvement with only small additional computational budgets.

Real-time object detecion Commonly, one-stage detectors are regarded as the key to real-time object detection, which directly predict class probabilities and bounding box offsets in a single pipeline without extra compared to two-stage detectors. For instance, YOLO series [17–19, 49] and SSD series [20–22] run in real time on GPU. One stage detection methods [17–21, 30, 35] formulate detection as a single regression problem, which does not need a complex pipeline. Redmon et al. [17] proposed YOLO, a unified detector casting object detection as a regression problem from image pixels to spatially separated bounding boxes and associated class probabilities. Another one-stage detector, Single-Shot Mulibox Detector (SSD), was proposed by Liu et al. [20] in 2016, which addressed the limitations of YOLO [17]. The main contribution of SSD is the introduction of the multi-resolution detection techniques, which significantly improves the detection performance of a one-stage detector, especially for some small objects. Redmon and Farhadi proposed YOLOv2 [18], an improved version of YOLO, in which the custom GoogLeNet [3] network is replaced with the simpler DarkNet19. The YOLOv3 [19] proposed three output scales and a deeper architecture Darknet-53. Each of the scale/feature-map has its own set of anchors. Compared with YOLOv2 [18], YOLOv3 [19] reaches higher accuracy, yet with sacrifice of inference speed due to the heavier backbone. The most recent version of YOLO [49] primarily considers various strategies such as bag of freebies and bag of specials, to greatly improve the performance of the detection, but still remains high efficiency. Although these methods achieve real-time inference, they only focus on how to simplify the detection pipeline but ignore the complexity of the network itself. Unlike those methods, we proposed a lightweight architecture RSANet for speeding up.

Lightweight backbone network is an efficient way for feature extracting, which focus on how to compress model size and save computational cost, as proposed by Google MobileNet series [12, 15, 23], Kuangshi Technology's ShuffleNet [13] and SqueezeNet [24], etc. Before MobileNet [12] is invented, bottleneck structure [4] is often used to reduce computational cost. Concretely, we can adopt pointwise convolution for channel dimension reduction [4, 29]. In MobileNetv1 [12], standard convolution is decomposed into two consequent parts: depthwise convolution and pointwise convolution. The depth separable convolution is different from the standard convolution. For the standard convolution, the convolution kernel filters all input channels. For depth separable convolution, it first uses depthwise convolution to convolve different input channels separately, and then uses pointwise convolution to linearly combine the outputs of depth separable convolution. The recent researches [12, 13, 15, 23] have demonstrate that depth separable convolution can achieve similar even better results with respect to standard convolution, but at the same time it is able to greatly save computational resources and very few model size. To our knowledge, MobileNetv1 [12] proposed depth separable convolution, and added a batch normalization layer [48] on the network structure. But the structure of Mobilnetv1 does not utilize shortcut to facilitate training. The residual connection is introduced in MobileNetv2 [15], which is easy to escape from gradient vanishing. Meanwhile, it is not difficult to find that the computation of the depth separable convolution is mainly dominated by pointwise convolution, which is different from the standard convolution. Therefore, in MobileNetv2, the researchers designed an inverted residual structure to increase the proportion of the computation of depthwise convolution. YOLO Nano [30] adopts the Residual Projection Expansion-Projection (PEP) unit for enabling computational complexity reductions further. The above methods have made great contributions to the lightweight design of the network. When coupled with small backbone networks, lightweight one-stage detectors, such as MobileNet-SSD [12], MobileNetV2-SSDLite [15], Pelee [25], achieve high frame rate of inference on mobile devices. Althoug these works achieve promising results from the perspective of reducing model size and computational cost, they ignore another important factor MAC. Unlike these methods, we design a lightweight backbone, named

LCNet, which considers the effect of MAC on inference speed.

Vision attention Motivated from the application of speech recognition, visual attention is widely-used in computer vision community in recent years. Attention mechanism can be used as global context to guide the feed forward network for improving performance. In recent years, there have been several attempts [26, 27] to incorporate attention processing to improve the performance of CNNs in large-scale classification tasks. Wang et al. [26] propose Residual Attention Network that uses an encoder-decoder style attention module. By refining the feature maps, the network not only performs well, but also is robust to noisy inputs. Hu et al. [27] introduce a compact module to exploit the inter-channel relationship. In their Squeeze-and-Excitation module, they use global average-pooled features to compute channel-wise attention. Unlike these models, our method employs RSAM. These works make significant progress on vision attention area. However, they only generate attention map independently at each detection layer for reweighting but ignore the importance of the deeper layer's semantic information to the shallower layer classification. In this work, we introduce the RSAM to FPN structure for constructing RSAFPN, encoding high-level features to produce strongly semantic information for improving low-level detection performance.

