IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received August 17, 2021, accepted August 30, 2021, date of publication September 1, 2021, date of current version September 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3109606

A Traffic-Sign Detection Algorithm Based on Improved Sparse R-cnn

JINGHAO CAO, (Member, IEEE), JUNJU ZHANG, AND XIN JIN
School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing, Jiangsu 210094, China
Corresponding author: Junju Zhang (junjuzhang@njust.edu.cn)

ABSTRACT Automatic traffic-sign detection is a hot topic in computer vision and one of the critical technologies of intelligent transportation. The Transformer structure has recently become a research hotspot due to its excellent performance. We hope to apply this structure to the design of traffic sign detection algorithms. Therefore, we make some improvements to Sparse R-cnn, a neural network model inspired by Transformer. Sparse R-cnn is a novel model, and its core idea is to replace hundreds of thousands of candidate anchors in the RPN network with a small set of proposal boxes. The experiments in our paper proved that the performance of the Sparse R-cnn model is better than other existing general object detection models. Based on the original Sparse R-cnn inspiration, an improved Sparse R-cnn model is proposed. First, a novel backbone for the task of traffic-sign detection is proposed. Multi-scale fusion structure is the essential method of improving the algorithm for small target detection, so improving the multi-scale capability of the backbone is a required method for designing traffic sign detection. So, we made further improvements to the existing backbone ResNest. We enhanced the multi-scale representation ability of the backbone by constructing hierarchical residual-like connections within each single radix block in the original ResNest. We call the improved backbone Res2Nest. The novel backbone proposed by us shows better performance without introducing excessive computational costs to the model. In addition, the attention mechanism is also an effective method to improve the detection of traffic signs, so we set up a branch network for recalibrating the channel feature response adaptively through the Global Average Pooling (GAP) operation and a fully connected layer. It can also be seen as the implementation of the cross-channel self-attention mechanism. After experiments by TT100K dataset, our method would attain a better accuracy and robustness.

INDEX TERMS Deep learning, object detection, traffic-sign detection, self-attention mechanism, improved Sparse R-cnn.

I. INTRODUCTION

In recent years, driving assistance systems and autonomous vehicles have been widely used. Correctly detecting and recognizing traffic signs in the field of vision can help drivers or autonomous vehicles effectively reduce driving risks. Therefore, automatic-traffic sign detection and recognition has become a hot topic in computer vision. Nevertheless, it remains challenging to detect and recognize traffic signs by the computer in the real world due to the unstable features of traffic signs in different occasions, such as self-damage, viewing angle changes, and bad weather. Moreover, the architecture of the traffic sign detection model is required to be as efficient and straightforward as possible because the promptness of the algorithm is essential for practical applications.

For researchers, there are several technical challenges in achieving automatic traffic sign detection and recognition. The first small object detection model based on deep learning has the problem of data imbalance, for example, the imbalance of samples in the dataset and the imbalance of positive and negative boxes in RPN (Region Proposal Network); compared with the feature map obtained by a shallower layer, the deeper one contains richer semantic information, but relatively lacks detailed features especially for small objects [1].

Additionally, the performance of the algorithm can be affected by multiple factors such as bad weather, the damage to the traffic sign itself, and the noise of the sensors.

Aiming at these problems, a traffic sign detection system based on improved Sparse R-cnn (Sparse Region-based convolutional network) [2] is proposed. Our detection model

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

accomplishes comparable detection and classification accuracy with state-of-the-art methods. Our model mainly contains three modules: a ResNest (Split-Attention Networks) [3] backbone network, an improved DIIH (Dynamic Instance Interactive Head) [2], and two task-specific prediction layers.

The Transformer structure has recently become a research hotspot due to its excellent performance. We hope to apply this structure to the design of traffic sign detection algorithms. Therefore, we make some improvements to Sparse R-cnn, a neural network model inspired by Transformer. Sparse R-cnn is a purely sparse method for object detection. It uses a series of learnable proposal boxes and features to replace the thousands of candidates generated by traditional RPN network in a two-stage(dense-to-sparse) model, such as Faster R-cnn (Faster Region-based convolutional network) [4] and Fast R-cnn (Fast Region-based convolutional network) [5].

The contributions can be summarized as follows:

We believe that the key to designing a traffic sign detection algorithm is to improve the small object detection ability of the algorithm as much as possible. Therefore, our idea is to improve the multi-scale capabilities of the backbone and add the attention mechanism to the RoI (Region of Interest) extraction process. Based on this idea, our improvements to Sparse R-cnn are as follows.

First, we propose a novel backbone for the task of traffic-sign detection. We made further improvements to the existing backbone ResNest. ResNest consists of a series of Split-Attention blocks that enables attention across feature-map groups, and these blocks are stacked in ResNet-Style. ResNest preserves the overall ResNet (Residual Network) [6] structure without incurring the high computational cost. We enhanced the multi-scale representation ability of the backbone by constructing hierarchical residual-like connections within each single radix block in the original ResNest. We call the improved backbone Res2Nest. The novel backbone proposed by us shows better performance without introducing high computational costs to the model.

Second, we design an improved DIIH (Dynamic Instance Interactive Head) as the RoI head of our model. We set up a branch network to recalibrate the channel feature response adaptively through the GAP (Global Average Pooling) [7] operation and a fully connected layer. The improved DIIH has better performance without incurring excessive calculation costs.

The model proposed in this paper shows a significant superiority compared to the state-of-the-art. The rest of this paper is organized as follows. Section 2 briefly reviews the related work for generic object detection and traffic sign detection. Section 3 presents the proposed traffic signs detection model based on a deep neural network. Section 4 discusses the results of our experiments and ablation research. Section 5 concludes our work.

II. RELATED WORK

Traditional traffic signs detection technology primarily relies on manually extracting image features of various attributes, such as color information, edge detection, and geometrical shapes.

Color-based methods like HSI(Hue-Saturation-Intensity) [8] and HCL(Hue-Chroma-Luminance) [9] mostly rely on threshold-based segmentation of traffic sign objects in a color space. However, one major disadvantage of these methods is that illumination change in the real world can easily affect algorithm performance. In order to solve the above problem, some shape-based approaches have been carried out, like SIFT (Scale Invariant Feature Transform) [10], SURF (Speeded-Up Robust Features) [11], HOG (Histogram-Oriented Gradients) [12], FFT (Fast Fourier Transform) [13] and Haar-Wavelet features [14]. Then, the features obtained by the color/shape-based extractor are usually utilized for training the classifier, such as SVM (Support Vector Machine) [15] and RF (Random Forest) [16]. Although these traditional image processing methods are still widely used in traffic sign and other object detection tasks, for example, Anant R [17] used a HOG-SVM model to recognize traffic signs, and Takaki Masanari [18] used a SIFT [10] model to detect road objects; however, there is still upside potential in the accuracy and robustness of the traffic signs detection algorithm.

In recent years, algorithms based on deep CNN (Convolutional Neural Networks) have been widely used in object detection tasks due to their excellent performance and plasticity [19], [20], and they perform much better than the traditional ones.

Deep learning detection algorithms include single-stage and two-stage algorithms. The single-stage algorithm is also known as the dense algorithm [2], which directly outputs the location and category of bounding boxes densely in a single-shot way, such as SSD (single shot multibox detector) [22], YOLO [23]–[25], and RetinaNet [26]. The two-stage algorithm is also called the dense-to-sparse algorithm [2]. This kind of algorithm obtains a small set of foreground regions proposals from dense candidates firstly, and then modify the localization of each proposal and predict category, such as R-cnn (Region-based convolutional network) [4], Fast R-cnn [4], Faster R-cnn [5].

The main factor that affects the traffic sign detection algorithm's performance is the algorithm's ability to detect small targets [21]. Therefore, researchers have taken many measures, such as designing a multi-scale feature fusion backbone, adding feature pyramids to the neural network, and due to the popularity of transformers, using the attention mechanism to improve the ability of neural networks to detect small targets has become a hot topic in research.

In response to this problem, Cao *et al.* [27] improved Faster R-cnn through the HRNet(High-Resolution Network) [28] backbone network and PISA (Prime Sample) sample strategy [29]. Hai *et al.* [30] used ResNet-101 to improve the backbone of Cascade R-cnn [31] for traffic-sign detection. Han *et al.* [32] tried to use the shallow layer of VGG(Visual Geometry Group) [33] as the backbone of Faster R-cnn and used OHEM [34] (Online Hard Example Mining)

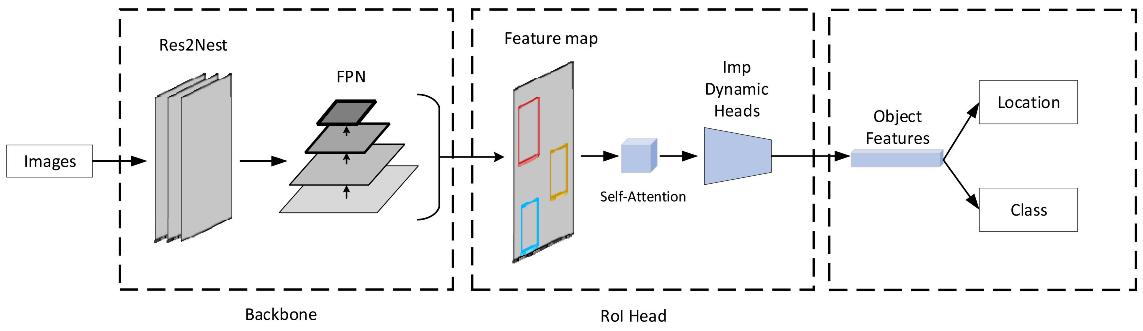


FIGURE 1. The architecture of our model. Our method is a deep neural network composed of a backbone network, a series of DIIHs, and two task-specific prediction layers. There are three inputs: an image, a set of proposal boxes, and proposal features. The backbone extracts the feature of an input image firstly, then inputs corresponding proposal boxes and proposal features into the improved dynamic head to generate object features, and finally outputs the prediction of objects.

to improve the sampling strategy. Liang *et al.* [35] designed a feature pyramid to improve the feature extractor of Faster R-cnn for traffic signs detection. Zhao *et al.* [36] proposed an improved Libra R-cnn [1] model based on the Guided-Anchor RPN [37]. These methods do improve the detection performance of the algorithm, but there are still some shortcomings. For example, the use of an overly complex backbone may lead to low inference time or ignoring the semantic information contained in the deeper feature maps, thereby may resulting in the deterioration of the robustness, and so on.

