Single-Shot Detection Based on Cyclic Attention

KEBAI HU, DAOCHUN XU, AND JIANGMING KAN
School of Technology, Beijing Forestry University, Beijing 100083, China
Key Laboratory of State Forestry Administration on Forestry Equipment and Automation, Beijing 100083, China
Corresponding author: Jiangming Kan (kanjm@bjfu.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFD1002401-3.

ABSTRACT The shallow feature map of the single-shot detector (SSD) is not always conducive to enhancing the recognition precision for a small object because of the lack of contextual information. In this research, a single-shot detection algorithm based on cyclic attention (CA-SSD) is proposed to construct a fast and accurate detector that efficiently obtains full-image contextual information. Our network is constructed by integrating ResNet-34 and proposed novel cyclic attention blocks. This type of building block aggregates different transformations, one of which includes an attention module that uses a long but narrow pooling kernel to acquire horizontal and vertical contextual information for each pixel of all pixels. Each pixel eventually captures the full-image dependencies by following an even further cyclic operation. Our design considers the variability of the gradient, which not only improves the reliability of the cyclic attention block but also cuts the number of parameters for computation. Additionally, by exploring the effects of the stem block and its stride on the performance of ResNet-based SSD algorithms, our network retains more detailed information. For an input size of $300 \times 300$, CA-SSD attained 82.5% mAP on PASCAL VOC 2007 test, 78.4% mAP on PASCAL VOC 2012 test, and 32.7% mAP on MS COCO. Experimental results achieved with CA-SSD surpass the best results achieved with the traditional SSD and other advanced object detection algorithms while real-time speed is maintained.

INDEX TERMS Object detection, single-shot detector, ResNet, cyclic attention.

I. INTRODUCTION

Object detection is a fundamental task in the field of computer vision and is critical in many recently developed applications, such as autonomous driving [1], [2], fault detection [3], [4], and medical decision-making [5], [6]. The goal is to find several targets in the complex background of the image and to estimate the position and category information of each object at the same time.

Object detection has been extensively and actively studied with continuous breakthroughs in technology. Most early detectors directly extract features of histogram of oriented gradients or deformable part model [7]–[9] and send them to a well-trained classifier for classification. Unfortunately, these hand-crafted features have weak generalization ability and struggle to meet the needs of the application. In recent years, benefiting from the comprehensive progress of deep convolutional neural networks (CNNs) [10]–[14] and graphical processing unit (GPU) computing capabilities and the accumulation of prior work on well-annotated data sets [15], [16], hand-crafted features have gradually been replaced by complex image characteristics extracted by CNNs. The performance of the object detector has improved. Among object detectors, Faster R-CNN [17] and R-FCN [18] select the top feature map in predicting candidate region proposals of various aspect ratios and scales. However, because the receptive field in the top feature map is fixed, there is a conflict between objects of different scales and the fixed receptive field in the natural image, which limits the prediction of objects that are too small or too large and may decrease the performance of object detection. Meanwhile, an excessively deep backbone network has been used to extract features and improve the detection accuracy but this greatly reduced the detection efficiency [13], [18]. To accelerate detection, the one-stage detection framework called “you look only once” (YOLO) [19] and single-shot detector (SSD) [20], which abandons the phase of object proposal generation, are proposed. YOLO [19] is similar to Faster R-CNN [17] and uses a fully connected layer with a fixed receptive field to predict the detection results of multi-scale objects. Although YOLO has speed sufficient for real-time detection, its accuracy is unsatisfactory. The SSD [20] is a
better trade-off between accuracy and speed. It utilizes feature maps in various layers to predict objects at distinct scales. The SSD is one of the first attempts to implement convolutional pyramid feature representation for object detection. In the feature pyramid of the SSD, the high-level feature maps of great receptive fields used to detect large targets provide stronger distinguishing ability for classification subtasks because of semantic information, and the low-level feature maps of small receptive fields have detailed information for regression subtasks, which is conducive to predicting small objects. However, less semantic information is contained in the shallow features, which may result in small objects being poorly detected. Therefore, studies [21]–[24] developed ways to further improve accuracy through ingenious design. The FPN [21] adopts a top–down structure in constructing new feature pyramids for prediction though fusing different level features. The DSSD [22] employs an hourglass structure to transfer contextual information and obtain better results. However, these improved features basically come from the very deep ResNet-101 [13] with heavy computational costs, which lowers the inference speed. Nevertheless, the RFB [23] and FSSD [24] did not have inferior detection performance on the VGG [12] when structures were carefully designed. The deep features of lightweight CNNs have been enhanced by designing modules that simulate the human visual system [23]. A lightweight feature fusion module [24] was proposed by Li et al. to fully utilize the features. Obviously, in addition to having a better feature network, adding more contextual information improves the performance of the detector.

On the basis of the discussion above, in building a detector that considers both speed and accuracy, a reasonable choice is to introduce an ingenious design that improves the feature representation capabilities and avoids using deeper models that slow the calculation. In this research, to enhance the expressive capability of shallow feature maps, a single-shot detection algorithm based on cyclic attention (CA-SSD) is proposed and a novel module, namely the cyclic attention block (CA block), is designed. Specifically, we replace the basic network of the SSD with ResNet-34 and residual blocks. By analyzing the network structure of VGGNet and ResNet, we redesign the stem block to reserve abundant detailed information for object detection and greatly improve the detection accuracy without affecting the detection speed. At the same time, the CA block that enhances the variability of the learned features within different layers while learning the full-image contextual information is used. The CA block has fewer parameters than the residual block, which contributes to the construction of a fast and accurate detector. In achieving the goal of collecting full-image contextual information, we simply perform repeated operations on the cyclic attention module (CAM) to capture the information of the remaining positions indirectly. Among them, the CAM uses a long but narrow pooling kernel to expand the receptive fields of network and maintains the portability of the module by sharing CAM parameters.