3 Our method

In designing our detection network, we keep in mind that both accuracy and computational complexity are important. In this section, we first introduce the whole network architecture, and then explain in detail about our network's each part: Backbone network LCNet depicted in Fig. 1a and Residual Semantic-guided Attention Feature Pyramid Network (RSAFPN), as shown in Fig. 1b.

3.1 Network architecture

In this work, our main motivation is to obtain one network that achieves the best possible trade-off between accuracy and efficiency. With this objective in mind, we follow the current trend of employing depthwise convolutions with residual connections as the core module of our network, for the purpose of leveraging their success in detection task. Our architecture consists of two parts: backbone LCNet shown in Fig. 1a, and detection head SRAFPN show in Fig. 1b. The backbone LCNet is built mainly based on CCM unit, which enables us to design stronger architecture, but with very smaller computational overhead. In the detection part, we adopt the SAFPN to fuse the high-level semantic information and low-level detail information for improving the detection performance efficiently.

3.2 LCNet

In this section, we introduce the backbone LCNet. LCNet follows the plain overall architecture like VGG [2]. The core units of the backbone are CCM and DSM, as depicted in Fig. 2d and e, respectively. As shown in Fig. 1a, the whole backbone LCNet is composed of 5 stages. The first stage begins with one 3×3 standard convolution with stride 2, the rest begins with our DSM. And in each stage, the numbers of CCM are 1, 1, 2, 6 and 3, respectively. The feature maps from last 3 stages are used for multi-scale detection. Note that in each module, the number in bracket represents the number of convolutional kernels.

3.2.1 CCM

The recent years has witnessed many efficient residual modules, such as residual bottleneck [4] (Fig. 2a), inverted residual bottleneck [15] (Fig. 2b) and PEP [30] (Fig. 2c). However, in these methods, the number of output feature channels of each convolutional layer is different from the number of input feature channels, leading to an increase in MAC. On the contrary, CCM maintains consistency of channel numbers between input and output features, resulting in the save of MAC and thus speeding up of inference process.

CCM combines the strength of Depthwise Convolution (DWC) [12] and residual connections [4]. More specifically, as depicted in Fig. 2d, a 3×3 filter kernel is convolved with input per each channel, resulting in the independent filtering responses along output channels. Thereafter, an 1×1 pointwise convolution is used to recover channel dependency by learning a linear combinations of channels. And these two operations are duplicated in each CCM. In contrast to previous works [4, 15, 30] that the number of channels is variational, the number of feature map channels remains constant across the block all the way in the CCM. We demonstrate the effectiveness of CCM in ablation study in Section 4.4. Although the focus of this paper is object detection, we believe that CCM can be easily transferred to any existing network architectures that are used for other visual tasks, such as image classification [4, 12] and semantic segmentation [31, 38–40].

3.2.2 DSM

The operation of downsampling is commonly-used in CNN architecture. The main drawback of downsampling is the reduction of feature resolution, but it also has two benefits: enabling deeper layers gather more context to improve classification, and helpful to reduce computation.

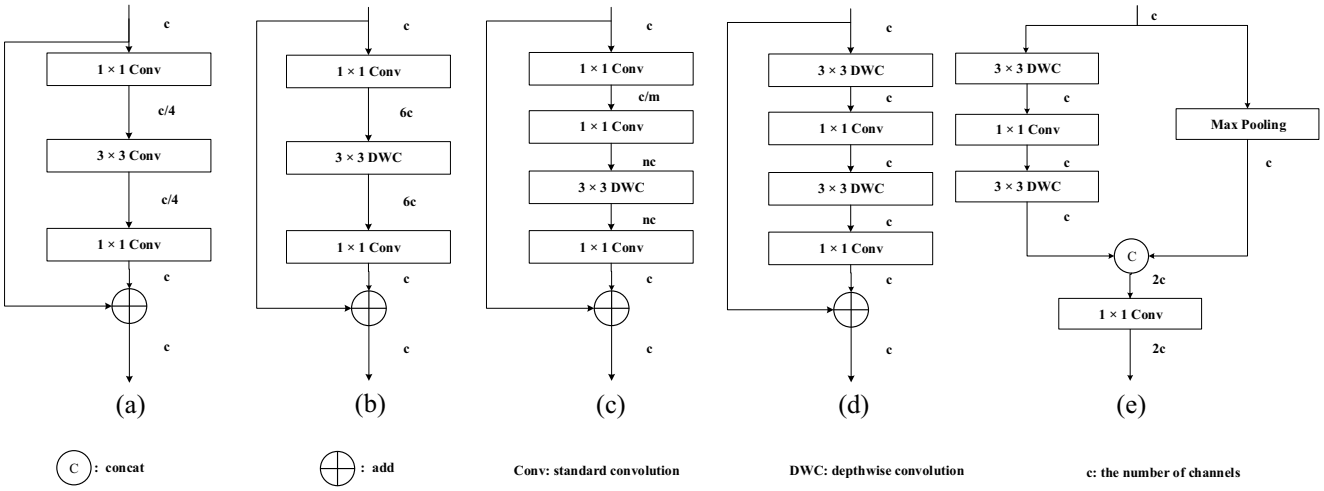

Fig. 2 Comparison of different types of residual modules. From left to right are (**a**) residual bottleneck [4], **b** inverted residual bottleneck [15], **c** PEP [30], **d** our CCM and **e** our DSM. "c" means the number of channels. In (c), the setting "c/m" means reducing the number of channels of the input feature by m times, and "nc" means increasing the number of channels of the input feature by n times, respectively

