Received March 27, 2020, accepted April 24, 2020, date of publication May 6, 2020, date of current version May 19, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2992532

Auxiliary Detection Head for One-Stage Object Detection

GUOZHENG JIN[1], RIN-ICHIRO TANIGUCHI[2], AND FENGZHONG QU[1], (Senior Member, IEEE)
[1]Key Laboratory of Ocean Observation-Imaging Testbed of Zhejiang Province, Zhejiang University–Zhoushan, Zhoushan 316021, China
[2]Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka 819-0395, Japan

Corresponding author: Fengzhong Qu (jimqufz@zju.edu.cn)

This work was supported in part by the project of China Scholarship Council under Grant 201806320324, in part by the National Natural Science Foundation of China for Excellent Young Scholars under Grant 61722113, in part by the Harbin Engineering University Open Project under Grant SSJSWDZC2018016, and in part by the Joint Fund of National Natural Science Foundation of China and Zhejiang Province under Grant U1809211.

ABSTRACT The auxiliary classifier can improve the performance of classification networks. However, the utility of the auxiliary detection head has not been explored in the object detection field. In this paper, we propose an auxiliary detection head to boost the performance of one-stage object detectors. Similar to other detection heads, the auxiliary detection head consists of a classification subnet and a regression subnet, which are essentially two convolution layers. Thus the auxiliary detection head is computationally efficient. Besides, the auxiliary detection head achieves implicit two-step cascaded regression. Specifically, the auxiliary detection head uses its output boxes as anchors for further regression. Within the auxiliary detection head, refinement of object localization corresponds to adjust the positions of its output boxes towards ground truth boxes, which helps the network learn more robust features. At inference, the auxiliary detection head can be removed without any adverse effect on the performance of the main detector head, which benefits from its independence and leads to two advantages: shrink the model size and shorten inference time. The proposed method is evaluated on Pascal VOC and COCO datasets. By incorporating the auxiliary detection head into a state-of-the-art object detector in parallel with the main detection head, we show consistent improvement over its performance on different benchmarks, whereas no extra parameters are introduced at inference time.

INDEX TERMS Auxiliary detection head, two-step cascaded regression, convolutional neural networks, object detection.

I. INTRODUCTION

Object detection has drawn a great deal of attention recently. It is a combination of classification and localization tasks. A detector can tell people the classes and locations of instances in an image. Object detection has a wide range of applications, including face detection [1], pedestrian detection [2], ship detection [3].

Recently, the deep-learning based object detectors are very popular and develop rapidly over a short period. Compared with the traditional methods, they can extract more robust features with the help of convolution neural networks (CNNs). The current state-of-the-art detectors can be divided into two types: two-stage detectors and one-stage detectors. The former firstly generates some region proposals, then classify and regress those proposals. Region proposals can be generated by an independent method [4] or a neural network embedded in the detector [5]. One-stage detectors eliminate proposal generation step and detect instances in a unified network. Thus, one-stage detectors have a great advantage in terms of computational efficiency. On the other hand, real-time processing is significant in some applications, which is closely linked with costs and efficiency. Therefore, this paper mainly explores one-stage detectors.

Though one-stage detectors can run at a relatively high speed, their performance tails two-stage detectors generally. After all, two-stage approaches can implement bounding box regression twice. Some methods have been proposed to improve the performance of one-stage detectors. [6]–[8] aggregated contextual information among different feature

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Liu.