In improving the backbone, we made further improvements to the existing backbone ResNest, a "split-transform-merge" network. This architecture can implement strong feature expression capability with fewer parameters and calculation costs. We enhanced the multi-scale representation ability of the backbone by constructing hierarchical residual-like connections within each single radix block in the original ResNest. In addition, we believe that a small number of proposal boxes (e.g., 100) is enough to predict all objects in an image according to [2], so we implement an improved Sparse R-cnn for traffic signs detection.

III. APPROACH

A. PIPELINE

Our method is a deep neural network composed of a backbone network, a series of DIIHs, and two task-specific prediction layers. There are three inputs in total: an image, a set of proposal boxes, and proposal features. The latter two are learnable and can be optimized with other parameters in the network. The backbone extracts feature map firstly, then inputs corresponding proposal boxes and proposal features into the improved dynamic head to generate object features, and finally outputs classification and location of objects. The overall architecture of the improved Sparse R-cnn is shown in Fig 1.

B. BACKBONE

Multi-scale fusion structure is the essential method of improving the algorithm for small target detection, so improving the multi-scale capability of the backbone is a required method for designing traffic sign detection. So, we made further improvements to the existing backbone ResNest. We enhanced the multi-scale representation ability of the backbone by constructing hierarchical residual-like connections within each single radix block in the original ResNest. We call the improved backbone Res2Nest.

SE-Net(Squeeze-and-Excitation Network) [39] introduces the channel-attention mechanism by recalibrating the channel feature response adaptively. SK-Net(Selective Kernel Network) [40] proposes two network branches to generalize feature- attention. ResNeXt [38] adopts group convolution in the ResNet [6] bottle block. This structure is also called "split- transform-merge." Inspired by the above networks, ResNest [2] is proposed, consisting of a series of Split-Attention blocks to enable attention across feature-map groups, and these blocks are stacked in ResNet-Style to generalize the attention mechanism. Reg2Net [6] improves the multi-scale ability of neural networks by constructing hierarchical residual-like connections within one single residual block. Based on the above inspiration, we have made further improvements to ResNest. In each radix of ResNest, we replace the 3×3 convolution layer with a group of 3×3 filters with smaller groups of filters while connecting different filter groups in a hierarchical residual-like style. The novel backbone our proposed has better performance without incurring excessive computational costs. In the following part of this section, we will introduce the Split Attention block and the structure of our backbone network in detail.

1) IMPROVED SPLIT-ATTENTION BLOCK

Split-Attention block is a computational unit composed of the feature-map group and split attention operations. We replaced the 3×3 convolution layer with our improved architecture called the MTF block(Multi-Scale Fusion Block). The overall structure diagram of our backbone is shown in Fig 2.

Like ResNext, the feature map input into the ResNest block will be divided into several groups, and the number of groups K is a hyperparameter. We regard the obtained feature map group as "cardinal groups," and each cardinal group is divided into R (R is the abbreviation of "Radix") splits, so the feature map which is input in ResNest block would be divided into $G = KR$ splits. In this paper, we set $K = 2$, $R = 2$, the same as the origin ResNest model, due to ablation experiment shows that this setting is the most efficient. We do some convolutional transformations to the features in each individual split, denoting them as $\{F_1, F_2 \ldots F_G\}$. As shown in Fig 3, for the feature maps in each Radix, after the 1×1 convolution, we evenly split the feature maps into s feature map subsets, denoted by x_i, where $i \in \{1, 2, \ldots s\}$. Each feature subset x_i has the same spatial size but $1/s$ number of channels compared with the input feature map. Except for x_1, each x_i has a corresponding 3×3 convolution, denoted by $K_i()$. We denote by y_i the output of $K_i()$. The feature subset x_i is added with the output of $K_{i-1}()$ and then fed into $K_i()$. The features obtained by transformation can be represented as U_i. Thus, U_i can be written as (1):

$$U_i = Concat(y_i)$$
$$y_i = \begin{cases} x_i & i = 1 \\ K_i(x_i) & i = 2 \\ K_i(x_i + y_{i-1}) & 2 < i \leq s \end{cases} \tag{1}$$

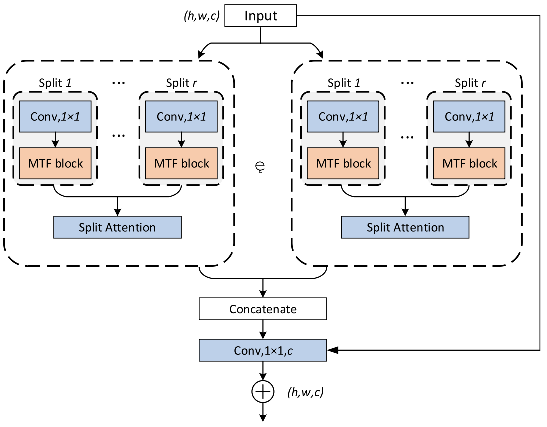


FIGURE 2. The overall architecture diagram of Res2Nest. The input feature map is first divided into R cardinal groups and split attention operation is performed on those features. Finally, the features of each cardinal group are concatenated to obtain a feature map with the same size as the input, then fused with the input through a neck bottle, similar to ResNet-D.

2) SPLIT ATTENTION IN CARDINAL GROUPS

For each cardinal group, we fuse the features of all its splits by element-wise summation, and then the K_{th} cardinal group can be expressed as (2):

$$\hat{U}^k = \sum\nolimits_{j=R(k-1)+1}^{Rk} U_j$$
$$\hat{U}^k \in \mathbb{R}^{H \times W \times (C/K)} \quad \text{for } K \in 1, 2, \ldots K \tag{2}$$

where H, W, C are the block output feature-map sizes, and then we perform split attention operations on the fused features. The overall flow is shown in Fig. 4.

The fused features are then fed into a Global Average Pooling (GAP) [7] layer across spatial dimensions $s^k \in \mathbb{R}^{C/K}$, which can effectively collect global context information with

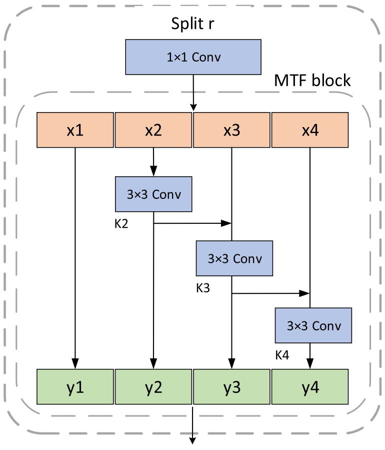


FIGURE 3. Schematic diagram of the structure of the MTF block, we evenly split the feature maps. Each feature subset x_i has the same spatial size but $1/s$ number of channels compared with the input feature map. Except for x_1, each x_i has a corresponding 3×3 convolution, denoted by $K_i()$. The feature subset x_i is added with the output of $K_{i-1}()$ and then fed into $K_i()$.

embedded channel statistics. Here the C_{th} composition is expressed as (3):

$$s_c^k = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} \widehat{U}_c^k(i, j) \tag{3}$$

Then we use two fully connected layers with ReLU (Rectified Linear Units) activation to get the mapping of s^k as the assignment weights of the feature combination. This operation is denoted as $\Psi(s^k)$. The representation of the cardinal group is obtained by a weighted combination over splits, the C_{th} channel is calculated as (4):

$$V_c^k = \sum_{i=1}^{R} a_i^k(c) U_{R(k-1)+i} \tag{4}$$

the coefficient a_i^k can be calculated as (5):

$$a_i^k(c) = \begin{cases} \dfrac{\exp(\Psi_i^c(s^k))}{\sum_{j=0}^{R} \exp(\Psi_i^c(s^k))} & \text{if } R > 1, \\ \dfrac{1}{1 + \exp(-\Psi_i^c(s^k))} & \text{if } R = 1 \end{cases} \tag{5}$$

the outputs from each cardinal group are then concatenated (6):

$$V = Concat\{V^1, V^2, \ldots V^K\} \tag{6}$$

The final output of the ResNest block is obtained by a shortcut connection, which can be expressed as $Y = V + X$ if the output and input feature map share the same shape. For blocks with a stride, we use an average pooling layer before a 1×1 convolutional layer with a stride of 2 to transform the input feature map: $Y = V + \Gamma(X)$. In the backbone design of this paper, in addition to replacing Residual Block with ResNest Block, we make the following three changes to the original ResNet.