Therefore, to keep a good balance between impalement efficiency and detection performance, LCNet adopts 5 downsampling operations. Most precious works adopt max-pooling layer or the convolution with stride 2 to reduce resolution without extra computational budgets, but they directly discard filtered responses without considering their relation. On the other hand, the stride convolution reduces feature resolution by adaptively learning pixel relations, yet resulting in additional computational cost. Our DSM combines DWC with stride 2 and max-pooling, adopting the advantage of two operations. As depicted in Fig. 2e, the DSM consists of two branches. The left branch adopts 3×3 DWC with stride 2, and the right branch is implemented using max-pooling. Then an 1×1 convolution is used to linearly combine the outputs of two branches.

3.3 RSAFPN

As shown in Fig. 1b, we introduce the RSAM to construct the RSAFPN detection head. RSAM utilizes semantic information from high-level features to guide low-level features for accurate detection. To our knowledge, low-level features contain local detail information for localization, and high-level features contain global semantic information for classification. Therefore, we abstract channel attention as semantics from high-level features to reweight low-level features. On the other hand, FPN [28] framework is mainstream way to fuse high-level features and low-level features. Therefore, we introduce our RSAM into FPN. As depicted in Fig. 3a, we first use pointwise convolution to reduce the feature channels, as the number of channels in high-level features is twice of low-level features. Meanwhile, we upsample the outputs of the pointwise convolution 2 times to match the resolution of low-level features. The LCCU is adopted to capture context information based on the concatenation of high-level and low-level features. Thereafter, CAM is utilized to abstract high-level semantics using global attention. Finally, the semantic information is used to guide the output of LCCU to produce reweighted feature maps. Concretely, motivated by CAN [31], LCCU is designed to save computational cost for speeding up inference. As shown in Fig. 3b, we adopt two 3×3 depthwise convolution in LCCU to capture context information. The first 1×1 convolution is used for channels reduction, and the second one is used to combine the output of depthwise convolution. As shown in Fig. 3c, the CAM helps the low-level feature maps to obtain the semantic information from high-level feature maps. Firstly, we employ a global average pooling on the high-level features to produce strong semantic information, then an 1×1 convolution is used for dimension reduction to match the feature dimension of the output of LCCU. Then we use sigmoid function to normalization the output of 1×1 convolution. Finally, the abstract attention vector is used to reweight the output of LCCU.

4 Experiments

In order to demonstrate the effectiveness of our method, we have conducted exhausted experiments on two widely-used general object detection datasets: PASCAL VOC 2007 [32] and MS COCO [33]. In addition, we carry on two ablation studies to uncover the benefit of our CCM and show the performance improvement from our RSAM. Experimental results show that, compared with recent state-of-the-art approaches, our RSANet achieves superior performance in terms of accuracy and efficiency trade-off on two datasets.

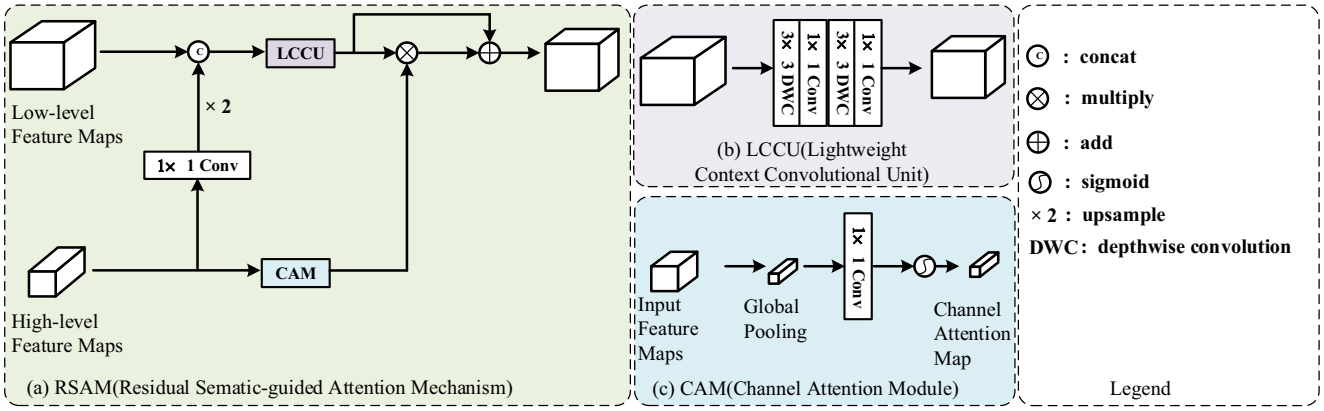

Fig. 3 **a** Detail architecture of RSAM. **b** and **c** are LCCU and CAM used in (**a**), respectively. Among them, LCCU is used to capture context information. And CAM is employed to abstract high-level semantics to guide the low-level feature maps classification (Best viewed in color)