Adopting the proposed architecture, our CA-SSD achieves excellent performance with a slight reduction in speed compared with the conventional SSD. We report extensive experiments on the PASCAL VOC 2007 and PASCAL VOC 2012 datasets, demonstrating that our CA-SSD has greatly improved detection of small objects compared with the conventional SSD and outperforms many detectors based on ResNet-101, such as R-FCN [18] and Faster R-CNN [13]. In conclusion, our major contributions are as follows.

1) We investigated factors that influence the detection performance of the ResNet-based SSD algorithm. Using the newly designed structure of the stem block can retain much detailed information for small-object detection.

2) We design the CA block, a novel and lightweight module. The computational cost is kept under control by simply replacing some components of ResNet with CA blocks without training them from scratch.

3) Quantitative and qualitative experiments demonstrate that our CA-SSD is better than the conventional SSD algorithm. Especially for small objects, our CA-SSD has competitive performance on common datasets (PASCAL VOC and MS COCO), but at a slightly reduced speed.

II. RELATED WORK

A. TWO-STAGE DETECTOR

The object detection method based on a region proposal can be called the two-stage algorithm, where the detection problem is divided into two stages. A two-stage detector called the region-based CNN (R-CNN) was first introduced by Girshick et al. [26] through integrating AlexNet [11] and a selective search [27]. The accuracy of this approach is much superior to that of the traditional approach, and the technique opens a new era of object detection in deep learning. Nevertheless, its low detection speed remains an urgent problem to be solved because R-CNN [26] needs to perform redundant feature calculations on a large number of overlapping proposals. He et al. subsequently suggested SPPNet [28] by incorporating conventional spatial pyramid pooling (SPP) in the R-CNN [26] architecture, so that the classification module can reuse the convolutional network features regardless of the resolution of the input image and improve the detection speed. Fast R-CNN [29] is a further improvement of SPPNet [28], allowing us to train classification and regression tasks at the same time, but it still relies on the proposals of external regions. Soon after, Faster R-CNN [17] emerged as the first detector that allows end-to-end training. Through the implementation of an effective and precise region proposal network (RPN) to produce region proposals, various parts of object detection are integrated into a unified deep learning framework. In addition, effective extension methods have been proposed to further enhance the prediction performance. ROI Align proposed by Mask R-CNN [30] solves the problem of mis-alignment between the feature map and the original image caused by ROI pooling, and improves the
effect of object detection and instance segmentation. Libra R-CNN [31] integrates the features of different levels more effectively through the balanced feature pyramid structure and uses the improved balanced L1 loss to make the detector achieve optimal convergence. D2Det [32] introduced a novel dense local regression strategy to achieve precise localization of objects.

B. ONE-SCALE DETECTOR

Different from the two-stage algorithm with higher precision but lower detection speed, the one-stage algorithm based on regression generates bounding boxes without proposals for external region and predicts the possibility of the object categories at the same time, which has an advantage in terms of speed. The YOLO [19], [33] and SSD [20] are the most representative one-stage detectors. YOLOv1 [19] divides an image into several grids to predict the bounding box at points near the center of the object instead of using anchor boxes. However, the recall of YOLOv1 is low, and YOLOv2 [33] thus employs anchor boxes and combines low-resolution and high-resolution feature maps via a pass through layer to achieve recognition of different scales. The SSD [20] uses multiple feature maps of different sizes and makes each layer concentrate on detecting targets of a specific size. It uses two $3 \times 3$ convolutional layers to calculate the category confidence and position offset of the default bounding boxes and then adopts non-maximum suppression (NMS) to filter the redundant boxes in the final prediction results. The DSSD [22] adopts deconvolution to aggregate contextual information and enhances the high-level semantics of shallow features. RetinaNet [34] proposes a focal loss through the reconstruction of standard cross-entropy loss, resulting in the detector paying more attention to samples that are hard to classify during the training process. The original lightweight backbones were updated with a deeper ResNet-101 [13], and the performance was comparable to or better than that of the two-stage method. However, the non-negligible cost of the increase in model complexity is the consumption of their advantages in speed.

Recently, more and more scholars have paid attention to improving the performance of the one-stage detector because of its convenience and efficiency [35]–[37]. Many works have proved that it is possible to balance accuracy and speed through some ingenious designs, such as STDN [38], RFB [23], and EfficientDet [37]. In addition, [39], [40] have confirmed the importance of contextual information in visual tasks such as object detection. Here, we propose to improve the detection performance by optimizing the network structure and capturing contextual information.

III. METHOD

To improve the performance of conventional SSD algorithms, a single-shot detection algorithm based on CA is proposed. We first introduce the backbone network in this chapter, which is part of our feature extraction network. In using ResNet-34 as the backbone network to obtain information, we change the elements in ResNet-34 that affect the performance of object detection. We then describe a new CA block comprising CA modules that extracts features that contain rich contextual information and are more favorable for identifying and positioning small objects. We finally describe the loss function and the network training details.

A. BACKBONE NETWORK

We use ResNet-34 to replace the VGG [12] in the original SSD paper [20] as our backbone network and explore the factors affecting the detection performance. The purpose is to overcome shortcomings in object detection while retaining powerful classification capabilities. Inspired by the DSOD [41], we replace the input $7 \times 7$ convolutional layer with a stack of three $3 \times 3$ convolutional layers. There are 64 output channels for all three convolutions, and the stride is 1. Next is the max pooling layer, and its kernel size is $2 \times 2$ and stride is 2. Different from the convolutional layer in the stem block of the DSOD, we have removed the downsampling operation of the first convolutional layer, considering the downsampling operation that already exists in the first convolutional layer is likely to have a greater influence on the detection accuracy, especially the accuracy of detection of small targets; see Table 2. Compared with the original ResNet design, a convolutional layer with a kernel size of $7 \times 7$ and a stride of 2 followed by a max pooling layer with a kernel size of $3 \times 3$ and a stride of 2, this simple substitution not only reduces parameter calculations but also reduces the information loss of the original image.

