Towards Effective Traffic Sign Detection via Two-Stage Fusion Neural Networks

Zhishan Li, Hongxu Chen, Battista Biggio, Yifan He✉, Haoran Cai✉, Fabio Roli, Lei Xie

***Abstract*—Automatic detection of traffic signs is crucial for Advanced Driving Assistance Systems (ADAS). Current two-stage approaches consist of a preliminary object detection step, where the traffic signs are categorized within broader families (e.g., speed limits), and then sub-classes (e.g., speed limit 40). However, these cascading methods fail to achieve satisfying performance, especially in more realistic driving scenarios where images are acquired under more challenging conditions. Under such conditions, the first-stage detection step is likely to provide inaccurate predictions, making the subsequent classification step useless. In this paper, we propose a simple yet effective two-stage fusion framework for traffic sign detection. Different from the previous cascading method, our framework directly predicts categories in the first-stage detection and fuse the two-stage category predictions to improves overall robustness. Besides, in order to filter the false detection boxes under low-resolution inputs, we also propose an effective post-processing method called Surrounding-Aware Non-Maximum Suppression (SA-NMS) as an alternative technique for the first-stage detection. After combining the above proposed methods, our framework obtains good detection performance. Experimental results on the widely used Tsinghua-Tencent 100K (TT100K) traffic sign dataset , which contains images of traffic signs collected under a variety of challenging conditions, show that the proposed framework outperforms current approaches in both accuracy and inference speed, achieving 89.7 mAP and 65 FPS for 608×608 low resolution images.**

***Index Terms*—Traffic Sign Detection, Two-Stage Fusion, SA-NMS, Lightweight**

1. Introduction

With the progress of deep learning theory, object detectors based on convolutional neural networks (CNNs), such as SSD [1], YoloV3 [2] and EfficientDet [3], have achieved excellent performance on open-source detection benchmarks [4]–[7]. Traffic sign detection is one of the most important applications of object detection in the field of intelligent transportation. Automatic detection of traffic sign is conducive to driving safely and preventing frequent traffic accidents.

For current traffic sign detectors, we divide them into two types. One is to directly design improved single detectors, which is also the first choice of most recent researches [8] [9] [10]. These approaches improve the fitting ability of the detectors to traffic sign detection by introducing more robust modules and structures. The other type detectors are those with two-stage cascading frameworks (detection families → classification categories). Taking Figure 1 as an example, the currently popular two-stage cascading traffic sign detection frameworks [11] [12] [13] are as follows: inputting the image into detector for first step detection, obtaining the location and family "Speed Limit" of traffic signs, then cropping the object part from the original image and acquiring the category "Speed Limit 40" through classifier. In the second-stage, the background of the cropped images is relatively simple, so the category predictions can be more accurate by using another classifier. However, the improvement is limited for this cascading manner. If the family prediction in the first-stage is wrong (e.g., Height Limit), the category predictions in the second-stage would belong to other family. The classification network in the second-stage cannot compensate for the first-stage. Since the first-stage is an object detection task and the background of traffic signs is relatively complex, it is difficult to accurately predict the families. Besides, the process of manually grouping traffic sign classes into several groups disrupts the inherent autonomous learning capacity of models. While human might intuitively take all speed limit signs as a group and all height restriction signs into another ones, the learning process of models based on neural networks may not inherently prioritize the feature similarity of two different speed limit signs over signs from different class groups. Consequently, this manual prior setting could indeed interfere with the training process.

In this paper, we discover that rather than using a cascading

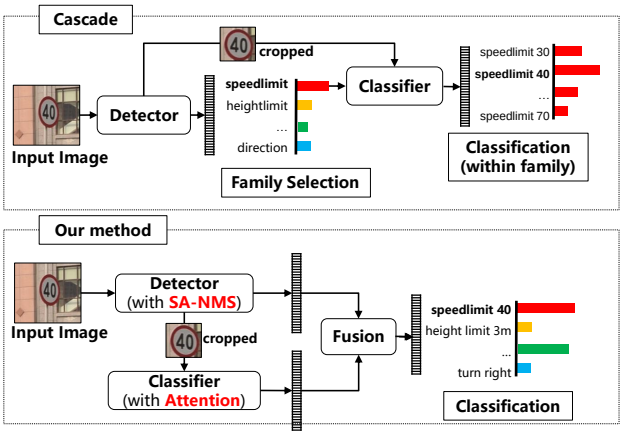


Fig. 1: Comparison of different two-stage traffic sign detection frameworks.

Zhishan Li, Yifan He are with Institute of Intelligence Science and Engineering, Shenzhen Polytechnic University, Shenzhen, China, 518055. (Email: zhishanli@zju.edu.cn; heyifan@reconova.com)

Hongxu Chen, Lei Xie are with State Key Laboratory of Industrial Control Technology and Institute of Cyber-systems and Control, Zhejiang University, Hangzhou 310027, China. Zhishan Li, Haoran Cai are with Huawei Technology Co., Ltd. (Email: 3180101444@zju.edu.cn; leix@iipc.zju.edu.cn; caihaoran1@huawei.com)

Battista Biggio, Fabio Roli are with the Department of Electrical and Electronic Engineering, University of Cagliari, Cagliari 09123, Italy. (Email: battista.biggio@unica.it; roli@unica.it)

Fig. 2: The red lines represents false detection for traffic sign. The heatmap area represents the probability of existence of traffic signs. The changement from blue to red indicates gradually higher probability.

method, it is better to directly predict the category results in the first and second-stages, and obtain the final results in a fusion method (detection categories $\leftrightarrows$ classification categories). The proposed fusion method possesses higher accuracy and utilizes the results of the second-stage with higher accuracy to revise category predictions in the first-stage detector. We use the first-stage detector to directly predict the detection results of each category, and crop out the predicted areas from the original images. These cropped pieces are then fed into a classification network (second-stage) to generate classification results. The final categories are fused through weighted summation or linear regression. From experimental results, we find that the prediction results of any stage model alone or cascading method cannot achieve the performance of our proposed fusion strategy.

Another aspect that needed further optimization is inference time. Traffic sign detection is a task requiring high real-time performance, yet most algorithms [11], [12], [14] are based on high-volume convolutional neural networks, such as Faster-RCNN [15], FPN [16] and Mask-RCNN [17], which require a huge amount of computation. Although some lightweight detectors [8] [18] possess high inference speed, their detection accuracy is relatively poor. Besides, resizing input images to a smaller resolution can improve the inference speed, but the objects will be smaller, which also causes a significant drop for overall detection performance. Small objects have always been the obstacle of object detection. For example, the image resolution of TT100K dataset [14] is 2048×2048, but the area of more than 40% of GroundTruths (GTs) is less than 32×32 pixels. To improve the detection ability of small targets, some scholars have proposed various optimization strategies [19] [10], but the improvement is still not enough. If we directly resize input images to a smaller resolution and utilize lightweight model to inference, there would be many false detection boxes for small objects.

