MI-YOLO: An Improved Traffic Sign Detection Algorithm Based on YOLOv8

Shuo Wang, Yang Xu

Abstract—**Traffic sign detection plays an essential role in the technology of self-driving vehicles. Recently, deep learning methods have significantly advanced the field of traffic sign recognition. Nevertheless, faced with increasingly complex traffic scenarios, practical applications of traffic sign detection still encounter challenges, including false detections, missed detections, and reduced accuracy. To tackle these challenges, we introduce an enhanced algorithm for traffic sign detection built on the YOLOv8 model, aimed at improving performance and accuracy. Firstly, a Multi-Scale Convolutional Attention (MSCA) module is embedded into the backbone architecture to improve the model's feature extraction capabilities at multiple scales, enhancing its focus on target areas. Furthermore, a small object detection layer is added during the detection phase, effectively reducing the false positive and missed detection rates for small objects. Finally, we present the Inner-WIoU loss function for bounding boxes, which integrates a dynamic non-monotonic focusing mechanism with auxiliary boxes. This boosts the model's capability to identify objects and enhances overall detection performance. The findings from the experiments demonstrate that the enhanced algorithm obtains an mAP$_{0.5}$ value of 83.8% on the TT100K dataset, indicating a 7.8% increase compared to the baseline YOLOv8 algorithm. When compared to existing algorithms, the proposed method demonstrates competitive performance.**

Index Terms—**Traffic Sign Detection, YOLOv8, Multi-Scale Attention, Small Object Detection, Bounding Box Loss**

I. Introduction

With the ongoing advancements in the automotive industry, autonomous driving technology has been advancing rapidly. Autonomous driving systems rely on sensors to detect the vehicle's condition and its surrounding environment in real time. Then, an intelligent system performs planning and decision-making, and finally, the control system executes driving operations [1], [2]. The technology for detecting traffic signs is crucial to autonomous driving. By recognizing and interpreting road traffic signs, it delivers key information to aid vehicles in precise navigation and decision-making, enhancing both driving safety and operational efficiency [3], [4], [5], [6]. Consequently, accurately and efficiently detecting traffic signs is of considerable research significance and value for the progression of autonomous driving technology.

Early traffic sign recognition methods primarily relied on traditional image processing techniques, emphasizing the analysis of visual characteristics such as hue and geometric form found in traffic signage. However, as the number of vehicles increases and road conditions become more complex, these methods encounter difficulties in delivering real-time and precise detection of traffic signs in intricate scenarios. For instance, if traffic signs are partially blocked, damaged, or impacted by harsh weather, detection accuracy declines significantly. Additionally, these algorithms often need considerable computational power, making it challenging to fulfill the criteria for instant detection. To resolve these challenges, numerous advanced algorithms have been created in recent years to enhance the effectiveness and precision of traffic sign recognition.

The rapid growth in deep learning technology has significantly accelerated the development of object detection algorithms. The detection of traffic signs has seen marked improvements in speed, accuracy, and efficiency thanks to the integration of deep learning techniques. Some well-known algorithms for object detection include R-CNN [7] and YOLO [8]. Of these, YOLO (You Only Look Once) is notable for achieving an exceptional trade-off between speed and precision, enabling it to quickly and reliably identify objects in images, thus significantly improving detection efficiency without sacrificing recognition accuracy.

YOLOv8, introduced as an open-source deep learning model by Ultralytics on January 10, 2023, was created to handle multiple applications in computer vision. Building upon YOLOv5, YOLOv8 significantly enhances computational speed and detection precision, allowing it to perform effectively across various computer vision applications, including segmentation, detection, tracking, as well as pose estimation. Although YOLOv8 performs well in traffic sign detection, it faces practical challenges, including difficulties in identifying small targets, false positives, and missed detections. To tackle these problems, this study introduces an enhanced traffic sign detection method (MI-YOLO) derived from YOLOv8, aiming to enhance the precision of detection. The key contributions of this research include:

This paper initially integrates a attention mechanism into YOLOv8. This mechanism functions as an adaptive selection process, guiding neural networks to concentrate on the most pertinent areas of the input. Given the highly complex environment of traffic sign detection, where numerous irrelevant objects can complicate the detection process, the focus mechanism equips the model with the ability to highlight important areas, thus improving detection accuracy

Manuscript received June 28, 2024; revised October 19, 2024. This research received funding from the National Natural Science Foundation of China (61775169) and the Education Department of Liaoning Province (LJKZ0310).

Shuo Wang is a postgraduate student at the School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan 114051, China (e-mail: 1243102566@qq.com).

Yang Xu is a professor in the School of Computer Science and Software Engineering at the University of Science and Technology Liaoning, Anshan 114051, China (corresponding author, phone: 86-13889785726; e-mail: 705739580@qq.com).

and efficiency. This research introduces an innovative Multi-Scale Convolutional Attention (MSCA) module [9], which adaptively merges feature maps across multiple scales using weighted fusion, enhancing the network's representational and generalization capabilities. This method enables the model to better combine multi-scale data, significantly boosting overall effectiveness.

Additionally, due to the considerable variation in size between traffic signs and surrounding objects in images, traffic signs captured by vehicle cameras often cover only a small area of the overall image. To tackle the difficulties posed by these small objects, this paper introduces a dedicated small object detection module within YOLOv8. This layer is specifically crafted to improve detection accuracy for smaller objects, helping to minimize missed and incorrect detections while ensuring accurate recognition and classification of small traffic signs. This enhancement greatly boosts the model's practical effectiveness and dependability in real-world traffic conditions.

The performance of object detection algorithms is greatly affected by how the loss function is structured, especially the important contribution of bounding box loss. This research proposes a new bounding box loss function called Inner-WIoU. Leveraging the dynamic non-monotonic focus mechanism of Weighted Intersection over Union (WIoU), this method employs the concept of 'outlierness' to evaluate anchor boxes instead of relying exclusively on IoU, while adopting a innovative gradient allocation approach [10]. Additionally, the auxiliary bounding box loss function (Inner-IoU) accelerates convergence by adjusting auxiliary box sizes through a scale factor ratio [11]. By combining WIoU with Inner-IoU to form Inner-WIoU, this method mitigates the issues found in traditional IoU-based metrics, including intense competition between top-tier anchor boxes and adverse gradient impacts from suboptimal samples, and boosts localization precision, thus greatly enhancing overall detector performance. Experimental findings reveal that this enhanced algorithm excels in both accuracy and robustness, underscoring its potential benefits for use in autonomous driving technologies.