Fig. 1 describes the architectural details of the CA-SSD model. To select feature maps with the same fixed space size as SSD instead of using a larger feature map for detection, we apply the output of the last two CA blocks as two of the feature maps with which to predict the location and confidences, and add three additional residual blocks at the end of ResNet to extract $10 \times 10$, $5 \times 5$, and $3 \times 3$ feature maps. The residual block comprises two branches, one is a $3 \times 3$ convolutional layer with a stride of 2 followed by a $3 \times 3$ convolutional layer with a stride of 1 and the other is a $1 \times 1$ convolutional layer with a stride of 2. Finally, the last feature map is extracted through a $1 \times 1$ convolutional layer and a $3 \times 3$ convolutional layer with a stride of 1. Among these convolutional layers, the number of output channels is 128.

Experimental results show that removing the downsampling operation of the stem block improves the performance of object detection (see Table 2).

B. CA BLOCK

Previous work has demonstrated that contextual information is crucial in object detection. We design a CA block to effectively capture the global contextual information. In this part, the specific component of the CA block improved on the basis of the residual block is first introduced. We then introduce the CA module that acquires horizontal and vertical contextual information. We finally introduce how to adopt a
cyclic operation in a CA block so that each pixel captures the full-image dependencies.

The overall structure of the CA block is shown in Fig. 2. First, we split the input feature map \( I \in \mathbb{R}^{C \times H \times W} \) into two parts \( x_1 \) and \( x_2 \) through two \( 1 \times 1 \) convolutional layers, where \( C \) is the number of channels of the input feature map and \( \{x_1, x_2\} \in \mathbb{R}^{C/2 \times H \times W} \). Here, feature map \( Q \in \mathbb{R}^{C \times H \times W} \) is acquired by \( x_2 \) after a \( 3 \times 3 \) convolutional layer. We input \( Q \) into the CAM to obtain a new attention map \( Q' \in \mathbb{R}^{C \times H \times W} \). The number of channels of \( Q \) and \( Q' \) is \( C/2 \), and we will equate it to \( C \) in the following. Each pixel of the \( Q' \) aggregates the contextual information on the crossing path where it is located. However, the attention map \( Q' \) only acquires sparse information. Therefore, \( Q' \) is input into the CAM again, so that the output feature \( Q'' \) can obtain richer contextual information. It means that each position on the feature map \( Q'' \) aggregates global contextual information. At this time, the output obtained by \( x_1 \) after a \( 3 \times 3 \) convolutional layer is concatenated with \( Q'' \). This allows us to collect both contextual information and short-range information. Then, the concatenated feature passes through a convolution operation with a kernel size of \( 3 \times 3 \). Finally, the final feature map is output through the shortcut connection. We named this structure a CA block.

To make calculations and memory more portable and use local features to establish full-image dependencies, inspired by the scene parsing method [42], we introduce a CA module that acquires horizontal and vertical contextual information. As shown in Fig. 3, we suppose the input tensor of the CAM is \( Q \). After the input feature map passes through a horizontal or vertical pooling layer and the average value of the elements in the pooling kernel is used as the output value of the pooling layer, two feature maps \( q^h \) and \( q^v \) are generated, where \( q^h \in \mathbb{R}^{C \times H} \) and \( q^v \in \mathbb{R}^{C \times W} \). The module then applies a convolutional layer with a kernel size of \( 3 \times 1 \) and \( 1 \times 3 \) on the two output feature maps \( q^h \) and \( q^v \) respectively.
and expands the two output feature maps in horizontal and vertical directions. After the expansion, the two feature maps are of the same size, and the element-wise sum of the two feature maps is $q \in \mathbb{R}^{C \times H \times W}$:

$$q_{i,j} = T(\phi_1(q_i^j)) + T(\phi_2(q_j^i)),$$

where $T(\cdot)$ represents expanding the feature maps to the same size and $\phi(\cdot)$ represents a convolution operation with a kernel size of $3 \times 1$ or $1 \times 3$.

The output of the CAM is then described as

$$Q' = \theta(Q, \sigma(g(q))),$$

where $\theta(\cdot, \cdot)$ is element-wise multiplication, $\sigma(\cdot)$ represents the sigmoid function, and $g(\cdot)$ denotes $1 \times 1$ convolution.

However, a single CA module only acquires contextual information on the crossing path where each pixel is located and there is no connection with the remaining locations. Some key details are thus lost. We therefore loop the CA module twice to obtain a new feature map with rich and useful contextual information. As shown in Fig. 4, it is known that any point $t$ on $Q'$ is associated with the feature points on $Q'$ corresponding to its row and column. The feature point on $Q'$ is associated with $Q$ in the same way. The two CA modules share the same parameters. At this time, the point $t$ on $Q''$ establishes a relation with the global information on $Q$.

Compared with a non-local module [43] having similar functions, our CA block needs few calculations and parameters to establish relationships between each pair of locations. The CA block can replace building blocks in ResNet without training them from scratch. We replaced the last residual block of each stage in ResNet-34 with a CA block.

The experimental data demonstrates that our CA-SSD has great performance in terms of the speed and accuracy speed of object recognition (see Table 5).