To make the overall inference speed faster, we use low-resolution (608×608 or 640×640) images instead of high-resolution (2048×2048 or 1024×1024) as the inputs. However, the type of a large number of false prediction boxes is that one prediction box contains multiple traffic signs, as shown in Figure 2. In this detection demo, the red box covering two traffic signs is a false detection box. Besides, the background area between two traffic signs also has the probability of being recognized as foreground. Furthermore, this false detection cannot be filtered out by Non-Maximum Suppression (NMS) [20]. Due to the low-resolution input, the size of traffic signs is smaller. With the down-sampling mechanism of neural network structure, the feature distinction between adjacent traffic signs is not obvious. Even with human eyes, it is difficult to distinguish those two traffic signs. We establish a new type of IoU called Surrounding-Aware IoU (SAIoU) to describe the surrounding relationship more clearly. Then we propose a new post-processing method called Surrounding-Aware NMS (SA-NMS) to delete the prediction boxes covering multiple traffic signs. As shown in Figure 4, we insert SA-NMS to the first-stage detector and greatly reduce false detection boxes.

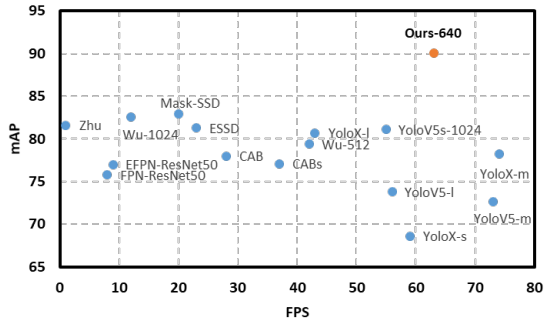


Fig. 3: Comparison with other detectors. The horizontal axis is inference FPS and the vertical axis is mAP.

In summary, our purpose is to achieve effective traffic sign detection for practical application, which means less inference time and high accuracy. To obtain higher FPS, we lower the resolution of input images to 608×608 and employ a lightweight framework. As shown in Figure 4, the first-stage model is based on YoloV5s [21] whose mAP on TT100K is 69.7 and the second-stage classification network is based on MobileNetV2. We insert attention mechanism on MobileNetV2 to improve the classification accuracy. Finally, we fuse the prediction results of the two stages and utilize the proposed SA-NMS post-processing method to increase the mAP to 89.7, with an inference speed of 65 FPS. To prove the generality of our overall framework, we also conduct experiments on SSD512 and ResNet18 [22]. Experimental results show the effectiveness of our method. A more intuitive comparison is shown in Figure 3. Our two-stage framework shows better performance in mAP and FPS than other generic detectors and traffic sign detectors. Finally, our main contribution can be summarized as follows:

- We propose a two-stage fusion strategy for traffic sign detection, which achieves higher detection accuracy than the improved single detector and two-stage cascading method.
- To make the overall inference speed faster, we use relatively low-resolution images and propose SA-NMS post-processing method to effectively filters out false detection

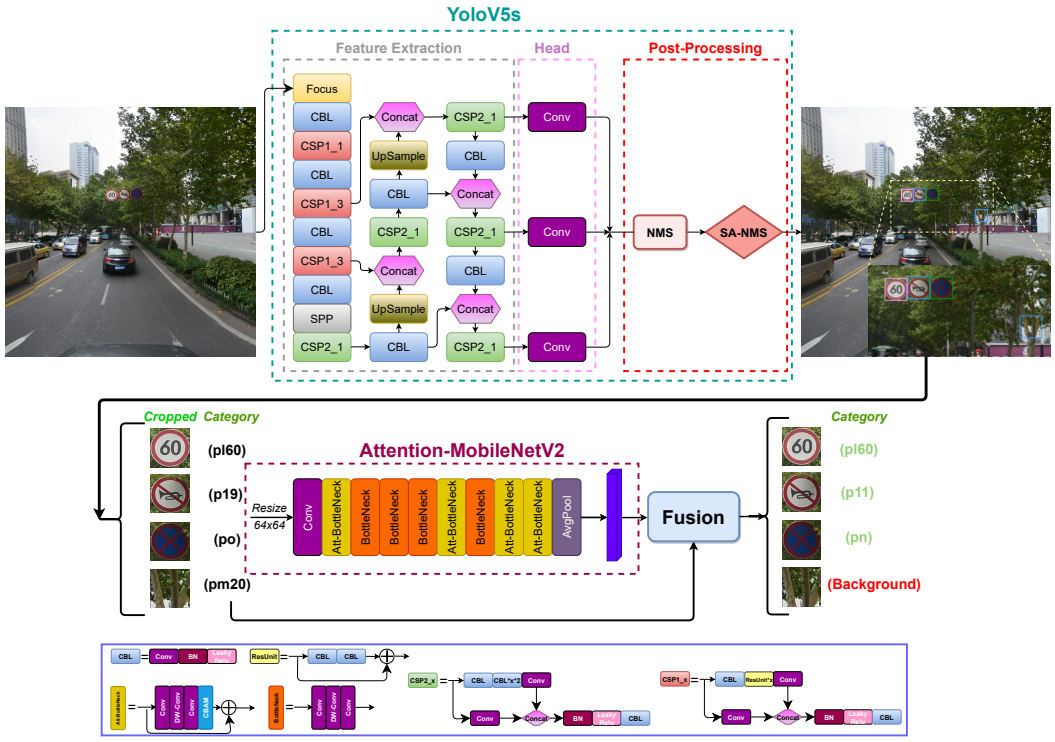


Fig. 4: Our proposed two-stage fusion framework for detection and recognition of traffic signs. Inside the rectangular box, Conv, DW-Conv, BN, and LeakyRelu represent Convolutional layer, Depthwise Convolutional layer, Batch Normalization layer, and LeakyRelu activation layer, respectively. Other modules, such as CBL, ResUnit, CSP2, etc., are all composed of upper basic modules according to the diagrams.

boxes.

- Compared with other traffic sign detectors, our proposed framework achieves higher detection accuracy with less inference time.
- Our framework is not limited to specific models. Other detection and classification models are also applicable to our framework.

2. Related Work

A. Traffic Sign Detectors

With the improvement of computing power, methods based on machine learning and deep neural networks have achieved great performance in traffic sign detection. As we illustrate in Introduction section, current traffic sign detectors can be divided into two types. The first type is improvement on single detectors. Shao *et al*. [23] proposed simplified Gabor Wavelets to achieve real-time traffic sign detection and recognition. Zhang *et al*. [24] proposed an end-to-end convolution network based on YoloV2 [25], which achieved great performance on the extended Chinese traffic sign dataset. In order to accurately detect small traffic signs, Liu *et al*. [26] presented a deconvolution region-based convolutional neural network (DR-CNN). Ahmed *et al*. [27] proposed Enhance-Net to improve the robustness of traffic sign detection under challenging weather conditions. Sun *et al*. [10] proposed a new detector called Mask-guided SSD, which integrates detection branch and segmentation branch to improve the accuracy of SSD. The others are two-stage cascading frameworks. Tabernik *et al*. [11] improved the accuracy of traffic sign detection based on improved Mask-RCNN [17] and data enhancement method. Serna *et al*. [12] proposed a multi-level network, which used Mask-RCNN [17] to detect the category groups and utilized a small CNN to subdivide the categories. Wu *et al*. [13] proposed a real-time traffic sign detector based on YoloV3. For those categories with less training data, Wu *et al*. proposed a data enhancement method to improve the diversity of training dataset. This optimization at the data level greatly improves the fitting ability of the original model. For digital types of traffic sign recognition, almost all detection frameworks belong to the two-stage type.

