IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 25, 2020, accepted March 26, 2020, date of publication March 31, 2020, date of current version April 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2984554

Real-Time Detection Method for Small Traffic Signs Based on Yolov3

HUIBING ZHANG[1], LONGFEI QIN[1], JUN LI[2], YUNCHUAN GUO[3], YA ZHOU[1], JINGWEI ZHANG[1], AND ZHI XU[1]

[1]Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China
[2]Department of Information Security, Beijing Information Science and Technology University, Beijing 100192, China
[3]Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100192, China

Corresponding author: Jun Li (lijun@bistu.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0803001, in part by the National Natural Science Foundation under Grant 61662013, Grant U1711263, Grant U1811264, Grant 61662015, and Grant 66162014, in part by the Beijing Information Technology University Fund Project under Grant 5221910933, in part by the Guangxi Science and Technology Department Base Talent Project under Grant 2018AD19083, and in part by the Guangxi Natural Science Foundation under Grant 2017GXNSFAA198372.

ABSTRACT It is very challenging to detect traffic signs using a high-precision real-time approach in realistic scenes with respect to driver-assistance systems for driving vehicles and autonomous driving. To address this challenge, in this paper, a new detection scheme (named MSA_YOLOv3) is proposed to accurately achieve real-time localization and classification of small traffic signs. First, data augmentation is achieved using image mixup technology. Second, a multi-scale spatial pyramid pooling block is introduced into the Darknet53 network to enable the network to learn object features more comprehensively. Finally, a bottom-up augmented path is designed to enhance the feature pyramid in YOLOv3, and the result is to achieve accurate localization of objects by utilizing fine-grained features effectively in the lower layers. According to the tests on the TT100K dataset (which is a dataset for traffic sign detection), the performance of the proposed MSA_YOLOv3 is better than that of YOLOv3 in detecting small traffic signs. The detection speed of MSA_YOLOv3 is 23.81 FPS, and the mAP (mean Average Precision) reaches up to 86%.

INDEX TERMS Convolutional neural network, small object detection, traffic sign detection, YOLOv3.

I. INTRODUCTION

As an essential part of an advanced driver-assistance system and autonomous driving, traffic sign detection is required for locating traffic signs from realistic scene images and classifying them into specified categories. For practical applications, reliable and real-time detection of small objects in complex backgrounds is required while the vehicle is running at high speeds [1].

Overall, traffic sign detection can be roughly divided into two categories: traditional schemes based on hand-crafted features and deep learning schemes based on convolutional neural networks (CNNs). In the traditional approaches, three steps are often involved: obtaining regional proposals that contain traffic signs, extracting hand-crafted features from these regions, and taking these features as the inputs to the classifiers (e.g., SVM) [2]–[5]. However, the hand-crafted features extracted by traditional algorithms are low-level, specific to specified objects and unable to well represent multi-class objects. Therefore, the robustness of the traditional approaches is poor for detecting traffic signs in a complex environment. In contrast, the CNNs can learn more generalized features from a large number of samples without preprocessing, and thus, they can avoid the difficulty of designing hand-crafted features. Overall, the CNN-based detection approach can be further divided into two categories: two-stage approaches and one-stage approaches. The two-stage approaches to detecting traffic signs have high precision [6], [7], but these approaches require high computing complexity and they do not satisfy the real-time requirement. In the one-stage approaches, regression is used to achieve real-time performance; for example, YOLOv3 [8] could reach 20 FPS under the 608 × 608 input size. However, their performance on detecting small objects (e.g., traffic signs) is often low.

Furthermore, the most common dataset used in the field of traffic sign detection is the German Traffic Sign Detection Benchmark (GTSDB) [9]. We argue that the schemes with perfect detection results on the GTSDB could have low performance in a practical environment [10], because the

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

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

GTSDB roughly divides the traffic signs into three categories (i.e., mandatory signs, danger signs, and prohibitory signs), and the schemes for the GTSDB detection task are required only to detect the 3 traffic signs. Obviously, the three types of signs are not enough in practice. To address this problem, the Tsinghua-Tencent 100K (TT100K) dataset [11], which has more practical traffic signs, has been published. This dataset covers natural weather factors (e.g., the weather conditions of foggy, cloudy, and rainy), partial occlusion, and significant variations (e.g., illuminance and viewing angle). This benchmark is closer to the real scenes than the GTSDB, which has more background and smaller traffic signs. However, the detetion methods [11]–[13] on this benchmark do not solve problems in real time.

To detect small traffic signs in time in real scenes using YOLOv3, we proposed a new detection scheme (named MSA_YOLOv3) to improve the detection efficiency. Our main contributions are as follows:

1) In the data preprocessing stage, image mixup technology is applied for data augmentation. Specifically, two images are randomly selected from the training set and mixed pixelwise. The mixed images are used as an interpolation between the training image pairs. MSA_YOLOV3 is trained on these random convex combinations of pairs of examples, which can effectively reduce the false and missed detection rates of the traffic signs in a complex background.
2) Three-scale spatial pyramid pooling (SPP) is added to the convolutional layer in the end of Darknet53. The SPP block performs pooling operations on the input feature map at different scales and connects the pooled three feature maps and input feature map, in such a way that MSA_YOLOv3 can learn the object features more comprehensively.
3) To utilize accurate localization signals in the lower layers, an augmented path, which is designed from bottom to top to enhance the feature pyramid structure in YOLOv3, is created to improve the locating accuracy on small objects in high-resolution images.

The remainder of this paper is organized as follows: We discuss the relevant work in Section 2 and propose the detection framework in Section 3. Section 4 presents the experimental results, and Section 5 concludes the paper.

II. RELATED WORK

Because deep learning-based object detection methods have achieved good results on public data, many researchers have begun to apply these methods to traffic sign data. Next, we briefly discuss the object detection algorithms based on CNNs and their applications in the field of traffic sign detection.

A. CNN-BASED OBJECT DETECTION

There are mainly two types of object detection methods that are based on CNNs: two-stage schemes (also named R-CNN series object detection) and one-stage schemes, where the two-stage schemes combine region proposals with the CNN network to detect objects. In the one-stage schemes, the object detection is transformed into a regression problem to perform end-to-end detection.