C. TRAINING

1) LOSS FUNCTION

The loss function of our CA-SSD is a combination of the classification loss function and localization loss function, and is expressed as

$$L(y, p, b, g) = \frac{1}{N_{pos}}(L_{cls}(y, p) + \lambda L_{loc}(y, b, g)), \quad (y = y_{ij}^k \in \{0, 1\}),$$

where $L_{cls}$ and $L_{loc}$ are the classification loss function and localization loss function respectively, and the number of positive samples is $N_{pos}$. If $N_{pos} = 0$, the loss value defaults to zero. When $y_{ij}^k = 1$, the $i$-th default box matches the $j$-th ground truth box of category $k$. $p$ represents the predicted value of category score. The predicted location of the bounding box, which corresponds to a default box, is defined as $b$ and the ground truth position is represented by $g$.

The classification loss function of multiple categories is calculated as

$$L_{cls}(y, p) = -\sum_i^{N} [y_i^k \ln(\hat{p}_i^k) + (1 - y_i^k) \ln(\hat{p}_i^0)],$$

where $\hat{p}_i^k = \exp(p_i^k)/\sum_k \exp(p_i^k)$.

Similar to Faster R-CNN [17], localization loss function $L_{loc}$ is a smooth L1 loss between the predicted position and the ground truth position. The center point coordinates of the two sets of boxes are calculated, and then bounding box regression is applied.

The localization loss function formula is expressed as

$$L_{loc}(y, b, g) = \sum_{i}^{N_{pos}} \sum_{m \in \{x, y, w, h\}} y_i^k \text{smoothL1}(b_{ij}^m - g_{ij}^m).$$

The weight coefficient $\lambda$ is set to 1 through cross validation.

2) TRAINING SETTINGS

A single NVidia TITAN Xp GPU with 12 GB of memory is used to train all models. Most of the training strategies
of the improved algorithm proposed in this paper follow the SSD [20]; e.g., adopting a matching strategy, setting the scale for default boxes, and encoding the bounding boxes by using the center code type. Notably, in detecting small objects, the random expansion augmentation included in the latest SSD has been verified to be exceedingly helpful. We adopt the same data augmentation strategy as the SSD during training to make the model more stable against different shapes and sizes of the input objects and to compare with the conventional SSD fairly. A hard negative mining strategy is applied to ensure fast optimization and more stable training. More details can be found in [20]. The experimental section below provides more parameter settings, such as the batch size and learning rate.

IV. EXPERIMENTS

We compare our approach proposed in this research with advanced detectors to objectively appraise whether our approach realistically enhances the precision of small-object detection. After introducing the experimental settings and implementation details, we evaluate quantitative indicators on the PASCAL VOC and MS COCO datasets.

In the experiment, we use the stochastic gradient descent with a weight decay of 0.0005 and momentum of 0.9 as the optimization function. Different datasets have different learning rate decay strategies, which we will introduce separately later. For a 300 × 300 input, the batch size is set to 16 considering GPU specifications. It is worth noticing that we apply an aspect ratio of [2.0, 3.0] in the 38 × 38 feature map, and the number of anchor boxes thus increases to 11,620. We remove the L2 normalization [44]. The “xavier” method [45] is used to initialize all new convolutional layers and the mean average precision (mAP) is used as a metric for evaluating the detection performance of the model. In the following sections, the experimental process and evaluation outcomes will be described in detail.

A. PASCAL VOC 2007

The VOC 2007 and 2012 datasets have a total of 20 object categories. They respectively contain 9963 and 22,531 images, and each image is annotated with the ground truth position and corresponding category information. In this experiment, we train the models on the union of VOC 2007 trainval and VOC 2012 trainval and test on the VOC 2007 test dataset. We use the “warm-up” strategy in the first five epochs to gradually increase the learning rate from $10^{-6}$ to 0.004 and then use a fixed learning rate of 0.004 for the next training until 150 epochs are completed. In the next 150 to 200 epochs, the learning rate is reduced to 0.0004, and in the last 50 epochs, it is 0.00004.

The evaluation results of competitive detectors and our method, which are measured in terms of the prediction performance of each object category on the VOC 2007 test set, are shown in Table 1. All object detection approaches are trained and tested on the consistent datasets. SSD300* and SSD512* refer to the latest SSD results obtained by