B. Post-Processing Methods

With regard to object detection frameworks based on convolutional neural networks, post-processing is an indispensable part. Whether anchor-based or anchor-free detectors, most of the prediction boxes acquired through detection head are repeated predictions. NMS [20] is the most widely used post-processing method which removes a large number of redundant detection boxes. For crowded scenarios, a common phenomenon is that IoU between different GTs is relatively high, and using traditional NMS is prone to eliminate those occluded objects. To address this issue, Bodla *et al*. [28] proposed Soft-NMS. Its principle is to reduce the confidence of those non-maximum detection boxes through penalty weights positively related to IoU, rather than discarding them completely. Later, scholars have successively proposed Softer-NMS [29] and

Adaptive-NMS [30], which are further optimization of post-processing. However, these optimizations have changed the original network structure, leading to a lack of generality.

3. Two-Stage Fusion Framework

A. Overall Construction

YoloV5 is the classical framework of Yolo series models [2], [25], [31]–[34]. The backbone of YoloV5 is based on two improved CSPNet [35]. In the detection neck part, YoloV5 configures a more robust feature fusion module by combining FPN [16] and PANet [36]. Through the detection head, we obtain three types of outputs corresponding to each anchor point, including detection box coordinates, probability of the existence of objects, and prediction confidence of each category. YoloV5s is the lightest model of the YoloV5 series. With the lowest model depth and width, the storage size of YoloV5s is only 15.1 MB in Linux OS. However, the disadvantage is that its feature extraction ability is not good enough. Especially for small traffic signs with less feature representations, its detection performance is relatively poor compared with other large-scale models.

The second-stage classifier is based on lightweight MobileNetV2 [18]. Under the preliminaries of depthwise separable convolution of MobileNet [37], MobileNetV2 borrows the idea of ResNet [22] and designs the structure of inverted residual block to reduce the feature degradation caused by ReLU. Besides, attention mechanism [38] [39] [40] have been demonstrated to enable networks to automatically learn the importance of each channel from global information through backpropagation. Then, the effective features are enhanced and the irrelevant features are suppressed. We introduce the Convolution Block Attention Module [41] (CBAM) into the Inverted Residual Block of MobileNetV2 to further improve the performance of the second-stage classifier. More detailed module composition is shown in Figure 5.

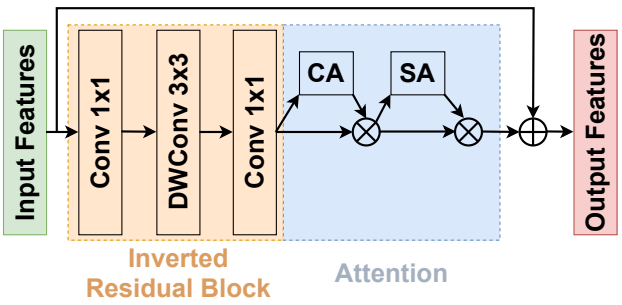


Fig. 5: Enhanced inverted residual block based on attention mechanism. CA indicates the channel attention module and SA represents the spatial attention module.

In CBAM, the channel attention and spatial attention can be described as follows:

$$F_c = \boldsymbol{\sigma}(MLP(AvgPool(F_{in})) + MLP(MaxPool(F_{in}))) \quad (1)$$

$$F_{out} = \boldsymbol{\sigma}(Conv7 \times 7([AvgPool(F_c); MaxPool(F_c)])) \quad (2)$$

where F_{in} and F_c indicate the input and output features of channel attention module, and F_{out} represent the output features of spatial attention block. $Conv7 \times 7$, MLP and $\boldsymbol{\sigma}$ represent Convolution layer with kernel size of 7×7, Multi-Layer Perception and Sigmoid function. Through the attention mechanism of two dimensions, we enhance the input features and improve the fitting ability of MobileNetV2 with little increase in calculation.

In addition, how to train the second-stage classifier is very important. We uniformly resize the cropped images according to the coordinates of prediction boxes in the first-stage as 64×64 to train the second classifier. To fuse the prediction results of the two-stage models, we keep the prediction outputs of classifier in the same shape as the category predictions of detector. We define the classifier labels according to the IoU between the position of detection boxes and GTs. If IoU is greater than 0.5, the label is the corresponding One-Hot vector. Otherwise, we define this object as background and set an All-Zero vector as the label. The second-stage label setting is shown in Figure 6.

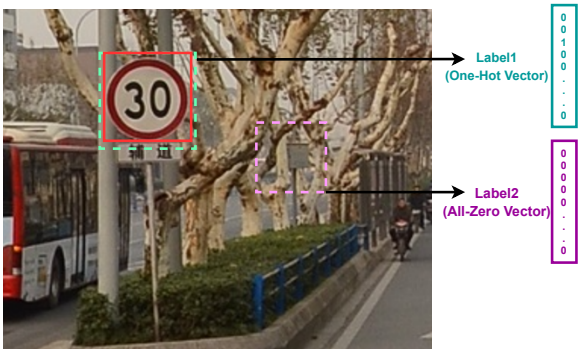


Fig. 6: Definition of labels for the second-stage classifier. Label1 is the One-Hot vector and Label2 is the All-Zero vector.The dashed line is the detection box for the first stage detector.

Because there are All-Zero vectors in the second-stage labels, we cannot take cross-entropy as classification loss, but use binary cross-entropy as the loss function of second-stage network, as shown in the following equation:

$$Loss_2 = \sum_{i=1}^{N_b} \sum_{j=1}^{N_c} \left(-y_{i,j} \times log(c_{i,j}^2) - (1 - y_{i,j}) \times log(1 - c_{i,j}^2)\right) \quad (3)$$

where N_c and N_b is the number of categories and prediction boxes, $y_{i,j}$ is the label of the j category of the i box and $c_{i,j}^2$ is the Sigmoid activated prediction output of the i box in the second-stage.

B. Model Fusion Strategy

Instead of the previous cascading methods, we propose a fusion strategy, which makes the prediction results of the two stages complement each other, so as to exceed the accuracy of any individual predictions. We propose two simple yet effective fusion strategies: one is the weighted summation and the other is obtained by linear regression.

The first method is the directly weighted fusion, which can be expressed as follows:

$$P_i^f = \lambda \times P_i^1 + (1 - \lambda) \times P_i^2 \qquad 0 \le \lambda \le 1 \quad (4)$$

where P_i^f, P_i^1 and P_i^2 represent the i box prediction of the first-stage, the second-stage and the final fusion respectively.