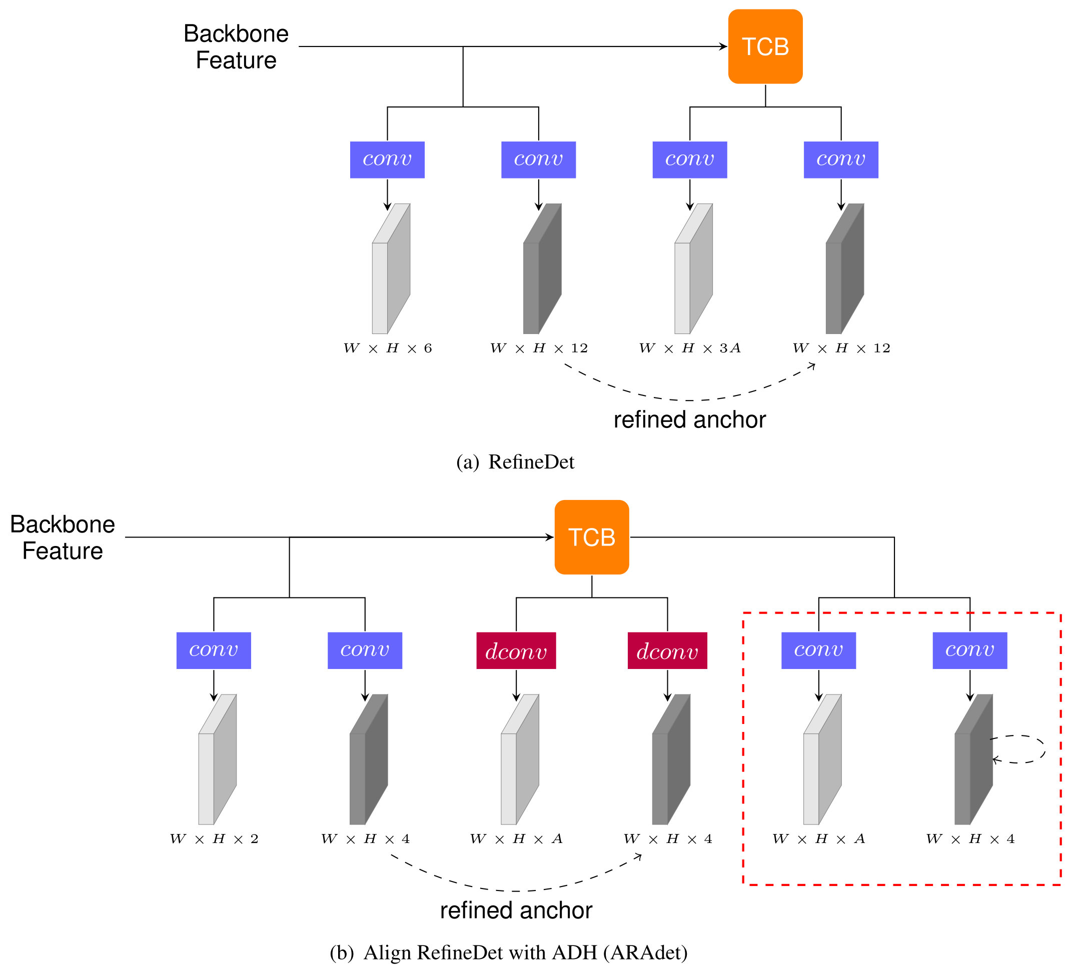

FIGURE 1. **The architectures of RefineDet and Align RefineDet with ADH (ARAdet). ARAdet contains three detection stages and each stage has two layers for classification and regression respectively. TCB is transfer connection block for feature fusion. *conv* and *dconv* are convolution and deformable convolution respectively. *A* is the number of classes. The first *conv* layer is for binary classification. The dashed red rectangle is ADH. One of the four detection branches is shown for clarity.**

maps to generate accurate results. [9], [10] focused on training a one-stage object detector from scratch. [11], [12] proposed new loss functions for bounding box regression to improve the performance of object detection algorithms. One challenging problem for object detection is rough localization. To locate objects accurately, two-step cascaded regression was proposed in [13]. The output boxes of one detection head are transferred to another detection head for further refinement. To the best of our knowledge, there still lacks studies on the auxiliary detection head to enhance one-stage detectors.

This paper aims to boost the performance of one-stage detectors from a new prospect: auxiliary detection head. The utility of auxiliary classifier has been discussed in [14], [15]. It has been proved that auxiliary classifiers can help the network to reach a slightly higher plateau. In this paper, a simple yet effective module named auxiliary detection head (ADH) is proposed to improve the performance of one-stage detectors. Similar to other prediction heads in a detector, ADH consists of a classification subnet and a regression subnet, which are essentially two convolution layers. Thus the structure of ADH is simple and ADH is computationally efficient. Besides, ADH can achieve implicit two-step cascaded regression in one detection head. As mentioned above, the two-step cascaded regression in [13] relies on two branches. The anchors of one branch are initialized by the output boxes of another branch. But ADH is an independent module and uses its output boxes to initialize anchors for further refinement. Within the auxiliary detection head, refinement of object localization corresponds to adjust the positions of its output boxes towards ground truth boxes, which aids in feature extraction. Thus ADH is effective. The idea is depicted in the red dashed rectangle in Fig. 1 (b).

ADH is easy to use in one-stage detectors. As an auxiliary module, ADH can be plugged into the state-of-the-art object detection frameworks in parallel with the existing prediction branch. When ADH is added into a detector, there are two groups of detection heads: the original detection head and ADH. To distinguish them, the original detection head is named the main detection head, which is used for final prediction. During training, ADH is jointly optimized with the main detection head and the parameters increase a few. However, as an auxiliary module, ADH can be omitted after training because its output does not affect the main detection head. This benefits from the independence of ADH. Thus, the model size will not increase after the application of ADH during inference, which is profitable. Experiments on Pascal VOC and COCO datasets will demonstrate the effectiveness of ADH. After plugging into a one-stage detector, ADH can consistently improve its performance by non-negligible margins whereas no extra parameters are introduced at inference time. For example, the proposed method surpasses its baseline RefineDet by 1.8% and 1.1% AP with VGG-16 and ResNet-101 backbones respectively on COCO. The code will be made publicly available. The main contributions of this paper are summarized as follows.

1) ADH is proposed to achieve implicit two-step cascade regression in one detection head.
2) ADH can be plugged into the state-of-the-art object detection frameworks in parallel with the existing prediction branch and trained jointly with the original detection head.
3) ADH can consistently improve the baseline by non-negligible margins whereas introducing no extra parameters at inference.

The remainder of this paper is organized as follows: In Section II, related works are discussed. The proposed method is introduced in detail in Section III. Experimental results and comparisons are presented in Section IV. Section V draws conclusions.

II. RELATED WORK

We present related works in three aspects: two-stage detectors, one-stage detectors, and auxiliary module.

A. TWO-STAGE DETECTORS

For two-stage detectors, a region proposal generation module is needed. [4] used selective search [16] to produce candidate object proposals. Afterward, [5] introduced a region proposal network (RPN) that shared features with a detection network to produce region proposals. [17] combined Faster R-CNN [5] with several data augmentation methods for marine organisms detection. [18] proposed position-sensitive score maps to address the dilemma between translation-invariance in classification and translation-variance in localization. [19] improved R-FCN by applying deformable convolution and more anchors for seafood detection and then a remote operated vehicle (ROV) was used for capture. [20] added a branch in Faster R-CNN [5] for predicting an object mask in parallel with the existing prediction branch. [21] presented a deep regionlet approach for object detection, which could select non-rectangular regions within a bounding box. To explore the local and global features, [22] proposed CoupleNet to couple the global structure with local parts for object detection. [23] presented Scale Aware Network (SAN) that mapped features from the different scales onto a scale-invariant subspace to make the detector more robust to the scale variation. [24], [25] aimed to explore more contextual information. [26] proposed a novel bounding box regression loss named KL Loss to learn bounding box transformation and variances of coordinates. [27] introduced Guided Anchoring scheme which generated non-uniform anchors of arbitrary shapes. [28]–[31] designed new network structures to extract more robust features.