II. Related Work

Identifying traffic signs is a key challenge within the expansive area of object detection, remaining a prominent subject of research. Traffic sign identification can utilize traditional image analysis techniques and state-of-the-art deep learning architecture. Sugiharto et al. [12] devised a detection framework centered on color segmentation strategies. They converted RGB images into the HSI color space to more effectively identify specific colored traffic signs. Morphological operations such as erosion and labeling were then applied to refine the search areas. Features from the Regions of Interest (ROI) were collected using histogram of oriented gradients. Consequently, classification into traffic signs or non-traffic signs was conducted using either Support Vector Machines (SVM) or clustering methods. Wang et al. [13] introduced a detection approach designed to identify non-red areas in prohibition and warning signs. They developed a new red bitmap extraction method that takes into account the color relationships among adjacent pixels to enhance accuracy. In the hole filtering phase, multiple weak classifiers were used in succession to effectively reduce false positives. In the last decision phase, a SVM was used to further reduce the false alarm rate. Shape-based methods for traffic sign detection generally focus on identifying potential regions based on sign shapes. Boujemaa et al. [14] used a distance transform matching technique, which creates DT images by calculating the proximity of each pixel to its closest edge. This pixel-based approach allows for detecting objects of any shape, rather than being limited to specific shapes.

The advent of deep learning has facilitated artificial neural systems to independently identify and retrieve intricate characteristics from extensive datasets, showcasing their impressive capacity to fit data accurately. This advancement has prompted the development of many high-efficiency target recognition systems. Deep learning models for target recognition can generally be divided into two categories: two-stage and single-stage methods. Methods that operate in two stages first create proposed bounding boxes and then refine them to produce the final detection outcome. While this process increases the complexity and time required, resulting in slower detection speeds, it delivers greater precision in object localization and identification. Representative algorithms in this class are R-CNN [7] and Mask R-CNN [15]. On the other hand, single-stage detectors perform object identification in a single step, bypassing the candidate region generation phase entirely. This method minimizes computational load and boosts detection speed, making it ideal for real-time uses. Examples of prominent one-stage detectors include YOLOv1 [16], YOLOv3 [17], YOLOv5, and YOLOv7 [18].

Improvements in technology for deep learning-based object recognition have resulted in significant advancements in the study of recognizing traffic signs. Numerous scholars are concentrating on investigating methods for detecting traffic signs using deep learning. Liu et al. [19] presented a neural network architecture known as MR-CNN, capable of fusing features from various scales, which boosts recall rates and improves detection precision by upsampling deep convolutional features and merging them with shallow features to create combined feature maps. This approach employs multi-scale contextual areas to enhance feature representation, thus boosting detection efficacy for minor traffic signs. Saxena et al. [20] developed an innovative method for detecting and recognizing traffic signs through the YOLOv4. They enhanced detection accuracy by optimizing the Path Aggregation Network (PAN) for better feature propagation. Furthermore, they utilized grouped convolutional layers to decrease the model's parameter count, which in turn boosts its efficiency.

In order to tackle the diverse obstacles in detecting traffic signs, this study employs the YOLOv8 algorithm as the main methodology. Within the YOLO family, YOLOv5 is extensively utilized, while YOLOv8 stands as the latest and most effective version introduced by Ultralytics after YOLOv5. The structure of YOLOv8 is consists of backbone, neck, and detection head, similar to YOLOv5's.

The backbone of this network incorporates the split idea derived from the Cross Stage Partial Network (CSPNet) and employs a version of CSPDarknet53 as its foundational structure. Nevertheless, in contrast to YOLOv5, YOLOv8 presents the Cross Stage Partial Fusion (C2f) module, which substitutes the Cross Stage Partial3 (C3) module utilized in YOLOv5. This enhancement enables the YOLOv8 backbone to stay lightweight while acquiring more comprehensive gradient information. Furthermore, YOLOv8 retains the Spatial Pyramid Pooling-Fast (SPPF) module.

Within the neck network, in contrast to YOLOv5, YOLOv8 removes the convolutional structure during the upsampling phase of the Path Aggregation Network for Feature Pyramid Networks (PAN-FPN) and substitutes the C2f module for the C3 module. These modifications are intended to improve network performance while decreasing model complexity. YOLOv8 has introduced notable enhancements in the detection head by utilizing the current decoupled head architecture, which distinguishes between the classification head and the detection head.

Concerning anchor strategies, while anchor-based techniques excel in certain data scenarios, they often face challenges like imbalance, manual configuration complexity, and prolonged time consumption. Thus, YOLOv8 adopts an anchor-free approach. In the training phase, the model learns different bounding box shapes directly, and during inference, it predicts object dimensions by utilizing learned bounding box distances and key point locations. Some adjustments have been made to the structure of the loss function in YOLOv8. The classification loss uses the Variational Focal Loss (VFL), while the regression loss combines the Distribution Focal Loss (DFL) and Complete Intersection over Union Loss (CIOU).

III. Improved Methods

YOLOv8 includes multiple variants with sizes such as n, s, m, l, and x. Considering the strict real-time requirements for detecting traffic signs, the YOLOv8n model, which achieves high detection accuracy and is remarkably lightweight, is better suited for this application. Nonetheless, applying YOLOv8n for detecting traffic signs encounters issues concerning low accuracy, false positives, and missed