We use an average pooling layer with a kernel size of 3×3 instead of the strided convolution at the transition block because the convolution layers require handling the zero-padding strategies, which is often suboptimal in small objects prediction tasks like traffic signs detection.

We replace the first 7×7 convolutional layer with three succeeding 3×3 convolutional layers (we set the stem channels of the backbone to 64), and a 2×2 average pooling layer is added to the shortcut connection before the 1×1 convolutional layer in the transitioning blocks with a stride of 2. The latter two changes of ResNet are two effective modifications introduced by ResNet-D [41].

We construct the Feature Pyramid Network (FPN) [42] with levels P_2 through P_5 to further fuse the feature maps output from ResNest, l indicates the levels of FPN, and P_l has 2^l lower than the input in resolution.

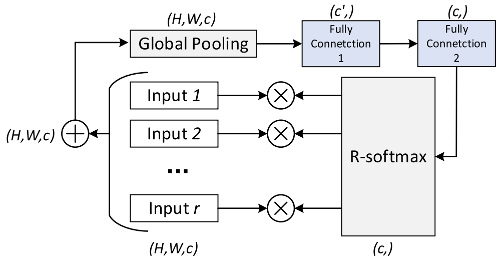


FIGURE 4. Split-attention block within a cardinal group. We use c = C/K in this figure for easy visualization, and c′ is the inter-channel which is expressed as c′ = c * radix/(reduction_factor), H, W are the sizes of the output feature map.

In the experiment of our paper, we resize the images of TT100K [43] to $(3 \times 1024 \times 1024)$ as the input of the model, and the channels of feature maps output from the four stages of ResNest are 256, 512, 1024 and 2048, respectively, then the backbone feeds these feature maps into FPN(Feature Pyramid Network), and we set the channels of all pyramid is 256. We visualize the feature map extracted by the backbone through HFM (Heat Feature Map) [44] shown in Fig. 5, and it can be seen that the features of the input image are accurately mapped to the feature map of ResNest block in each stage.

C. IMPROVED SPARSE R-CNN

In addition to the feature maps outputs by FPN, Sparse R-cnn also requires two inputs: Proposal boxes and Proposal features. The attention mechanism is also an effective method to improve the detection of traffic signs, so we set up a branch network for recalibrating the channel feature response adaptively through the Global Average Pooling (GAP) operation and a fully connected layer. It can also be seen as the implementation of the cross-channel self-attention mechanism. The rest of this section explains the workflow of improved Sparse R-cnn in detail.

1) LEARNABLE PROPOSAL BOX

Instead of using the Region Proposal Network (RPN) to generate proposals, Sparse R-cnn [2] adopts a small set of learnable proposal boxes with a size of $(N \times 4)$ as region proposals. Each proposal box contains four values, representing the center coordinates, height, and width of the predicted object. All proposal boxes are initialized as the size of the whole image before they are fed into DIIH (the shapes of each initialized proposal box are slightly different to make the model more flexible), as shown in Fig 10(a). Those proposal boxes can be seen as the initial guess of the object location, and the parameters of these boxes are updated in DIIH during the training or inference process in each iteration.

2) LEARNABLE PROPOSAL FEATURE

The 4-d proposal box can only describe the location of the object but does not contain detailed information, such as posture and shape. Therefore, Sparse R-cnn needs another input, "proposal feature" as a supplement. It is a feature vector that encodes the rich instance characteristics with a size of $N \times d$, where N is the number of proposal boxes and d is the dimension of the proposal features. We set its dimension to 256 as the original Sparse R-cnn in our paper. Those features have a one-to-one correspondence with the proposal boxes, representing the detailed features of the corresponding proposal boxes.

Sparse R-cnn utilizes the self-attention mechanism of Transformer [45] to process the proposal features before them fed into DIIH. We use Multi-head attention with a residual connection followed by layer normalization to implement this process. It can be expressed as (7), where x is the proposal feature and the Matten(x) is the Multi-head attention function.

$$x = \text{LayerNorm}\,(x + \text{MAtten}\,(x)) \tag{7}$$

The attention function can be described as mapping a set of key-value pairs and a query to an output. The output can be computed as a weighted sum of the values, where the weight can be calculated by the compatibility function of the query with its corresponding key. We divide the proposal features into h parts equally, where h is the number of heads (we set $h = 8$), and perform self-attention operations in parallel, then the output vectors from each head are concatenated to get the final values. The multi-head self-attention function requires three inputs, namely keys, queries and values. We simply set them as $x = keys = values = queries = proposal\ features$ in this work.

As shown in Fig. 6, the input values, queries, and keys are divided into h parts. After these inputs enter a fully connected layer, they are performed the Scaled Dot-Product Attention operation in parallel, and then we concatenate the output vector of each head and feed it into a fully connected layer to get the final value. The calculation process of Scaled Dot-Product Attention can be expressed as (8), where d_k represents the dimension of the key matrix.

$$\text{Atten}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{8}$$

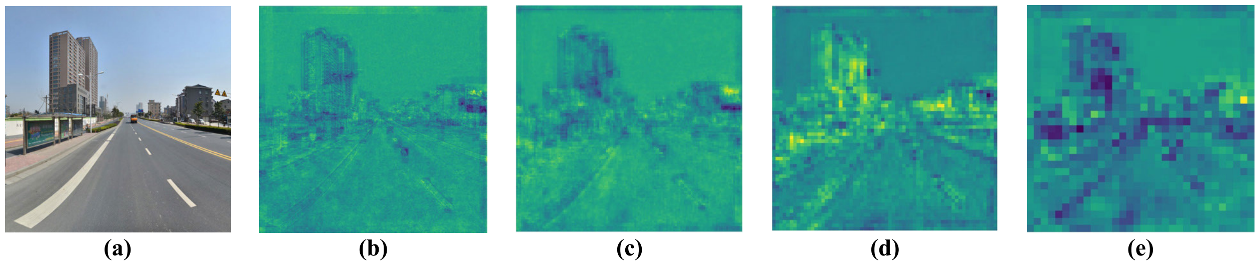


FIGURE 5. We visualize the feature map extracted by the backbone through HFM, (a) is a sample of the dataset with a size of 3 × 1024 × 1024, (b)-(e) are the HFM of ResNest from the second to the fourth stage. It can be seen that the features of the input image are accurately mapped to the feature map of the ResNest block in each stage.

The calculation formula of the Multi-head attention mechanism can be expressed as (9)

$$\begin{aligned}\text{Multi\_h}(Q, K, V) &= \text{Concat}(head_1, \ldots head_h)W^O\\ head_{\text{i}} &= \text{Atten}(QW_i^Q, KW_i^K, VW_i^V) \qquad (9)\end{aligned}$$

where h is the number of heads, W_i is the learnable weight, and $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$, where d_{model}, d_k, and d_v are the channels of x (proposal feature), and the *key* and *value* channels in each head. Proposal features that implemented self-attention operation will be fed into DIIH.

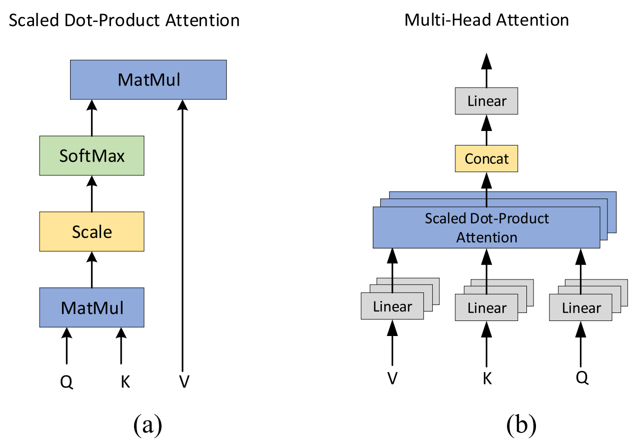


FIGURE 6. (a) The process of scaled dot-product attention. (b) Multi-head attention consists of several attention layers running in parallel.

3) IMPROVED DIIH

Dynamic Instance Interactive Head (DIIH) is the core of Sparse R-cnn. It generates the positions and features of the predicted objects through the input feature maps, proposal boxes, and proposal features. As shown in Fig. 7, we improve the DIIH and set up a branch network to recalibrate the channel feature response adaptively through the GAP (Global Average Pooling) [7] operation and a fully connection layer. It can also be seen as the implementation of the cross-channel self-attention mechanism. Moreover, we stack 6 cascaded DIIHs in the whole model to generate the final object prediction. The object features and elegant proposal boxes output by each DIIH stage will be used as the input of the next DIIH stage.

Given N proposal boxes, Sparse R-cnn first remaps these boxes to the FPN feature map and uses the RoIrEAligns operation to extract the features of each box, the size of each RoI feature is $(S \times S \times C)$ (We set $S = 7$, $C = 256$ in this paper). Similarly, the RoI feature in each box has a one-to-one correspondence with its proposal feature $\text{P}_i(C)$, and they will be fused in the instance interactive module.