1) TWO-STAGE SCHEMES

The main idea of these schemes is to first generate a large number of region proposals through heuristic algorithms (e.g., selective search) [14] or CNN networks (e.g., Region Proposal Networks) for each image, and then, they classify and regress these candidate regions. For example, as a classic two-stage scheme, R-CNN [15], which obtains approximately 2k region proposals by selective search [16], extracts the features of the region proposals by the CNN, and finally, it determines the classes of the objects by multiple SVMs [17] and adopts linear regression to fine-tune the bounding boxes. The SPP-Net [18] convolves the whole image at one time to extract the features and avoids the problem of having the enormous and redundant computation when the R-CNN extracts features for all of the candidate regions separately. In addition, the SPP-Net adds a spatial pyramid pooling layer between the last convolutional layer and the fully connected layer to extract the fixed-length feature vectors and to avoid the normalization of the region proposals. However, the SPP-Net follows almost the same multi-stage pipeline as the R-CNN. The steps of the region proposals are determination, feature extraction, object classification, and bounding-box regression, which are still separated. Thus, additional expense on storage space is still required. Based on the SPP-Net, Fast R-CNN [19] simplifies the SPP layer into the ROI Pooling layer and applies singular value decomposition (SVD) on the outputs of the fully connected layer to accelerate the testing procedure. Finally, the classification score of the object and the regression between the predicted bounding boxes (bboxes) and the ground-truth boxes are obtained through two sibling output layers. Fast R-CNN combines classification with bounding box regression to achieve single-stage training instead of the original separated training of the object classification and object localization, Fast R-CNN suffers from the problem of too much calculation (because it uses selective search to determine the region proposals, this step involves many calculations). For example, when running on a CPU, it takes 2 s on average to obtain the region proposals for each image. To address this problem, Ren *et al.* [20] replaced the selective search algorithm by Region Proposal Networks (RPN) to extract the region proposals. Through this approach, they realize end-to-end computation on the object detection and greatly improved the detection efficiency by sharing the convolutional layers.

2) ONE-STAGE SCHEMES

Both R-CNN and Faster R-CNN adopt the idea of ''region proposal + CNN feature extraction + SVM or softmax classification'' to detect objects. The two-stage schemes are slightly insufficient in terms of real-time performance.

To address this problem, YOLO (which is a "one-stage" model) [21] is proposed to divide the input image into S × S grids, where each grid cell predicts 2 bounding boxes, their confidence scores, and the conditional class probabilities. In the YOLO scheme, a region proposal is replaced with a grid-centered multi-scale region. Through this approach, the detection efficiency is substantially increased to satisfy the real-time requirement at the cost of low accuracy. Faster R-CNN features high detection accuracy but slow detection speed, while the detection accuracy of YOLO is not high, but its detection speed is fast. SSD [22] combines the advantages of YOLO and Faster R-CNN to perform object detection on different feature maps at the same time and to achieve both high accuracy and high efficiency. However, the SSD suffers from the following problems. (1) The number of default boxes increases linearly with the resolution of the input image. (2) The scale and ratio of the default box in each layer of the network varies, and they cannot be obtained by learning; instead, they must be set manually. In practice, their set seriously relies on the experience-dependent debugging process. YOLOv2 [23] further improves the detection efficiency and accuracy by adding batch normalization after each convolution layer, performing multi-scale training, and applying the K-means dimension clustering on the training set bounding boxes to automatically determine the suitable priors. The algorithm could achieve up to 78.6% mAP (mean Average Precision) on the PASCAL VOC 2007 dataset [24] with a detection speed of 40FPS. In 2018, by adding a feature pyramid structure on the basis of YOLOv2, Joseph released YOLOv3 [8]. The feature extractor is updated from the original darknet19 to darknet53. YOLOv3 further improves the ability to detect small objects by using top-down multi-level predictions. However, the images in both the PASCAL VOC [24] and COCO [25] datasets have small resolution. In practical traffic sign detection tasks, the images usually have large resolution, and the object sizes are small (e.g., the image resolution is 2048 × 2048, and the size of the traffic sign is 40 × 40). Therefore, YOLOv3 cannot be directly used on such datasets.

B. TRAFFIC SIGN DETECTION

Yang *et al.* [12] combined traditional computer vision algorithms with CNNs and presented a new traffic sign detection network. In their work, a two-stage adjustment strategy is used to extract the region proposals, and an Attention Network (AN) is introduced in the Faster-RCNN to find the potential regions of interest; then, these regions are further classified roughly according to their color characteristics. Finally, the final region proposals are generated by an FRPN (Fine Region Proposal Network). The experimental results tested on the TT100k benchmark dataset showed that its mAP was 80.31%. However, the detection accuracy on small objects is quite low (49.81%), and the detection efficiency also fails to meet the real-time requirements. Lu *et al.* [13] applied the visual attention model to improve the detection effect of Faster R-CNN. The visual attention model can generate a set of region proposals with appropriate sizes to locate and classify small objects. Their mAP on TT100K is 87.0%, and the efficiency is 3.85 FPS. Zhu *et al.* [11] adopted the object localization in [26] to detect traffic signs. Their detection framework has three branches. The first layer (i.e., the pixel layer) used Overfeat's [27] efficient "sliding window" to detect the probability with which a 4 × 4 pixel region of the input image contains a target object. Each result of the second layer (i.e., the bounding box layer) represents the distance between the 4 × 4 pixel region and the four sides of the predicted bounding box of the target. GroupRectangles in OpenCV is applied to merge the bounding boxes. The third layer (i.e., the label layer) outputs the category probability of the target object. Their mAP on the TT100K dataset is 87.5%. Their model can locate the objects efficiently, because the mask detection proposed in [28] is introduced to highlight the object positions by the object mask outputted by the pixel layer. Their work relies on image pyramids to detect objects of different sizes, and thus, their detection speed should be improved. Li *et al.* [29] proposed a Perceptual Generative Adversarial Network (Perceptual GAN) model that improves the detection accuracy of small traffic signs in the TT100K data set through narrowing the representation difference of small objects from large objects. Meng *et al.* [30] used an expensive image pyramid and sliding window approach to achieve a recall of 0.93 and an accuracy of 0.90 on TT100K. Unfortunately, they did not provide the inference time.

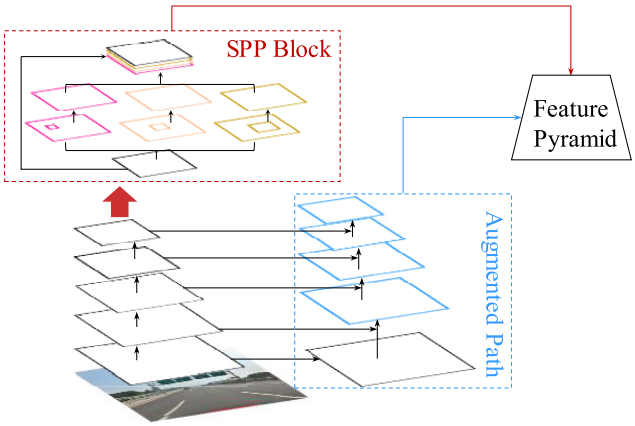


FIGURE 1. The simplified network structure of MSA_YOLOv3. (Note that the bottom feature map in the augmented path is not processed; it comes from the lower convolution layer in Darknet53).