### TABLE 1. PASCAL VOC 2007 test detection results.

| Method    | Backbone | mAP  | aeroplane | bicycle | bird | boat | bottle | bus | car | cat | chair | cow | table | dog | horse | motorbike | person | plant | sheep | sofa | train | tv |
|-----------|----------|------|-----------|---------|------|------|--------|-----|-----|-----|-------|-----|-------|-----|-------|-----------|--------|-------|-------|------|-------|----|
| Faster[17]| VGG      | 73.2 | 76.5      | 79.9    | 65.9 | 52.1 | 83.1   | 84.7| 86.4| 52  | 81.9  | 65.7| 84.8  | 84.6| 77.5  | 76.7       | 73.6   | 73.9  | 83   | 72.6 |
| ION[25]   | VGG      | 75.6 | 79.2      | 83.1    | 77.6 | 65.6 | 54.9   | 85.4| 85.1| 87  | 54.4  | 80.6| 73.8  | 85.3| 82.2  | 82.2       | 74.4   | 47.1  | 75.8 | 80.4 |
| MR-CNN[46]| VGG      | 78.2 | 80.3      | 84.1    | 78.5 | 70.8 | 68.5   | 88  | 85.9| 87.8| 60.3  | 85.2| 73.7  | 87.2| 86.5  | 85         | 76.4   | 48.5  | 76.3 | 75.5 |
| Faster[13]| ResNet-101| 76.4 | 79.8      | 80.7    | 76.2 | 68.3 | 55.9   | 85.1| 85.3| 89.8| 56.7  | 87.8| 69.4  | 88.3| 89.9  | 80.9       | 78.4   | 41.7  | 78.6 | 79.8 |
| R-FCN[18] | ResNet-101| 80.5 | 79.9      | 82.7    | 81.5 | 72.6 | 69.8   | 86.8| 88.5| 89.8| 67    | 88.1| 74.5  | 89.8| 90.6  | 79.9       | 81.2   | 53.7  | 81.8 | 81.5 |
| SSD300*[20]| VGG      | 77.5 | 79.5      | 83.9    | 76   | 69.6 | 50.5   | 87  | 85.7| 88.1| 60.3  | 85.1| 77   | 86.1| 87.5  | 83.9       | 79.4   | 52.3  | 77.9 | 79.5 |
| SSD512*[20]| VGG      | 79.5 | 84.8      | 85.1    | 81.5 | 73   | 57.8   | 87.8| 88.3| 87.4| 63.5  | 85.4| 73.2  | 86.2| 86.7  | 83.9       | 82.5   | 55.6  | 81.7 | 79   |
| RFB300*[23]| VGG      | 80.5 | 85       | 86.1    | 77.7 | 75.7 | 60.6   | 88.9| 87.6| 86.8| 64.2  | 85.3| 77.9  | 86.1| 89    | 87.1       | 82.3   | 58.7  | 81.5 | 88.3 |
| Retina640[34]| ResNet-50| 79.1 | 80.6      | 83.9    | 82.1 | 71.5 | 67.4   | 85.9| 87.3| 89  | 63.1  | 82.5| 73.8  | 84.7| 85.9  | 84.7       | 84.3   | 53.4  | 82.3 | 75.5 |
| DSSD321[22]| ResNet-101| 78.6 | 81.9      | 84.9    | 80.5 | 68.4 | 53.9   | 85.6| 86.2| 88.9| 61.1  | 83.5| 78.7  | 86.7| 86.7  | 86.7       | 79.7   | 51.7  | 78   | 80.9 |
| DSSD513[22]| ResNet-101| 81.5 | 86.6      | 86.2    | 82.6 | 74.9 | 62.5   | 89  | 88.7| 88.8| 65.2  | 87  | 78.7  | 88.2| 89    | 87.5       | 83.7   | 51.1  | 86.3 | 81.6 |
| STDN321[38]| DenseNet-169| 79.3 | 81.2      | 88.3    | 78.1 | 72.2 | 54.3   | 87.6| 86.5| 88.8| 63.5  | 83.2| 79.4  | 86.1| 89.3  | 88          | 77.3   | 52.5  | 80.3 | 80.8 |
| STDN513[38]| DenseNet-169| 80.9 | 86.1      | 89.3    | 79.5 | 74.3 | 61.9   | 88.5| 88.3| 89.4| 67.4  | 86.5| 79.5  | 86.4| 89.2  | 88.5       | 79.3   | 53    | 77.9 | 81.4 |
| CA-SSD300| ResNet-34-CA| **82.5** | 87.2 | 88.8      | 82.6  | 77.1 | 65.6   | 88.4| 88.4| 89.2| 68.5  | 88.2| 79.8  | 87   | 89    | 88.4       | 85.1   | 61.2  | 86.7 | 79.8 |
| CA-SSD512| ResNet-34-CA| **83.2** | 88.2 | 89       | 84.8   | 78   | 69.7   | 89.3| 89.1| 88.8| 68.5  | 87.6| 77.5  | 87.5| 89.2  | 88.7       | 86.1   | 62.5  | 86.3 | 81.6 |

VOLUME 9, 2021
TABLE 2. Ablation study on the PASCAL VOC 2007 test set.

| Method                                      | mAP  | FPS  | Anchor boxes | Parameters |
|---------------------------------------------|------|------|--------------|------------|
| SSD300*                                     | 77.5 | -    | 8732         | 26.3M      |
| ResNet-34 + SSD + extra ResBlock            | 75   | 32.4 | 11620        | 24.2M      |
| ResNet-34 + SSD + extra ResBlock + stem block | 76.5 | 43.4 | 11620        | 24.2M      |
| ResNet-34 + SSD + extra ResBlock + stem block + w/o downsmaping | 81.5 | 53.0 | 11620        | 24.6M      |
| ResNet-34 + SSD + extra ResBlock + stem block + w/o downsmaping + CA Block | **82.5** | 33.5 | 11620        | 24.0M      |
| ResNet-34 + SSD + extra ResBlock + stem block + w/o downsmaping + RCCA(R=2)[48] | 82   | 17.6 | 11620        | 23.6M      |

