Enhancing Steel Surface Defect Detection: A Hyper-YOLO Approach with Ghost Modules and Hyper FPN

Guinan Wu, Qinghong Wu*

***Abstract*—Steel surface defect detection poses a significant challenge in the steel industry, aiming to enhance product quality and production efficiency. Traditional mechanical and optical detection methods exhibit relatively low efficiency and poor real-time performance in detecting acceptable defects on the surface of steel strips. This paper proposes a new model named Hyper-YOLO for steel surface defect detection in steel strips. Firstly, the CSP module in the conventional YOLO backbone network is replaced with the Ghost module. The Ghost module, a lightweight convolutional module, enhances model efficiency by reducing parameter count and computational load while maintaining satisfactory performance. Secondly, researchers replace the PAFPN module in YOLO V5 in the bottleneck section with the Hyper FPN module. Hyper FPN, an improved feature pyramid network module, leverages features at different scales for multi-level feature fusion, enhancing the model's capability to detect targets at various scales. Lastly, improvements are made in the loss functions for both training and prediction stages. The α-CIoU loss function is introduced during training to substitute the original CIoU loss function, and the α-DIoU loss function is utilized during prediction instead of the original DIoU loss function. These enhanced loss functions effectively measure the accuracy and position precision of target boxes, thereby improving the detection performance of the model.**

Through these enhancements, the Hyper YOLO model achieves an overall performance improvement of 4.58% over the baseline model. This indicates that Hyper YOLO performs outstanding surface defect detection in steel strips, providing innovative insights for the YOLO V5 model. These improvements not only elevate the accuracy and efficiency of the model but also hold significant guidance for similar research and applications.

***Index Terms*—YOLO, Object Detection, Steel Defect Detection, Computer Vision**

I. Introduction

STEEL Surface Defect Detection(SSDD) refers to the process of detecting, identifying, and locating surface defects or anomalies during steel production. These defects include cracks, scratches, dents, and other issues that may impact the quality and performance of the steel. Traditional SSDD methods typically rely on human visual inspection, which is inefficient and prone to missed detections or false positives[1–3].

Object detection technology holds significant potential for SSDD. By employing deep learning algorithms, models can be trained to automatically recognize and locate defects on steel surfaces[4]. These models can learn from large amounts of labelled data, improving detection accuracy and robustness. Object detection technology can be combined with traditional image processing methods to achieve more precise detection results. For instance, preprocessing methods can enhance image contrast and clarity before utilizing object detection algorithms for detect localization and identification. Furthermore, various object detection algorithms can be compared and selected to determine the most suitable method for SSDD.

Based on this objective, this paper proposes an improved YOLO V5 model, referred to as Hyper-YOLO, which includes the following three main enhancements:

1) Lightweight modifications to the convolutional modules in the backbone network significantly reduce module parameters and further improve computational efficiency.
2) Introduction of an improved Feature Pyramid Network (Hyper FPN) module in the bottleneck section, which utilizes multi-level feature fusion from different scales of feature maps to enhance the model's ability to detect objects of various scales.
3) New loss functions α-CIoU and α-DIoU are utilized during the training and prediction phases. Compared to the original CIOU and DIoU[5, 6], these new loss functions better measure the accuracy and positional precision of the bounding boxes, thereby improving the model's detection performance.

These enhancements enable the Hyper-YOLO model to achieve a 4.37% performance improvement over the baseline model, demonstrating its excellent detection capabilities in SSDD. The improvements enhance the model's accuracy and efficiency and provide valuable insights for research and applications in similar tasks.

II. Related work

A. YOLO V5

YOLO (You Only Look Once) V5[7] is a prominent object detection algorithm representing the latest iteration in the YOLO series. YOLO V5 transforms the object detection task into a regression problem within a single neural network, achieving rapid and accurate real-time object detection. Compared to its predecessors, YOLO V5 exhibits notable advancements in accuracy and speed[8, 9]. It adopts a lightweight network structure with fewer parameters and faster inference speed while maintaining high detection precision.

Manuscript received Feb 9, 2024; revised 7 July, 2024.