B. ONE-STAGE DETECTORS

One-stage detectors do not need a step to generate region proposals. [32] combined predictions from six feature maps with different resolutions to detect objects of various sizes. Later, [33] introduced additional context by adding deconvolution layers to improve SSD, especially for small object detection. [34] proposed reverse connection block to combine fine-grained details with highly-abstracted information and objectness prior to reduce the searching space of objects. [11] designed focal loss to address the imbalance between foreground and background classes by assigning different weights to different examples. [35] imitated the structure of Receptive Fields (RFs) in human visual systems and proposed RFB net to enhance the feature discriminability and robustness. [12] introduced generalized intersection over union (GIoU) as a new loss for bounding box regression to improve the performance of object detection algorithms. [36] solved object detection in a per-pixel prediction fashion without the need for an anchor. Current state-of-the-art object objectors are fine-tuned from pretrained classification networks, [9], [10] focused on training a one-stage object detector from scratch. [6]–[8], [37] aggregated contextual information among different feature maps to generate robust features. To inherit the merits of one-stage and two-stage detectors, [13] designed a single-shot neural network detector, called RefineDet. It contains two important modules, anchor refinement module (ARM) and object detection module (ODM). ARM imitates RPN to coarsely adjust anchors and implement binary classification. The output of ARM is used to provide a better initialization for ODM to further refine boxes and predict multi-class labels. Later, this coarse-to-fine manner is also explored in [38]. [39], [40] tried to solve the misalignment problem between refined boxes and features.

C. AUXILIARY MODULE

In [14], [15], the application of the auxiliary classifier was explored to improve the performance of classification networks. PC-TDM [41] used two subnets for group activity recognition. [42] introduced a set of auxiliary tasks to boost the accuracy of two-stage detectors. It is worth noting that the auxiliary tasks in [42] are different from our ADH. First of all, the auxiliary tasks in [42] are applied for two-stage detectors. Besides, the auxiliary branches can not be removed after training, because their outputs are used by the main task branch. Whereas ADH aims to improve one-stage detectors and can be excluded after training. To the best of our knowledge, there has been no previous work about designing an auxiliary module to boost the performance of one-stage detectors.

III. METHOD

A. MOTIVATION

ADH is inspired by the previous auxiliary classifier and the two-step cascaded regression. [14], [15] has proved that auxiliary classifiers can boost the performance of classification networks. But there is no work about the application of auxiliary detection head in one-stage detectors. Thus we try to design ADH.

As an auxiliary module, ADH works as an independent module to improve one-stage detectors. During training, ADH is optimized jointly with the main detection head. After training, ADH can be omitted. On the other hand, ADH achieves implicit two-step cascaded regression within itself. ADH does not rely on the output of other prediction branches but uses its output boxes as anchors. This novel design can assist the network to learn more robust features and will be demonstrated by experiments. ADH and the main detection head share the feature extraction layers. Therefore the main detection head can benefit from those robust features. The idea, as shown in Fig. 1 (b), suggests that ADH uses its output boxes to initialize anchors and refine these boxes again. It is worth noting that ADH is different from ODM in [13]. In [13], coarse boxes from ARM are used to initialize anchors in ODM for further refinement as shown in Fig. 1 (a). Consequently, ODM relies on the output boxes of ARM. However, ADH is an independent module and applies its output boxes as anchors. Besides, transfer connection blocks (TCBs) are adopted to link ARM and ODM, whereas ADH is agnostic to the structure and needs no other connection module. ADH is plugged into the one-stage detection framework in parallel with the main detection head. We can regard ADH as an enhanced module and train it jointly with the main detection head.

B. ARCHITECTURE

ADH is plugged into RefineDet [13]. Because RefineDet proposed the two-step cascaded regression, whereas ADH introduces the implicit two-step cascaded regression. We will demonstrate ADH can boost RefineDet though it applies two-step cascaded regression. In addition, RefineDet has misalignment problem between refined boxes and features. Inspired by [40], we use deformable convolution [43] for feature alignment. Without loss of generality, ADH can be also used in other detectors, e.g., SSD [32]. The network architectures of RefineDet and Align RefineDet with ADH (ARAdet) are depicted in Fig. 1.

There are four prediction branches in ARAdet. One branch contains six layers. The first two convolution layers form ARM, which performs binary classification and provides refined anchors to the main detection head. ARM is the same as in RefineDet. The main detection head is composed of the middle two deformable convolution layers. Deformable convolution [43] is adopted for feature alignment. Note that deformable convolution is only used in the main detection head. For simplicity, the difference between anchors and refined anchors are applied as offsets for deformable convolution. For example, Fig. 2 depicts a refined anchor and an anchor. As 3×3 deformable convolution is adopted in this paper, an offset map with 18 channels is needed. The generation process of the offset map is simple and needs no extra layers. Specifically, calculate the coordinates of nine points on the anchor and its corresponding refined anchor respectively through bilinear interpolation, then do a minus operation for those two sets of coordinates of points accordingly and get the final offset map. Note that the anchor and its corresponding refined anchor have the same center coordinate. Besides, one anchor is equipped with our method. Because 3×3 deformable convolution can only receive one offset map with 18 channels. If more anchors are applied, there will be multiple offset maps and more deformable convolution layers are required, which is inefficient. Using more deformable convolution layers will slow down the running speed. The final detection results are produced by the main detection head. ADH is formed by the last two convolution layers. One is for classification and the other is for regression. We attach ADH at the end of the network in parallel with the main detection head. All detection heads are optimized jointly during training. After training, ADH is removed because its output does not affect the main detection head. This benefits from the independence of ADH. As ADH employs implicit two-step cascaded regression, it has a novel loss function, which will be described in the next session.

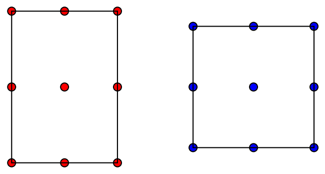

FIGURE 2. A refined anchor (left) and an anchor(right).