For each proposal region, the RoI feature and its corresponding proposal feature are fused through an architecture termed "Dynamic Layer." Firstly, the RoI feature is resized to a 2-d vector with a size (S^2, C) as one of the inputs of the Dynamic Layer, and the proposal feature is directly fed into a fully connected layer to get the parameters with length $(2 \times C' \times C)$ of Dynamic Layer,
where C' is a hyperparameter named "Dynamic Channel," which is set to 64 in this paper. Then the vector is divided equally into two parts with length $(C' \times C)$, and we reshape them into 2-d vectors of size (C, C') and (C', C) respectively named "P_{in}" and "P_{out}." Dynamic layer performs matrix multiplication on the ROI_{2d}, P_{in}, and P_{out} in turn, then makes batch normalization and ReLU activation to the output after each multiplication, then the fusion feature (S^2, C) is obtained, this process can be expressed as (10), where $P_{\text{in}} \in \mathbb{R}^{C \times C'}$, $P_{\text{out}} \in \mathbb{R}^{C' \times C}$, $RoI_{2\text{d}} \in \mathbb{R}^{S^2 \times C}$.

$$\begin{aligned}Out &= \text{M}(\text{M}(RoI_{2\text{d}}, P_{\text{in}}), P_{\text{out}})\\ \text{M}(A, B) &= \text{ReLU}(\text{BN}(AB)) \qquad (10)\end{aligned}$$

In the original Sparse R-cnn model, the fusion feature is flattened into a 1-d vector and then fed into a fully connected layer to obtain the final output of the Dynamic Layer. However, this fully connected layer is prone to overfitting because the fully connected layer has a lot of learnable parameters, and more importantly, it acts as a black box in the process of generating object features [7]. Therefore we improved this architecture. SE-Net [39] and ResNest [2] implement the cross-channel attention mechanism through recalibrate the channel feature response adaptively. Based on the inspiration of the above works, we design a branch network to implement the cross-channel self-attention mechanism in the object feature generation process without additional computational costs of the entire network.

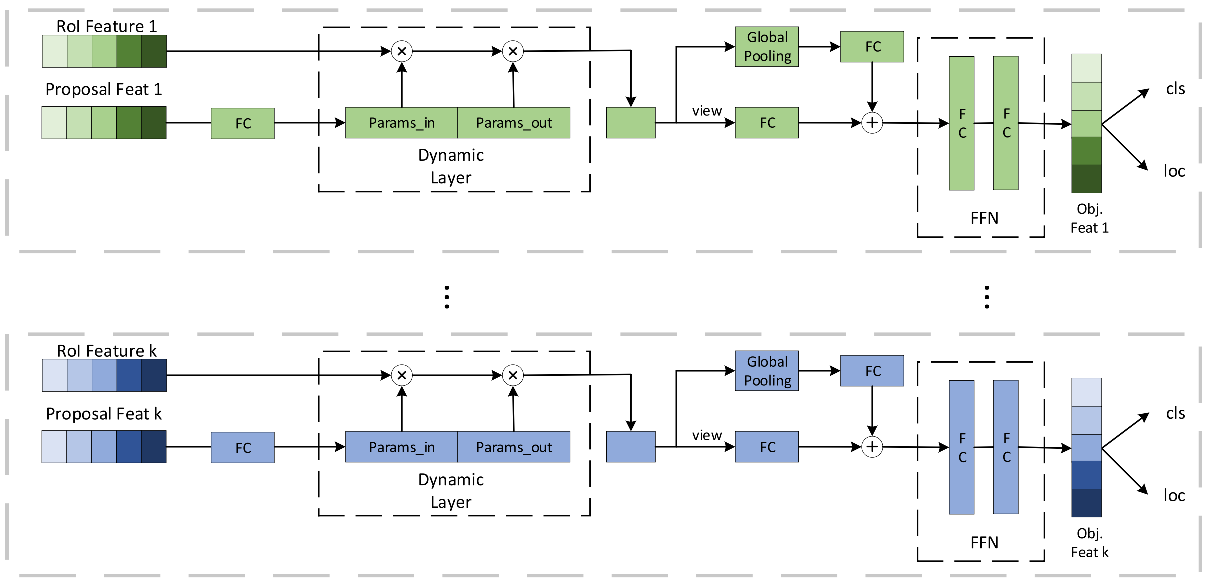


FIGURE 7. The architecture of improved dynamic instance interactive head, RoI features, and Proposal features are fused through dynamic module, and then fed into two branch networks to implement the self-attention mechanism, and then object features and predictions of object classifications and localizations are generated, object features are used as the proposal features in next stage.

As shown in Fig. 7, the features output by the Dynamic Layer are processed separately in two branches, one is to flatten the features and then feed them into a fully connected layer, and the other is to perform the Global Average Pooling (GAP) [7] operation to the features, the GAP layer averages the feature maps of each layer. This operation can be regarded as a structured regularization to obtain the spatial average of multiple feature layers. Then we match the output channels with another branch through a fully connected layer, as shown in Fig. 8, and finally fuse the outputs of the two branches in an element-wise add manner, thus implementing a self-attention mechanism and inhibiting the overfit of the neural network.

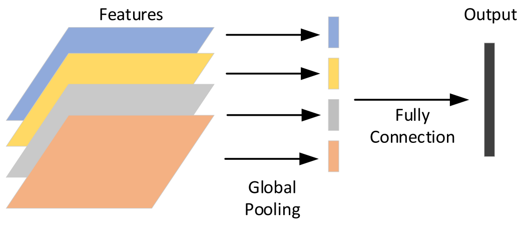


FIGURE 8. The process of global average pooling. The GAP layer averages the feature maps of each layer and then sends them into a fully connected layer to match the size of the other branch network.

The feature outputs from Dynamic Layer are fed into two consecutive C-channels fully connected layers. This architecture is borrowed from the FFN (Feed Forward Network) of Transformer [45] to increase the flexibility of the model. The feature outputs from FFN are called object features, which are used to be the proposal feature of the next DIIH stage. The object feature is fed into two fully connected layers respectively to obtain the final prediction of the object's class and location. The input proposal box refined through location prediction is used as the proposal box in the next DIIH stage. After six refinements by DIIHs, the output proposal boxes and their corresponding class labels are used as the final prediction of the image, we set the threshold to filter out the proposal boxes with low scores, and finally the model outputs appropriate prediction boxes. We select three stages of proposal boxes from a particular iteration and visualized them as Fig 10(b)-(e). It can be seen that the proposal boxes in the model are gradually positioned on the traffic signs of the input image.

4) SET PREDICTION LOSS

The loss function in our paper can be expressed as (11)

$$L = \lambda_{\mathrm{cls}} \times L_{\mathrm{cls}} + \lambda_{\mathrm{L1}} \times L_{\mathrm{L1}} + \lambda_{\mathrm{giou}} \times L_{\mathrm{giou}} \tag{11}$$

where L_{cls} is focal loss ($\alpha = 0.25, \gamma = 2$) of predicted classifications and ground truth category labels. L_{L1} and L_{giou} are L1 loss and generalized IoU loss of predicted boxes and ground truth bounding boxes. λ_{cls}, λ_{L1} and λ_{giou} are hyper-parameters, and we set $\lambda_{\mathrm{cls}} = 2$, $\lambda_{\mathrm{L1}} = 5$ and $\lambda_{\mathrm{giou}} = 2$ respectively same as the origin Sparse R-cnn model.

IV. EXPERIMENTS

A. DATASET

The experiments on traffic signs used the TT100K [43] dataset, a traffic-sign benchmark from 100K Tencent Street View panoramas. The dataset contains 9167 images (6105 for training and 3071 for testing). These images cover significant variations in illuminance and weather conditions with a size of 2048 × 2048. Each traffic sign in the benchmark is annotated with a class label, gt-bbox (ground truth bounding box), and pixel mask.

TABLE 1. Statistical table of TT100K.

Name	Tsinghua-Tencent 100K
Images	9176 (6105 for training, 3071 for testing)
Gt Bboxes	23193
Categories	125
Size (Pixels)	2048×2048
Small Bboxes	9396 (40.40%)
Middle Bboxes	12119 (52.25%)
Large Bboxes	1705 (7.35%)

Although this benchmark includes a total of 234 categories of traffic signs, in the experiment, we found that some categories only appear in the training dataset or the test dataset. Using such annotations cannot objectively evaluate the performance of the algorithm. Therefore we selected 125 categories that are contained in both the training dataset and test dataset.

We perform statistical analysis on the benchmark and summarize the statistical results in Table 1, and some statistical results are visualized as shown in Fig 9.

According to the convention proposed by the commonly used large-scale dataset MS COCO benchmark [49]. We divided bboxes(bounding boxes) into three groups based on their area, namely, small(area $\in$ [1, 1024)), middle(area $\in$ [1024, 9216)), and large(area $>=$ 9216). It can be seen that the areas of bbox in the TT100K are mostly smaller than 9216 pixels, the total proportion of large objects is only 7.35%, the proportion of medium objects is slightly higher, about 52.25%, and the small objects are second, accounting for about 40.40 %. This result shows that the key to improving the performance of the traffic-sign detection model is to improve its ability to detect small objects.

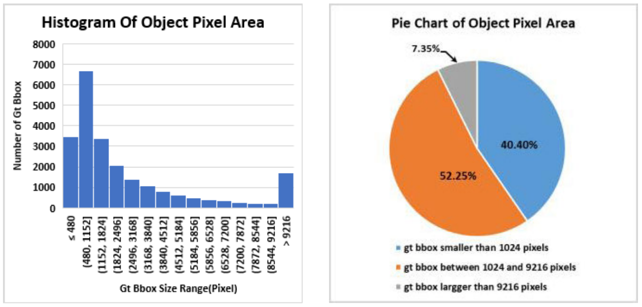


FIGURE 9. Visualizations of some statistical results about TT100K. (a) is the histogram of object pixel areas, and (b) is the pie chart of the object pixel area.

B. TRAINING DETAILS

The experimental environment of our approach was NVIDIA TITAN XP graphics, Ubuntu 16.04LTS system, CUDA 10.2, PyTorch 1.5.1 programming framework based on Python 3.8.1.