Guinan Wu is a Postgraduate of School of Electronic Information Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, China. (e-mail: 2608942251@qq.com).

Qinghong Wu* is a Professor of School of Electronic Information Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, China. (corresponding author to provide e-mail: aswqh@163.com).

A novel feature extraction module, termed "Focus," is introduced in YOLO V5 to enhance the detection performance for small targets. Additionally, YOLO V5 incorporates a feature fusion mechanism known as "PAFPN" that contributes to improved detection outcomes for multiscale targets. The model is designed with user-friendliness and flexibility in mind. It offers pre-trained model weights that support transfer learning across various computer vision tasks. Furthermore, YOLO V5 is compatible with multiple mainstream frameworks and programming languages, such as PyTorch and TensorFlow, facilitating ease of integration and utilization.

YOLO V5 has garnered widespread success in numerous application domains, including autonomous driving, intelligent surveillance, object recognition, and industrial inspection[10–12]. Its swift inference speed and accurate detection performance make it an ideal choice for real-time applications. Moreover, its open-source nature allows researchers and developers to undertake further improvements and extensions based on the YOLO V5 foundation.

B. BiFPN

BiFPN is a feature fusion mechanism used for object detection, initially proposed in the EfficientDet model and widely adopted in subsequent object detection algorithms [13].

The primary objective of BiFPN is to address the challenge of multiscale feature fusion to enhance object detection accuracy. In traditional feature pyramid networks, multiscale features are typically fused upward or downward. However, this unidirectional feature propagation may lead to information loss or blurring, especially when dealing with small targets [14, 15].

BiFPN achieves bidirectional feature propagation by introducing skip connections and incorporating upward and downward connections. It comprises a series of fusion layers, each comprising an upsampling path and a downsampling path. In the upsampling path, low-resolution feature maps are enlarged through upsampling operations and fused with high-resolution feature maps. In the downsampling path, high-resolution feature maps are reduced through downsampling operations and fused with low-resolution feature maps. BiFPN effectively retains rich information from multiscale features and merges them cohesively through this bidirectional feature propagation.

Moreover, BiFPN introduces a mechanism called "Attention" to adaptively adjust the importance of feature maps. This allows the network to automatically learn the weight distribution of feature maps based on different targets and scenes, enhancing focus on crucial targets [16].

BiFPN has demonstrated significant performance improvements in object detection. By effectively fusing multiscale features, it captures targets of varying sizes and provides more accurate predictions of positions and categories. As a result, BiFPN has been widely applied in various object detection algorithms, including EfficientNet[17] and YOLOv4[18].

In summary, BiFPN serves as a feature fusion mechanism for object detection, leveraging bidirectional feature propagation and adaptive attention mechanisms to effectively merge multiscale features, thereby enhancing the accuracy and robustness of object detection.

C. Ghost Net

GhostNet is a convolutional neural network proposed by Huawei Noah's Ark Lab. It constructs a lightweight network suitable for hardware and mobile devices, exhibiting superior performance compared to MobileNet [19]. Ghost Net comprises a series of Ghost bottlenecks, with the Ghost module as its foundational building block. The Ghost module represents a novel neural network basic unit that generates more feature maps using fewer parameters to enhance efficiency. Specifically, the Ghost module divides a regular convolutional layer into two parts: the first part involves a standard convolution but rigorously controls the number of convolution kernels, while the second part applies a series of simple linear operations to generate additional feature maps known as "ghost" feature maps. These ghost feature maps are