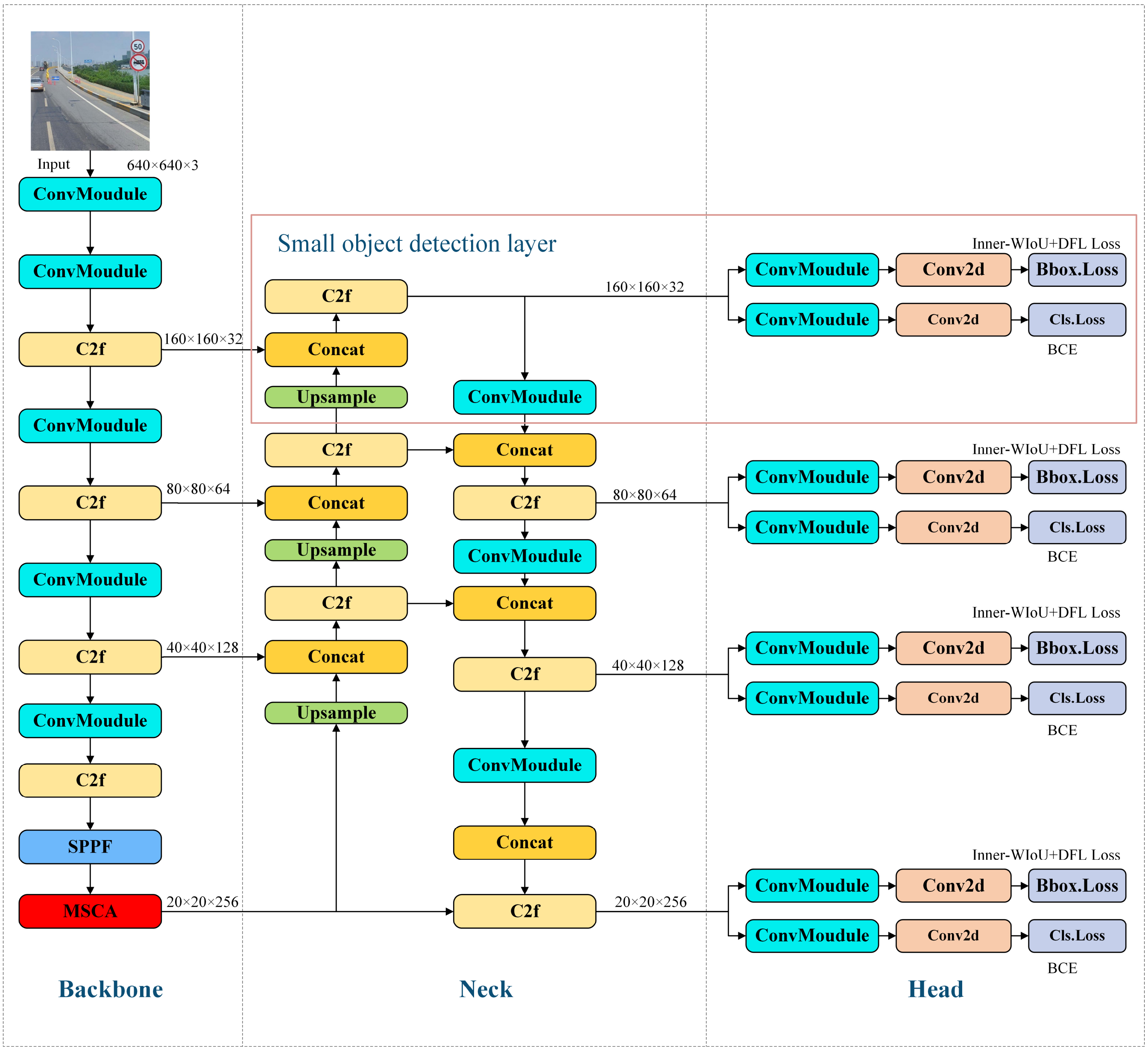

Fig. 1. MI-YOLO structural diagram

detections. To resolve these challenges, this study presents three significant improvements to the YOLOv8n algorithm. First, we incorporate a MSCA (Multi-Scale Convolutional Attention) module following the SPPF layer within the backbone architecture. This module conducts a weighted combination of feature maps from multiple scales, thus improving the backbone architecture's ability to generalize over various scales. Moreover, the MSCA module is lightweight, allowing for improved model performance without significantly increasing computational costs. Second, we introduce a detection component for identifying small objects. This component significantly reduces false positives and missed detections for minor objects, thereby improving the model's accuracy in recognizing small traffic signs. Finally, we replace the original regression loss function, transitioning from DFL Loss and CIOU Loss to DFL Loss and Inner-WIoU Loss. Inner-WIoU employs a dynamic, non-monotonic approach combined with auxiliary bounding boxes to accelerate convergence, improving the algorithm's precision in identifying targets. The enhanced model (MI-YOLO) is depicted in Figure 1.

A. *Multi-Scale Convolutional Attention*

To effectively leverage features and allow the model to concentrate on the most prominent characteristics of targets, attention mechanisms have been extensively employed in numerous fields. In this research, we integrate a Multi-Scale Convolution Attention (MSCA) within the backbone network to enhance the model's ability to capture representations across multiple scales, thus effectively retrieving contextual information. The configuration of MSCA is depicted in Figure 2. The MSCA module comprises three elements: a depthwise convolution for initial feature extraction, multiple depthwise separable convolutions for capturing features at different scales, and a 1×1 convolution for weighting the input.

$$Att = Conv_{1\times1}(\sum_{i=0}^{3} Scale_i(DW - Conv(F))) \tag{1}$$

$$Out = Att \otimes F \tag{2}$$

In Formula (1), Att represents the output attention feature map, The input feature is denoted as F, $DW-Conv$ stands for depthwise convolution. $Scale_i$, $i \in \{0,1,2,3\}$ represents the feature branch corresponding to the i-th position after applying depthwise convolution in Figure 2. $Scale_0$ denotes the skip connection, which element-wise adds the input feature with the features processed by the other three branches. In the other three branches, each branch employs two depthwise stripe convolutions to simulate a conventional depth convolution with larger kernel sizes of 7, 11, and 21, respectively. Implementing depthwise stripe convolutions in this context diminishes computational demands when compared to a conventional 2D convolution utilizing a 7×7 kernel size, by utilizing a combination of 7×1 and 1×7 convolutions, which are lighter in terms of resource usage. Furthermore, depthwise stripe convolutions improve lattice convolutions, facilitating the extraction of features resembling stripes.

In Formula (2) the features processed by the multi-branch depthwise stripe convolutions undergo an additional 1×1 convolution, which is used to compute the attention weights. Research has shown that MSCA efficiently combines features across different scales, allowing the model to dynamically modify its attention towards targets of various scales, thereby significantly improving model performance.

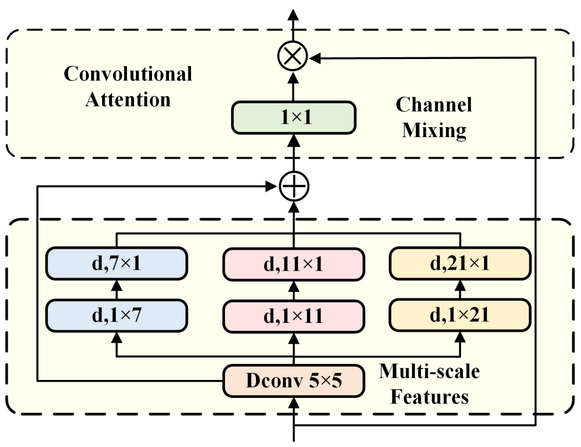

Fig. 2. MSCA structural diagram

B. *Small object detection layer*

In YOLOv8, the model automatically adjusts the input image size to fit the computational process, with a default size of 640×640 pixels. After traversing the backbone and neck networks, feature maps measuring 80×80, 40×40, and 20×20 are generated, enabling detection at these three scales. Although this architecture supports object detection across various scales, continuous downsampling significantly reduces the size of feature maps. This may result in a loss of features related to small objects, impairing the model's capability to understand their attributes, which can cause misdetections or failures to detect small objects. Furthermore, in real-world traffic sign detection situations, traffic signs appear comparatively small in relation to the entire image size, necessitating careful consideration of the requirements for detecting small objects in such tasks.