In the preprocessing of the dataset, we first resized the input images to 1024 × 1024. In all training and inference experiments, we uniformly used the 1024 × 1024 images as input images, and the data augmentation includes random horizontal and random flip.

In the training process, the default training schedule was 36 epochs, and the initial learning rate was set to 2.5×10^{-5}, but we used the "linear-warmup" method to increase the learning rate to 2.5 × 10.5 slowly, the warm-up iteration was 500, and the learning rate was divided by 10 at epoch 27 and 33, respectively. The optimizer was "AdamW [46]" with a weight decay of 0.0001. The mini-batch was 2 images, and all models were trained with 2 GPUS. The partial weights of our backbone Res2Nest are initialized as the pre-trained weights of ResNest on ImageNet [47], and the weights of other modules of our algorithm were initialized with Xavier [48].

The default number of proposal boxes, proposal features, and DIIH stages was 300,300,6, respectively. We will specify it in the ablation analysis.

C. INFERENCE DETAILS

Given an input image, our model directly predicts 100 bounding boxes associated with their scores, and without any post-processing, the origin Sparse R-cnn set the number of predictions per image is equal to the input proposal boxes. However, we set it to 100 directly because we found that this hyperparameter has little effect on the results. We proved the above points in the ablation experiment and listed the relevant ablation experiment data in Table 6. The scores indicate the probability of boxes containing an object.

D. DETECTION PERFORMANCE AND EFFICIENCY

In response to different situations, many previous scholars have proposed different algorithm evaluation methods, such as [52]. Based on the inspiration of these works, we evaluated the traffic sign detection model from the three aspects of algorithm complexity, accuracy, and robustness. We compared our method with three representative generic object detectors Faster R-cnn, Cascade R-cnn, Sparse R-cnn, and three state-of-the-art traffic-sign detectors proposed by Cao *et al.*, Wang *et al.*, and Zhao *et al.* Their general structure is shown in Table 2. The purpose of comparing with the general object detection model is to prove why we improved Sparse Rcnn instead of other approaches. In addition, for each general object detection approach, we used ResNet, ResNest, and our improved backbone for training to prove that our improvement is practical. Furthermore, the specific impact of each improvement is analyzed in the ablation experiment.

TABLE 2. Comparison of the state-of-the-art traffic-sign detection model in architecture.

Model	Backbone	Neck	RPN Head	RoI Head
Ours	Res2N50	FPN	\	IMP-DIIH
Wang et al. [30]	Res101	FPN	Dense RPN	Rand
Zhao et al. [36]	Res50	\	Guide-Anchor	Rand
Cao et al. [27]	HR_w18	HRFPN	Dense RPN	PISA

In this section, we used the inference time to represent the complexity of the overall model to compare with

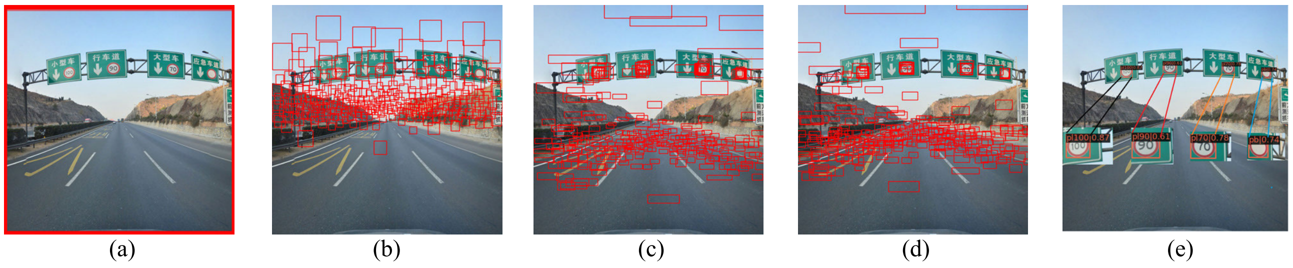


FIGURE 10. Visualization of proposal boxes. (a) is the initialized proposal boxes, all proposal boxes are initialized as the size of the whole image. (b)-(d) are the visualization of the proposal boxes in 1,3,5 DIIH stages of a particular iteration. It can be seen that the proposal boxes in the model are gradually positioned on the traffic signs of the input image, and (e) is a visualization of the final prediction. The predictions are partially enlarged.

TABLE 3. Comparison of the state-of-the-art model in architecture.

No.	Approach	Backbone	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	Inf.Speed
1	Ours	Res2N50	**49.0**	**62.3**	**56.8**	**36.8**	**57.7**	**77.4**	21FPS
2	Wang et al. [30]	Res101	43.7	54.7	51.7	23.6	51.4	75.4	18FPS
3	Zhao et al. [36]	Res50	44.3	56.2	52.4	35.5	51.9	71.8	20FPS
4.	Cao et al. [27]	Hr_w18	46.3	58.0	55.6	25.4	54.7	75.8	22FPS
5.	CRCNN [31]	Res50	45.4	56.5	53.5	27.8	54.3	75.5	19FPS
6.	FRCNN [4]	Res50	40.5	51.1	48.2	23.1	49.9	72.6	22FPS
7.	SRCNN [2]	Res50	44.2	56.9	52.3	35.8	54.2	75.9	21FPS
8	CRCNN [31]	ResN50	45.8	56.9	53.6	28.2	54.6	75.8	18FPS
9	FRCNN [4]	ResN50	42.4	53.0	50.5	20.3	53.5	73.4	22FPS
10	SRCNN [2]	ResN50	46.2	59.0	55.0	36.4	55.1	76.4	21FPS
11	CRCNN [31]	Res2N50	46.5	57.6	54.3	29.2	55.3	76.3	18FPS
12	FRCNN [4]	Res2N50	44.0	54.8	52.0	24.2	55.1	76.0	22FPS
13	SRCNN [2]	Res2N50	47.8	60.4	56.3	36.7.	56.0	77.2	21FPS

state-of-the-art more intuitively. However, we adopted Floating point operations (FLOPs) and the number of parameters to represent algorithm complexity for more detailed analysis in the ablation analysis.

We used AP(Average Precision) (AP_s, AP_m, AP_l), AP_{50}, and AP_{75}, which were introduced by MS COCO [49] benchmark to evaluate the accuracy of methods. The results are shown in Table 3. The AP obtained by our model is 49.0, and the AP_{50}, and AP_{75} are 62.3 and 56.8, respectively. According to the results of experiments 1-4, our model performs the best accuracy compared with other state-of-the-art traffic sign detection models. Experiments 5-10 prove that for general target detection models, directly replacing ResNet with ResNest can improve the algorithm's performance, while experiments 8-13 prove that our improvement to ResNest is practical, our MST block further improves the performance of the algorithm. We will explain in more detail in the ablation analysis. In addition, in terms of the detection ability of small and medium targets, the AP obtained by our algorithm was significantly higher than other traditional two-stage algorithms.

Fig. 11 illustrates the P-R(Precision-Recall) [49] curves of our model and other state-of-the-art models of AP_s, AP_m, and AP_l. We only use ResNet50 for the backbone of the general target detection model in order to make the experimental results more intuitive and concise. The P-R curve is a standard measure to evaluate the performance of object detectors. The larger the coverage area of the P-R curve, the better performance of the algorithm. It can be seen in Fig. 11 that the coverage area of our P-R curve is generally higher than other methods. The visualization of our detection results is shown in Fig. 13. As shown in Fig. 13, our method can effectively detect small and multiple traffic-sign objects in images, but our method does not significantly improve the detection ability of large objects.

In addition, we also evaluated the robustness of our methods. We simulated some severe weather images and noisy images by the corruption image generation method introduced by Hendrycks and Dietterich [50]. We generated foggy, frost, snowy images and images with Gaussian noise based on the original TT100K test dataset as our robustness test dataset and divided each severity into 3 levels according to [51]. Fig. 12 shows some generated corruption images.

We considered the AP obtained by detecting the original dataset as "AP Clean" and that by corruption as "AP Corr." and used the percentage of "AP Corr." in "AP Clean" to