C. LOSS FUNCTION

The whole loss function of ARAdet includes two parts: loss of ADH and loss of RefineDet. The loss function of RefineDet remains unchanged. Thus we mainly introduce the loss function of ADH. Similar to the detection head in SSD [32], ADH consists of a classification subnet and a regression subnet. The objective loss is a combination of localization loss (loc) and confidence loss (conf). The loss function of ADH is defined as follows.

$$L\left(p, p^{*}, g, g^{*}\right)=\frac{1}{N}\left(L_{conf}\left(p, p^{*}\right)+\alpha L_{loc}\left(g, g^{*}\right)\right) \quad (1)$$

where N is the number of matched boxes. Softmax is used for confidence loss L_{conf} in this paper. The softmax results show probability estimates over all object classes and a background class. p and p^{*} are predicted probability and true label of matched boxes, respectively. α is a weight to balance two losses and set to 1 in all experiments. The localization loss is computed by Smooth L1 [44]. Different from SSD [32], ADH applies its output boxes as anchors and regresses to offsets for the center, width, and height of its output boxes towards ground truth boxes. g and g^* are the offsets of a predicted box and a ground-truth bounding box, respectively. g is the output of the regression subnet in ADH. To get g^*, g need to be decoded by the following (2).

$$\begin{aligned} x &= g_x * w_a + x_a \\ y &= g_y * h_a + y_a \\ w &= w_a * exp(g_w) \\ h &= h_a * exp(g_h) \end{aligned} \tag{2}$$

where x_a, y_a, w_a, h_a are the center coordinates (x,y) of an anchor and its width (w) and height (h), respectively. g_x, g_y, g_w, g_h are predicted offsets for center coordinates (x,y) and width (w) and height (h), respectively. Equation (2) is the decode equation and used to get real size boxes through the corresponding anchor and offsets. After obtaining the output boxes of ADH, they are employed as anchors. Then (3) is applied to encode ground truths by those anchors.

$$\begin{aligned} g_x^* &= \frac{x^* - x}{w} \\ g_y^* &= \frac{y^* - y}{h} \\ g_w^* &= log\left(\frac{w^*}{w}\right) \\ g_h^* &= log\left(\frac{h^*}{h}\right) \end{aligned} \tag{3}$$

where x, y, w, h are the center coordinates (x,y) of an output box by ADH and its width (w) and height (h), respectively. It is generated by (2). Equation (3) is applied to encode a ground-truth bounding box by the result of (2). Note that SSD uses anchors to encode ground truth bounding boxes, whereas ADH uses its output boxes to encode ground truth bounding boxes. Fig. 3 depicts different ways to generate g^* in SSD and ADH. Compared with SSD, ADH applies both the encode and decode process. With ADH, refinement of object localization corresponds to adjust the positions of its output boxes towards ground truth boxes.

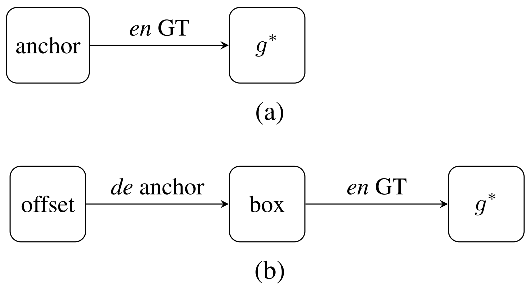

FIGURE 3. **Different ways to encode a ground truth box in (a) SSD and (b) ADH. *en* and *de* refer to encode and decode respectively. GT is a ground truth box. offset is the output of ADH.**

When ADH is plugged into an object detector and jointly optimized with the original detection head, we need to combine the original loss function and the loss function of ADH as follows.

$$L_{sum} = L_{ori} + L\left(p, p^*, g, g^*\right) \tag{4}$$

where L_{ori} denotes the original loss function of a detector, e.g. RefineDet [13].

D. INFERENCE

As an independent auxiliary module, the output of ADH has no use for the detection result. Thus ADH is removed to shrink module size during inference. The main detection head is responsible to generate the detection result. Predictions from four feature maps are merged. We then apply non-maximum suppression (NMS) with a threshold of 0.45 and keep the top 200 detections per image, yielding the final detection result.

IV. EXPERIMENTS

A. DATASETS

The proposed method is evaluated on PASCAL VOC and COCO datasets. PASCAL VOC is a popular dataset which contains 20 classes [45]. For VOC 2007, models are trained on union of VOC 2007 `trainval` set and VOC 2012 `trainval` set, then tested on VOC 2007 `test` set. COCO is a bigger dataset for object detection. For COCO 2017, it contains ∼118k images for training, 5k for validation (`val`) and ∼20k for testing without annotations (`test-dev`). The proposed method is trained on the training set and reported the performance on `test-dev` set. The detection result of COCO is submitted to the public testing server for evaluation.

B. IMPLEMENTATION DETAILS

VGG-16 [46] and ResNet-101 [47] are used as the backbone networks in our experiments, which are pretrained on the ImageNet [48]. The settings are the same as RefineDet [13]. The last two fully connected layers of fc6 and fc7 in VGG-16 are replaced by two convolution layers. Then two extra convolution layers are added at the end of VGG-16. In addition, conv4_3 and conv5_3 are scaled by L2 normalization [49]. For ResNet-101, the last fully connected layer is removed and one extra residual block is added to the end of the truncated Net. Four feature maps with strides [8,16,32,64] are applied for detection. Besides, TCB in RefineDet is also used to fuse features from multiple branches.

During training, we match ground truth boxes with anchors based on jaccard overlap. In some paper, anchors are also called default boxes. The Matching process is the same as in SSD [32]. Specifically, each ground truth is matched to an anchor with the maximum overlap score, then anchors that have a jaccard overlap higher than the threshold of 0.5 with any ground truth are also set to positives, and others are negatives. After matching, a majority of anchors are negatives. To mitigate this imbalance, hard negative mining in [32] is applied and negatives are selected according to the classification loss value with a ratio of 1:3 between the positives and negatives. The same matching strategy is applied for both the main detection head and ADH.