III. MSA_YOLOV3 DETECTION FRAMEWORK

To achieve real-time detection of traffic signs and improve the detection ability of YOLOv3 on small objects, while considering MSA_YOLOv3 (as shown in Fig. 1), an algorithm for small traffic sign detection in a realistic environment is proposed in this paper. First, the generalization of the new model is improved by the image mixup technique during the model training. Then, by utilizing global features and multi-scale local region features, a multi-scale spatial pyramid

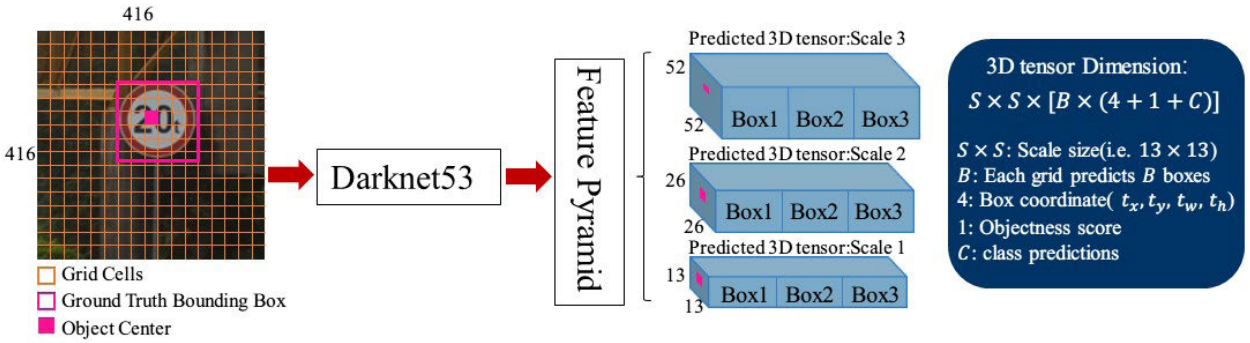


FIGURE 2. YOLOv3 detection method.

pooling module is introduced to Darknet53 to improve the detection accuracy of small targets with rich local region features. Finally, by utilizing the fine-grained features of the lower convolutional layer, the augmented path is designed in Darknet53 to improve the localization ability for small traffic signs.

A. YOLOv3

In YOLOv3 detection pipelines, all of the bounding boxes and category probabilities from the entire image are generated by a single convolutional network at once, as shown in Fig. 2. First, the network divides each image in the training set into $S \times S$ (e.g., $S = 13$) grids. Each grid is given candidate boxes of different sizes. If the center of the object ground truth falls in a grid, then the grid is responsible for detecting the object. Then, the features are extracted through the convolutional layer (Darknet53). Finally, the yolo layer is used for multi-scale prediction. Each grid predicts B bounding boxes and their confidence scores, as well as C class conditional probabilities.

B. DATA AUGMENTATION

According to the Vicinal Risk Minimization (VRM) principle, the generalization capacity should be improved by creating samples that are similar to the training samples for data augmentation. In this paper, data augmentation is conducted from two aspects: On the one hand, since YOLOv3 does not have a fully connected layer, the prediction results are generated from every single cell in the feature map and hence preserve the spatial alignments. Therefore, during the model training stage, images are randomly flipped and cropped to increase the spatial position transformation of the objects. On the other hand, we apply mixup [31] of the classification tasks into YOLOv3. After preliminary experiments, we choose a distribution for the blending ratio in mixup that is drawn from beta distributions B(1.5, 1.5). In a nutshell, mixup constructs virtual training examples.

$$img\_C = \lambda img\_A + (1-\lambda) img\_B \tag{1}$$

where, represent two input images, is the mixed image, and λ is the blending ratio.

$$\boldsymbol{C}_{img\_C} = \lambda \boldsymbol{C}_{img\_A} \cup (1-\lambda)\boldsymbol{C}_{img\_B} \tag{2}$$

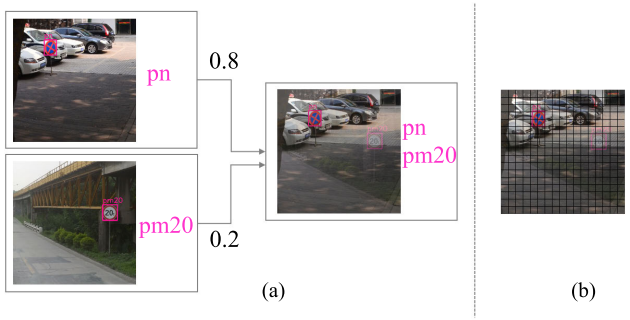


FIGURE 3. (a) Mixup visualization with a blending ratio at 0.8:0.2. Object labels are merged as a new array (pm20 and pn represent traffic sign labels). (b) YOLOv3's grid cell prediction mechanism, the confidence of the ground truth bounding box in the training stage is set to be the corresponding blending weight.

where $\boldsymbol{C}_{img\_A}$ represents the confidence of the ground truth boxes in image img_A, $\mathbf{C}_{img\_B}$ represents the confidence of the ground truth boxes in image img_B, $\mathbf{C}_{img\_C}$ and represents the confidence of the ground truth boxes in the mixed image.

An example of mixup in the proposed approach is illustrated in Fig. 3. Two images from the training set are randomly selected and mixed pixelwise. The mixed images are used as an interpolation between the training image pairs, as shown in Fig. 3a. In Fig. 3b, the confidence of the ground truth box in the mixed image is set to be the corresponding blending weight (the confidence of the ground truth box of pn is set to 0.8, and the confidence of the ground truth box of pm20 is set to 0.2). Compared with [32], we not only mix the pictures but also adjust the confidence of the ground truth box. Three training images are produced after mixup. The model is trained on random convex combinations of pairs of examples and their confidences, which can effectively improve the generalization of the model.

C. MULTI-SCALE LOCAL REGION FEATURES

YOLOv3 utilizes the feature pyramid structure to improve the detection accuracy of the multi-scale objects by fusing feature maps of different scales. However, these feature maps are only the global features of different convolutional layers of the network, and the multi-scale local region features of the convolutional layer are not effectively utilized. To effectively