Consequently, this paper presents an additional layer specifically for detecting small objects in YOLOv8, as illustrated in Figure 1. This detection layer retrieves additional details regarding small objects from shallow features, thus minimizing the loss of feature information for those objects. The specific procedures are outlined as follows: initially, the 80×80 feature map generated by the second C2f block within the neck architecture is upsampled to a size of 160×160, related to the first C2f block's output within the backbone architecture, thus retaining detailed information about small targets. Subsequently, the 160×160 feature map obtained from the first C2f block within the backbone architecture is combined with the upsampled deep feature map, strengthening the integrated feature layer at the 160×160 scale to better capture semantic features and positional details of small targets. Following that, the combined feature map is input into the C2f block to merge features across different branches in the channel dimension. Ultimately, this feature representation is directed into the small object localization module and the subsequent Pyramid Attention Network (PAN).

C. *Optimization of Loss Function*

The object detection task encompasses determining the precise location and dimensions of objects and categorizing them into defined classes. YOLOv8 employs binary cross-entropy (BCE) loss for categorization objectives, which is particularly effective for binary classification scenarios where the output indicates probabilities for only two classes. In YOLOv8, BCE Loss generates confidence scores for each category, selecting the highest score as the certainty value of the present anchor. The formula for BCE Loss can be expressed as:

$$L_{BCE} = -w_n[y_n \cdot \log x_n + (1-y_n)\cdot \log(1-x_n)] \quad (3)$$

Within this formula, n represents a specific class to be identified, w_n is the weight for each class. x_n denotes the probability output of the model for that class, and y_n represents the true value of the class. The rectangular regression loss incorporates DFL Loss along with CIoU. The main goal of DFL Loss is to rectify inaccuracies related to predicting object boundaries. Optimizing this loss function can significantly improve object detection accuracy, particularly in images that are blurry or poorly focused. During training, a reduced DFL Loss indicates enhanced model performance in bounding box predictions. The formula for calculating DFL Loss is presented below:

$$DFL(F_n, F_{n+1}) = -((b_{n+1}-b)\log(F_n) + (b-b_n)\log(F_{n+1})) \quad (4)$$

$$F_n = \frac{b_{n+1}-b}{b_{n+1}-b_n}, F_{n+1} = \frac{b-b_n}{b_{n+1}-b_n} \quad (5)$$

The CIoU represents a more comprehensive metric than DIoU, effectively balancing the critical aspects of distance, overlap, and aspect ratio. This multifaceted approach results in enhanced performance in object detection tasks, as it not only improves the accuracy of bounding box predictions but also stabilizes the regression process. The equation for determining CIoU is presented below:

$$L_{CIOU} = 1 - IOU(A,B) + \rho^2(A_{ctr}, B_{ctr})/c^2 + \alpha\upsilon \quad (6)$$

$$\upsilon = \frac{4}{\pi^2}(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h})^2, \alpha = \frac{\upsilon}{(1-IoU)+\upsilon} \quad (7)$$

In formula (6), A represents the target bounding box detected by the algorithm or model, while B refers to the target bounding box that is real and physically present, A_{ctr} and B_{ctr} indicate the centers of A and B respectively, $\rho^2(A_{ctr}, B_{ctr})/c^2$ defines the normalized distance separating the centers of the two boxes, ensuring υ consistency in aspect ratios, and the paramete α is a positive weight.

CIoU serves as an effective method for calculating IoU; however, it presents several limitations, including a significant computational load and reduced sensitivity to small objects. Consequently, using WIoU to replace CIoU can improve the model's effectiveness and computational efficiency. WIoU employs a dynamic non-monotonic adjustment method for anchor box quality assessment and focusing that utilizes "outlierness" rather than conventional IoU for evaluating anchor box quality. This method not only diminishes the emphasis on top anchor boxes but also mitigates the negative impact of lower-quality samples on gradient allocation. As a result, WIoU can focus more on the majority of anchor boxes that better align with the data distribution, thereby improving overall detection performance. The formula for calculating WIoU is presented below:

$$L_{WIOU} = \gamma R_{WIoU} L_{IoU}, \gamma = \frac{\beta}{\delta\alpha^{\beta-\delta}} \quad (8)$$

$$R_{WIoU} = \exp(\frac{(x-x_{gt})^2 + (y-y_{gt})^2}{(W_g^2 + H_g^2)^*}), \beta = \frac{L_{IoU}^*}{L_{IoU}} \in [0,+\infty) \quad (9)$$

In formula (8), R_{WIoU} substantially improves the loss L_{IoU} for anchor boxes of moderate quality, whereas L_{IoU} reduces the influence of high-quality anchor boxes upon R_{WIoU}. This mechanism lessens the emphasis on the center-point distance in cases where anchor boxes align properly with the target boxes. In formula (9), β serves to quantify "outlierness," representing the quality of anchor boxes. High-quality anchor boxes have lower "outlier" values, resulting in reduced gradient gain during gradient allocation for these boxes, allowing intermediate-quality anchor boxes to receive more attention. For anchor boxes exhibiting larger "outlierness" values, diminished gradient gains are likewise allocated to mitigate the effects of harmful gradients generated by lower-quality samples.

This paper introduces a new bounding box calculation method called Inner-IoU, based on WIoU. During the bounding box regression phase, various regression samples are distinguished, and losses are calculated with the help of auxiliary boxes of varying dimensions. The size of the auxiliary box can be adjusted according to the size of the detected target. Larger auxiliary boxes help the model capture hard-to-recognize objects, while smaller auxiliary boxes can accelerate the convergence of the bounding box loss. The diagrammatic representation of Inner-IoU is shown in Figure 3.