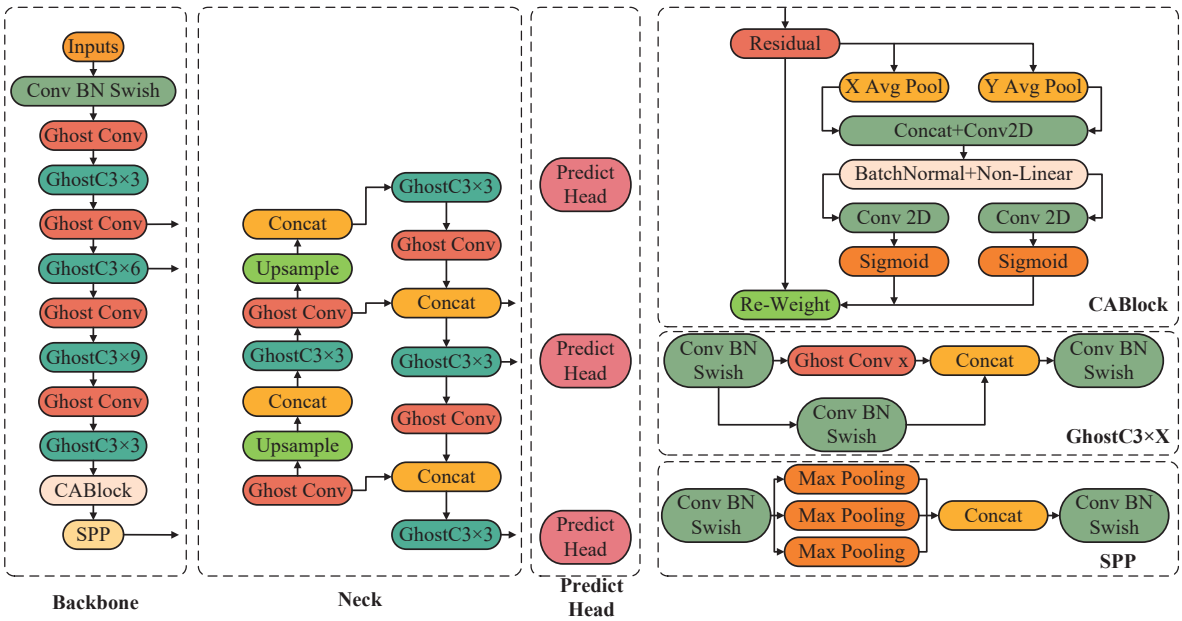

Fig. 1. Overview of Hyper YOLO's Workflow

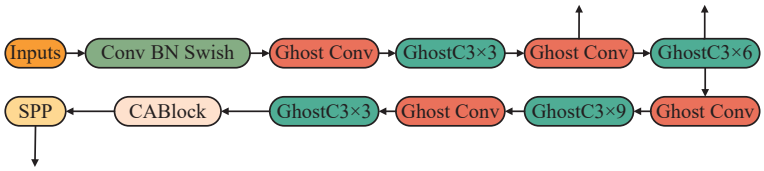

Fig. 2. Overview of Backbone

obtained through cost-effective operations from the original feature maps [20, 21].

GhostNet is a lightweight neural network constructed based on Ghost bottlenecks. In ImageNet classification tasks, GhostNet achieves a Top-1 accuracy of 75.7% under similar computational constraints, surpassing MobileNetV3's accuracy of 75.2%. The design of GhostNet aims to address the computational cost and performance challenges when applying neural networks on mobile devices. Using the Ghost module, GhostNet achieves efficient feature extraction, enabling rapid inference on mobile devices[22].

III. Methods

A. Overall Structure

In this paper, further enhancements are proposed for the YOLO V5 model. The overall flow chart is shown in Fig.1 Firstly, the CSP module in the backbone network is replaced with the Ghost module. The Ghost module, characterized by its lightweight convolutional design, enhances the model's efficiency by reducing parameter quantity and computational workload while maintaining satisfactory performance.

Secondly, researchers replace the PAFPN module in YOLO V5 in the bottleneck section with the Hyper FPN. Hyper FPN, an improved feature pyramid network module, leverages feature maps of different scales for multi-level feature fusion, thereby augmenting the model's capability to detect targets at various scales.

Lastly, improvements are introduced to the loss functions during both the training and prediction phases. The α-CIoU loss function is incorporated during training to replace the original CIoU loss function, and the α-DIoU loss function is utilized during prediction to replace the original DIoU loss function. These refined loss functions better measure target boxes' accuracy and positional precision, enhancing the model's detection performance.

Through these enhancements, the Hyper YOLO model achieves a performance improvement ranging from 3% to 5% over the baseline model. This improvement suggests that, in surface defect detection on steel strips, Hyper YOLO exhibits outstanding detection performance and introduces innovative perspectives to the YOLO V5 model. These improvements elevate the model's accuracy and efficiency and guide the research and application of similar tasks.