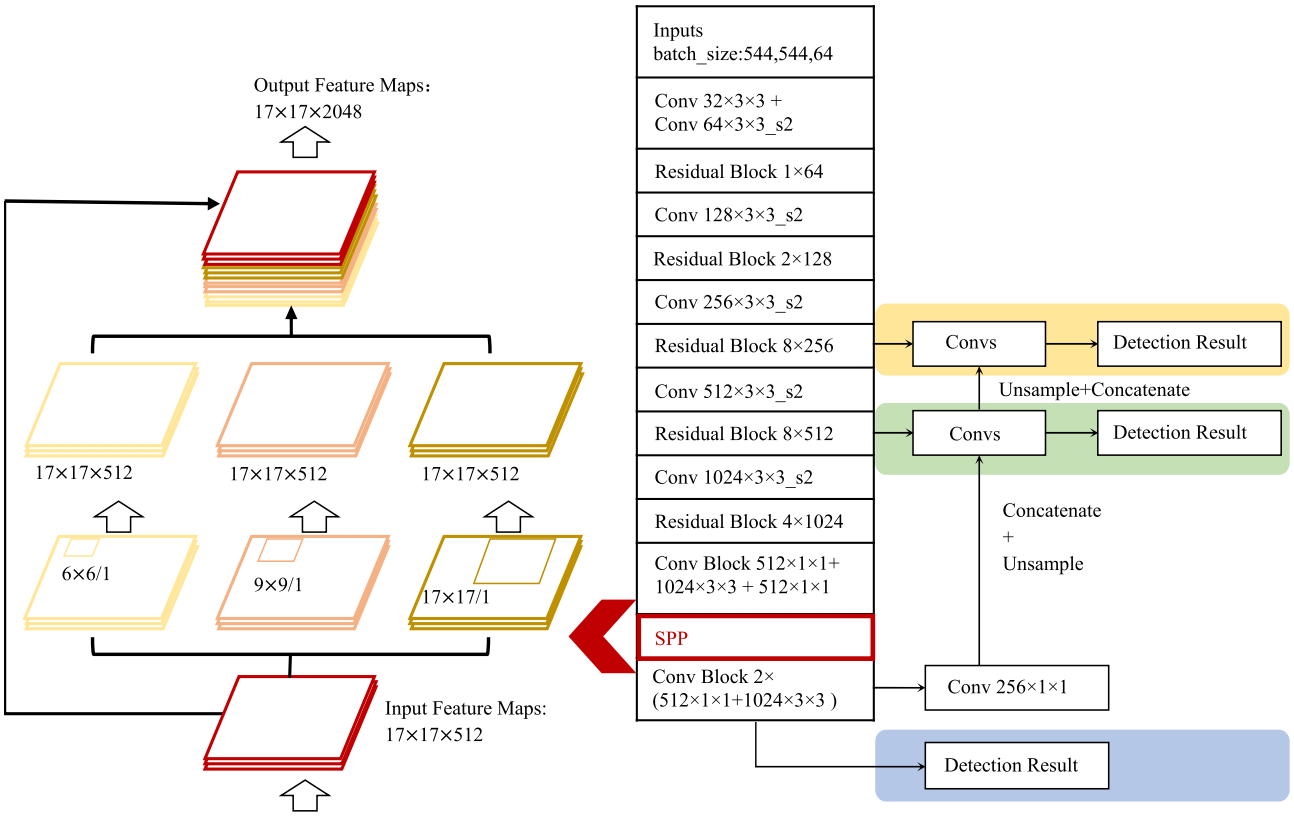


FIGURE 4. SPP_YOLOv3 (Conv represents the convolutional layer, _s2 represents that the stride is 2).

make use of the local region features of the final convolutional layer of Darknet53, the spatial pyramid pooling block (SPP Block) [33] (as shown in Fig. 4) is adopted to pool the local regions of the feature maps. Both the local multi-scale and global features are utilized together to improve the accuracy of the object detection. Here, the multi-scale spatial pyramid pooling block is composed of three max-pooling layers, and the size of the pooling window can be computed from (3)

$$size_p = \lceil size_f / n_i \rceil \tag{3}$$

where $size_p$ represents the size of the pooling windows, and $size_f$ represents the size of the feature maps, $n_i = 1, 2, 3$.

We add the spatial pyramid pooling block in front of the detection layer of YOLOv3. The resolution of the input image is 544×544. After downsampling 5 times, the size of the input feature map of the SPP Block is $17 \times 17 \times 512$. The sizes of the pooling windows obtained from (3) are 6×6, 9×9 and 17×17. The strides of the pooling windows are all 1, and the input feature maps are padded with 0 to ensure that the output feature maps after pooling are the same size as the input.

D. BOTTOM-UP AUGMENTATION PATH

Although the fine-grained features in the lower layer map can accurately locate small targets, they are not suitable for object recognition (because of their weak representation capacity) [34]. Therefore, an augmentation path (AP) is designed from

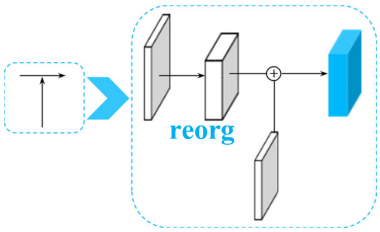


FIGURE 5. AP_YOLOv3 (White cubes represent convolutional layers, red cubes represent residual modules, and yellow cubes represent upsampling layers). The blue dotted line represents the augmented path, where each blue cube is obtained by merging a large feature map after processing by the reorg layer with a small feature map. P1, P2 and P3 constitute the FPN structure of YOLOv3.

bottom to top. As shown in Fig. 5: {P1, CP2, CP3} represents the feature pyramid structure of YOLOv3, and the feature maps after performing the augmentation path are concatenated with P1, P2, and P3 of the feature pyramid, respectively.

In Darknet53, the feature maps that are outputted by many consecutive layers are of the same size. We consider these layers to be in the same network stage. The final feature map of each network stage has the strongest semantic information. Therefore, a new feature map can be generated utilizing a high-resolution feature map and a feature map after downsampling in a network stage. As shown in Fig. 6, a user-defined reorg layer is adopted to process the high-resolution

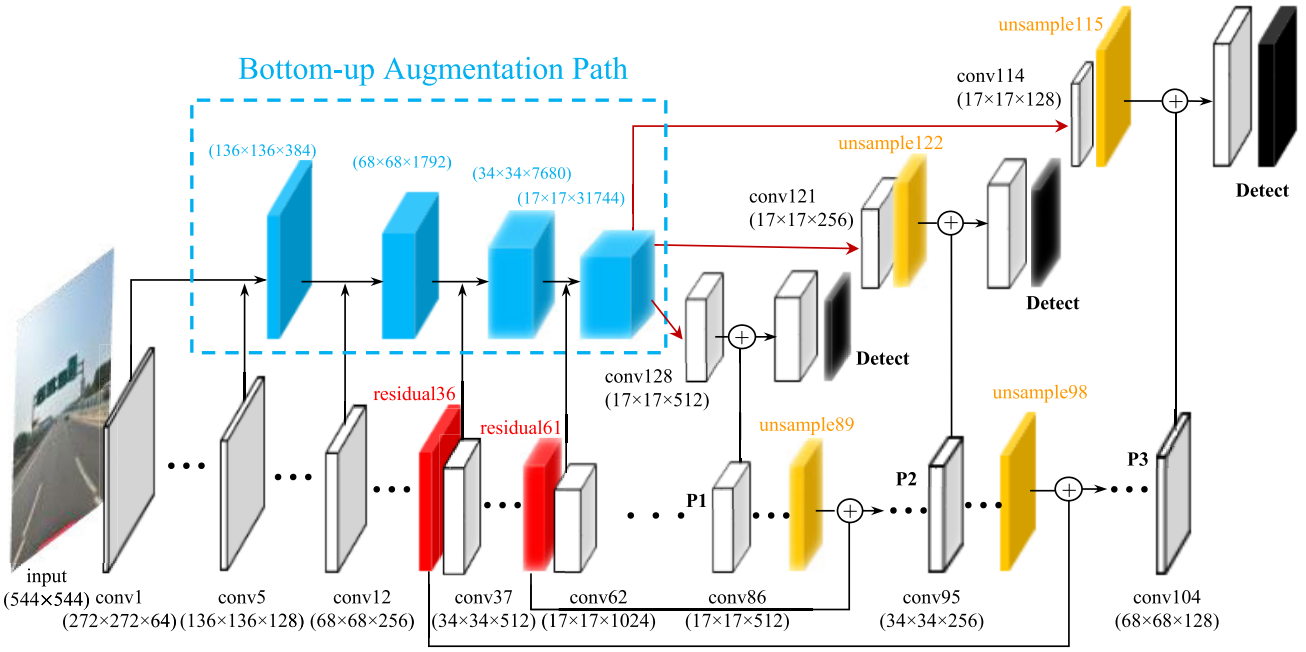


FIGURE 6. A building block illustrating the bottom-up pathway and the vertical connection.

feature maps to make it be the same size as the downsampled feature map, and then, they are connected vertically. The reorg layer can preserve fine-grained features without introducing trainable parameters. The specific process of the reorg layer is described with pseudo-code in algorithm 1, in which 'width', 'height', 'channel', and 'num', respectively, represent the width, height, channel of the image, and number of the image in a batch. The parameter stride in each reorg layer is 2.