The weighted fusion method is straightforward. By selecting an appropriate λ of this equation, we change the proportion of different stages in the final fusion predictions. When λ is equal to 0, the category predictions completely depend on the second-stage classifier. On the contrary, the predictions of category completely depends on the detector in the first-stage when λ is equal to 1. Through experiments, we find that when λ is 0.4, the overall detection performance is the best, and the final result is determined by both stages.

The other fusion method is based on Multiple Linear Regression [42]. We build a linear regression model for the two stage prediction results, and take the prediction outputs of each category in the two-stage model as an independent variable. Since the prediction confidence of each category of the final result is a value greater than 0 and less than 1, we control the regression coefficient between 0 and 1 through the sigmoid function. This can be mathematically expressed as:

$$P_{i,j}^f = \sum_{n=1}^{2} W_j^n \times P_{i,j}^n = \sum_{n=1}^{2} \boldsymbol{\sigma}(\theta_j^n) \times P_{i,j}^n \tag{5}$$

where $P_{i,j}^f$ is the i box predicted value of the fusion result for j catgory. W_j^n and θ_j^n represent the j category regression coefficient of the n stage model.

The objective loss function is the mean square deviation, as shown below:

$$e_j(\theta) = \frac{1}{2} \sum_{i=1}^{N_b} (P_{i,j}^f - y_{i,j})^2 \tag{6}$$

We use Gradient Descent to make the predicted value $P_{i,j}^f$ approach label $y_{i,j}$. The partial derivative of the loss function is as follows:

$$\frac{\partial e_j(\theta)}{\partial \theta_j^n} = \sum_{i=1}^{N_b} (P_{i,j}^f - y_{i,j}) \times P_{i,j}^n \times \boldsymbol{\sigma}(\theta_j^n) \times (1 - \boldsymbol{\sigma}(\theta_j^n)) \tag{7}$$

The new θ_j^n is as follows:

$$\theta_j^n = \theta_j^n - \alpha \frac{\partial e_j(\theta)}{\partial \theta_j^n} \tag{8}$$

where α is the learning rate and all equations are based on one image.

By the Multiple Linear Regression, the regression coefficient of each category W_j^n is obtained by backpropagation and the fused results can be automatically acquired. Experiments show that both the above two methods effectively improve the accuracy of the overall framework, and significantly surpass the performance of either isolated stage.

C. Surrounding-Aware Post-Processing Method

To reduce the inference time, we resize original images to lower resolution as the inputs of the model and establish a lightweight detection framework. Under this circumstance, a large number of false detection boxes are as shown in Figure 2. In the detection area, each traffic sign corresponds to one detection box, but there is a large detection box surrounding all traffic signs. Moreover, this false detection cannot be filtered out by NMS due to not large IoU with any other detection box. This false detection box appears because of the relatively lower resolution input. For small traffic signs in the original resolution, the size of these objects under the new resolution is smaller. Therefore, convolution kernels fuse several targets feature together during sliding windows. As a result, the detector cannot distinct extracted features. Considering that there is no overlap between different traffic signs, we propose the SA-NMS post-processing algorithm.

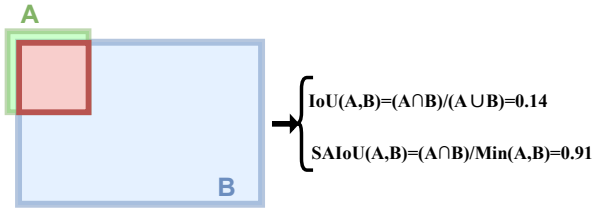


Fig. 7: Comparison of calculation methods between SAIoU and IoU.

We first define Surrounding-Aware IoU (SAIoU). As shown in Figure 7, the large box B almost surrounds the small box A, and the area proportion of its interaction area $(A \cap B)$ in the joint area $(A \cup B)$ is only 0.14. As a result, it's not convenient to filter out B through NMS because of the small IoU. However, the interaction area between box A and box B occupies a high proportion of the small one, so box B can be filtered out through this relationship. We define SAIoU as follows:

$$SAIoU(A, B) = \frac{A \cap B}{Min(A, B)} \tag{9}$$

Compared with the original IoU, SAIoU more clearly reflects this surrounding false detection. For comparison, $SAIoU(A, B)$ is 0.91 which is more obvious than $IoU(A, B)$. By setting the SAIoU threshold T_s, we conduct further post-processing called SA-NMS, and more detailed steps are shown in Algorithm 1. The core of SA-NMS is to use SAIoU to filter detection boxes. For detection box M, if there is a detection box $\mathbf{B}_i$ with smaller area but the SAIoU between M and $\mathbf{B}_i$ is larger than threshold T_s, the detection box M would be filtered out.

4. Experimental Results

A. Experiment Settings

Dataset. TT100K is a large-scale dataset of traffic sign detection and recognition, which was recorded by Tsinghua University and Tencent based on real road scenarios of Chinese cities. There are 9180 labeled images with 30000 traffic-sign instances in this dataset, including 6107 images for training and 3073 images for testing respectively. In addition, the image resolution of TT100K is 2048×2048. As shown in Table 1, more than 40% of the instances in this dataset are small objects, which makes TT100K a challenging benchmark. Besides, according to the definition of large $(96 \times 96 < area)$, medium$(32 \times 32 < area < 96 \times 96)$ and small $(area < 32 \times 32)$ in COCO benchmark [5], the proportion of large traffic signs in testset is only 7.6%, and the number of large

Algorithm 1 Surrounding-Aware NMS.

Input:
$\mathbf{B}$ is a set of detection boxes after NMS
$\mathbf{A}$ is a set of box area of detection boxes $\mathbf{B}$
T_s is the SAIoU threshold
Output:
$\mathbf{D}$ is a set of detection boxes after SA-NMS

Initialize $\mathbf{D}$ as an empty list: $\mathbf{D} \leftarrow \varnothing$;
while $\mathbf{B} \neq \varnothing$ **do**
  Acquire index with the largest area: $m \leftarrow argmax\mathbf{A}$;
  Acquire the box with the largest area: $M \leftarrow \mathbf{B}_m$;
  Initialize the status of M: $sur \leftarrow 0$;
  Remove M from $\mathbf{B}$ and $\mathbf{A}$:
  $\mathbf{B} \leftarrow \mathbf{B} - M$; $\mathbf{A} \leftarrow \mathbf{A} - \mathbf{A}_m$;
  for each b_i in $\mathbf{B}$ **do**
    if $SAIoU(M, b_i) >= T_s$ **then**
      Change the status of M: $sur \leftarrow 1$;
      break
    end if
  end for
  if $sur \neq 1$ **then**
    Add M to $\mathbf{D}$: $\mathbf{D} \leftarrow \mathbf{D} \cup M$;
  end if
end while
return $\mathbf{D}$;

instances in many categories is no more than 3 in the testset. For example, for the category 'pl20' and 'pm30', there is only one large target, which causes the detection performance of these categories to fluctuate greatly. Therefore, we attribute the targets defined as large and medium in the way of COCO to medium targets, as Table 1 shows.

TABLE 1: GTs distribution of different object sizes under different specified methods in the TT100K testset.