4.1 Implementation details

Dataset The PASCAL VOC 2007 dataset contains images provided by Microsoft Research Cambridge and collected from the photo-sharing web-site "flickr", including 20 object classes, which is selected to evaluate our RSANet. For PASCAL VOC 2007, all models are trained on the VOC 2007 and VOC 2012 trainval set (16,551images), and tested on the VOC 2007 test set (4,952 images). We use the same settings and configurations. Note that the resolution of images in PASCALVOC 2007 is about 500×375 and 375×500. We also evaluate RSANet on MS COCO dataset. The model is trained from scratch on the MS COCO trainval35k set (118,287 images) and tested on the test-dev set (81,434 images). The full name of MS COCO is Microsoft Common Objects in Context, which originated from the Microsoft COCO dataset that Microsoft funded and annotated in 2014. Like the ImageNet competition, it is regarded as one of the most watched and authoritative competitions in the computer vision field, which has 80 object classes.

Hyper-parameters settings The proposed RSANet is implemented in Tensorflow library [34], which is end-to-end trained based on one GeForce RTX 2080ti GPU using Adam optimizer. Each training mini-batch has 24 images for input size 320×320 or 12 images for input size 416×416. Learning rate is warm up to 1e-3 for first epoch in order to make training stable initially, and follows cosine scheduler to decrease to 1e-6 for the rest iterations. It relies on the observation that we might not want to decrease the learning rate too drastically in the beginning and moreover, that we might want to "refine" the solution in the end using a very small learning rate. Note that for the restrict of computational resource, we do not rely on pretrian model on imagenet dataset. In other words, we train the network from scratch so as to refine every layer of the whole network without any restrict.

Evaluation metrics We adopt mean average precision (mAP) averaged across all classes to evaluate detection accuracy, while computational cost (FLOPs), and model size (number of parameters) to measure implementing efficiency.

4.2 Evaluation on Pascal VOC 2007

PASCAL VOC dataset consists of natural images drawn from 20 classes. Our RSANet is trained on the union set of VOC 2007 trainval and VOC 2012 trainval, and we report single-model results on VOC 2007 test at the input size of 320×320 and 416×416. The results are exhibited in Table 1.

In Table 1, we have reported the quantitative results on Pascal VOC 2007 [32] test compared with other mainstream methods. Our RSANet yields 72.9% mAP when input size is 320×320 and 75.8% mAP when input size is 416×416. The results show that RSANet achieves the best available trade-off in terms of accuracy and efficiency. With easy data augmentation, our RSANet obtains more excellent results with respect to Pelee [25] and YOLO Nano [30]. And Our RSANet's accuracy is near to YOLOv2 [18]'s, but the model size and computational cost are much smaller than YOLOv2 [18]'s. Regarding to the efficiency, RSANet is nearly 2×, 2×, 2×, 5×, 5× and 2× smaller than Mobilenet-SSD [12], DSOD-small [35], Pelee [25], D-YOLO [36], Tiny-YOLOv2 [18] and Tiny-YOLOv3 [19], respectively. Note that our RSANet surpasses Tiny-YOLOv2 [18] and Tiny-YOLOv3 [19] both in terms of efficiency or accuracy. Although Pelee [25], an anther efficient network, needs only nearly 2× less FLOPs than our RSANet, but delivers poor detection accuracy of 2% drops in terms of mAP. Another interesting result is the comparison with YOLO Nano [30], where it has 3× fewer parameters, while needs more FLOPs and performs 6.7% mAP lower than our RSANet. This is probably because that RSANet has better utilization of computational cost than YOLO Nano [30], yielding more efficient in inference process.