E. LOSS FUNCTION

The loss function of MSA_YOLOv3 is defined as follows:

$$Loss = Error_{boxes} + Error_{objectness} + Categorical\ cross - entropy \quad (4)$$

$Error_{boxes}$ is the error sum of squares (SSE) of the coordinate regression, which is used to locate the bounding box. $Error_{objectness}$ is also the SSE, which is the loss of confidence of the bounding box. MSA_YOLOv3 predicts a confidence score for each bounding box, in other words, how likely it is that the bounding box is the target, which can remove unnecessary anchors and reduce the amount of calculation. $Categorical\ cross - entropy$ is used as the cross entropy loss for the object classification.

The localization loss of bounding box is defined as follows,

$$Error_{boxes} = \lambda_{coord} \sum_{i=0}^{S} \sum_{j=0}^{S} \sum_{k=0}^{A} 1_{ijk}^{obj} \sum_{r\in(x,y,w,h)} (truth^r - pre^r)^2 \quad (5)$$

In (5), *truth* and *pre* represent the label and prediction result. Here, λ_{coord} is the weight of the coordinate error. S is the number of girds of YOLOv3. A refers to the number of bounding boxes generated by each grid cell (it is 3 in this paper), and $truth^r$ represents the coordinates of the ground truth box. In addition, pre^r represents the coordinates of the predicted bounding box. $1_{ijk}^{obj} = 1$ denotes that the object falls into the *kth* bounding box in grid *(i, j)*, and otherwise, $1_{ijk}^{obj} = 0$. In addition, $1_{ijk}^{obj} = 1$ means that only the maximum of IOU_{pre}^{truth} is taken as the prediction result of each grid among the A anchor boxes.

The confidence loss of the bounding box is defined as follows:

$$Error_{objectness} = \sum_{i=0}^{S} \sum_{j=0}^{S} \sum_{k=0}^{A} 1_{ijk}^{obj} (1^m - pre^o)^2 + \lambda_{noobj} \sum_{i=0}^{S} \sum_{j=0}^{S} \sum_{k=0}^{A} 1_{ijk}^{noobj} (-pre^o)^2 \quad (6)$$

In (6), pre^o is the bounding box confidence, which represents the probability that there exists an object in an anchor box. If IOU_{pre}^{truth} of the *kth* predicted bounding box of grid cell *(i, j)* is smaller than the threshold IOU_{thres}, then $1_{ijk}^{noobj} = 1$. Since mixup is applied in data augmentation, then if the object is generated by the mixup, then $1^m = blending\ weight$; otherwise $1^m = 1$.

The categorical cross-entropy is defined as follows:

$$Categorical\ cross - entropy = \sum_{i=0}^{S} \sum_{j=0}^{S} \sum_{k=0}^{A} 1_{ijk}^{obj} \sum_{c=0}^{C} (-truth^c \cdot \log(pre^c)) \quad (7)$$

Algorithm 1 Reorg Layer

Input: input_batch(width, height, channel, num) and stride **Output:** output_batch(width/stride, height/stride, channel × stride × stride, num)

```
for n ← 0 to num-1 do
  for c ← 0 to channel × stride × stride-1 do
    for h ← 0 to height/stride-1 do
      for w ← 0 to width/stride-1 do
        // index of each feature location of the output feature map
        output_index ← w + (width/stride) × (h + (height/stride) × (c + channel × stride × stride × n))
        c2 ← c mod channel
        offset ← c / channel
        w2 ← w × stride + offset mod stride
        h2 ← h × stride + offset / stride
        // index of each feature location of the input feature map
        input_index ← w2 + width × (h2 + height × (c2 + channel × n))
        output_batch[output_index] ← input_batch[input_index]
return output_batch
```

TABLE 1. Ablation experiment.

	YOLOv3(544× 544)				MSA_YOLOv3(544× 544)
Mixup		√	√	√	√
SPP			√		√
Augmented Path				√	√
TT100K test mAP(IOU=0.5)	0.804	0.829	0.842	0.858	**0.863**

TABLE 2. Test results of SPP_YOLOV3 on TT100K.

	TP	FP	FN	Precision	Recall
YOLOv3	6242	1820	1464	0.77	0.81
SPP_YOLOv3	6557	2243	1149	**0.75**	**0.85**

TABLE 3. Test results of YOLOv3 on TT100K after applying image mixup technology (P: Precision, R: Recall, Iters: Iterations).

	TP	FP	FN	P	R	Iters
YOLOv3	5989	2317	1717	0.72	0.78	32500
Mixup+YOLOv3	6242	1820	1464	**0.77**	**0.81**	**30700**

In (7), $truth^c = 1$ if the label of an object is c; otherwise, $truth^c = 0$; pre^c refers to the predicted probability that an object belongs to class c. YOLOv3 adopts binary cross-entropy as the loss function for the object classification. During the experiments, we determined that multiple prediction categories of traffic signs can be generated for the same predicting box, which is because the binary cross-entropy with logistic activation is used for multi-label classification. Therefore, the categorical cross-entropy with softmax activation is introduced to replace the binary cross-entropy in the original loss function when performing multi-class classification.

IV. TEST ANALYSIS

The detection performance of MSA_YOLOV3 on small objects is evaluated on the TT100K [11] dataset. The model is implemented based on the Darknet52 neural network [35] and runs on the Ubuntu 16.04 PC equipped with Tesla P100 GPU and CUDA 9.2. During the process of training MSA_YOLOv3 and YOLOv3, the initial learning rate is set to 0.001, and the learning strategy of steps is adopted. The SGD optimizer with a momentum of 0.9 is utilized to adjust the parameters of the network. Moreover, we use a weight decay of 0.0005 to prevent model overfitting. We initialize the weight using the pre-trained model on ImageNet [36], and each training batch contains 64 images. The input resolutions of training and testing images are both 544 × 544.

TABLE 4. Comparison between YOLOv3 and MSA_YOLOv3.