Average precision (AP) [50] is used as the metric to evaluate the performance of detectors. AP is computed across the intersection of union (IoU) thresholds from 0.5 to 0.95 with an interval of 0.05. This metric tends to measure the detection results through various qualities and reflects the result more comprehensively than using one threshold. For Pascal VOC 2007, mAP is adopted as metric and mAP is equal to AP 50.

The resolution of the training image is resized to 320×320. Training and inference are performed on the same image scale. As data augmentation is crucial for one-stage detectors, the same data augmentation method in [32] is applied. For Pascal VOC, models are trained for 240 epochs with an initial learning rate of 10^{-3}, which is divided by 10 at 160 and again at 200 epochs with a batch size of 32. For COCO 2017, detectors are trained on the training set and reported the performance on `test-dev` set. The result is submitted to a server for evaluation. Models are trained for 130 epochs with an initial learning rate of 10^{-3}, which is divided by 10 at 80 and again at 110 epochs with a batch size of 32. In addition, the warmup strategy is applied to linearly increase the learning rate from 10^{-6} to 10^{-3} at the first 5 epochs. Weight decay and momentum are set to 0.0005 and 0.9 respectively. All models are trained end-to-end with stochastic gradient descent (SGD) on a Titan X GPU. To get a faster detection speed, ADHs are omitted during the inference phase.

C. ABLATION STUDY

Ablation studies are conducted on Pascal VOC 2007. The `trainval` set are used for training then models are tested on VOC 2007 `test` set. For fair comparisons, all methods are trained for 240 epochs with an initial learning rate of 10^{-3}, which is divided by 10 at 160 and again at 200 epochs with a batch size of 32.

To demonstrate the versatility of ADH, we incorporate it into three one-stage detectors, SSD [32], RefineDet [13], and align RefineDet, in parallel with the original detection head. The same backbone network VGG-16 is used. As the feature fusion module can boost performance, TCBs are adopted for the three baselines. RefineDet has a misalignment problem between refined boxes and features. We use deformable convolution for feature alignment and call it Align RefineDet. The feature alignment method has been described in Section III-B. Fig. 4 depicts the simplified architectures of SSD and SSD with ADH. The only difference between them is whether the model has ADH. Fig. 5 presents the architectures of RefineDet and RefineDet with ADH. RefineDet with ADH is obtained by adding ADH to RefineDet. After plugging ADH into Align RefineDet, it is ARAdet. The main detection head and ADH are optimized simultaneously during training. AP is used as a metric

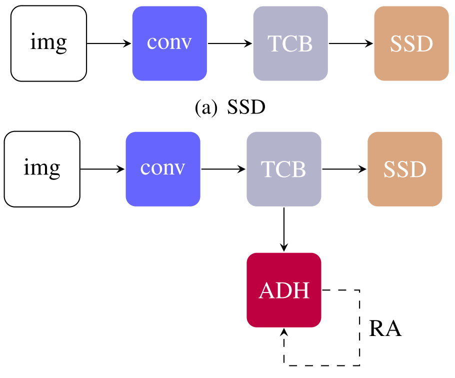

FIGURE 4. The architectures of (a) SSD and (b) SSD with ADH. "img" and "conv" refer to input image and backbone convolutions respectively. "TCB" and "SSD" are transfer connection block and detection head in SSD, respectively. "ADH" and "RA" are auxiliary detection head and refined anchors.

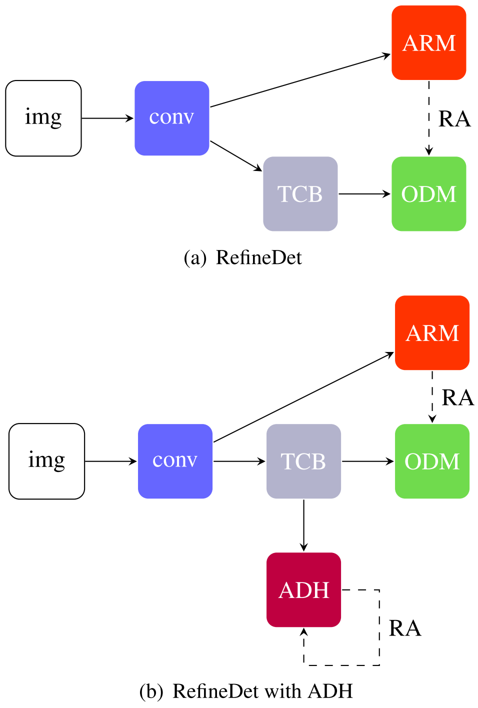

FIGURE 5. The architectures of (a) RefineDet and (b) RefineDet with ADH. "img" and "conv" refer to input image and backbone convolutions respectively. "TCB", "ARM" and "ODM" are transfer connection block, anchor refinement module and object detection module (ODM), respectively. "ADH" and "RA" are auxiliary detection head and refined anchors.

because it reflects performance with different threshold. All models are reimplemented with Pytorch [51] on the same platform for fair comparisons.

The results are presented in Table 3. With the help of ADH, the performance of all three baselines is improved, which firmly demonstrates the effectiveness of ADH. For the three baselines, Align RefineDet achieves the best result. RefinDet outperforms SSD due to the two-step cascaded regression. The reason is that RefineDet has two regression operations to achieve accurate localization. After using the

TABLE 1. Comparison with other methods on PASCAL VOC 2007.

Method	Backbone	Model size	Input size	Boxes	FPS	mAP (%)
RefineDet [13]	VGG-16	-	320×320	6375	40.3	80.0
APLNet [38]	VGG-16	-	320×320	6375	41.2	81.2
PASSD [40]	VGG-16	-	320×320	6375	50	81.0
RefineDet	VGG-16	138M	320×320	6375	50.0	79.9
Align RefineDet	VGG-16	135M	320×320	2125	47.1	79.9
ARAdet	VGG-16	135M	320×320	2125	47.1	80.4

TABLE 2. Detection results on COCO test-dev.