Definition	Object Size	Instances	proportion
COCO	small($area < 32 \times 32$)	3281	43.5%
	medium($32 \times 32 < area < 96 \times 96$)	3683	48.9%
	large($96 \times 96 < area$)	572	7.6%
Ours	small($area < 32 \times 32$)	3281	43.5%
	medium($32 \times 32 < area$)	4255	56.5%

TT100K contains more than 200 traffic sign categories, which is divided into three types according to colors: yellow for warning, red for prohibitory, and blue for mandatory. In order to define categories more simply, TT100K dataset describes them in the format of combination of type and serial number. For example, 'w64' is a warning sign indicating that there may be animals on the road, and '64' is its serial number in warning sign list. 'pl40' is the abbreviation of 40 km/h speed limit, which is the type of prohibitory traffic sign. Although there are more than 200 traffic sign categories, the training samples of most categories are insufficient. Following the original analysis method [14], we also ignore the categories with training data less than 100, and the number of categories analyzed is 45.

Training Details. The experiments are carried out under Pytorch deep learning framework. Our hardware mainly includes 2 × NVIDIA TITAN XP GPUs and 1 × Inter Core I7-9700K CPU. Each stage of our framework is trained separately. In the first-stage training process, we set the batch size as 32 and the total epochs as 300. For training the second-stage network, the batch size of cropped images is 160 and the total epochs are 150. We use the linear learning rate setting for backpropagation, as follows:

$$lr_{epoch} = lr_0 \times (1 - \frac{epoch}{TotalEpoch}) \tag{10}$$

where lr_{epoch} is the learning rate under current $epoch$ and lr_0 is the initial learning rate which we set as 0.01.

Evaluation Metrics. The most commonly used metric in object detection is Average Precision (AP) whose value is equal to the area under the Precision-Recall (PR) curve and mAP is the most important indicator for comparing detection capabilities, which can be calculated as:

$$AP = \int_0^1 PdR \tag{11}$$

$$mAP = \frac{1}{N_c} \sum_{j=0}^{N_c} AP_j \tag{12}$$

where P and R represent precision and recall respectively. Their definition is as follows:

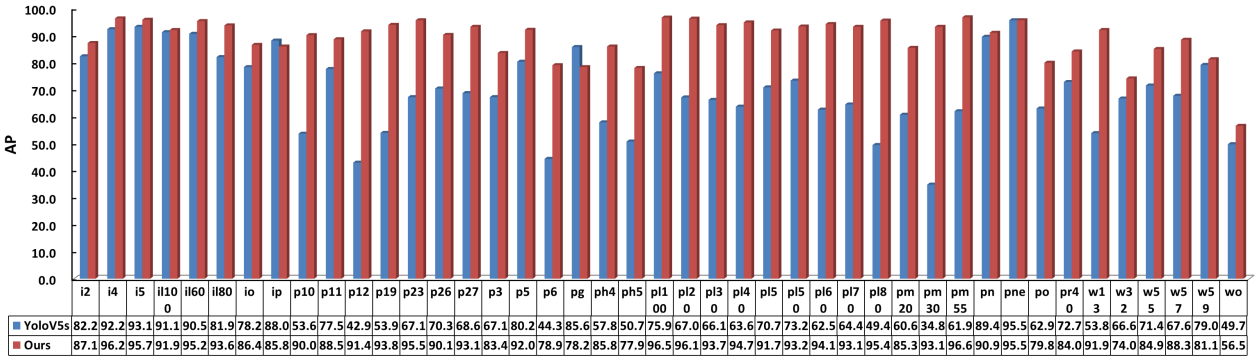
$$P = \frac{TP}{TP + FP} \tag{13}$$

$$R = \frac{TP}{N_{GT}} \tag{14}$$

where TP, FP, N_{GT} represent the number of truly predicted boxes, false positive predicted boxes, and number of GTs. As previous analysis method [14], we define the prediction box whose IoU with GT is greater than 0.5 as TP. In object detection tasks, mAP is set as the main criteria for comparison. In addition, we also show AP_s (Average Precision for small objects) and AP_m (Average Precision for medium objects) to make the analysis more thorough.

B. Ablation Experiments

Analysis of different fusion methods. We conduct ablation experiments about the proposed fusion methods, as shown in Table 2. The input resolution is 608×608. When λ is 1.0, the prediction of the category is completely determined by the first-stage YoloV5s, and the mAP is only 69.7. With the decrease of λ, the proportion of the second branch in the prediction results gradually increases, and the mAP also gradually increases. Until λ is 0.4, the mAP of this framework reaches the maximum of 81.1. However, when λ is 0.0, the prediction results are completely determined by the second-stage classification network, the overall mAP decreases to 75.0. As shown in Table 2, the accuracy of detection with YoloV5s only is the lowest. By constructing the second-stage network and cropping out the box-area images of the YoloV5s detection for reclassification, the mAP is improved to 75.0. However, the performance of solely relying on any stage is not as good as that of integrating the two stages. By simply weighted summing the two-stage outputs, our framework boosts the mAP from 69.7 to 81.1. In addition, we



	i2	i4	i5	il100	il60	il80	io	ip	p10	p11	p12	p19	p23	p26	p27	p3	p5	p6	pg	ph4	ph5	pl100	pl20	pl30	pl40	pl5	pl50	pl60	pl70	pl80	pm20	pm30	pm55	pn	pne	po	pr40	w13	w32	w55	w57	w59	wo
YoloV5s	82.2	92.2	93.1	91.1	90.5	81.9	78.2	88.0	53.6	77.5	42.9	53.9	67.1	70.3	68.6	67.1	80.2	44.3	85.6	57.8	50.7	75.9	67.0	66.1	63.6	70.7	73.2	62.5	64.4	49.4	60.6	34.8	61.9	89.4	95.5	62.9	72.7	53.8	66.6	71.4	67.6	79.0	49.7
Ours	87.1	96.2	95.7	91.9	95.2	93.6	86.4	85.8	90.0	88.5	91.4	93.8	95.5	90.1	93.1	83.4	92.0	78.9	78.2	85.8	77.9	96.5	96.1	93.7	94.7	91.7	93.2	94.1	93.1	95.4	85.3	93.1	96.6	90.9	95.5	79.8	84.0	91.9	74.0	84.9	88.3	81.1	56.5

Fig. 8: Comparison between our method and YoloV5s under each category of AP. The blue bar represents the detection performance of YoloV5s, and the red bar represents the AP of our framework.

also carry out experiment based on linear regression fusion. It can be found that the overall mAP has been further improved to 82.4. Through the above two fusion methods, we confirm the ability to integrate the two-stage results to improve overall performance.

TABLE 2: Analysis of different fusion methods.

Fusion Strategy		mAP	AP_s	AP_m
BaseLine (YoloV5s)		69.7	69.1	70.9
Weighted	λ=0.0*	75.0	74.0	77.7
	λ=0.2	80.1	77.7	82.9
	λ=0.4	**81.1**	**79.0**	**83.6**
	λ=0.6	77.7	77.6	78.7
	λ=0.8	73.4	73.0	74.5
	λ=1.0*	69.7	69.1	70.9
Linear Regression		**82.4**	**79.0**	**84.2**

* $\lambda = 0.0$ means that final category predictions are only the output of second-stage classifier.
* $\lambda = 1.0$ means that final category predictions are only the output of first-stage detector.