	Bn Ops	Speed	FPS	mAP
YOLOv3 544×544	112.196	0.024 s	41.67	0.804
Ours 544×544	137.975	0.042 s	23.81	**0.863**

A. ABLATION STUDIES

We investigate the effectiveness of different components of MSA_YOLOv3. As shown in Table 1, "Mixup" refers to the use of the mixup algorithm to train YOLOv3, in which the distribution of the blending ratio is drawn from a beta distribution B (1.5, 1.5). The mixup helps to improve the detection mAP of YOLOv3 for 2.5%, because the model is trained on virtual examples that are constructed as the linear interpolation of two random examples from the training set and their

FIGURE 7. The upper row presents the experiment results of YOLOv3, and the lower row presents the detection results of YOLOv3 after performing data augmentation. The green box presents the correct prediction of the traffic signs, the red box represents the missed detections, and the blue box represents the false detections.

TABLE 5. Detection results of traffic signs with different sizes.

	(0,32]	(32,96]	(96,400]
YOLOv3 mAP	0.703	0.857	0.886
YOLOv3 precision	0.641	0.765	0.784
YOLOv3 recall	0.669	0.836	0.883
Ours mAP	**0.782**	**0.910**	**0.903**
Ours precision	0.721	0.842	0.826
Ours recall	0.742	0.895	0.894

confidence. ''SPP'' means that we added an improved SPP block to YOLOv3 (i.e., SPP_YOLOv3). Table 1 presents that the mAP of SPP_YOLOv3 is 84.2%, which is higher than that of YOLOv3. The precision and recall of YOLOv3 and SPP_YOLOv3 are compared in Table 2. From the experiment, we determined that the local features obtained through the multi-scale spatial pyramid have rich semantic information. If local features similar to traffic signs appear in the feature map in the test stage, the detection network will predict that it is a traffic sign. Through this approach, both the truth positives (TPs) and the general recall are enhanced. ''Augmented Path'' represents a bottom-up augmented path. Because the feature map generated by the augmented path not only preserves the fine-grained features that improve the object localization capability but also improves the accuracy of the object classification by its strong representational capacity, the detection accuracy of AP_YOLOv3 is improved by 2.9%.

B. DATA AUGMENTATION EXPERIMENT

As shown in Fig. 7, it can be seen that in cloudy weather conditions (Column 1), the vanilla model misses some detection; and under incomplete object conditions (Column 2), there is also missed detection. For images with a complex background (Column 3), there are many false detections in the vanilla model. Because the background of the images becomes complex, and the objects increase after performing mixup, and the images at the bottom of Column 1 and Column 3 show good detection results. Since random cropping makes objects in the training set incomplete, the model can detect the incomplete objects (images at the bottom of Column 2).

Table 3 further proves the validity of the mix technology. The model trained with the mix approach shows a 3%

TABLE 6. Performance of different models on the TT100K test set.

Method	mAP	Precision	Recall	Speed	FPS	GPU
AN+FRPN [12]	0.803	——	——	0.128 s	7.8	Tesla K20
Lu_model [13]	0.870	0.917	0.834	0.26 s	3.85	GTX980
Zhu_model [11]	0.927	0.905	0.928	10.83 s	——	Tesla P100
Ours	0.863	0.825	0.841	**0.042 s**	**23.81**	Tesla P100

TABLE 7. The recall and precision on each category of TT100K using the model of Zhu *et al.* [11] and MSA_YOLOv3.

Class	i2	i4	i5	il100	il60	il80	io	ip	p10	p11	p12	p19	p23	p26
Zhu recall	0.82	0.96	0.95	0.97	0.95	0.96	0.90	0.87	0.96	0.92	0.95	0.93	0.96	0.92
Zhu precision	0.83	0.87	0.95	1.00	0.97	0.97	0.79	0.87	0.90	0.91	0.88	0.91	0.91	0.89
Ours recall	0.88	0.97	0.96	0.96	0.97	0.97	0.89	0.83	0.80	0.79	0.84	0.86	0.90	0.86
Ours precision	0.85	0.84	0.92	0.85	0.95	0.89	0.85	0.90	0.74	0.72	0.78	0.73	0.82	0.81

Class	p27	p3	p5	p6	pg	ph4	ph4.5	ph5	pl100	pl120	pl20	pl30	pl40	pl5
Zhu recall	0.97	0.91	0.96	0.89	0.90	0.78	0.88	0.89	0.97	0.98	0.94	0.95	0.95	0.93
Zhu precision	0.95	0.81	0.89	0.83	0.91	0.90	0.82	0.83	0.99	1.00	0.92	0.90	0.93	0.91
Ours recall	0.82	0.84	0.93	0.78	0.94	0.73	0.88	0.73	0.89	0.86	0.82	0.82	0.77	0.86
Ours precision	0.83	0.81	0.77	0.72	0.92	0.82	0.83	0.63	0.88	0.80	0.75	0.74	0.74	0.78

Class	pl50	pl60	pl70	pl80	pm20	pm30	pm55	pn	pne	po	pr40	w13	w32	w55
Zhu recall	0.92	0.93	0.95	0.97	0.87	0.96	1.00	0.93	0.94	0.78	0.98	0.90	0.79	0.93
Zhu precision	0.91	0.96	0.89	0.91	0.93	0.91	0.77	0.91	0.93	0.79	0.93	0.90	0.90	0.76
Ours recall	0.79	0.76	0.83	0.83	0.72	0.88	0.84	0.90	0.97	0.73	0.92	0.78	0.79	0.83
Ours precision	0.72	0.76	0.79	0.72	0.76	0.67	0.71	0.83	0.96	0.76	0.85	0.70	0.91	0.79

Class	w57	w59	wo
Zhu recall	0.95	0.96	0.60
Zhu precision	0.90	0.87	0.63
Ours recall	0.92	0.92	0.57
Ours precision	0.85	0.73	0.53

improvement in the recall rate and a 5% improvement in the precision rate. Since the visually deceptive training images are generated randomly within the neighborhood of the training sample, the model becomes more robust. From Table 3, it is easy to deduce that mixup will work better as the training time increases.