applying new expansion data augmentation step, where the numbers 300 and 512 represent the input size. These mAP results are respectively 1.1% and 3.1% higher than those of Faster R-CNN using ResNet-101 as the base network (77.5% vs. 76.4% and 79.5% vs. 76.4%). However, they are lower than those of other algorithms that use ResNet-101 as the base network; e.g., DSSD321 as 78.6% mAP and DSSD513 81.5% mAP and input sizes similar to that of the SSD, while R-FCN has 80.5% mAP. Although both the DSSD and R-FCN have improved accuracy compared with the SSD, the excessively deep backbone network undoubtedly limits the detection speed. The STDN [38] enhances the efficiency of object detection by embedding a scale-transfer module into DenseNet with almost no increase in running time, but it has an mAP 0.6% lower than that of DSSD513 (80.9% vs. 81.5%). The RFB through a multi-branch convolutional block that enhances feature representation of the lightweight network can even reach the accuracy of the advanced model R-FCN operating under the two-stage framework (80.5% mAP), and it is much superior to other methods, such as ION [25] and MR-CNN [46], which try to include contextual information. However, when our CA-SSD300 uses ResNet-34-CA as the backbone network, it attains 82.5% mAP, which is 5%, 3.9%, and 2% higher than the mAP results of SSD300, DSSD321, and RFB300, respectively. The result is even better than those for models with larger input sizes, such as SSD512, STDN513, DSSD513, and Retina640 implemented by us. Our mAP result of CA-SSD300 is respectively 3%, 1.6%, 1%, and 3.4% higher than the mAP results of these methods. Moreover, CA-SSD300 shows great improvement in small-object detection. It achieves the highest ranking in the categories of aero, bird, boat, chair, cow, person, plant, and sheep. Especially compared with the conventional SSD, the performance is improved for 19 categories, with the exception being the train category. We show the precision-recall curve of CA-SSD300 on the PASCAL VOC 2007 test set in Fig. 5. Experimental results show that our CA-SSD512 is only 0.7% higher than CA-SSD300. We think the possible reason is due to hardware limitations. When training CA-SSD512, we can only use a batch size of 8 to train the model at most. In addition, finding the optimal results for the extracted multi-scale features through the loss function is difficult to greatly affect the detection effect. Nevertheless, our algorithm still achieved the best performance. These results indicate that the weaknesses of small-object detection in the SSD are effectively solved by our proposed CA-SSD model.

B. ABLATION STUDY ON VOC 2007

The ablation experiments on VOC 2007 were designed and their evaluations were recorded in Table 2 to prove the effectiveness of each constituent in CA-SSD. The term “w/o
downsampling” means to delete the downsampling operation in the first convolutional layer of the stem block. The CA block is as described in section III.

In the second row of Table 2, the original backbone network and additional layers in the SSD algorithm with an input size of 300 × 300 are replaced with ResNet-34 and residual blocks. This method has 75.0% mAP, which is worse than the result obtained using VGG16.

To improve the prediction results, we replace the first convolutional layer and max-pooling layer in ResNet-34 with a stem block. The first convolutional layer applies a relatively large kernel size of 7 × 7 with a stride of 2. The results in the second and third rows of Table 2 reveal that the stem block improves the target detection accuracy from 75.0% to 76.5%. We believe that this improvement is due to the input having undergone two consecutive downsampling operations (conv and max pooling with stride 2) in ResNet, which resulted in too much information being lost, thereby impairing the performance of the detector. Although the results are improved, the mAP of ResNet-34 is still 1.0% (76.5% vs. 77.5%) lower.

FIGURE 6. Detection examples comparison of the SSD300 and our designed CA-SSD300 on the PASCAL VOC 2012 test. We display the results using a score threshold of 0.6. Different colors of boxes represent various categories. For each pair of pictures, the left side represents the recognition of the SSD, and the right side represents the recognition of our CA-SSD300.
than that of the SSD algorithm based on VGG16 at this time.

Comparing the structure in VGG16, we find that ResNet-34 performs a downsampling operation in the first convolutional layer, which leads to a considerable loss of circumstantial information and greatly affects the detection performance, especially in the recognition of small objects. The fourth row of Table 2 shows that after eliminating the downsampling operation of the first convolution, the performance of our detector reaches 81.5% mAP, which is an astonishing 5% mAP improvement over the previous modification (76.5% mAP). Experiments show that the downsampling step of the first convolution hinders better detection performance. When training the ResNet-based SSD, we need to delete this operation to decrease the lack of much important information from the original input image.

In the fifth row of Table 2, we replace the last building block of each stage in ResNet-34 with CA blocks to construct the backbone network, namely ResNet-34-CA, and CA-SSD is used as the basic network for feature extraction. When other components are the same, the mAP of CA-SSD is 82.5%, which is 1% higher than that in the fourth row of Table 2 and 5% higher than that for the SSD. At the same time, fewer parameters are required for calculation.

In the last experiment listed in Table 2, we compare the performance between the CAM in the CA block proposed in this paper and the recurrent criss-cross attention (RCCA) [48] with similar functions, which has superior performance in semantic segmentation. We replace the two CA modules in Fig. 2 with two criss-cross attention modules. Reference [48] has pointed out that when RCCA has two loops (R = 2), it can better balance efficiency and resource usage. The experimental results show that although RCCA can help us improve the performance of the algorithm, it only improves by 0.5%, and RCCA greatly slows down the detection speed (17.6 vs. 33.5). This verifies the effectiveness of the CA-SSD.
method. By combining ResNet-34 with CA blocks, we still have excellent performance on the VOC 2007 test dataset when the input size of the image is only $300 \times 300$.

C. PASCAL VOC 2012

We use the same training hyperparameters as used in the experiment on VOC 2007 for training. The difference is that we use 07++12 composed of VOC 2012 trainval and VOC 2007 trainval and test for training, and test on the VOC 2012 test dataset. Table 3 shows the accuracy of CA-SSD and the experimental results of advanced detector methods.

CA-SSD300 has 78.4% mAP, which exceeds the precision of some one-stage methods with similar input sizes, such as SSD300-VGG16 (75.8%, 2.6% higher mAP) and DSSD321-ResNet101 (76.3%, 2.1% higher mAP). CA-SSD512 is 0.5% higher than DSSD513-ResNet101. Simultaneously, CA-SSD300 maintains an excellent detection performance superior to that of the two-stage detector, although R-FCN has an input size of approximately $1000 \times 600$ (77.6%, 0.8% higher mAP).

To evaluate our method further in an intuitive manner, we qualitatively compare our method with the conventional SSD. When using an input size of $300 \times 300$, the recognition examples for the SSD and CA-SSD models is displayed in Fig. 6. The detection results clearly show that our CA-SSD model is more capable of detecting and identifying small-scale targets and some classes with different contexts.