B. Backbone

In YOLOv5, the default backbone network is CSPDarknet53, a lightweight convolutional neural network structure. CSPDarknet53 adopts the Cross Stage Partial (CSP) architecture, wherein the input is split into two branches. One branch undergoes convolution operations, while the other engages in skip connections, enhancing feature representation and model performance. This paper takes an alternative approach by replacing the Ghost Module in the network with the CSP Module. Firstly, GhostNet surpasses CSPNet in both accuracy and speed. Its performance on the ImageNet dataset exhibits superior classification accuracy compared to CSPNet. Secondly, the Ghost Module possesses fewer parameters and computational costs than the CSP Module does . This implies that, under similar hardware conditions, the Ghost Module can facilitate faster inference and achieve commendable performance on resource-constrained devices.

Furthermore, the Ghost Module excels in model compression. It can achieve a more petite model size through low-scale network pruning and quantization without significantly compromising performance. Lastly, GhostNet demonstrates notable generalization capabilities across various tasks and datasets. It yields outstanding results in multiple computer vision tasks, including image classification, object detection, and semantic segmentation. The Ghost Module structure is illustrated in Fig. 2.

C. Hyper FPN

In YOLO V5, the Neck module introduces multiscale feature information within the Feature Pyramid Network (FPN). Its structure is shown in Fig. 3. It merges feature maps from different hierarchical levels to facilitate object detection and localization across various scales. While YOLO V5 utilizes PAFPN, this paper replaces it with Hyper FPN. Hyper FPN represents a further improvement based on BiFPN, concurrently substituting the C3 module with the GhostC3 module. Experimental evidence demonstrates that Hyper FPN adopts a bidirectional feature propagation mechanism, enabling more effective capture of feature information across different scales. This facilitates proficient feature fusion among diverse features. Hyper FPN exhibits superior performance in object detection tasks. It balances feature representation across different scales, enhancing the accuracy and recall of object detection.

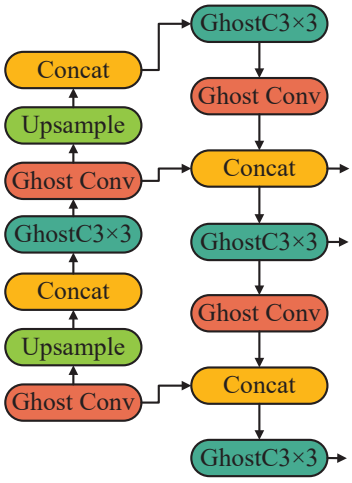

Fig. 3. Overview of Neck

D. CA Block(Coordinate Attention Block)

This paper introduces a novel efficient attention mechanism at the end of the backbone network to alleviate the loss of spatial information caused by 2D global pooling. Its structure is shown in Fig. 4. The channel attention is first decomposed into two parallel processes (in the x and y directions) during 1D feature encoding. This effectively integrates spatial coordinate information into the generated attention map. Subsequently, these two feature maps, encoding direction-specific information, are transformed into two attention maps. Each attention map captures long-range dependencies along a spatial direction of the input feature map.

Consequently, the generated attention maps preserve positional information. These two attention maps are multiplied with the input feature map to enhance its representational capacity. This attention operation distinguishes spatial directions and generates coordinate-aware feature maps. This efficient attention mechanism enables better retention of positional information within the backbone network, enhancing the model's perception of spatial features. This is particularly crucial for tasks demanding accurate spatial information, such as object detection and image segmentation.