Table 1 Comparison with the recent approaches in terms of object detection accuracy and implementing efficiency on PASCAL VOC2007 test

Mothod	Input size	Params(M)	FLOPs(B)	AP(%)
SSD [20]	300 × 300	26.30	31.75	77.3
YOLOv2 [18]	416 × 416	48.20	34.90	76.8
Mobilenet-SSD [12]	300 × 300	5.5	1.14	68.0
DSOD-small [35]	300 × 300	5.9	5.29	73.6
Pelee [25]	300 × 300	5.43	1.21	70.9
D-YOLO [36]	416 × 416	15	-	67.6
Tiny-YOLOv2 [18]	416 × 416	15.12	6.97	57.1
Tiny-YOLOv3 [19]	416 × 416	8.35	5.52	58.4
YOLO Nano [30]	416 × 416	1	4.57	69.1
Ours	320 × 320	3.243	2.128	72.9
Ours*	416 × 416	3.243	3.536	75.8

"*" means the input size is 416×416 for our method

Figure 4 shows some visual comparisons of detection outputs from different methods on the PASCAL VOC 2007 [32] test dataset. It is evident that our RSANet is not only more robust to small object classification, i.e., aeroplane, but also produces promising results with imcomplete objects, i.e., potted plant. other baselines could not detect above objects because the confidence output with respect to them is small, which is easy filtered out. It is also discovered that our method produces more accurate classification for different objects and regions, such as bicycle in the second example, potted plant in the third example. All the results on this dataset show that our algorithm can capture more accurate context information and fuse multi-level features for more accurate detection.

4.3 Evaluation on MS COCO

MS COCO dataset consists of natural images from 80 object categories. Following common practice, we use trainval35k for training, minival for validation, and report single-model results on test-dev.

(a) (b) (c) (d)

Fig. 4 Some visual comparisons to other baselines on Pascal VOC 2007 test dataset. From left to right are the corresponding detection outputs from Tiny-YOLOv3 [19], MobileNet-SSD [12], Pelee [25], and our RSANet. Different colored bounding boxes represent different categories. The upper left corner of the bounding box refers to the category and its recognition confidence

Table 2 Comparison with the recent approaches in terms of object detection accuracy and implementing efficiency on COCO test-dev

Mothod	Input size	Params(M)	FLOPs(B)	AP(%)	AP50(%)	AP75(%)
SSD [20]	300 × 300	34.30	34.36	25.1	43.1	25.8
YOLOv2 [18]	416 × 416	67.43	17.50	21.6	44.0	19.2
Light-head RCNN [37]	300 × 300	–	5.65	23.7	–	–
MobileNet-SSD [12]	300 × 300	6.8	1.2	18.8	–	–
MobileNet-SSDLite [12]	300 × 300	5.1	1.3	22.2	–	–
MobileNetv2-SSDLite [15]	300 × 300	4.3	0.8	22.1	–	–
Pelee [25]	300 × 300	5.98	1.29	22.4	38.3	22.9
Tiny-YOLOv3 [19]	416 × 416	12.3	–	–	33.1	–
Ours	320 × 320	4.348	2.341	23.7	41.1	23.2
Ours*	416 × 416	4.348	3.927	24.9	42.5	24.7

"*" means the input size is 416×416 for our method

As shown in Table 2, our RSANet achieves Light-head [37] level accuracy with half of the FLOPs. Although there is a small gap between our model and SSD [20], our model size is much smaller than its, and the computational cost our model need is much less than its. In terms of accuracy, our RSANet surpasses YOLOv2 [18] but with much smaller model size and much less computational cost. Our model is smaller than MobileNet-SSD [12], MobileNet-SSDLite [12], MobileNetv2-SSDLite [15], Pelee [25] and Tiny-YOLOv3 [19]. In contrast, our model is stronger than all of them.

Also, we show some visual examples of detection outputs from other different methods on the MS COCO [33] test dataset in Fig. 5. It is demonstrated that, compared with other different methods, our RSANet not only more correctly classifies object with different scales but also produces more accurate regression results for all objects, which is consistent with the quantative results reported in Table 2. For example, the players in the first example, the foods and drinks in the second example, and the tv and the chair in the third example. Moreover, our method is very effective for correctly classifying tiny objects, such as the small sports balls in the distance in the first example, the small cup on the table in the second example, and the small laptop on the desk in the third example, which is omitted by other baselines. The key reason for this situation is that our