Method	Backbone	Input Size	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RefineDet [13]	VGG-16	320×320	38.7	29.4	49.2	31.3	10.0	32.0	44.4
APLNet [38]	VGG-16	320×320	-	30.6	51.3	32.3	10.6	35.1	45.9
PASSD [40]	VGG-16	320×320	40.0	31.4	51.6	33.6	12.0	35.1	45.8
Align RefineDet	VGG-16	320×320	40.0	30.3	50.6	32.0	11.6	33.2	43.1
ARAdet	VGG-16	320×320	40.0	31.2	52.0	32.9	11.8	34.2	44.8
RefineDet [13]	ResNet-101	320×320	-	32.0	51.4	34.2	10.5	34.7	50.4
PASSD [40]	ResNet-101	320×320	34.5	32.7	52.1	35.3	10.8	36.5	50.2
ARAdet	ResNet-101	320×320	35.9	33.1	53.5	35.1	11.5	35.7	50.4

TABLE 3. Effectiveness of ADH.

	AP	AP_{50}	AP_{60}	AP_{70}	AP_{80}	AP_{90}
SSD	40.6	69.6	62.5	49.4	30.9	8.3
SSD ADH	42.3	71.1	64.1	51.7	33.3	9.0
RifineDet	41.7	72.1	65.4	51.8	30.5	7.3
RifineDet ADH	42.4	73.0	66.4	53.2	31.5	6.7
Align RifineDet	43.7	73.0	66.7	54.8	33.7	9.0
ARAdet	45.2	73.5	67.3	56.2	37.1	10.8

deformable convolution for feature alignment in RefineDet, Align RefineDet surpasses RefineDet by 2.0% AP.

ADH promotes SSD by 1.7% AP. For RefineDet, the improvement became less and ADH brings a 0.7% increase of AP. For the strongest baseline aligned RefineDet, ADH improves it by 1.5% AP. We also observe that the improvement for a high threshold metric is more significant than that for the low threshold metric in Align RefineDet. For example, AP 80 and AP 90 are gained by 3.4% and 1.8%, respectively. We emphasize that those improvements are only introduced by ADH. The explanation is that ADH uses its output boxes as anchors for further refinement leading the network to learn more effective features. The main detection head shares feature extraction layers with ADH. Thus they can benefit from those effective features. ADH is widely applicable across different detectors, achieving consistent gains independently of the baseline detector strength. Among all methods, ARAdet gets the best result. Later, it will be tested on other datasets.

D. RESULTS ON PASCAL VOC 2007

In this section, ARAdet is compared with other methods on the union of VOC 2007 `trainval` set and VOC 2012 `trainval` set. As ARAdet is built upon Align RefineDet and applies two-step cascaded regression, it is mainly compared with other methods that also use two-step cascaded regression. APLNet [38] used an attention enhancement module (AEM) to generate more semantically meaningful information. PASSD [40] proposed offseted convolution to align the receptive field of the convolution filters.

The results are presented in Table 1 and mAP is used as the metric. Our reimplemented RefineDet gets a 79.9% mAP compared with the published result 80.0% in [13]. It seems that feature alignment does not affect RefineDet, because the mAP of Align RefineDet is also 79.9%. ARAdet exceeds Align RefineDet by 0.5% mAP as a result of ADH. APLNet achieves the best result with 81.2% mAP.

We argue that ADH and feature alignment mainly boost performance on the high threshold metric. As mentioned above, mAP only considers a threshold of 0.5. Compared with mAP, AP is more comprehensive than mAP and can reflect the ability of detectors on various qualities. Table 4 shows the results of AP. The baseline RefineDet gets a 50.5% AP. After applying deformable convolution for feature alignment, it is improved by 1.3% AP. ADH can boost performance further. ARAdet outperforms Align RefineDet by 0.9% AP. Thus ADH and feature alignment perform better on the high threshold metric and are capable of accurate localization. Besides, Table 1 also presents the model size. After training, ADH is removed, thus ARAdet has the same size as Align

TABLE 4. Results on Pascal VOC 2007 with AP metric.

	AP	AP_{50}	AP_{60}	AP_{70}	AP_{80}	AP_{90}
RifineDet	50.5	79.9	75.3	63.7	42.1	12.0
Align RifineDet	51.8	79.9	75.0	64.4	44.9	15.8
ARAdet	52.7	80.4	75.2	65.5	46.3	16.1

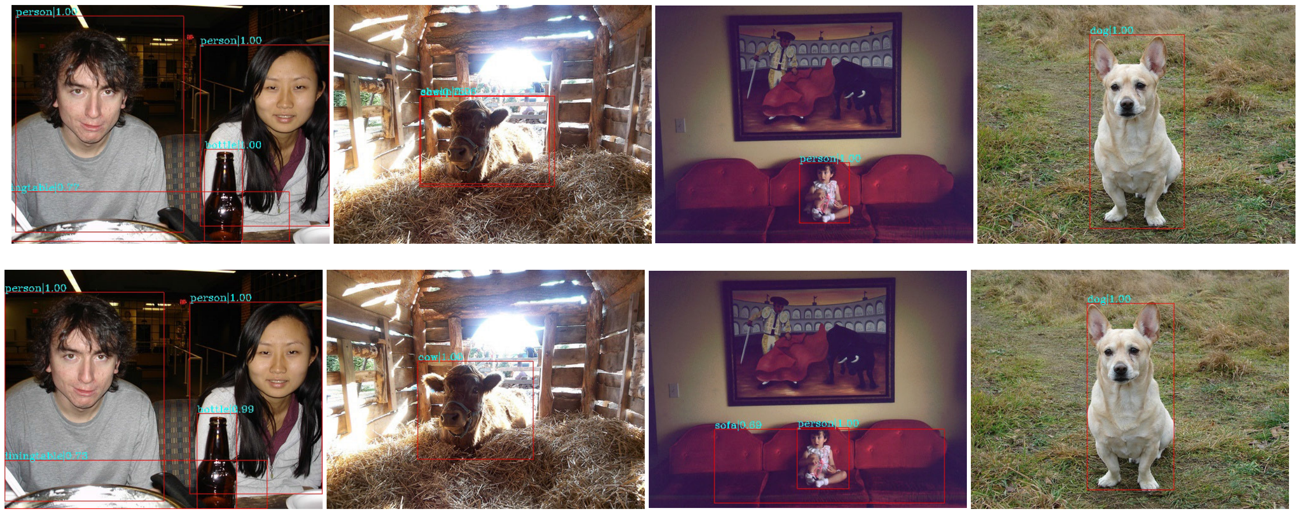

FIGURE 6. Some qualitative detection results of RefineDet (first row) and ARAdet (second row) on the Pascal VOC 2007 `test` set. Detected instances with confidence scores higher than 0.5 are shown.