In Figure 3, the locations of the center q_{gt} for both the ground truth and the internal auxiliary boxes are denoted as

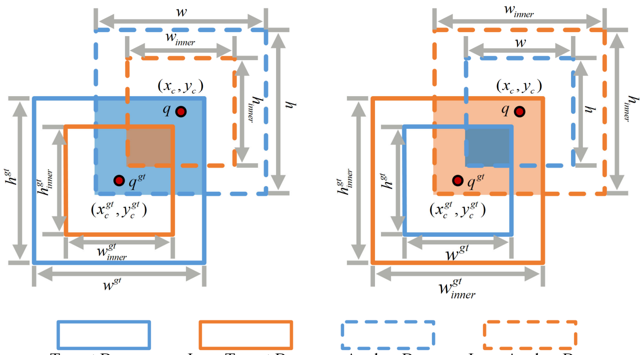

Fig. 3. Inner-IoU diagram

(x_c^{gt}, y_c^{gt}), while for the anchor and auxiliary boxes, the center point coordinates are marked as (x_c, y_c). The dimensions of the ground truth bounding box are labeled as w_{gt} and h_{gt}, while the dimensions of the detected bounding box are labeled as w and h. This scaling factor k, used as a variable, generally falls within a specific range. The Inner-IoU formula is computed as follows:

$$q_l^{gt} = q_c^{gt} - \frac{w^{gt} * k}{2}, q_r^{gt} = x_c^{gt} + \frac{w^{gt} * k}{2} \quad (1)$$

$$q_t^{gt} = y_c^{gt} - \frac{h^{gt} * k}{2}, q_b^{gt} = y_c^{gt} + \frac{h^{gt} * k}{2} \quad (2)$$

$$q_l = x_c - \frac{w*k}{2}, q_r = x_c + \frac{w*k}{2} \tag{3}$$

$$q_t = y_c - \frac{h*k}{2}, q_b = y_c + \frac{h*k}{2} \tag{4}$$

$$\begin{gathered} inn = (\min(q_r^{gt}, q_r) - \max(q_l^{gt}, q_l)) * \\ (\min(q_b^{gt}, q_b) - \max(q_t^{gt}, q_t)) \end{gathered} \tag{5}$$

$$union = (w^{gt} * h^{gt}) * (k)^2 + (w*h) * (k)^2 - inn \tag{15}$$

$$IoU^{inn} = \frac{inn}{union} \tag{16}$$

The scaling factor in Inner-IoU is the key difference from other IoU loss functions. By controlling the size of this scaling factor, the size of the auxiliary bounding boxes can be adjusted. The inclusion of auxiliary bounding boxes allows the model to focus on intermediate detection boxes, helping it learn the general features of the detected objects. Below is the definition of Inner-WIoU:

$$L_{Inner-WIoU} = L_{WIoU} + IoU - IoU^{inn} \tag{17}$$

IV. Experimental Evaluation

A. Dataset

This study utilizes TT100K as the dataset [21] for traffic sign detection experiments. TT100K, developed by Tsinghua University in collaboration with Tencent Joint Laboratory, is an extensive collection created specifically to handle traffic sign detection and classification tasks. It contains 100,000 images. The images, taken in various Chinese cities, were captured with six high-resolution, wide-angle DSLR cameras for Tencent Street View panoramas, under varying lighting and weather conditions. The original panoramas, with an 8192×2048 resolution, were cropped into smaller images of 2048×2048 pixels. Covering 221 unique traffic sign categories, the dataset has a notable class imbalance, as category frequencies vary significantly. To address this, we preprocessed the dataset by selecting categories with enough instances, resulting in 9738 images for training and testing, focusing on 45 categories with over 100 instances each. The final split resulted in 7789 images for training and 1949 for testing.

B. Experimental environment

The hardware platform for the experiments is built on an Intel Xeon Platinum 8255C processor alongside an RTX 3080 graphics card (10GB). The software environment consists of Ubuntu 18.04, Pytorch 2.0.0-gpu, and Jupyter Notebook. In the experiments, some adjustable parameters include a learning rate starting from 0.001, a batch size of 16, and an input image size of 640×640.

C. Evaluation Metrics

This paper selects precision and recall as key metrics. When evaluating model performance, we aim for a strong balance between precision and recall, so the mAP metric is also chosen to account for both factors comprehensively. Additionally, the number of model parameters and computational complexity (FLOPs) are selected as indicators to measure the model's complexity. These metrics are employed to thoroughly assess the performance of the enhanced algorithm. The equations used for calculations are presented below:

$$Precision = \frac{TP}{TP + FP} \tag{18}$$

In formula (18), TP indicates the number of accurately identified positive bounding boxes, while FP stands for those negative boxes misclassified as positive. The sum of TP and FP results in the overall number of positively predicted instances.

The recall measures the percentage of accurately predicted targets among all actual targets. The formula for this calculation is presented below:

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

In formula (19), FN refers to the count of real targets that the model failed to detect.

Average Precision (AP) can be understood as the area under the precision-recall curve. For Mean Average Precision (mAP), it is calculated by averaging the AP values across all detected object classes. The formula used to calculate mAP is presented below:

$$mAP = \frac{1}{n}\sum_{i=1}^{n} AP(i) \tag{20}$$

$$AP = \int_0^1 P(R)dR \tag{21}$$

With the IoU set to 0.5, $mAP_{0.5}$ first calculates the Average Precision (AP) for each category across all images, and then computes the average of these values for all categories. In contrast, $mAP_{0.5:0.95}$ is a more stringent criterion, measuring the mAP value with IoU thresholds that incrementally rise from 0.5 to 0.95. In the well-known object detection dataset MS COCO, the mAP evaluation metric is further categorized by object size into AP^{small}, AP^{medium}, and AP^{large}, which correspond to objects with areas below 1024 pixels, between 1024 and 9216 pixels, and above 9216 pixels, respectively. This differentiation by object size aids in evaluating the model's effectiveness in identifying objects of different sizes.

The total count of parameters in a model indicates the sum of all its learnable weights and biases, with a larger count reflecting increased complexity. FLOPs measure the total count of floating-point computations executed throughout training or inference, providing a useful metric for assessing a model's computational complexity and efficiency.

D. experimental analysis

The P-R curve combines recall and precision metrics, providing a comprehensive performance evaluation of a classification model. Particularly valuable in handling imbalanced datasets and positive class classification problems, the P-R curve more accurately reflects the model's true performance.

Figure 4 presents the P-R curves for YOLOv8 and its enhanced version, MI-YOLO, on the TT100K dataset. Two perpendicular coordinate axes represent precision and recall, respectively. From the graph, it is clear that MI-YOLO's P-R curve consistently exceeds that of YOLOv8. MI-YOLO shows marked performance improvements in the mid-to-high recall range (0.2 to 0.9), indicating its capability to maintain greater accuracy at higher recall levels for most targets. This highlights the success of the enhancements made to the YOLOv8 algorithm in this research.