Analysis of different post-processing thresholds. We analyze the influence of the hyper-parameter T_s of our proposed SA-NMS algorithm on the overall performance, as shown in Table 3. The baseline model is the best result obtained by our weighted fusion method. With NMS only, the mAP of this framework is 81.1, as Table 2 shows. Since there are a large number of false detection boxes like B in Figure 7, we attach the SA-NMS post-processing algorithm to achieve a great increase. When T_s is equal to 0.8, our framework reaches the best mAP of 89.7. Since this method is made after NMS and the number of detection boxes is not large, the calculation cost is negligible.

TABLE 3: Analysis of different T_s in our post-processing method.

Post-Processing Method		mAP	AP_s	AP_m
NMS		81.1	79.0	84.2
SA-NMS	T_s=0.2	87.9	82.0	90.8
	T_s=0.4	88.4	83.2	91.0
	T_s=0.6	89.2	84.1	91.7
	T_s=0.8	**89.7**	**84.6**	**92.1**
	T_s=1.0	81.1	79.0	84.2

Step by step optimization for the overall performance. As shown in Table 4, we intuitively show the mAP improvement brought by the optimization strategies. When we directly utilize the first-stage detector to separately predict 45 classes of the dataset, the mAP is only 69.7 while the FPS reaches 144 due to the lightweight model structure and low input resolution of 608×608. The best performance of our proposed framework reaches 89.7 mAP with the two-stage weighted fusion method and the SA-NMS post-processing algorithm, which is 29% higher than that of the baseline model. A more detailed AP comparison is shown in Figure 8 and the AP of our method is higher than baseline in each category. Although the inference FPS is reduced to 65, the overall performance is still better than other latest generic detectors and traffic sign detectors, as Table 7 and Table 6 shows.

C. Comparative Experiments

Comparison between our method and YoloV5s with larger input resolution. Our purpose is to achieve accurate and fast detection performance, so we design such a two-stage lightweight framework, and further improve the detection speed through the low-resolution input. Comparing with high-resolution input, as shown in Table 5, we verify the significance of this idea. As we illustrated in the ablation experiments, the mAP of YoloV5s at the low resolution of 608×608 is only 69.7. In terms of object detection tasks, the higher resolution not only means the improvement in detection accuracy, but also accompanies a greater amount of calculation. With the increase of input resolution, the overall mAP of YoloV5s is significantly improved, but the FPS also decreased sharply. When the input resolution is 1024×1024, the mAP of YoloV5s is increased to 81.1, but FPS is reduced to 55. In contrast, our method achieves better mAP and higher FPS on low-resolution inputs at 608×608 and 640×640. With this comparison, we verify the effectiveness of the proposed idea. Under low-resolution inputs, we acquire better detection performance than large-scale inputs in inference speed and accuracy by proposing the two-stage detection framework.

Comparison with other traffic sign detectors. We also compare our detection performance with other methods on TT100K dataset, as shown in Table 6. For fair comparison, we compare the detectors without loading any pre-trained weights,

TABLE 4: Step by step optimization for the overall performance

Model	First only*	Second only⋆	Weighted Fusion	Linear Regression	SA-NMS	mAP	AP_s	AP_m	FPS
Our Framework	✓					69.7	69.1	70.9	144
		✓				75.0	74.0	77.7	65
			✓			81.1	79.0	83.6	65
				✓		82.4	79.0	84.2	65
			✓		✓	**89.7**	**84.6**	92.1	65
				✓	✓	88.5	82.6	**92.9**	65

* means that the final category predictions are only determined by the first-stage detector (YoloV5s).
⋆ means that final category predictions are only determined by the second-stage classifier (Attention-MobileNetV2).

TABLE 5: Comparison between our method and YoloV5s with large input resolution

Method	Input Resolution	mAP	AP_s	AP_m	FPS
YoloV5s	608*608	69.7	69.1	70.9	144
	800*800	75.2	74.6	77.0	90
	1024*1024	81.1	78.3	82.4	55
Ours	608*608	**89.7**	**84.6**	**92.1**	65
	640*640	**90.1**	**84.9**	**94.4**	63

TABLE 6: Comparison with other traffic sign detectors

Model	Resolution	mAP	FPS
YoloV4-tiny [43]	608×608	52.1	87
Wang et al. [8]	608×608	65.1	95
EFPN-ResNet50 [19]	1400×1400	77.0	9
CABs [44]	512×512	77.1	37
EFPN-ResNet101 [19]	1400×1400	77.6	5
CAB [44]	512×512	78.0	28
Wu et al. [13]	512×512	79.4	42
ESSD [9]	512×512	81.3	23
Zhu et al. [14]	2048×2048	81.6	1
VATSD [45]	608×608	82.8	22
Wu et al. [13]	1024×1024	82.6	12
Mask-SSD [10]	640×640	82.9	20
FE-YOLOv5 [46]	640×640	82.6	-
Gao Improved Faster-RCNN [47]	1333×800	86.5	8
AD-RCNN Lite [48]	1024×1024	86.3	16
Our Method	608×608	**89.7**	65
	640×640	**90.1**	63

and do not consider the results of those using multi-scale inputs for evaluation. The best performance is achieved by AD-RCNN Lite [48], which is 86.3 mAP and 16 FPS. Compared with AD-RCNN Lite and other traffic sign detectors, our algorithm achieves 90.1 mAP and 63 FPS under 640×640 input resolution, which is more superior both in accuracy and inference speed.

Comparison with other generic detectors. We train and test other generic object detection networks in TT100K dataset, and the results are shown in Table 7. For fair comparison, we keep the input resolution consistent for EfficientDet [3], YoloV3 [2], YoloV5 and YoloX [49]. For SSD [1], M2Det [50] and FPN [16], we keep the input resolution consistent with the original network. Through comparative experiments, it is apparent that the YoloX-l achieves the best mAP under the input resolution of 608×608 while the inference speed is just 43 FPS. The mAP of our method is much higher than that of other generic detectors. Besides, our approach also surpasses most models in inference speed.

TABLE 7: Comparison with other generic object detectors.