C. EVALUATION

1) EXPERIMENTAL COMPARISON BETWEEN MSA_YOLOV3 AND YOLOV3

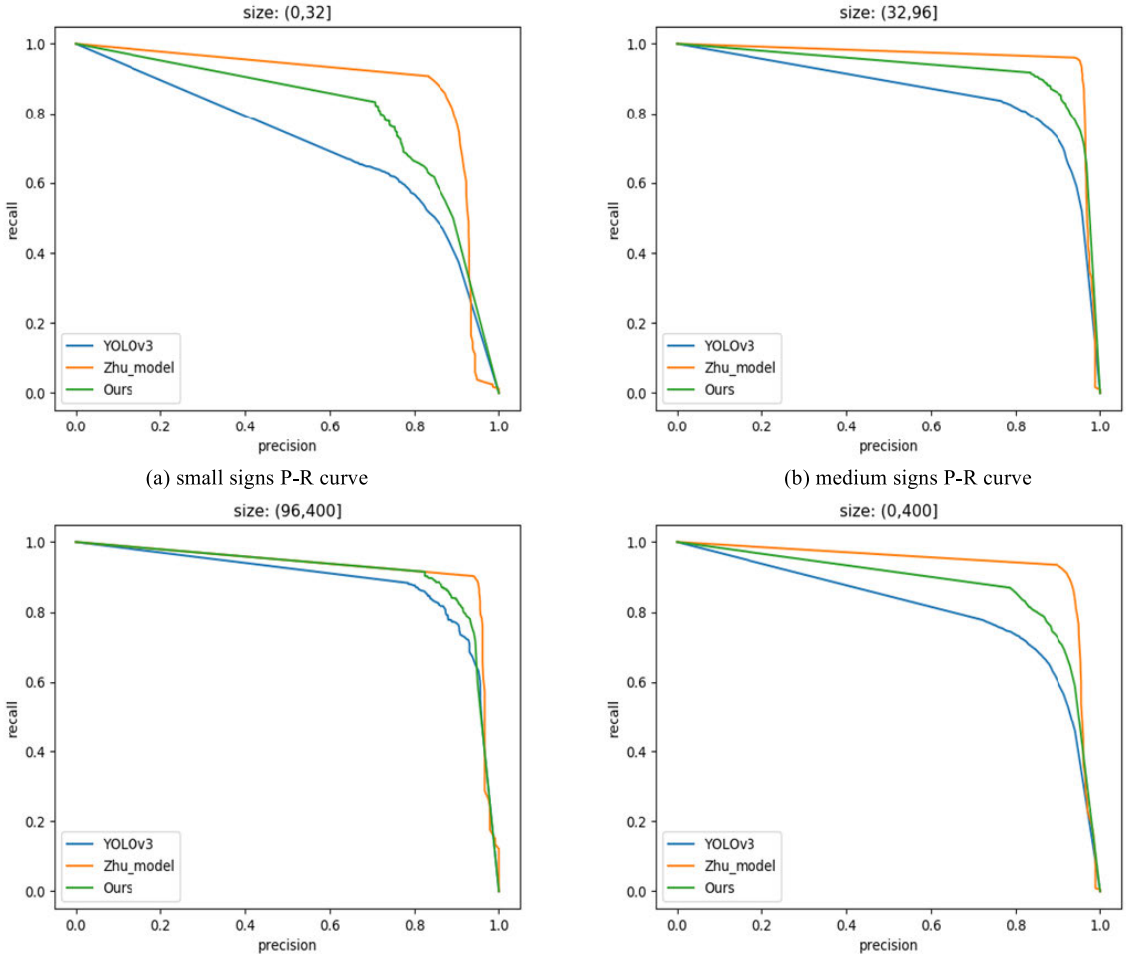
Table 4 reveals the complexity, speed, and accuracy of YOLOv3 and MSA_YOLOv3. Bn Ops (Billions of floating point operations) describes the complexity of the model. As shown in the table, the proposed MSA_YOLOv3 is more complicated, and the detection time is nearly doubled, but the detection accuracy is greatly improved.

To verify the efficiency of our scheme for small object detection, the Microsoft COCO benchmark [25] evaluation metrics are adopted, which divide the traffic signs into three types according to their sizes: small objects (the length and width of the object are (0, 32] pixels), medium objects (the length and width of the object are (32, 96] pixels), and large objects (the length and width of the object are (96, 400] pixels). Such an evaluation scheme can evaluate the detection capacity of the detectors on objects of different sizes. As suggested in Table 5, our model improves the detection capacity of YOLOv3 on small and medium-sized traffic signs.

2) EXPERIMENTAL COMPARISON BETWEEN MSA_YOLOV3 AND THE OTHER REFERENCE MODELS

Currently, the most accurate model on TT100K is the OverFeat framework as improved by Zhu *et al*. [11]. We re-trained their model under the same experimental environment. The results show that the mAP of their model reaches as high as 0.93, the recall reaches 0.93, and the precision reaches 0.91. However, they must establish an image pyramid to detect traffic signs of different sizes. The original image size is 2048 × 2048, while the largest image in the image pyramid is 8192 × 8192. The detector scans multiple high-resolution images in the image pyramid, which causes enormous amounts of computation. For the model proposed by Lu *et al*. [13], since it avoids scanning and multi-scale detection on all high-resolution images, their model is two orders of magnitude faster than that of Zhu *et al.* [11], taking only 0.3 s to process the same image. Although the experimental results of Lu *et al.* [13] appear to be perfect, they fail to meet the real-time performance requirement. In Table 6, the mAP of MSA_YOLOv3 is lower than that of [11] and [13], and MSA_YOLOv3 achieves real-time performance.

Fig. 8 shows the P-R curves of the YOLOv3 model, the model of Zhu *et al.* [11] and the proposed model on large/medium/small traffic signs. Although compared with the YOLOv3 model, the detection accuracy on small/medium traffic signs of the proposed model is increased, it is worse than that of the state-of-the-art model of Zhu *et al.* [11],



(a) small signs P-R curve

(b) medium signs P-R curve

(c) large signs P-R curve

(d) all signs P-R curve

FIGURE 8. P-R curves of the MAS_YOLOv3 and other reference methods.

especially on detecting small objects. We also present the recall and precision for each category for Jaccard similarity coefficient 0.5 in Table 7.

V. CONCLUSION

In the field of traffic sign detection, it is truly a challenge to reliably detect small signs in real time in high-resolution images. To address this problem, we use MSA_YOLOv3 to improve the generalization of the model through data augmentation. Furthermore, the multi-scale spatial pyramid pooling and augmentation path are added to the original Darknet53, which makes MSA_YOLOv3 outperform YOLOv3 in detecting small/medium-sized traffic signs. During the experiments, we determined that the augmentation path of MSA_YOLOv3 will greatly increase the computational complexity of the model. In the future, we will prune the augmentation path to eliminate the useless and redundant features. At the same time, to deploy MSA_YOLOv3 in mobile scenarios, we must also turn Darknet53 into a highly compact convolutional neural network. Through these methods, the computation cost is reduced to make the MSA_YOLOv3 applicable on a low/medium-end graphics card.

REFERENCES

[1] Q. Chen, "F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds," 2019, *arXiv:1909.06459*. [Online]. Available: http://arxiv.org/abs/1909.06459

[2] V. Q. Dinh, Y. Lee, H. Choi, and M. Jeon, "Real-time traffic sign recognition," in *Proc. IEEE Int. Conf. Consum. Electron. Asia (ICCE-Asia)*, Jun. 2018, pp. 206–212.

[3] M. Liang, M. Yuan, X. Hu, J. Li, and H. Liu, "Traffic sign detection by ROI extraction and histogram features-based recognition," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Aug. 2013, pp. 1–8.

[4] J. Greenhalgh and M. Mirmehdi, "Real-time detection and recognition of road traffic signs," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1498–1506, Dec. 2012, doi: 10.1109/TITS.2012.2208909.

[5] Y. Yang, H. Luo, H. Xu, and F. Wu, "Towards real-time traffic sign detection and classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 2022–2031, Jul. 2016, doi: 10.1109/TITS.2015.2482461.