FIGURE 7. Some qualitative detection results on the COCO `test` set. Detected instances with confidence scores higher than 0.5 are shown.

RefineDet. ARAdet is also smaller than RefineDet because it applies one anchor per position over feature maps. Some qualitative detection results of RefineDet and ARAdet on the Pascal VOC 2007 `test` set are depicted in Fig. 6. It is observed that ARAdet can detect instances by a smaller box than RefineDet in some cases. Thus ARAdet gives a better localization.

E. RESULTS ON COCO

In this section, ARAdet is compared with other methods on COCO. Note that there are lots of methods that improve detectors form different aspects [11], [12], [31], [35]. But their contributions are orthogonal to ours. As ARAdet applies two-step cascaded regression, we compare it with models that also use two-step cascaded regression.

The results are shown in Table 2. Align RefineDet outperforms baseline RefineDet by 0.9% AP. Thus the use of deformable convolution for feature alignment is effective. Since ARAdet is generated by plugging ADH into Align RefineDet, the comparison with it can demonstrate the effectiveness of ADH convincingly. ARAdet surpasses Align RefineDet by 0.9% AP.

Compared with other methods, we also observe that Align RefineDet is not on par with PASSD (30.3% AP vs. 31.4% AP). One reason is that PASSD proposed offseted convolution layer to align the receptive field of the convolution filters, which is more powerful than the deformable convolution in Align RefineDet. We use ADH to enhance align RefineDet and ARAdet is comparable with PASSD. APLNet [38] employs an attention enhancement module (AEM) to generate more semantically meaningful information. But ARAdet still surpasses APLNet and gains 0.6% AP. Though PASSD gets a slightly better result than ARAdet with VGG backbone, ARAdet outperforms PASSD by 0.4% AP with Resnet backbone. Hence ADH also works excellently under a strong backbone. ADH can consistently improve the detection accuracy no matter what kind of backbones are used.

ARAdet with ResNet-101 runs at nearly 36 frames per second (FPS) with an input size of 320×320 in inference, which fulfills real-time processing. The inference time is evaluated with a batch size of 1 on an NVIDIA Titan X GPU, CUDA 10.0, and cuDNN v7.6. It is hard to perform fair comparisons due to inconsistent environments. Some qualitative detection results on the COCO `test` set are depicted in Fig. 7. We believe ADH can promote other one-stage detectors as well.

F. RESULT ANALYSIS

Experiments have demonstrated that ARAdet outperforms Align RefineDet on different datasets. The only difference between them is ADH. Thus ADH can boost the performance of the baseline. We explain that ADH can help the network learn more robust features. In SSD [32], only dozens of anchors are positives after the matching process, because anchors are fixed. But ADH uses its output boxes as anchors to match ground truths. Those output boxes have a high overlap with ground truths resulting in more positives and a better matching result. More positives can mitigate sample imbalance. On the other hand, the matching result is related to the loss function directly, then the update of parameters is affected. ADH and the main detection head share the same convolutional features from the backbone and feature fusion module. Thus those robust features are also helpful to the main detection head. There is no direct relation between ADH and the main detection head. Because ADH is an independent module and can be removed after training. The main detection head is influenced by ADH through updating parameters in the feature extraction layers.

V. CONCLUSION

In this paper, ADH is proposed to improve the performance of current one-stage detectors. It is a simple module that only contains two convolution layers for classification and regression respectively. In addition, ADH achieves implicit two-step cascaded regression. Compared with two-step cascaded regression, ADH uses its output boxes as anchors. Within ADH, refinement of object localization corresponds to adjust the positions of its output boxes towards ground truth boxes, which helps feature extraction. ADH can be regarded as an enhanced module which builds upon the state-of-the-art object detection frameworks. We can plug ADH into a single-shot detector in parallel with the original detection head. In the training stage, ADH is jointly optimized with the main detection head. At the time of inference, ADH is removed without any adverse effect on the performance of the main detection head. Thus ADH plays a role as an auxiliary module. This benefits from the independence of ADH and has two advantages: shrink the model size and shorten inference time. Experiments are carried out on Pascal VOC and COCO datasets to demonstrate the effectiveness of ADH. By incorporating ADH into a one-stage detector, we show consistent improvement to its performance without introducing any parameters at inference time.

In the future, we plan to introduce ADH into two-stage detectors. We believe that ADH is beneficial to the research of object detection.

REFERENCES

[1] Y. Bai, Y. Zhang, M. Ding, and B. Ghanem, "Finding tiny faces in the wild with generative adversarial network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 21–30.

[2] G. Brazil and X. Liu, "Pedestrian detection with autoregressive network phases," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 7231–7240.

[3] Z. Shao, W. Wu, Z. Wang, W. Du, and C. Li, "SeaShips: A large-scale precisely annotated dataset for ship detection," *IEEE Trans. Multimedia*, vol. 20, no. 10, pp. 2593–2604, Oct. 2018.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural nf. Process. Syst.*, 2015, pp. 91–99.

[6] J. Ren, X. Chen, J. Liu, W. Sun, J. Pang, Q. Yan, Y.-W. Tai, and L. Xu, "Accurate single stage detector using recurrent rolling convolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5420–5428.