D. MS COCO

In order to further verify the effectiveness of our algorithm, we train the model on the more challenging MS COCO. There are 80 categories in MS COCO. We use trainval35k set for training and use standard COCO metric to report the results on test-dev.

Compared with PASCAL VOC, the COCO dataset has more small objects, and there are more objects to be detected in a single image, which makes detection more difficult.

Therefore, during the training process, we retain the strategy of reducing the size of default box when training with the COCO in the SSD. At the beginning of training, we still apply the “warm-up” strategy to increase the learning rate from $10^{-6}$ to 0.002 in the first 500 iterations, divide by 10 at 80 and 110 epochs, and end training at 125 epochs.

The test results on MS COCO are shown in Table 4. The experimental results show that SSD300* has been better than the two-stage detectors Faster R-CNN and ION by 3.2% and 1.5%, respectively. Although SSD512* improves the original result by 3.7%, it is still lower than R-FCN. Our CA-SSD300 attains 32.7% mAP on test-dev, not only surpassing traditional SSD, but also outperforming R-FCN. We also can see

---

**TABLE 5. Comparison of speed and accuracy on the PASCAL VOC 2007 test set.**

| Method             | pre-train | Backbone | GPU   | Speed | Input size | Anchor boxes | mAP  |
|--------------------|-----------|----------|-------|-------|------------|--------------|------|
| Faster R-CNN[17]   | √         | VGG16    | Titan X | 7     | 1000×600   | 6000         | 73.2 |
| Faster R-CNN[13]   | √         | ResNet-101 | K40 | 2.4   | 1000×600   | 300          | 76.4 |
| R-FCN[18]         | √         | ResNet-101 | Titan X | 9     | 1000×600   | 300          | 80.5 |
| DSSD300[41]       | ×         | DS/64-192-48-1 | Titan X | 17.4   | 300×300    | -            | 77.7 |
| YOLOv2[33]        | √         | Darknet-19 | Titan X | 40    | 544×544    | -            | 78.6 |
| SSD300[20]        | √         | VGG16    | Titan X | 46    | 300×300    | 8732         | 77.5 |
| SSD512[20]        | √         | VGG16    | Titan X | 19    | 512×512    | 24564        | 79.5 |
| DSSD321[22]       | √         | ResNet-101 | Titan X | 9.5   | 321×321    | 17080        | 78.6 |
| DSSD513[22]       | √         | ResNet-101 | Titan X | 5.5   | 513×513    | 43688        | 81.5 |
| STDN300[38]       | √         | DenseNet-169 | Titan Xp | 41.5  | 300×300    | 13888        | 78.1 |
| STDN321[38]       | √         | DenseNet-169 | Titan Xp | 40.1  | 321×321    | 17080        | 79.3 |
| STDN513[38]       | √         | DenseNet-169 | Titan Xp | 28.6  | 513×513    | 43680        | 80.9 |
| RSSD300[47]       | √         | VGG16    | Titan X | 35    | 300×300    | 8732         | 78.5 |
| RSSD512[47]       | √         | VGG16    | Titan X | 16.6  | 512×512    | 24564        | 80.8 |
| FSSD300[24]       | √         | VGG16    | 1080Ti | 65.8  | 300×300    | 8732         | 78.8 |
| FSSD512[24]       | √         | VGG16    | 1080Ti | 35.7  | 512×512    | 24564        | 80.9 |
| CA-SSD300         | √         | ResNet-34-CA | Titan Xp | 33.5  | 300×300    | 11620        | 82.5 |
| CA-SSD512         | √         | ResNet-34-CA | Titan Xp | 19.6  | 512×512    | 32756        | 83.2 |
that the CA-SSD300 can achieve the best performance no matter what the threshold is. This means that our algorithm has stronger object detection and recognition capabilities than other algorithms, although they may have a larger image input size. Similarly, in terms of small object detection, CA-SSD300 is still 6.6%, 5.3%, 5.8% and 3.8% higher than SSD300, STDN321, DSSD321 and DSOD300 respectively, which is an advantage that cannot be ignored. These results prove that the detection performance of our algorithm for small objects has surpassed algorithms such as DSSD300. We show examples of object detection results on test-dev in Fig. 7. As can be seen from the figure, benefiting from CA-SSD, we can still identify more small objects even on COCO dataset.

E. COMPARISON OF SPEED AND ACCURACY

The batch size of 1 was used on a single Titan Xp to test the speed of CA-SSD. CA-SSD300 and CA-SSD512 run at 33.5 and 19.6 FPS, respectively. The speed of inference of the various models is shown in the fifth column of Table 5. The values of FPS and mAP in Table 5 were plotted in Fig. 8 as the abscissa and ordinate respectively, so that the advantages and disadvantages of various algorithms in terms of speed and accuracy can be compared more clearly. The figure shows that
our CA-SSD300 is faster than most methods used for target recognition and has the best performance.

V. CONCLUSION
We proposed a single-shot detection algorithm based on cyclic attention (CA-SSD). Different from the common use of the deepened backbone network to extract features to achieve better detection performance, we propose a novel and lightweight module, namely the CA block, and the algorithm uses ResNet-34 and the proposed CA block to effectively capture contextual and detailed information. We also explored factors that affect the recognition accuracy of the ResNet-based SSD algorithm and redesigned the stem block, which greatly improved the detection accuracy. We compared the proposed CA-SSD with many competitive methods through a series of quantitative and qualitative experiments on PASCAL VOC. The experimental results strongly prove that the CA-SSD300 model we proposed is much better than the conventional SSD framework, achieving 82.5% mAP on VOC 2007 test and 32.7% mAP on COCO test-dev; excellent detection results were obtained for small objects and context-specific objects. At the same time, CA-SSD300 reached 33.5 frames per second on a single Titan Xp GPU and thus had a speed comparable to that of other detectors.