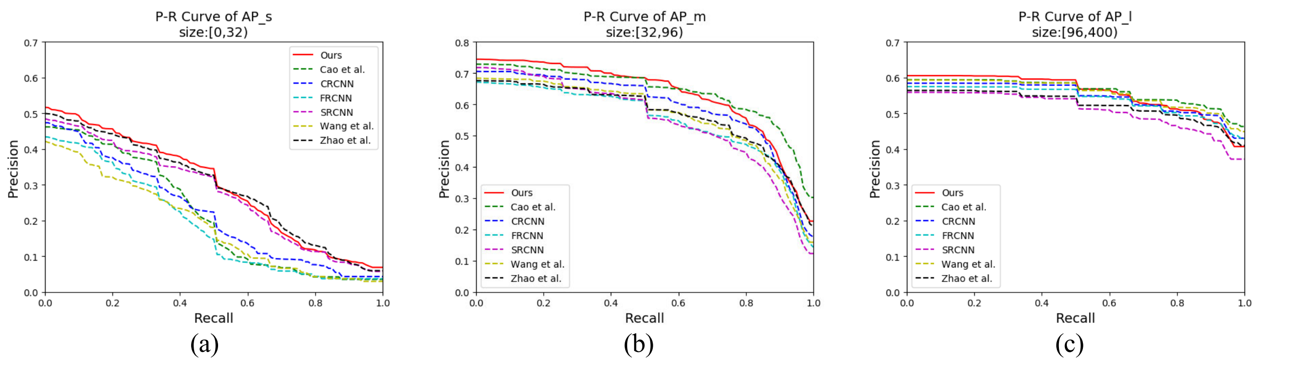


FIGURE 11. **Precision-recall curves for AP_s, AP_m, and AP_l of seven object detection methods.**

represent the method robustness (12). The results of the robustness experiment are shown in Table 5. In general, our method can maintain good performance for traffic-sign detection under bad weather and noise. For fog corruption, our method can almost ignore the corruption of bad weather. The AP percentages for the severity levels 1 to 3 were 96.5%, 92.1%, and 90.6%, respectively, which are the highest compared to other state-of-the-art methods. Compared to the original Sparse R-cnn, the AP percentage of our algorithm increased by an average of 8%, indicating that the robustness under foggy weather has been significantly improved. For frost corruption, our method is still the most robust of all stage-of-the-art in our experiments. Our method obtains AP percentages of 94.9%, 87.8%, and 77.4%, and compared to the original Sparse R-cnn, the AP percentage of our algorithm also increased by an average of 8%, indicating that the robustness under frost weather has been significantly improved. For Gaussian noise corruption, our method obtained AP percentages of 89.3%, 75.9%, and 55.3%, respectively, and for snow corruption, the AP percentages were 76.6%, 57.6, and 52.9, respectively. For Gaussian noise and snowy weather interference, our model does not have the highest robustness ratio (second only to the model of Cao *et al.*), we believe that it may be caused by the nature of the noise being different from the previous noises, but the AP obtained by our model under this interference is still the highest. Therefore we believe that our algorithm can still work normally under the interference of these noises.

$$Percentage = \frac{AP\ Corr.}{AP\ Clean} \tag{12}$$

E. ABLATION ANALYSIS

1) THE COMPLEXITY OF BACKBONE

This section mainly analyzes the complexity of the backbone and the improvement effect of improved Sparse R-CNN. We used FLOPs (Floating-point Operations) and the number of parameters to represent the complexity of the neural network architecture. We compared the FLOPs and number of parameters of ResNet50, ResNest50, and our Res2Nest50. The results are shown in Table 4, ResNest50 has about 1.8M more parameters than ResNet50, and when the input image size is 1024 × 1024, ResNest50 is 26.4G FLOPs more than ResNet50, and our Res2Nest50 has about 1.7M parameters, and 13.2G FLOPs more than ResNest50. We believe that the increased computational costs are within an acceptable range. In addition, although we believe that a deeper feature extraction network may improve the accuracy of the detector, it will affect the inference time and does not meet the real-time requirements of traffic sign detection tasks. Therefore, we only use a feature extraction network with a depth of 50 as the backbone of our model.

TABLE 4. **Comparison of the model complexity and computational efficiency of backbones.**

Backbone	Input size	GFLOPs	Param.
1. Res50		86.10	23.28M
2. ResN50	1024×1024	112.50	25.11M
3. Res2N50		125.70	26.82M

2) THE IMPACT OF PREDICTION BOXES

The number of prediction boxes in the final output of the original Sparse R-cnn neural network was set to be equal to the proposal boxes. However, we found that the number of prediction boxes has nothing to do with the proposal boxes, and too many prediction boxes will prolong the training time of the model, as shown in Table 6. We used the improved Sparse R-cnn with a Res2Nest50 as the model of the ablation experiment, but according to the original Sparse R-cnn paper, we only used 100 and 300 proposal boxes as the main configuration of the improved Sparse R-cnn. It can be concluded from Table 5 that the number of proposal boxes in DIIH will affect the performance of the algorithm, but the final prediction boxes have nothing to do with the accuracy, and too many predict boxes can make the training time longer.

We analyzed this because the number of objects in the image is usually much smaller than the number of the maximum prediction boxes, so most of the prediction boxes are filtered by the threshold, which does not affect the final detection accuracy. Therefore, we set the prediction boxes to a constant of 100.

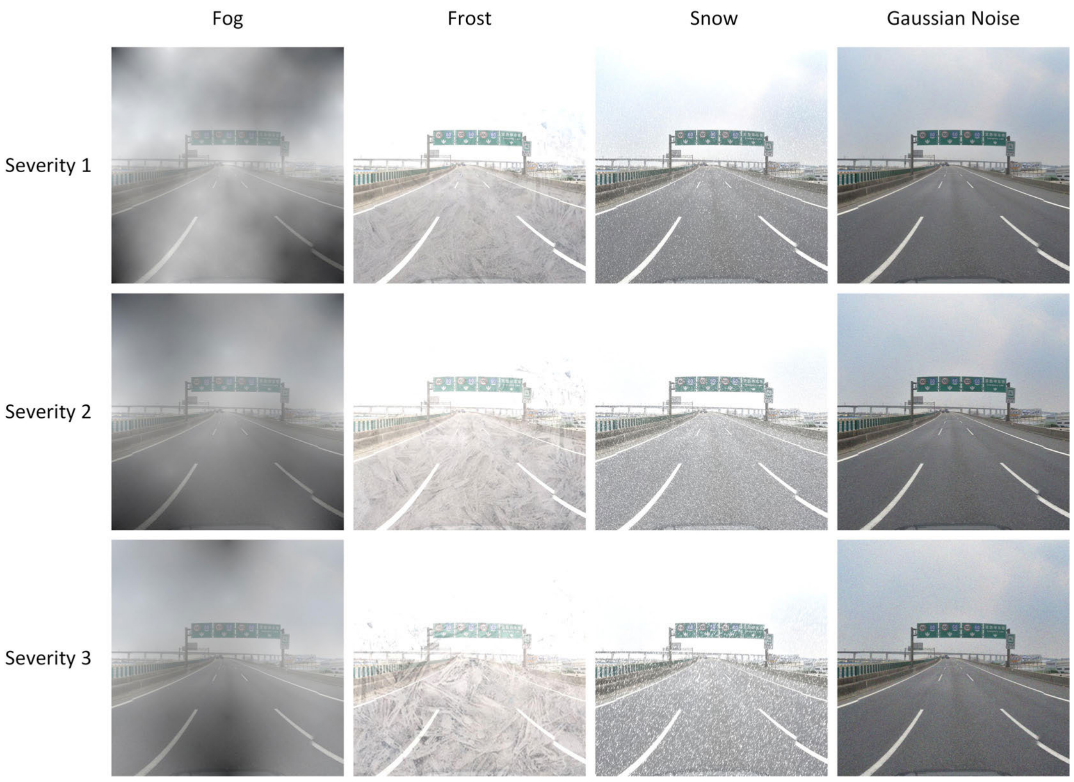


FIGURE 12. Corruption image simulation of different severities. We generated foggy, frost, snowy images and images with Gaussian noise based on the original TT100K test dataset as our robustness test dataset.

TABLE 5. Comparison of the robustness of detection methods.

	Fog1	Fog2	Fog3	Frost1	Frost2	Frost3
1: ours	**96.5%**	**92.1%**	**90.6%**	**94.9%**	**87.8%**	**77.4%**
2. Cao et al. [27]	87.4%	84.9%	77.1%	91.6%	78.2%	68.8%
3. CRCNN [31]	77.1%	88.3%	84.1%	89.4%	80.6%	73.3%
4. FRCNN [4]	87.4%	82.0%	77.3%	92.6%	81.5%	73.8%
5. SRCNN [2]	85.3%	80.3%	75.1%	86.0%	74.0%	64.9%
6. Wang et al. [30]	85.1%	79.6%	73.7%	91.8%	83.1%	73.7%
7. Zhao et al. [36]	89.8%	85.1%	81.7%	88.9%	83.7%	75.8%
	Gauss.1	Gauss.2	Gauss.3	Snow1	Snow2	Snow3
1: ours	**89.3%**	**75.9%**	**55.3%**	**76.6%**	**57.6%**	**52.9%**
2. Cao et al. [27]	91.0%	79.2%	56.2%	80.5%	59.7%	58.5%
3. CRCNN [31]	87.4%	70.7%	48.2%	59.7%	75.6%	52.2%
4. FRCNN [4]	88.6%	75.1%	49.9%	73.6%	53.3%	51.1%
5. SRCNN [2]	83.9%	66.1%	46.2%	68.1%	45.9%	48.0%
6. Wang et al. [30]	85.4%	71.4%	49.2%	72.3%	51.0%	54.0%
7. Zhao et al. [36]	86.9%	75.2%	55.8%	77.2%	58.0%	55.5%

3) THE ABLATION OF DESIGN CHOICES

This section mainly introduces some ablation experiments for hyperparameter selection and model design. First, we explained our choice of parameters in the backbone design through hyperparameter ablation experiments. In addition, we proved that each of our improvements is effective through model ablation experiments.