Model	Input Resolution	mAP	FPS
MobileNetV2-SSD [18]	512×512	32.0	44
M2Det [50]	512×512	46.6	12
EfficientDet-D0 [3]	608×608	57.9	26
RetinaNet [51]	1000×800	59.0	19
YoloV3 [2]	608×608	61.9	27
YoloX-s [49]	608×608	68.6	59
YoloV5s [21]	608×608	69.7	144
YoloV5m [21]	608×608	72.6	73
YoloV5l [21]	608×608	73.8	56
FPN-ResNet101 [16]	1333×800	75.5	5
FPN-ResNet50 [16]	1333×800	75.8	8
YoloX-m [49]	608×608	78.2	74
YoloX-l [49]	608×608	80.7	43
YoloV7 [52]	608×608	72.7	161
Our Method	608×608	**89.7**	65

D. Discussions

Why is the second-stage classification network more accurate than the first-stage detector? The amount of computation of YoloV5s in the first-stage is almost the same as that of MobileNetV2 + CBAM in the second-stage, but the mAP relying solely on the second-stage reaches 75.0, which is higher than that of relying solely on the first-stage. The reason is that the detection task is to find the coordinates of the objects and get the categories on the whole image. Due to the large resolution of the original images, most of the features belong to 'background', and the proportion of foreground features is relatively small. Therefore, it is difficult to accurately judge the objects classes because of the uneven proportion of features. Compared with object detection tasks, image classification is relatively simple. When we cropped the object images according to the first-stage outputs, convolution neural networks only need to judge which categories these cropped images belong to. For the cropped images with large background areas, the extracted background features can be directly classified as background. For those images that belong to foreground, the background area is small, so they can be easily identified as object classes. However, it is not the best strategy to completely rely on the second-stage outputs. Although the accuracy of the first-stage is relatively poor, the result of integrating the two stages is higher than that predicted by any stage alone.

Generality of our proposed framework. This framework is not only limited to YoloV5s and MobileNetV2, but also applicable to other models to achieve higher detection performance on traffic signs. We utilize the classic models, e.g.

TABLE 8: Detection performance of our framework with SSD and ResNet18

Method		mAP	Improvement
SSD(baseline)		64.1	-
+ResNet18 Classifier		73.9	↑9.8
+Weighted Fusion	λ=0.0	73.9	↑9.8
	λ=0.2	80.6	↑16.5
	λ=0.4	**81.3**	↑**17.2**
	λ=0.6	78.9	↑14.8
	λ=0.8	74.4	↑10.3
	λ=1.0	64.1	-
+SA-NMS		**84.6**	↑**20.5**

TABLE 9: Detection performance of our framework with YoloV7 and Attention-MobileNetV2

Method		mAP	Improvement
YoloV7(baseline)		72.7	-
+Attention-MobileNetV2 Classifier		76.8	↑4.1
+Weighted Fusion	λ=0.0	76.8	↑4.1
	λ=0.2	80.2	↑9.5
	λ=0.4	**85.4**	↑**12.7**
	λ=0.6	83.1	↑10.4
	λ=0.8	76.2	↑3.5
	λ=1.0	72.7	-
+SA-NMS		**92.0**	↑**19.3**

SSD512 as the first-stage detector and ResNet18 as the second-stage classifier, to construct this framework. The results are as shown in Table 8. The mAP using SSD512 for the first-stage is only 64.1. Without any fusion method, using ResNet18 as the second-stage classifier, the mAP improves to 73.9. After we fuse the two stage results, the mAP is increased to 81.3, which is much better than any individual predictions. In the process of weighted fusion, the change of overall accuracy with parameters λ is consistent with the previous YoloV5s and MobileNetV2 in Table 2. With the insertion of SA-NMS, we acquire the maximum 84.6 mAP. Compared to the original SSD512 with only 64.1 mAP, our framework improves the overall accuracy to 84.6 mAP. It's an improvement of 20.5 mAP. Therefore, although deploying some classical models with less powerful detection performance, we still acquire a more obvious improvement with our framework. Besides, YoloV7 is a relatively new version to the Yolo series. However, after deploying it for retraining and testing on the TT100K dataset, it did not achieve a satisfactory level of accuracy. We apply the strategies proposed in this paper to improve the performance of YoloV7. The experimental results are as shown in Table 9. Notably, our method continues to enhance accuracy, achieving an impressive mAP of 92.0. Our proposed framework not only yields noticeable accuracy enhancements on previous models such as YoloV5 and SSD512 but also substantially elevates accuracy on new generic object detector YoloV7. This reflects the generality and effectiveness of our proposed methods.

Comparison with cascading framework. To compare our fusion method with previous cascading approach, we conduct comparative experiments with the same detector and classifier. According to colors, we divide 45 types of traffic signs into 3 groups and each group has 15 types traffic signs. We use the first stage YoloV5s to identify the category group, then use the second stage classification network to recognize the subcategory index, and finally output the category prediction by combining the results of the two stages. The experimental results are as shown in Table 10. It's apparently that although the cascading framework possesses higher accuracy compared to solely rely on classifier, the overall performance is still inferior to the method of proposed two-stage fusion. This also verifies that our framework possesses better robustness.

TABLE 10: Comparison of different framework

Method		mAP	Improve
YoloV5(baseline)		69.7	-
Framework	Only classifier*	75.0	↑5.3
	Cascade⋆	77.4	↑7.7
	Weighted Fusion	81.1	↑11.4
	Linear Regression	**82.4**	↑**12.7**

* means that final category predictions are only determined by the second stage classifier.
⋆ means that final category predictions are formed by cascading the category group in the first stage detector and the subcategories in the second stage classifier.

E. Detection Demos

Visualization of step-by-step optimization process. In order to show the effectiveness of our approach more intuitively, we show the visualization results of several images at different stages, as shown in Figure 9. We enlarge the area with the potential detected objects to show more clearly. For the first row images, the false detection box covers the positive detection box, and the fusion results of the two stages still cannot recognize the false detection box as 'background'. After inserting SA-NMS post-processing algorithm, the false detection box is filtered out. For the second row images whose detection demos of the second stage exist false detection boxes, we finally acquire the non-false detection boxes after fusing the two-stage results. The step-by-step detection effect of our whole framework is clearly shown in the third row images. Through our fusion method and SA-NMS, the two false detection boxes in original detection image are filtered out.

Comparison in challenging scenarios. There are also some challenging scenarios in TT100K dataset, such as blur, occlusion, distortion and dilapidation. We make a further comparison, as shown in Figure 10. For the scenarios with blur and occlusion, the original YoloV5s is interfered and cannot detect the traffic signs inside the images. In contrast, whether there are blurred warning signs or prohibited traffic signs covered by trees, our framework recognizes them and predicts the correct categories. For traffic signs with distortion or dilapidation, the original YoloV5s predicts the location of traffic signs, but the categories predictions are wrong. For comparison, our framework obtains the accurate categories. In addition, this also reflects that the robustness of the original detector is relatively poor. It cannot accurately identify traffic signs if there is distortion or dilapidation. For comparison, our method possesses stronger robustness to obtain accurate predictions.

Fig. 9: Detection demo at different stages of our framework. The first row images reflect the improvement through the SA-NMS post-processing method. All the second, third and fifth rows images all show the effect of the two-stage fusion method. The fourth row images indicate the detection improvement of the whole framework.

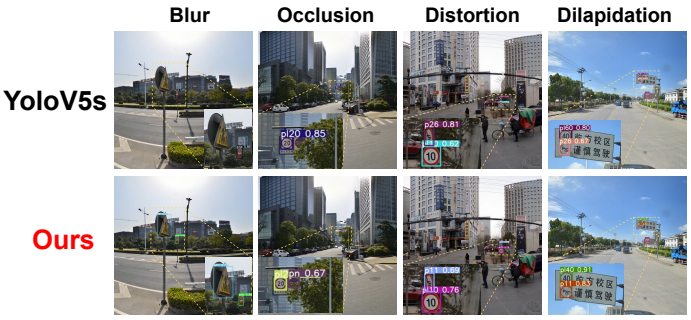


Fig. 10: Comparison of detection performance between YoloV5s and our method in challenging scenarios.