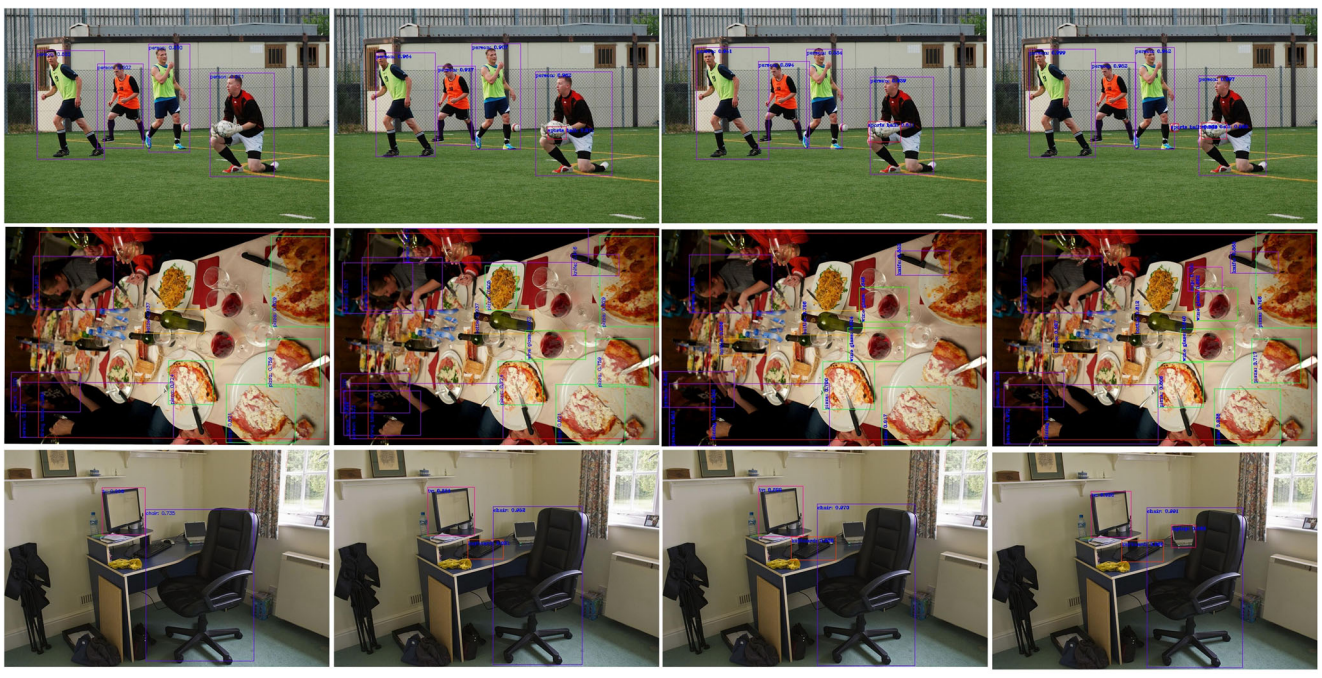

Fig. 5 Some visual comparisons to other baselines on MS COCO test dataset. From left to right are the corresponding detection outputs from Tiny-YOLOv3 [19], MobileNetv2-SSDLite [15], Pelee [25], and our RSANet. Different colored bounding boxes represent different categories. The upper left corner of the bounding box refers to the category and its recognition confidence

Table 4 Validation of RSAM on PASCAL VOC2007 test

Mothod	Input size	Params(M)	FLOPs(B)	mAP(%)
Baseline	320 × 320	3.21	2.08	71.6
Baseline+RSAM	320 × 320	3.243	2.128	72.9
Baseline*	416 × 416	3.21	3.48	74.1
Baseline*+RSAM	416 × 416	3.243	3.536	75.8

"*" means the input size is 416×416 for our method

RSAFPN is more beneficial to classifier the small objects from low-level features, because the RSAFPN is designed to extract the high-level semantics for guiding the low-level features classifications.

4.4 Ablation study

To understand the underlying behavior of our RSANet, this section reports the results of two ablation studies.

4.4.1 Ablation study for CCM design

In this section, we design the following experiment to explore the best setting of the ratio by keeping the floating point of operations (FLOPs) constant. Motivated from ShuffleNetv2 [16], we prove it in whole backbone network LCNet that keeping the number of input and output channels of dot convolution consistent can minimize the amount of memory access and speed up the speed of inference. We use our module CCM to form backbone LCNet and change the ratio of input and output channels of point convolution of pointwise convolution feature map. Meanwhile, we keep the FLOPs constant. And then we test the speed of network inference. The experimental results are shown in the Table 3.

In Table 3, the first column refers to the different channel number settings(in brackets) for each convolutional layer in backbone under a fixed computational costs. The second refers to the inference speed corresponding to different settings, respectively. The first row represents our setting that the number of input and output channels keep consistent. For fair comparison, we keep the floating point of operations (FLOPs) constant and change the setting of the number of channels. The second to fourth rows are baseline architecture using the inverted bottleneck residual setting, where numbers of feature channels are first increased and then reduced to the numbers of input. Conversely, the fifth to seventh rows represent the bottleneck residual setting. Experiment results show that, compared with these settings, our method achieves highest inference speed.

4.4.2 Ablation study for RSAM

In order to verify the effectiveness of RSAM, we do a simple experiment with Pascal VOC 2007. We adopt LCNet and FPN with LCCU to constructing our baseline. And we validate the effectiveness of attention by adding RSAM in RSAFPN. It can be seen in Table 4 that when we use RSAM, the accuracy could be improved by 1.3% and 1.7% compared to our baseline when the input size is 320x320 and 416x416, respectively. Experimental results show that our RSAM could achieve considerable performance improvement with only small additional computational budgets. Therefore, we also use the best settings to evaluate MS COCO.