[6] Y. Zhu, M. Liao, M. Yang, and W. Liu, "Cascaded segmentation-detection networks for text-based traffic sign detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 209–219, Jan. 2018, doi: 10.1109/TITS.2017.2768827.

[7] Y. Zhu, C. Zhang, D. Zhou, X. Wang, X. Bai, and W. Liu, "Traffic sign detection and recognition using fully convolutional network guided proposals," *Neurocomputing*, vol. 214, pp. 758–766, Nov. 2016, doi: 10.1016/j.neucom.2016.07.009.

[8] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[9] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The german traffic sign recognition benchmark: A multi-class classification competition," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2011, pp. 1453–1460.

[10] Q. Yang, A. Lim, S. Li, J. Fang, and P. Agrawal, "ACAR: Adaptive connectivity aware routing for vehicular ad hoc networks in city scenarios," *Mobile Netw. Appl.*, vol. 15, no. 1, pp. 36–60, Feb. 2010, doi: 10.1007/s11036-009-0169-2.

[11] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2110–2118.

[12] T. Yang, X. Long, A. K. Sangaiah, Z. Zheng, and C. Tong, "Deep detection network for real-life traffic sign in vehicular networks," *Comput. Netw.*, vol. 136, pp. 95–104, May 2018, doi: 10.1016/j.comnet.2018.02.026.

[13] Y. Lu, J. Lu, S. Zhang, and P. Hall, "Traffic signal detection and classification in street views using an attention model," *Comput. Vis. Media*, vol. 4, no. 3, pp. 253–266, Sep. 2018, doi: 10.1007/s41095-018-0116-x.

[14] D. Wu, B. Liu, Q. Yang, and R. Wang, "Social-aware cooperative caching mechanism in mobile social networks," *J. Netw. Comput. Appl.*, vol. 149, Jan. 2020, Art. no. 102457, doi: 10.1016/j.jnca.2019.102457.

[15] 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.*, Columbus, OH, USA, Jun. 2014, pp. 580–587.

[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, doi: 10.1007/s11263-013-0620-5.

[17] J. Xiong, M. Zhao, M. Bhuiyan, L. Chen, and Y. Tian, "An AI-enabled three-party game framework for guaranteed data privacy in mobile edge crowdsensing of IoT," *IEEE Trans. Ind. Informat.*, early Access, Dec. 2, 2019, doi: 10.1109/TII.2019.2957130.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proc. Eur. Conf. Comput. Vis.*, Zurich, Switzerland: Springer, 2014, pp. 346–361.

[19] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[20] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[22] W. Liu, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands, Oct. 2016, pp. 21–37.

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

[24] 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, doi: 10.1007/s11263-009-0275-4.

[25] T.-Y. Lin, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[26] B. Huval, T. Wang, S. Tandon, J. Kiske, W. Song, J. Pazhayampallil, M. Andriluka, P. Rajpurkar, T. Migimatsu, R. Cheng-Yue, F. Mujica, A. Coates, and A. Y. Ng, "An empirical evaluation of deep learning on highway driving," 2015, *arXiv:1504.01716*. [Online]. Available: http://arxiv.org/abs/1504.01716

[27] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated recognition, localization and detection using convolutional networks," 2013, *arXiv:1312.6229*. [Online]. Available: http://arxiv.org/abs/1312.6229

[28] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2553–2561.

[29] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1951–1959.

[30] Z. Meng, X. Fan, X. Chen, M. Chen, and Y. Tong, "Detecting small signs from large images," in *Proc. IEEE Int. Conf. Inf. Reuse Integr. (IRI)*, Aug. 2017, pp. 217–224.

[31] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2018, pp. 1–13.

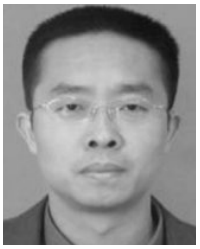
[32] Z. Zhang, T. He, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of freebies for training object detection neural networks," 2019, *arXiv:1902.04103*. [Online]. Available: http://arxiv.org/abs/1902.04103

[33] Z. Huang and J. Wang, "DC-SPP-YOLO: Dense connection and spatial pyramid pooling based YOLO for object detection," 2019, *arXiv:1903.08589*. [Online]. Available: http://arxiv.org/abs/1903.08589

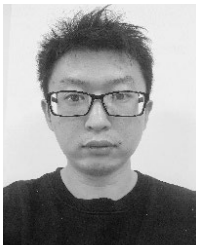
[34] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.

[35] J. Redmon. (Jun. 2013). *Darknet: Open Source Neural Networks in C*. [Online]. Available: http://pjreddie.com/darknet

[36] 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, doi: 10.1007/s11263-015-0816-y.



HUIBING ZHANG was born in Nanyang, China, in 1976. He received the B.S. and M.S. degrees in computer science and technology from the Guilin University of Electronic Technology, Guilin, China, in 2007, and the Ph.D. degree in computer science and technology from the Beijing University of Technology, Beijing, China, in 2012. From 2012 to 2016, he was a Lecturer with the Guilin University of Electronic Technology, where he has been an Associate Professor with the Guangxi Key Laboratory of Trusted Software, since 2016. His research interests include trust evaluation and management in the Internet of Things and social computing.



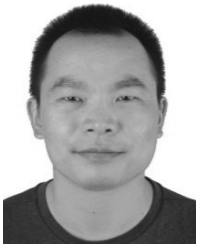
LONGFEI QIN was born in Weifang, China, in 1992. He is currently pursuing the M.S. degree with the Guilin University of Electronic Technology. His current research interests include deep learning, object detection, and small target detection.

JUN LI received the B.S. degree in computer science and technology from Shandong University, in 2007, and the Ph.D. degree in computer science and technology from the Beijing University of Posts and Telecommunications, in 2012. He is currently a Senior Engineer with the Department of Information Security, Beijing Information Science and Technology University. His research interests include network traffic management in network security, the accurate mining of network big data, and security issues in blockchain.

YUNCHUAN GUO received the B.S. and M.S. degrees in computing science and technology from the Guilin University of Electronic Technology, Guilin, China, in 2000 and 2003, respectively, and the Ph.D. degree in computing science and technology from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2011. He is currently an Associate Professor with the Institute of Information Engineering, Chinese Academy of Sciences. His current research interests include network security and access control.

YA ZHOU was born in 1966. She received the M.S. degree in computer science from Fudan University, China. She is currently a Professor with the Guilin University of Electronic Technology. Her research interests include distributed systems, database theory, data mining, and Web service technology.

JINGWEI ZHANG received the Ph.D. degree from East China Normal University, China, in 2012. He is a Professor with the School of Computer and Information Security, Guilin University of Electronic Technology, China. His research interests include massive data management, distributed computing frameworks, Web data analysis, and big data analytics for emerging applications.



ZHI XU received the Ph.D. degree from Sichuan University, Chengdu, China, in 2013. He is currently an Associate Professor with the School of Computer Science and Information Safety, Guilin University of Electronic Technology. His research interests include computer vision, machine learning, and pattern recognition.