5. Conclusions

In this paper, we propose a lightweight two-stage fusion object detection framework for detection and recognition of traffic signs. By fusing the two-stage outputs, our method achieves high mAP which exceeds the latest generic detectors and other traffic sign detection algorithms. We further improve the detection performance by inserting SA-NMS post-processing method. Our framework consists of simple and lightweight structures and the input resolution is relatively lower, but it achieves great performance both in accuracy and inference speed. Experimental results show the efficiency of our framework.

In future, we will continue to propose a more general traffic sign detection algorithm to promote the development of intelligent transport systems.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable feedback and suggestions. This work was supported by Jianbing Lingyan Foundation of Zhejiang Province, P.R. China (Grant No. 2023C01022). Zhishan Li has graduated from Zhejiang University and joined Huawei Technologies Co., Ltd. We want to thank colleagues from Huawei for their guidance and suggestions.

References

[1] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[2] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[3] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.

[4] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, Jan. 2015.

[5] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[6] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[7] S. Shao, Z. Zhao, B. Li, T. Xiao, G. Yu, X. Zhang, and J. Sun, "Crowdhuman: A benchmark for detecting human in a crowd," *arXiv preprint arXiv:1805.00123*, 2018.

[8] J. Wang, Y. Chen, M. Gao, and Z. Dong, "Improved yolov5 network for real-time multi-scale traffic sign detection," *arXiv preprint arXiv:2112.08782*, 2021.

[9] C. Sun, M. Wen, K. Zhang, P. Meng, and R. Cui, "Traffic sign detection algorithm based on feature expression enhancement," *Multimedia Tools and Applications*, vol. 80, no. 25, pp. 33 593–33 614, 2021.

[10] C. Sun, Y. Ai, S. Wang, and W. Zhang, "Mask-guided ssd for small-object detection," *Applied Intelligence*, vol. 51, pp. 3311–3322, 2021.

[11] D. Tabernik and D. Skočaj, "Deep learning for large-scale traffic-sign detection and recognition," *IEEE transactions on intelligent transportation systems*, vol. 21, no. 4, pp. 1427–1440, 2019.

[12] C. G. Serna and Y. Ruichek, "Traffic signs detection and classification for european urban environments," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 10, pp. 4388–4399, 2019.

[13] Y. Wu, Z. Li, Y. Chen, K. Nai, and J. Yuan, "Real-time traffic sign detection and classification towards real traffic scene," *Multimedia Tools and Applications*, vol. 79, no. 25, pp. 18 201–18 219, 2020.

[14] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2110–2118.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[16] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[19] C. Deng, M. Wang, L. Liu, Y. Liu, and Y. Jiang, "Extended feature pyramid network for small object detection," *IEEE Transactions on Multimedia*, vol. 24, pp. 1968–1979, 2021.

[20] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders, "Segmentation as selective search for object recognition," in *2011 international conference on computer vision*. IEEE, 2011, pp. 1879–1886.

[21] Ultralytics, "Yolov5: 5x faster than yolov4, state-of-the-art object detection at 57 fps," https://github.com/ultralytics/yolov5, 2020.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[23] F. Shao, X. Wang, F. Meng, T. Rui, D. Wang, and J. Tang, "Real-time traffic sign detection and recognition method based on simplified gabor wavelets and cnns," *Sensors*, vol. 18, no. 10, p. 3192, 2018.

[24] J. Zhang, M. Huang, X. Jin, and X. Li, "A real-time chinese traffic sign detection algorithm based on modified yolov2," *Algorithms*, vol. 10, no. 4, p. 127, 2017.

[25] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[26] Z. Liu, D. Li, S. S. Ge, and F. Tian, "Small traffic sign detection from large image," *Applied Intelligence*, vol. 50, pp. 1–13, 2020.

[27] S. Ahmed, U. Kamal, and M. K. Hasan, "Dfr-tsd: A deep learning based framework for robust traffic sign detection under challenging weather conditions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5150–5162, 2021.

[28] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-nms–improving object detection with one line of code," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5561–5569.

[29] Y. He, X. Zhang, M. Savvides, and K. Kitani, "Softer-nms: Rethinking bounding box regression for accurate object detection," *arXiv preprint arXiv:1809.08545*, 2018.

[30] S. Liu, D. Huang, and Y. Wang, "Adaptive nms: Refining pedestrian detection in a crowd," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6459–6468.

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[32] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[33] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie *et al.*, "Yolov6: A single-stage object detection framework for industrial applications," *arXiv preprint arXiv:2209.02976*, 2022.

[34] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint arXiv:2207.02696*, 2022.

[35] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "Cspnet: A new backbone that can enhance learning capability of cnn," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 390–391.

[36] H. Li, P. Xiong, J. An, and L. Wang, "Pyramid attention network for semantic segmentation," *arXiv preprint arXiv:1805.10180*, 2018.

[37] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[38] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[39] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 510–519.

[40] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3146–3154.

[41] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision*, 2018, pp. 3–19.

[42] D. A. Freedman, *Statistical models: theory and practice*. cambridge university press, 2009.

[43] L. Wang, K. Zhou, A. Chu, G. Wang, and L. Wang, "An improved light-weight traffic sign recognition algorithm based on yolov4-tiny," *IEEE Access*, vol. 9, pp. 124 963–124 971, 2021.

[44] L. Cui, P. Lv, X. Jiang, Z. Gao, B. Zhou, L. Zhang, L. Shao, and M. Xu, "Context-aware block net for small object detection," *IEEE Transactions on cybernetics*, vol. 52, no. 4, pp. 2300–2313, 2020.

[45] J. Wang, Y. Chen, X. Ji, Z. Dong, M. Gao, and C. S. Lai, "Vehicle-mounted adaptive traffic sign detector for small-sized signs in multiple working conditions," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[46] M. Wang, W. Yang, L. Wang, D. Chen, F. Wei, H. KeZiErBieKe, and Y. Liao, "Fe-yolov5: Feature enhancement network based on yolov5 for small object detection," *Journal of Visual Communication and Image Representation*, vol. 90, p. 103752, 2023.

[47] X. Gao, L. Chen, K. Wang, X. Xiong, H. Wang, and Y. Li, "Improved traffic sign detection algorithm based on faster r-cnn," *Applied Sciences*, vol. 12, no. 18, p. 8948, 2022.

[48] Z. Ou, Z. Wang, F. Xiao, B. Xiong, H. Zhang, M. Song, Y. Zheng, and P. Hui, "Ad-rcnn: Adaptive dynamic neural network for small object detection," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4226–4238, 2022.

[49] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[50] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, "M2det: A single-shot object detector based on multi-level feature pyramid network," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 9259–9266.

[51] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[52] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7464–7475.