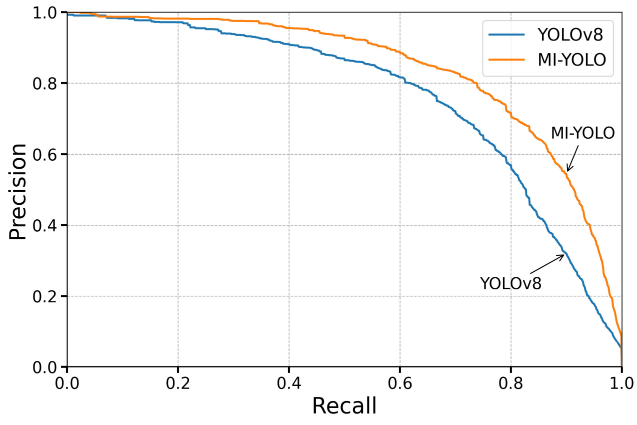

Fig. 4. P-R curve graphs of the original YOLOv8 and MI-YOLO

The MI-YOLO model demonstrates superior performance in managing complex scenes or multi-target detection tasks, achieving high detection coverage while ensuring greater detection accuracy. This capability is especially crucial for applications that require high target detection coverage rates, such as traffic sign identification.

TABLE I
COMPARISON OF SMALL OBJECT DETECTION RESULTS

Model	AP^{small}(%)	AP^{medium}(%)	AP^{large}(%)
YOLOv8n	38.4	67.5	77.0
MI-YOLO(Ours)	49.2	71.9	77.7

The scenario of detecting traffic signs is intricate because of the numerous unrelated objects present in the images. Moreover, there are issues such as unclear targets and small target sizes in both practical applications and datasets, creating challenges for detecting small objects. Figure 5 shows the distribution of traffic sign dimensions in the TT100K dataset studied here. Most traffic signs fall within the range of 0 to 96×96 pixels, whereas the total image size is 2048×2048 pixels. Consequently, most targets occupy a minor portion of the image, highlighting the necessity of recognizing small objects for the model. Thus, we conducted experiments on the detection performance of the original YOLOv8 and MI-YOLO on targets of different sizes in the TT100K dataset, as detailed in Table 1.

From Table 1, it is evident that the enhanced YOLOv8 algorithm (MI-YOLO) demonstrates significant improvements in small object detection, showing a 10.8% boost in AP^{small}, a 4.4% rise in AP^{medium}, and a 0.7% enhancement in AP^{large}. These results indicate that the MI-YOLO model has a greater advantage in detecting small traffic signs and demonstrates better generalization.

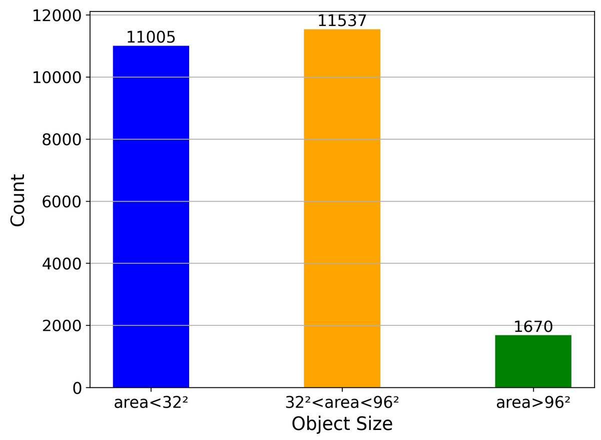

Fig. 5. Distribution of sizes for traffic signs in the TT100K dataset

E. Ablation Study

We conducted multiple ablation experiments on the TT100K dataset to verify the effectiveness of our various improvements to YOLOv8 for traffic sign detection. The experiments consisted of the following configurations: (1) the original YOLOv8n algorithm, (2) Only the MSCA module was added to the feature extraction network of the original algorithm, (3) Only change the IoU loss used in the original algorithm to Inner-WIoU loss, (4) the independent addition of the Small Object Detection Layer (SODL), (5) simultaneous incorporation of MSCA, SODL, alongside Inner-WIoU into YOLOv8n's framework. All experiments were conducted under uniform conditions using the TT100K dataset, and the comprehensive findings are presented in Table 2.

Drawing from the experimental findings shown in Table 2, the integration of the MSCA component into the backbone network resulted in a 1.8% increase in $mAP_{0.5}$, with an additional 0.1 million parameters and a 0.1 GFLOPs increase in computational complexity. Incorporating SODL resulted in a 6.3% enhancement in $mAP_{0.5}$, raising the parameter count by 0.1 million and adding 5.6 GFLOPs to the computational load. Although SODL increases computational cost, the 6.3% enhancement in $mAP_{0.5}$ justifies its inclusion.

TABLE II
COMPARISON OF ABLATION EXPERIMENTAL DATA

MSCA	Inner-WIoU	SODL	P(%)	R(%)	$mAP_{0.5}$(%)	$mAP_{0.5:0.95}$(%)	#Params(M)	FLOPs(G)
×	×	×	77.0	68.4	76.0	57.7	3.0	8.2
√	×	×	81.3	67.4	77.8	59.4	3.1	8.3
×	×	√	81.1	74.4	82.3	62.8	3.1	13.8
×	√	×	79.3	68.8	77.3	58.7	3.0	8.2
√	√	√	83.6	73.7	83.8	63.7	3.2	13.8