For Res2Nest, although the increase of group, radix, and split will increase the complexity of its algorithm, the increase is minimal. However, the difference in AP obtained by different hyperparameter training is more pronounced. Therefore, our strategy is to choose a smaller group, radix, or split hyperparameters if the obtained AP is not much different. In the experiment, we only change the backbone of the model,

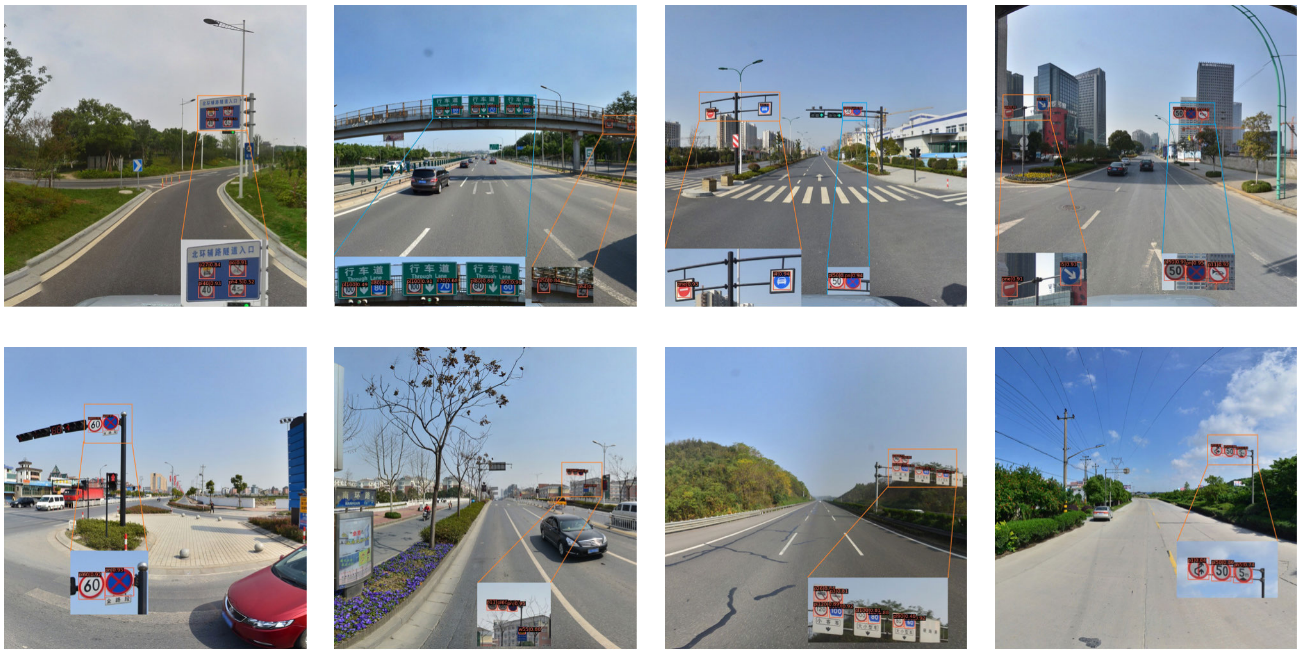


FIGURE 13. **Visualization of the detection results attained by our method. We partially magnified the detected traffic signs in order to facilitate observation.**

TABLE 6. **Ablation on the final predict boxes.**

Prop. boxes	Pred. boxes	AP	AP_{50}	AP_{75}	Training time
100	100	45.7	58.7	53.9	8h
100	300	46.0	60.5	53.4	10h
300	100	49.0	62.3	56.8	10h
300	300	48.7	61.6	55.9	13h

and the rest of the model is the improved Sparse R-cnn architecture with 300 proposal boxes. As shown in Table 7, we find that the hyperparameters *Group* (K) and *Radix* (R) have a relatively significant impact on the experimental results, but their increase causes a relatively significant increase in computational cost. The hyperparameter *Split(S)* has a relatively small impact on the experimental results, and the calculation cost brought about by its growth is relatively small. In addition, we find that when $K > 2$, the increase of K is not evident to the increase of the algorithm. We analyze that the overfitting of the algorithm causes this. Similarly, when $R > 2$, or $S > 4$, the above situation also appears, so we believe that $K = 2$, $R = 2$, $S = 4$ is the most efficient hyperparameter combination. This ablation experiment indirectly proves that our improvement is effective, but we make a more specific explanation in the next ablation experiment.

In addition, we compared the impact of the Res2Nest backbone and the improved DIIH head. Since the results in Table 3 have proved that ResNest is superior to ResNet in performance, we only compared Res2Nest with ResNest in this experiment to prove the superiority of our Res2Nest. As shown in Table 8, when the number of proposal boxes is 100, and the RoI head of model was composed of our improved DIIHs, the AP obtained by using Res2Nest50 was 1.1 higher than ResNest50, and when the RoI head was composed of the original DIIHs, it was 1.0 higher. When the number of proposal boxes was 300, and the RoI head of the model was composed of improved DIIHs, the AP obtained by using Res2Nest50 was 0.9 higher than ResNest50, and when the RoI head was composed of the original DIIHs, it was 1.0 higher.

For the RoI head of the model, our improved DIIH has better performance than the original DIIH. When we used ResNest50 as the backbone of the model, and when the number of proposal boxes was 100, replacing the original DIIH with our improved DIIH increased the AP by 0.9, and when the number of proposal boxes was 300, it increased by 2.9. When we used Res2Nest50 as the backbone of the model, and when the number of proposal boxes was 100, replacing the original DIIH with our improved DIIH increased the AP by 1.0, and when the number of proposal boxes was 300, it increased by 2.8. When the number of proposal boxes was 100, the improvement effect of the improved DIIH was not pronounced, we analyzed that the reason for the result is the underfitting of the neural network, so we set the number of proposal boxes to 300 by default.

Moreover, the APs of small objects detection by different models are also listed in Table 8. It can be seen that our improvements to the backbone and HIID head are effective. When the number of proposal boxes is 300, Our model can increase the AP for the small object by about 0.6, and the improved HIID head can increase the AP for the small object

TABLE 7. Hyperparameter ablation experiment results, the content of the table are the APs trained by the model choosing different hyperparameters.

	R=1 (K=1, K=2, K=3)	R=2 (K=1, K=2, K=3)	R=3 (K=1, K=2, K=3)
S=1	46.3, 46.7, 46.7	47.7, 48.1, 48.1	47.8, 48.1, 48.2
S=2	46.7, 47.1, 47.2	47.9, 48.4, 48.4	48.0, 48.4, 48.4
S=3	47.2, 47.6, 48.1	48.1, 48.7, 48.7	48.3, 48.7, 48.7
S=4	47.9, 48.2, 48.3	48.5, **49.0,** 49.0	48.4, 48.9, 49.0
S=5	47.8 ,48.2, 48.2	48.5, 48.0, 49.0	48.3, 49.0, 49.0

TABLE 8. The impact of ResNest50 backbone and improved DIIH head.

Backbone	RoI Head	Prop. boxes	AP	AP_{50}	AP_{75}	AP_s	Inference Speed
ResN50	DIIH	100	43.9	56.0	51.7	34.7	23FPS
ResN50	IMP-DIIH	100	44.8	57.7	52.5	35.0	23FPS
ResN50	DIIH	300	45.2	58.0	54.0	36.0	21FPS
ResN50	IMP-DIIH	300	48.1	61.3	56.1	36.2	21FPS
Res2N50	DIIH	100	44.9	56.8	52.6	35.5	23FPS
Res2N50	IMP-DIIH	100	45.9	58.5	53.4	35.7	23FPS
Res2N50	DIIH	300	46.2	59.1	54.9	36.5	21FPS
Res2N50	IMP-DIIH	300	**49.0**	**62.3**	**56.8**	**36.8**	21FPS

by about 0.3. When the number of proposal boxes is 100, the AP of the small target detected by our Backbone model is increased by about 0.8, and the improved HIID head can increase it by about 0.3.

V. CONCLUSION

The Transformer structure has recently become a research hotspot due to its excellent performance. We hope to apply this structure to the design of traffic sign detection algorithms. Therefore, we make some improvements to Sparse R-cnn, a neural network model inspired by Transformer. Sparse R-cnn is a novel model, and its core idea is to replace hundreds of thousands of candidate anchors in the RPN network with a small set of proposal boxes. The experiments in our paper proved that the performance of the Sparse R-cnn model is better than other existing general object detection models. Based on the original Sparse R-cnn inspiration, an improved Sparse R-cnn model is proposed.

We believe that the key to designing a traffic sign detection algorithm is to improve the small object detection ability of the algorithm as much as possible. Therefore, our idea is to improve the multi-scale capabilities of the backbone and add the attention mechanism to the algorithm's RoI (Region of Interest) extraction process.

First, we propose a novel backbone for the task of traffic-sign detection. We made further improvements to the existing backbone ResNest. We enhanced the multi-scale representation ability of the backbone by constructing hierarchical residual-like connections within each single radix block in the original ResNest. In addition, we set up a branch network for recalibrating the channel feature response adaptively through the Global Average Pooling (GAP) operation and a fully connected layer. It can also be seen as the implementation of the cross-channel self-attention mechanism.

We evaluated the traffic sign detection model from the three aspects of algorithm complexity, accuracy, and robustness. We used FLOPs (Floating-point Operations), the number of parameters, and inference time to represent the complexity of the neural network architecture. We used AP(Average Precision), which was introduced by the MS COCO [49] benchmark, to evaluate the accuracy of methods. We considered the AP obtained by detecting the original dataset as "AP Clean" and that by corruption as "AP Corr." and used the percentage of "AP Corr." in "AP Clean" to represent the method robustness.

It can be concluded that our model performs the best accuracy compared with other state-of-the-art traffic sign detection models, and our method can maintain good performance for traffic-sign detection under bad weather and noise.