ACKNOWLEDGMENT
The authors would like to thank Glenn Pennycook, M.Sc., from Liwen Bianji, Edanz Group China, for editing the English text of a draft of this manuscript.

REFERENCES
[1] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? The KITTI vision benchmark suite,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Providence, RI, USA, Jun. 2012, pp. 3354–3361.
[2] X. Dai, “HybridNet: A fast vehicle detection system for autonomous driving,” Signal Process., Image Commun., vol. 70, pp. 79–88, Feb. 2019.
[3] Y. Wu, B. Jiang, and N. Lu, “A descriptor system approach for estimation of incipient faults with application to high-speed railway traction devices,” IEEE Trans. Syst., Man, Cybern. Syst., vol. 49, no. 10, pp. 2108–2118, Oct. 2019.
[4] Y. Wu, B. Jiang, and Y. Wang, “Incipient winding fault detection and diagnosis for squirrel-cage induction motors equipped on CRH trains,” ISA Trans., vol. 99, pp. 488–495, Apr. 2020.
[5] Z. Li, M. Dong, S. Wen, X. Hu, P. Zhou, and Z. Zeng, “CLU-CNNs: Object detection for medical images,” Neurocomputing, vol. 350, pp. 53–59, Jul. 2019.
[6] S.-G. Lee, J. S. Bae, H. Kim, J. H. Kim, and S. Yoon, “Liver lesion detection from weakly-labeled multi-phase CT volumes with a grouped single shot multibox detector,” in Medical Image Computing and Computer-Assisted Intervention (Lecture Notes in Computer Science). Granada, Spain, Sep. 2018, pp. 693–701.
[7] R. Ranjan, V. M. Patel, and R. Chellappa, “A deep pyramid deformable part model for face detection,” in Proc. IEEE 7th Int. Conf. Biometrics Theory, Appl. Syst. (BTAS), Arlington, VA, USA, Sep. 2015, pp. 1–8.
[8] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), San Diego, CA, USA, Jun. 2005, pp. 886–893.
[9] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained part-based models,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, pp. 1627–1645, Sep. 2010.
[10] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5987–5995.
[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in Proc. Adv. Neural Inf. Process. Syst. (NIPS), Lake Tahoe, NV, USA, Dec. 2012, pp. 1097–1105.
[12] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014, arXiv:1409.1556. [Online]. Available: http://arxiv.org/abs/1409.1556.
[13] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[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), Boston, MA, USA, Jun. 2015, pp. 1–9.
[15] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal visual object classes (VOC) challenge,” Int. J. Comput. Vis., vol. 88, no. 2, pp. 303–338, Jun. 2010.
[16] T. Y. Lin, M. Maire, S. Belongie, J. Hays, and A. C. Berg, “SSD: Single shot multibox detector,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Amsterdam, The Netherlands, Oct. 2016, pp. 779–788.
[17] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[18] J. Dai, Y. Li, K. He, and J. Sun, “R-FCN: Object detection via region lar single shot detector,” 2017, arXiv:1701.06659.
[25] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, “Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Las Vegas, NV, USA, Jun. 2016, pp. 2874–2883.

[26] 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. (CVPR), Columbus, OH, USA, Jun. 2014, pp. 580–587.

[27] 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, Apr. 2013.

[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] R. Girshick, “Fast R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Santiago, Chile, Dec. 2015, pp. 1440–1448.

[30] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Venice, Italy, Oct. 2017, pp. 2980–2988.

[31] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, “Libra R-CNN: Towards balanced learning for object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Long Beach, CA, USA, Jun. 2019, pp. 821–830.

[32] J. Cao, H. Cholakkal, R. M. Anwer, F. S. Khan, Y. Pang, and L. Shao, “D2Det: Towards high quality object detection and instance segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 2020, pp. 11482–11491.

[33] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Honolulu, HI, USA, Jul. 2017, pp. 6517–6525.

[34] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Venice, Italy, Oct. 2017, pp. 2999–3007.

[35] N. Samet, S. Hicsonmez, and E. Akbas, “HoughNet: Integrating near and long-range evidence for bottom-up object detection,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Glasgow, U.K., Aug. 2020, pp. 406–423.

[36] Y. Li, Y. Pang, J. Shen, J. Cao, and L. Shao, “NETNet: Neighbor erasing and transferring network for better single shot object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 2020, pp. 13346–13355.

[37] M. Tan, R. Pang, and Q. V. Le, “EfficientDet: Scalable and efficient object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 2020, pp. 10778–10787.

[38] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu, “Scale-transferable object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Salt Lake City, UT, USA, Jun. 2018, pp. 528–537.

[39] Z. Qin, Z. Li, Z. Zhang, Y. Bao, G. Yu, Y. Peng, and J. Sun, “ThunderNet: Towards real-time generic object detection on mobile devices,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Seoul, South Korea, Oct. 2019, pp. 6717–6726.

[40] C. Guo, B. Fan, Q. Zhang, S. Xiang, and C. Pan, “AugFPN: Improving multi-scale feature learning for object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 2020, pp. 12592–12601.

[41] 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), Venice, Italy, Oct. 2017, pp. 1937–1945.

[42] Q. Hou, L. Zhang, M.-M. Cheng, and J. Feng, “Strip pooling: Rethinking spatial pooling for scene parsing,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 2020, pp. 4002–4011.

[43] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Salt Lake City, UT, USA, Jun. 2018, pp. 7794–7803.