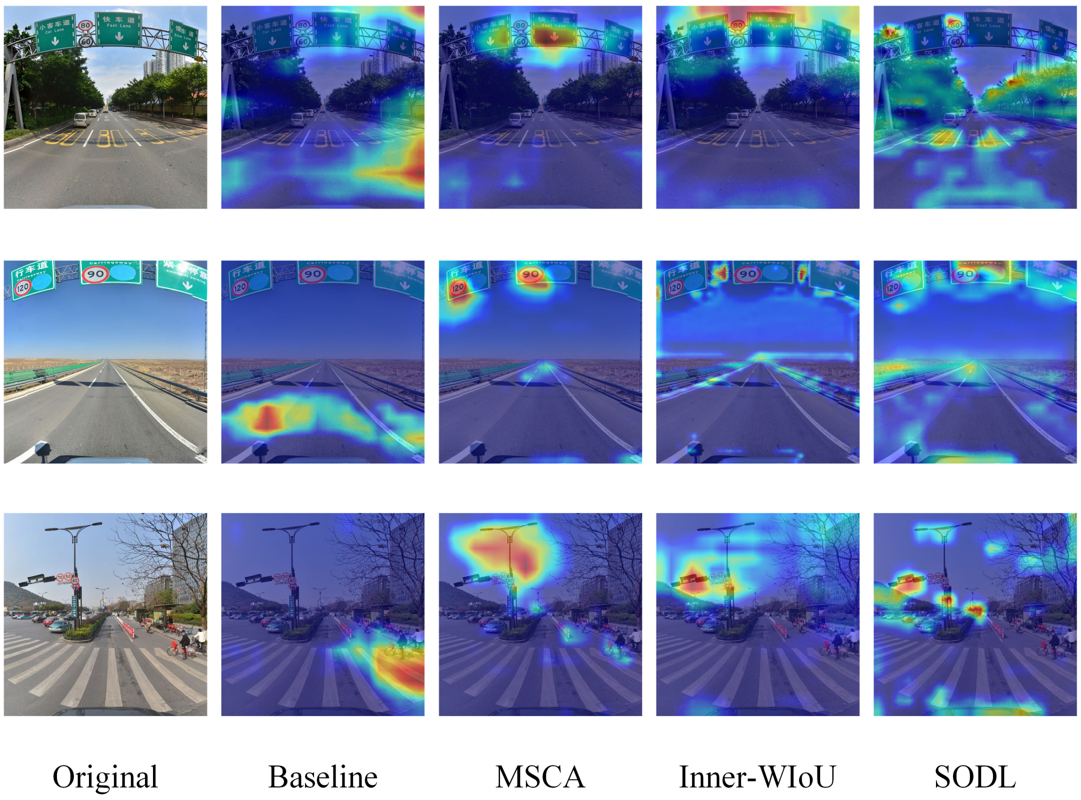

Fig. 6. Comparison of heatmaps for various improved modules

The application of Inner-WIoU to YOLOv8 yielded a 1.3% increase in $mAP_{0.5}$ without altering the parameter count or computational load. Simultaneous integration of MSCA, SODL, and Inner-WIoU into the YOLOv8 algorithm resulted in the model reaching optimal performance, improving $mAP_{0.5}$ by 7.8% compared to the original YOLOv8, with an increase of 0.1 million parameters and 5.6 GFLOPs in computational complexity. These ablation study results validate the efficacy of the enhancements made to YOLOv8 as presented in this research.

Heatmaps are commonly used to visually represent feature maps of convolutional neural networks, highlighting where the model focuses on input images. This helps clarify the model's decision-making process and enhances its interpretability. This paper uses heatmaps to display various enhancement components, selecting three random images from the dataset for visualization. From these figures, it becomes clear that the original YOLOv8 algorithm distributes its focus widely across the image, paying insufficient attention to traffic signs. Incorporating the MSCA module increases the model's attention toward traffic signs. Additionally, the integration of Inner-WIoU and the SODL further refines the model's focus on traffic signs. These findings indicate that the enhancements introduced in this research result in a significant boost for the detection capability of the model.

F. Comparison with Existing Methods

This study compares the enhanced YOLOv8 algorithm (MI-YOLO) with the original YOLOv8 and other leading detection approaches aimed at identifying traffic signs. In order to highlight the advantages of MI-YOLO, comparative experiments were performed using the TT100K dataset. The other competing advanced algorithms include YOLOv3-tiny [17], YOLOv5n, YOLOv6n [22], TOOD [23], Mask R-CNN [15], DDQ[24] and YOLOv9[25]. Multiple metrics are used to compare the performance of different detection algorithms.

As shown in Table 3, YOLOv5n and YOLOv6n have lower computational complexity compared to the enhanced algorithms; however, they are significantly lower than MI-YOLO in terms of precision and recall. Additionally, compared to YOLOv5n and YOLOv6n, MI-YOLO achieves better values for both $mAP_{0.5}$ and $mAP_{0.5:0.95}$. Overall, considering all metrics, MI-YOLO outperforms the existing algorithms and delivers the best performance. Although the improved algorithm increases parameter count by 0.1M and computational complexity by 5.6 GFLOPs compared to the original YOLOv8, the slight increase in these metrics leads to a 7.8% improvement in $mAP_{0.5}$, which is deemed acceptable given the substantial performance gains.

The comparative experiments clearly demonstrate that MI-YOLO outperforms existing detection methods for identifying traffic signs, establishing its superiority in performance.

G. Traffic Sign Detection Results

To visually demonstrate the detection outcomes of the original YOLOv8 model and the enhanced MI-YOLO version concerning traffic sign detection, this paper randomly selected two sample images from the TT100K dataset for evaluation.

Figure 7 shows the detection results, with the top two images displaying the performance of YOLOv8 and the bottom two images illustrating the performance of MI-YOLO.

The detection results presented in the images clearly indicate that MI-YOLO has a significant edge in identifying

TABLE III
COMPARATIVE EXPERIMENTAL DATA

Model	P(%)	R(%)	$mAP_{0.5}$(%)	$mAP_{0.5:0.95}$(%)	#Params(M)	FLOPs(G)
YOLOv3-tiny	74.8	51.4	59.6	46.2	12.2	19.1
YOLOv5n	74.8	66.5	74.0	56.8	2.5	7.2
YOLOv6n	65.7	57.8	64.0	49.1	4.3	12.1
TOOD	54.9	0.49	68.4	51.7	32.1	125
Mask R-CNN	68.0	50.0	60.3	47.4	44.2	187
DDQ	70.5	63.3	79.1	57.6	48.2	356.6
YOLOv9-t	73.6	63.7	72.1	54.4	3.2	14.0
YOLOv8n	77.0	68.4	76.0	57.7	3.0	8.2
MI-YOLO（ours）	83.6	73.7	83.8	63.7	3.2	13.8