In the future, our research focus is to reduce the complexity of the algorithm as much as possible while ensuring the performance of the algorithm. First of all, based on the previous work and the experimental results of our algorithm in this paper, we believe that improving the backbone's multi-scale capabilities, especially the fusion of shallow information, is an effective way to improve the algorithm's detection of small objects. However, the complicated fusion method will cause the calculation cost to be too large, and the inference speed of the algorithm is too slow. In the future, we will try to design more efficient fusion feature structures, such as deleting redundant pooling layers and skipping connection layers in backbone to save computational costs and reduce unnecessary resolution loss, or design a feature pyramid network with weight sharing.

In addition, the attention mechanism is also a key method to solve difficult object detection tasks. In the future, we will optimize the loss function or further optimize the RoI head to realize the self-attention mechanism of the algorithm.

REFERENCES

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

[2] P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, C. Wang, and P. Luo, "Sparse R-CNN: End-to-end object detection with learnable proposals," 2020, *arXiv:2011.12450*. [Online]. Available: http://arxiv.org/abs/2011.12450

[3] Z. Hang, C. Wu, Z. Zhang, Y. Zhu, Z. Zhang, H. Lin, Y. Sun, T. He, J. Mueller, R. Manmatha, and M. Li, "ResNest: Split-attention networks," Apr. 2020, *arXiv:2004.08955*. [Online]. Available: https://arxiv.org/abs/2004.08955

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

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

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[7] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, *arXiv:1312.4400*. [Online]. Available: http://arxiv.org/abs/1312.4400

[8] S. Xu, "Robust traffic sign shape recognition using geometric matching," *IET Intell. Transp. Syst.*, vol. 3, no. 1, pp. 10–18, Mar. 2009.

[9] J. F. Khan, S. M. A. Bhuiyan, and R. R. Adhami, "Image segmentation and shape analysis for road-sign detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 83–96, Mar. 2011.

[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 69, pp. 91–110, Nov. 2004, doi: 10.1023/B:VISI.0000029664.99615.94.

[11] H. Bay, T. Tuytelaars, and G. L. Van, *SURF: Speeded Up Robust Features*, vol. 3951. Berlin, Germany: Springer, 2006, doi: 10.1007/11744023_32.

[12] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, no. 1, Jun. 2005, pp. 886–893, doi: 10.1109/CVPR.2005.177.

[13] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," Microsoft Res., Bengaluru, India, Tech. Rep. MSR-TR-98-14, Apr. 1998.

[14] P. Gil-Jiménez, S. Maldonado-Bascón, H. Gómez-Moreno, S. Lafuente-Arroyo, and F. López-Ferreras, "Traffic sign shape classification and localization based on the normalized FFT of the signature of blobs and 2D homographies," *Signal Process.*, vol. 88, no. 12, pp. 2943–2955, 2008.

[15] X. Baro, S. Escalera, J. Vitria, O. Pujol, and P. Radeva, "Traffic sign recognition using evolutionary AdaBoost detection and forest-ECOC classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 1, pp. 113–126, Mar. 2009.

[16] F. Zaklouta and B. Stanciulescu, "Real-time traffic sign recognition in three stages," *Robot. Auto. Syst.*, vol. 62, no. 1, pp. 16–24, 2014.

[17] A. R. Dubey, N. Shukla, and D. Kumar, *Detection Classification Road Signs Using HOG-SVM Method*, vol. 766. Singapore: Springer, 2020, doi: 10.1007/978-981-13-9683-0_6.

[18] M. Takaki and H. Fujiyoshi, "Traffic sign recognition using SIFT features," *IEEJ Trans. Electron. Inf. Syst.*, vol. 129, no. 5, pp. 824–831, 2009.

[19] U. Kamal, T. I. Tonmoy, S. Das, and M. K. Hasan, "Automatic traffic sign detection and recognition using SegU-Net and a modified Tversky loss function with L1-constraint," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 4, pp. 1467–1479, Apr. 2020.

[20] S.-K. Tai, C. Dewi, R.-C. Chen, Y.-T. Liu, X. Jiang, and H. Yu, "Deep learning for traffic sign recognition based on spatial pyramid pooling with scale analysis," *Appl. Sci.*, vol. 10, no. 19, p. 6997, Oct. 2020.

[21] D. Chutian, "A review on the extraction of region of interest in traffic sign recognition system," in *Proc. Int. Conf. Comput. Data Sci. (CDS)*, Aug. 2020, pp. 19–22, doi: 10.1109/CDS49703.2020.00010.

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," 2015, *arXiv:1512.02325*. [Online]. Available: http://arxiv.org/abs/1512.02325

[23] 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.

[24] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7263–7271.

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

[26] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020, doi: 10.1109/TPAMI.2018.2858826.

[27] J. Cao, J. Zhang, and W. Huang, "Traffic sign detection and recognition using multi-scale fusion and prime sample attention," *IEEE Access*, vol. 9, pp. 3579–3591, 2021, doi: 10.1109/ACCESS.2020.3047414.

[28] K. Sun, Y. Zhao, B. Jiang, T. Cheng, B. Xiao, D. Liu, Y. Mu, X. Wang, W. Liu, and J. Wang, "High-resolution representations for labeling pixels and regions," 2019, *arXiv:1904.04514*. [Online]. Available: http://arxiv.org/abs/1904.04514

[29] Y. Cao, K. Chen, C. C. Loy, and D. Lin, "Prime sample attention in object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11583–11591.

[30] W. Hai, W. Kuan, C. Yingfeng, L. Ze, and C. Long, "Traffic sign recognition based on improved cascade convolution neural network," *Automot. Eng.*, vol. 42, pp. 1256–1262, Sep. 2020.

[31] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6154–6162, doi: 10.1109/CVPR.2018.00644.

[32] C. Han, G. Gao, and Y. Zhang, "Real-time small traffic sign detection with revised faster-RCNN," *Multimedia Tools Appl.*, vol. 78, no. 10, pp. 13263–13278, May 2019, doi: 10.1007/s11042-018-6428-0.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[34] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 761–769.

[35] Z. Liang, J. Shao, D. Zhang, and L. Gao, "Traffic sign detection and recognition based on pyramidal convolutional networks," *Neural Comput. Appl.*, vol. 32, no. 11, pp. 6533–6543, Jun. 2020.

[36] Z. Zhao, X. Li, H. Liu, and C. Xu, "Improved target detection algorithm based on libra R-CNN," *IEEE Access*, vol. 8, pp. 114044–114056, 2020, doi: 10.1109/ACCESS.2020.3002860.

[37] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, "Region proposal by guided anchoring," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2960–2969, doi: 10.1109/CVPR.2019.00308.

[38] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5987–5995, doi: 10.1109/CVPR.2017.634.

[39] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 2011–2023, Aug. 2020, doi: 10.1109/TPAMI.2019.2913372.

[40] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 510–519, doi: 10.1109/CVPR.2019.00060.

[41] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of tricks for image classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 558–567, doi: 10.1109/CVPR.2019.00065.

[42] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.

[43] 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.

[44] J. Zhang, J. Zhang, and S. Yu, "Hot anchors: A heuristic anchors sampling method in RCNN-based object detection," *Sensors*, vol. 18, no. 10, p. 3415, Oct. 2018.

[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*. [Online]. Available: http://arxiv.org/abs/1706.03762

[46] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*. [Online]. Available: http://arxiv.org/abs/1711.05101

[47] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

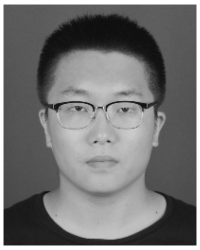
[48] X. Glorot and B. Yoshua, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intell. Statist. (ICAIS)*, Sep. 2019, pp. 249–256.

[49] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, *Microsoft COCO: Common Objects in Con- Text*, vol. 8693. Cham, Switzerland: Springer, 2014, doi: 10.1007/978-3-319-10602-1_48.

[50] C. Michaelis, B. Mitzkus, R. Geirhos, E. Rusak, O. Bringmann, A. S. Ecker, M. Bethge, and W. Brendel, "Benchmarking robustness in object detection: Autonomous driving when winter is coming," 2019, *arXiv:1907.07484*. [Online]. Available: http://arxiv.org/abs/1907.07484

[51] D. Hendrycks and T. G. Dietterich, "Benchmarking neural network robustness to common corruptions and surface variations," 2018, *arXiv:1807.01697*. [Online]. Available: http://arxiv.org/abs/1807.01697

[52] M. Lopez-Montiel, U. Orozco-Rosas, M. Sanchez-Adame, K. Picos, and O. H. M. Ross, "Evaluation method of deep learning-based embedded systems for traffic sign detection," *IEEE Access*, vol. 9, pp. 101217–101238, 2021, doi: 10.1109/ACCESS.2021.3097969.



JINGHAO CAO (Member, IEEE) was born in Shanxi, China, in 1997. He is currently pursuing the master's degree with the School of Electronic Engineering and Optoelectronic Technology, Nanjing University of Science and Technology. His main research interests include photoelectric detection and image engineering, especially in the object detection based on deep learning algorithm.

JUNJU ZHANG was born in 1979. He received the Ph.D. degree in optical engineering from Nanjing University of Science and Technology.

In recent years, he has been responsible for participated in more than 20 national, provincial or horizontal development research projects. He has published more than 30 articles in journals and conferences, of which more than 20 were indexed by SCI and EI, applied for five national invention patents, and compiled one national textbook. His main research interests include photoelectric information detection, image signal processing, photoelectric emission material theory, and preparation technology.

XIN JIN was born in Inner Mongolia, China, in 1997. He is currently pursuing the master's degree with the School of Electronic Engineering and Optoelectronic Technology, Nanjing University of Science and Technology. He is mainly engaged in the research of digital image processing.