5 Conclusion

This paper has introduced an architecture that achieves accurate and fast object detection. In contrast to top accurate networks that are computationally expensive with complex and deep architectures, our RSANet focuses more on developing lightweight network backbone and strong multi-scale features fusion head, achieving one trade-off between accuracy and efficiency. The CCM is adopted to redesign the commonly-used residual modules, which is more

Table 3 Inference speed comparison between different channel number ratio with constant computational cost

The number of channel in backbone	FPS
(32,32,32,32),(64,64,64,64),(128,128,128,128), (256,256, 256,256),(512,512, 512,512)	155
(22,44,44,22),(46,92,92,46),(90,180,180,90), (180,360,360,180),(362,724,724,362)	144
(13,78,78,13),(26,152,152,26),(52,312,312,52), (104,624,624,104),(208,1248,1248,208)	132
(9,108,108,9)(18,216,216,18),(36,432,432,36), (74,888,888,74),(148,1776,1776,148)	127
(44,22,22,44),(92,46,46,92),(180,90,90,180), (360,180,180,360),(724,362,362,724)	126
(78,13,13,78),(152,26,26,152),(312,52,52,312), (624,104,104,624),(1248,208,208,1248)	122
(108,9,9,108)(216,18,18,216),(432,36,36,432), (888,74,74,888),(1776,148,148,1776)	105

The first column refers to the different channel number settings(in brackets) for each convolutional layer in backbone under a fixed computational costs. The second refers to the inference speed corresponding to different settings, respectively. The first row represents our setting that the number of input and output channels keep consistent. The second to fourth rows are baseline architecture using the inverted bottleneck residual setting. Conversely, the fifth to seventh rows represent the bottleneck residual setting

efficient while maintaining a similar learning performance. Meanwhile, we use the RSAM to construct RSAFPN for multi-scale features fusion efficiently. The experimental results show that our RSANet achieves comparative available trade-off on PASCAL VOC 2007 and MS COCO dataset in terms of detection accuracy and implementing efficiency. Besides, one may achieve better detection results using the model pre-trained on ImageNet dataset. The future work includes using the method for other computer vision tasks such as real-time semantic segmentation [38–40] and real-time object tracking [41–43].

Acknowledgments National Natural Science Foundation of China (61876093, 61671253), National Natural Science Foundation of Jiangsu Province(BK20181393), and China Scholarship Council(201908320072).

References

1. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
2. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556
3. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1–9
4. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
5. Nakayama Y, Lu H, Li Y, Kamiya T (2020) WideSegNeXt: semantic image segmentation using wide residual network and NeXt dilated unit. IEEE Sensors Journal
6. Lu W, Zhang X, Lu H, Li F (2020) Deep hierarchical encoding model for sentence semantic matching. Journal of Visual Communication and Image Representation, 102794
7. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 580–587
8. Van de Sande KE, Uijlings JR, Gevers T, Smeulders AW (2011) Segmentation as selective search for object recognition. In: 2011 International conference on computer vision. IEEE, pp 1879–1886
9. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 1440–1448
10. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems, pp 91–99
11. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 2961–2969
12. Howard A, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861
13. Zhang X, Zhou X, Lin M, Sun J (2018) Shufflenet: an extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6848–6856
14. Zoph B, Vasudevan V, Shlens J, Le QV (2018) Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8697–8710
15. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC (2018) Mobilenetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4510–4520
16. Ma N, Zhang X, Zheng HT, Sun J (2018) Shufflenet v2: practical guidelines for efficient cnn architecture design. In: Proceedings of the European conference on computer vision (ECCV), pp 116–131
17. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
18. Redmon J, Farhadi A (2017) YOLO9000: better, faster, stronger. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7263–7271
19. Redmon J, Farhadi A (2018) Yolov3: an incremental improvement. arXiv:1804.02767
20. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) Ssd: single shot multibox detector. In: European conference on computer vision. Springer, Cham, pp 21–37
21. Fu CY, Liu W, Ranga A, Tyagi A, Berg AC (2017) Dssd: deconvolutional single shot detector. arXiv:1701.06659
22. Zhang S, Wen L, Bian X, Lei Z, Li S (2018) Single-shot refinement neural network for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4203–4212
23. Howard A, Sandler M, Chu G, Chen LC, Chen B, Tan M, Wang W, Zhu Y, Pang R, Vasudevan V, Le QV, Adam H (2019) Searching for mobilenetv3. In: Proceedings of the IEEE international conference on computer vision, pp 1314–1324
24. Iandola FN, Han S, Moskewicz MW, Ashraf K, Dally WJ, Keutzer K (2016) SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size. arXiv:1602.07360
25. Wang RJ, Li X, Ling CX (2018) Pelee: a real-time object detection system on mobile devices. In: Advances in neural information processing systems, pp 1963–1972
26. Wang F, Jiang M, Qian C, Yang S, Li C, Zhang H, Wang X, Tang X (2017) Residual attention network for image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3156–3164
27. Hu J, Shen L, Sun G (2018) Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7132–7141
28. Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2117–2125
29. Lin M, Chen Q, Yan S (2013) Network in network. arXiv:1312.4400
30. Wong A, Famuori M, Shafiee MJ, Li F, Chwyl B, Chung J (2019) YOLO nano: a highly compact you only look once convolutional neural network for object detection. arXiv:1910.01271
31. Cong D, Zhou Q, Cheng J, Wu X, Zhang S, Ou W, Lu H (2019) CAN: contextual aggregating network for semantic segmentation. In: ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 1892–1896
32. Everingham M, Van Gool L, Williams CK, Winn J, Zisserman A (2007) The PASCAL visual object classes challenge 2007 (VOC2007) results
33. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollar P, Zitnick CL (2014) Common objects in context. In: European conference on computer vision. Springer, Cham, pp 740–755

34. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Deven M, Ghemawat S, Irving G, Isard M, Kudlur M, Levenberg J, Monga R, Moore S, Murray DG, Steiner B, Tucker P, Vasudevan V, Warden P, Wicke M, Yu Y, Zheng X (2016) Tensorflow: a system for large-scale machine learning. In: 12th USENIX symposium on operating systems design and implementation (OSDI 16), pp 265–283
35. Shen Z, Liu Z, Li J, Jiang YG, Chen Y, Xue X (2017) Dsod: learning deeply supervised object detectors from scratch. In: Proceedings of the IEEE international conference on computer vision, pp 1919–1927
36. Mehta R, Ozturk C (2018) Object detection at 200 frames per second. In: Proceedings of the European conference on computer vision (ECCV)
37. Li Z, Peng C, Yu G, Zhang X, Deng Y, Sun J (2017) Light-head r-cnn: in defense of two-stage object detector. arXiv:1711.07264
38. Wang Y, Zhou Q, Liu J, Xiong J, Gao G, Wu X, Latecki LJ (2019) Lednet: a lightweight encoder-decoder network for real-time semantic segmentation. In: 2019 IEEE International conference on image processing (ICIP). IEEE, pp 1860–1864
39. Liu J, Zhou Q, Qiang Y, Kang B, Wu X, Zheng B (IEEE) FDDWNet: a lightweight convolutional neural network for real-time semantic segmentation. In: ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 2373–2377
40. Li S, Zhou Q, Liu J, Wang J, Fan Y, Wu X, Latecki LJ (2020) IEEE 27th Int. conf. on image processing (ICIP) virtual
41. Wang Z, Zheng L, Liu Y, Wang S (2019) Towards real-time multi-object tracking. arXiv:1909.12605
42. Zhan Y, Wang C, Wang X, Zeng W, Liu W (2020) A simple baseline for multi-object tracking. arXiv:2004.01888
43. Lu Z, Rathod V, Votel R, Huang J (2020) RetinaTrack: online single stage joint detection and tracking. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 14668–14678
44. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. IEEE, pp 248–255
45. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks, pp 4700–4708
46. Xie S, Girshick R, Dollár P, Tu Z, He K (2017) Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1492–1500
47. He K, Zhang X, Ren S, Sun J (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans Pattern Anal Mach Intell 37(9):1904–1916
48. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167
49. Bochkovskiy A, Wang CY, Liao HYM (2020) YOLOv4: optimal speed and accuracy of object detection. arXiv:2004.10934
50. Lin TY, Goyal P, Girshick R, He K, Dollár P (2017) Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision, pp 2980–2988
51. Tian Z, Shen C, Chen H, He T (2019) Fcos: fully convolutional one-stage object detection. In: Proceedings of the IEEE international conference on computer vision, pp 9627–9636
52. Zhao Q, Sheng T, Wang Y, Tang Z, Chen Y, Cai L, Ling H (2019) M2det: a single-shot object detector based on multi-level feature pyramid network. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 9259–9266
53. Tan M, Pang R, Le QV (2020) Efficientdet: scalable and efficient object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10781–10790

Affiliations

Quan Zhou[1] · Jie Wang[1] · Jia Liu[1] · Shenghua Li[1] · Weihua Ou[2] · Xin Jin[3]

Jie Wang
jiewangnirvana@163.com

Jia Liu
lj107024@163.com

Shenghua Li
1040308786@qq.com

Weihua Ou
ouweihuahust@gmail.com

Xin Jin
jinxinbesti@foxmail.com

[1] College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing, China

[2] School of Big Data and Computer Science, Guizhou Normal University, Guiyang, China

[3] Department of Computer Science and Technology, Beijing Electronic Science and Technology Institute, Beijing, China