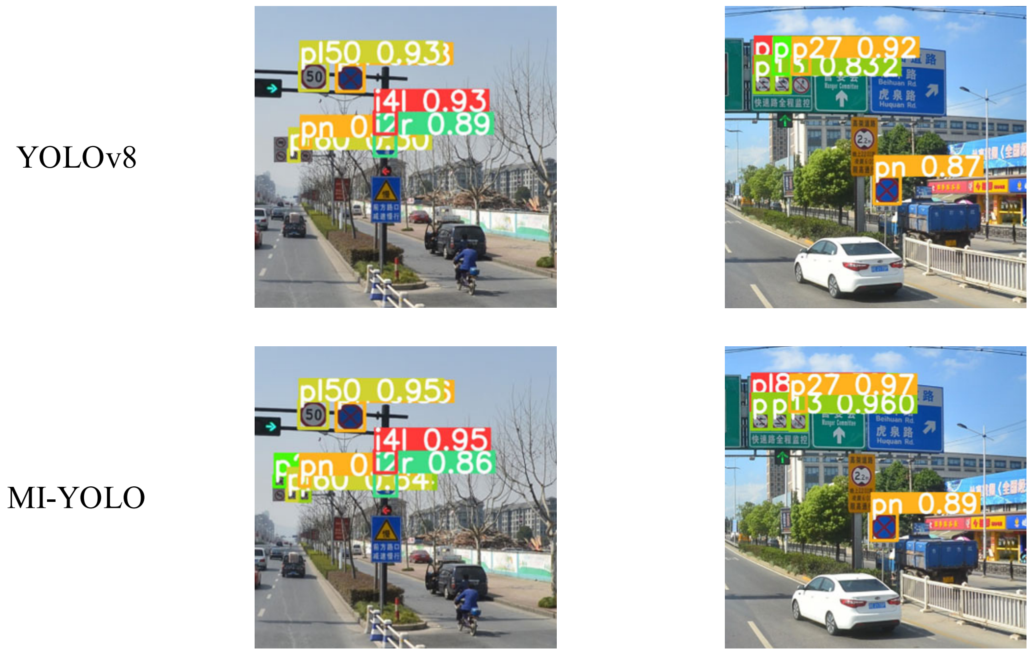

Fig. 7. Comparison of YOLOv8's detection outcomes with those of the MI-YOLO

small objects, minimizing false positives and avoiding missed detections of small traffic signs, and providing more accurate detection outcomes. In comparison to the original YOLOv8 algorithm, the proposed approach demonstrates enhanced effectiveness in detecting traffic signs.

V.CONCLUSION

This study introduces several improvements to YOLOv8 to enhance its performance in traffic sign detection. The improved algorithm, MI-YOLO, demonstrates outstanding performance. A Multi-Scale Convolution Attention (MSCA) module is integrated into the feature extraction network to capture target features across various scales and enhance its focus on relevant targets. The implementation of a Small Object Detection Layer aimed at accurately identifying small targets significantly reduces false alarms and lowers the chances of missing small targets. Lastly, Use Inner-WIoU with a new gradient allocation method and auxiliary boxes, the model achieves more accurate target localization, thereby improving overall performance. Future studies will focus on optimizing model lightweighting and improving traffic sign detection in various weather conditions.

REFERENCES

[1] L.-H. Wen and K.-H. Jo, “Deep learning-based perception systems for autonomous driving: A comprehensive survey,” Neurocomputing, vol. 489, pp. 255–270, Jun. 2022

[2] Balasubramaniam and S. Pasricha, “Object Detection in Autonomous Vehicles: Status and Open Challenges,” arXiv, 2022.

[3] M. Flores-Calero et al., “Traffic Sign Detection and Recognition Using YOLO Object Detection Algorithm: A Systematic Review,” Mathematics, vol. 12, no. 2, Art. no. 2, Jan. 2024

[4] X. R. Lim, C. P. Lee, K. M. Lim, T. S. Ong, A. Alqahtani, and M. Ali, “Recent Advances in Traffic Sign Recognition: Approaches and Datasets,” Sensors, vol. 23, no. 10, Art. no. 10, Jan. 2023

[5] Kim, J. Park, Y. Park, W. Jung, and Y. Lim, "Deep Learning-Based Real-Time Traffic Sign Recognition System for Urban Environments," Infrastructures, vol. 8, no. 2, Art. no. 2, Feb. 2023

[6] M. A. Khan, H. Park, and J. Chae, "A Lightweight Convolutional Neural Network (CNN) Architecture for Traffic Sign Recognition in Urban Road Networks," Electronics, vol. 12, no. 8, Art. no. 8, Jan. 2023

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587.

[8] Nazir and Mohd. A. Wani, "You Only Look Once - Object Detection Models: A Review," in 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom), Mar. 2023, pp. 1088–1095.

[9] M.-H. Guo, C.-Z. Lu, Q. Hou, Z. Liu, M.-M. Cheng, and S.-M. Hu, "Segnext: Rethinking convolutional attention design for semantic segmentation," Advances in Neural Information Processing Systems, vol. 35, pp. 1140–1156, 2022.

[10] Z. Tong, Y. Chen, Z. Xu, and R. Yu, "Wise-IoU: Bounding Box Regression Loss with Dynamic Focusing Mechanism." arXiv, 2023.

[11] H. Zhang, C. Xu, and S. Zhang, "Inner-IoU: More Effective Intersection over Union Loss with Auxiliary Bounding Box." arXiv, 2023.

[12] Sugiharto and A. Harjoko, "Traffic sign detection based on HOG and PHOG using binary SVM and k-NN," in 2016 3rd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), Oct. 2016, pp. 317–321.

[13] G. Wang, G. Ren, L. Jiang, and T. Quan, "Hole-based traffic sign detection method for traffic signs with red rim," Vis Comput, vol. 30, no. 5, pp. 539–551, May 2014

[14] K. S. Boujemaa, I. Berrada, A. Bouhoute, and K. Boubouh, "Traffic sign recognition using convolutional neural networks," in 2017 International Conference on Wireless Networks and Mobile Communications (WINCOM), IEEE, 2017, pp. 1–6.

[15] 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.

[16] 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.

[17] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement." arXiv, 2018.

[18] 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.

[19] Z. Liu, J. Du, F. Tian, and J. Wen, "MR-CNN: A multi-scale region-based convolutional neural network for small traffic sign recognition," IEEE Access, vol. 7, pp. 57120–57128, 2019.

[20] S. Saxena, S. Dey, M. Shah, and S. Gupta, "Traffic sign detection in unconstrained environment using improved YOLOv4," *Expert Systems with Applications*, vol. 238, p. 121836, 2024.

[21] 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.

[22] Li et al., "YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications." arXiv, 2022.

[23] Feng, Y. Zhong, Y. Gao, M. R. Scott, and W. Huang, "Tood: Task-aligned one-stage object detection," in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE Computer Society, 2021, pp. 3490–3499.

[24] S. Zhang et al., "Dense distinct query for end-to-end object detection," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 7329–7338.

[25] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "Yolov9: Learning what you want to learn using programmable gradient information," arXiv, 2024.