[7] S.-W. Kim, H.-K. Kook, J.-Y. Sun, M.-C. Kang, and S.-J. Ko, "Parallel feature pyramid network for object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 234–250.

[8] T. Kong, F. Sun, C. Tan, H. Liu, and W. Huang, "Deep feature pyramid reconfiguration for object detection," in *The Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 169–185.

[9] Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, and X. Xue, "DSOD: Learning deeply supervised object detectors from scratch," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1919–1927.

[10] R. Zhu, S. Zhang, X. Wang, L. Wen, H. Shi, L. Bo, and T. Mei, "ScratchDet: Training single-shot object detectors from scratch," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2268–2277.

[11] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *The IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[12] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *The IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 658–666.

[13] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4203–4212.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[16] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Sep. 2013.

[17] H. Huang, H. Zhou, X. Yang, L. Zhang, L. Qi, and A.-Y. Zang, "Faster R-CNN for marine organisms detection and recognition using data augmentation," *Neurocomputing*, vol. 337, pp. 372–384, Apr. 2019.

[18] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 379–387.

[19] L. Ji-yong, Z. Hao, H. Hai, Y. Xu, W. Zhaoliang, and W. Lei, "Design and vision based autonomous capture of sea organism with absorptive type remotely operated vehicle," *IEEE Access*, vol. 6, pp. 73871–73884, 2018.

[20] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2961–2969.

[21] H. Xu, X. Lv, X. Wang, Z. Ren, N. Bodla, and R. Chellappa, "Deep regionlets for object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 798–814.
[22] Y. Zhu, C. Zhao, J. Wang, X. Zhao, Y. Wu, and H. Lu, "CoupleNet: Coupling global structure with local parts for object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4126–4134.
[23] Y. Kim, B.-N. Kang, and D. Kim, "San: Learning relationship between convolutional features for multi-scale object detection," in *The Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 316–331.
[24] Z. Chen, S. Huang, and D. Tao, "Context refinement for object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 71–86.
[25] X. Chen and A. Gupta, "Spatial memory for context reasoning in object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4086–4096.
[26] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding box regression with uncertainty for accurate object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2888–2897.
[27] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, "Region proposal by guided anchoring," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2965–2974.
[28] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.
[29] T. Kong, A. Yao, Y. Chen, and F. Sun, "HyperNet: Towards accurate region proposal generation and joint object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016.
[30] X. Zeng, W. Ouyang, J. Yan, H. Li, T. Xiao, K. Wang, Y. Liu, Y. Zhou, B. Yang, Z. Wang, H. Zhou, and X. Wang, "Crafting GBD-net for object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 9, pp. 2109–2123, Sep. 2018.
[31] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.
[32] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 21–37.
[33] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD : Deconvolutional single shot detector," 2017, *arXiv:1701.06659*. [Online]. Available: http://arxiv.org/abs/1701.06659
[34] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, "Ron: Reverse connection with objectness prior networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5936–5944.
[35] S. Liu, D. Huang, and A. Wang, "Receptive field block net for accurate and fast object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 385–400.
[36] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9627–9636.
[37] Z. Guo, W. Zhang, Z. Liang, Y. Shi, and Q. Huang, "Multi-scale object detection using feature fusion recalibration network," *IEEE Access*, vol. 8, pp. 51664–51673, 2020.
[38] H. Zhang, D. Kang, H. He, and F.-Y. Wang, "Aplnet: Attention-enhanced progressive learning network," *Neurocomputing*, vol. 371, pp. 166–176, Oct. 2020.
[39] X. Chen, X. Yang, S. Kong, Z. Wu, and J. Yu, "Dual refinement network for single-shot object detection," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 8305–8310.
[40] H.-D. Jang, S. Woo, P. Benz, J. Park, and I. S. Kweon, "Propose-and-attend single shot detector," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2020, pp. 815–824.
[41] R. Yan, J. Tang, X. Shu, Z. Li, and Q. Tian, "Participation-contributed temporal dynamic model for group activity recognition," in *Proc. ACM Multimedia Conf. Multimedia Conf.*, 2018, pp. 1292–1300.
[42] W. Lee, J. Na, and G. Kim, "Multi-task self-supervised object detection via recycling of bounding box annotations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4984–4993.
[43] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.
[44] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[45] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jan. 2015.
[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556
[47] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[48] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[49] W. Liu, A. Rabinovich, and A. C. Berg, "ParseNet: Looking wider to see better," 2015, *arXiv:1506.04579*. [Online]. Available: http://arxiv.org/abs/1506.04579
[50] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2014, pp. 740–755.
[51] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.

GUOZHENG JIN received the bachelor's degree from Dalian Maritime University, Dalian, China, in 2015. He is currently pursuing the Ph.D. degree with the Ocean College, Zhejiang University–Zhoushan, Zhoushan, China. His research interests include computer vision and image processing.

RIN-ICHIRO TANIGUCHI received the B.E., M.E., and D.Eng. degrees from Kyushu University, in 1978, 1980, and 1986, respectively. Since 1996, he has been a Professor with the Graduate School of Information Science and Electrical Engineering, Kyushu University, where he directs several projects, including multiview image analysis and software architecture for cooperative distributed vision systems. His current research interests include computer vision, image processing, and parallel and distributed computation of vision-related applications.

FENGZHONG QU (Senior Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from Zhejiang University, Hangzhou, China, in 2002 and 2005, respectively, and the Ph.D. degree in electrical and computer engineering from the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA, in 2009. From 2009 to 2010, he was an Adjunct Research Scholar with the Department of Electrical and Computer Engineering, University of Florida. Since 2011, he has been with the Ocean College, Zhejiang University–Zhoushan, Zhoushan, China, where he is currently a Professor and the Chair of the Institute of Ocean Sensing and Networking. His current research interests include underwater acoustic communication and networks, wireless communications, signal processing, and intelligent transportation systems. Dr. Qu is an Associate Editor of the IEEE Transactions on Intelligent Transportation Systems, *IET Communications*, and *China Communications*.