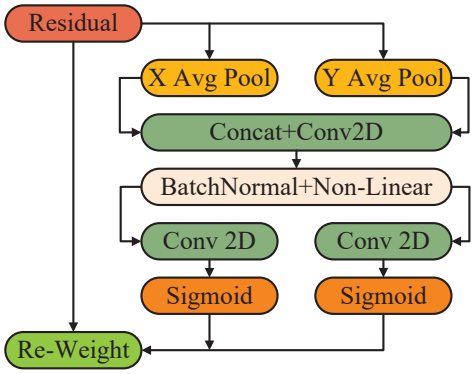

Fig. 4. Overview of CA Block

E. IoU Loss

The loss function of Hyper-YOLOv5 consists of two main components: classification loss and bounding box regression loss. The bounding box regression loss utilizes different variants of IoU to measure the matching degree between predicted and target bounding boxes. These IoU variants include:

1) GIoU Loss [23]: Builds upon IoU Loss to address the issue when the boxes do not overlap.
2) DIoU Loss [24]: Extends IoU Loss and GIoU Loss by considering the distance between the centre points of the bounding boxes.
3) CIoU Loss: Further incorporates the aspect ratio and scale information of the bounding boxes based on DIoU Loss.

These loss functions are designed to improve the matching accuracy of the bounding boxes and assist the model in better localizing and predicting target objects. In Hyper-YOLOv5, α-CIoU is employed as the IoU variant for the bounding box regression loss, which comprehensively considers the overlapping area, cases when the boxes do not overlap, the distance between the centre points of the boxes, and the aspect ratio and scale information of the boxes. The calculation for α-CIoU is shown in Equations(1) to (8):

$$\mathcal{L}_{\mathrm{IoU}} = 1 - \mathrm{IoU} \tag{1}$$

$$\mathcal{L}_{\alpha-\mathrm{IoU}} = 1 - \mathrm{IoU}^{\alpha} \tag{2}$$

$$\mathcal{L}_{\mathrm{GLOU}} = 1 - \mathrm{IoU} + \frac{|C - (B \cup B^{\mathrm{gt}})|}{|C|} \tag{3}$$

$$\mathcal{L}_{\alpha-\mathrm{GIoU}} = 1 - \mathrm{loU}^{\alpha} + \left(\frac{|C - (B \cup B^{\mathrm{gt}})|}{|C|}\right)^{\alpha} \tag{4}$$

$$\mathcal{L}_{\mathrm{DIoU}} = 1 - \mathrm{IoU} + \frac{\rho^2(\boldsymbol{b}, \boldsymbol{b}^{\mathbf{gt}})}{\mathbf{c}^2} \tag{5}$$

$$\mathcal{L}_{\alpha-\mathrm{DIoU}} = 1 - \mathrm{IoU}^{\alpha} + \frac{\rho^{2\alpha}(\boldsymbol{b}, \boldsymbol{b}^{\boldsymbol{gt}})}{\mathbf{c}^{2\alpha}} \tag{6}$$

$$\mathcal{L}_{\mathrm{CIoU}} = 1 - \mathrm{IoU} + \frac{\rho^2(\boldsymbol{b}, \boldsymbol{b}^{\boldsymbol{gt}})}{\mathbf{c}^2} + \beta\mathbf{v} \tag{7}$$

$$\mathcal{L}_{\alpha-\mathrm{CIoU}} = 1 - \mathrm{loU}^{\alpha} + \frac{\rho^{2\alpha}(\boldsymbol{b}, \boldsymbol{b}^{\boldsymbol{gt}})}{c^{2\alpha}} + (\beta v)^{\alpha} \tag{8}$$

Where C in $\mathcal{L}_{\mathrm{DIoU}}$ denotes the most minor convex shape enclosing B and B^{gt}; b and b^{gt} in $\mathcal{L}_{DIoU}$ denote central points of B and B^{gt} with $\rho()$ being the Euclidean distance and c being the diagonal length of the smallest enclosing box; and in $\mathcal{L}_{DIoU}$.

$$v = \frac{4}{\pi^2}(arctan\frac{W^{gt}}{h^{gt}} - arctan\frac{W}{h})^2 \tag{9}$$

$$\beta = \frac{\nu}{(1 - IoU) + v} \tag{10}$$

They give us the family of power IoU losses for box regression with their original versions recovered at α = 1. Note that the above α-IoU generalization can be easily extended to more complex loss functions that have multiple IoU or penalty terms.

IV. Experiment Setting

A. Datasets

Multiple mainstream metal surface defect datasets were selected for the experiments to evaluate the performance of the proposed Hyper YOLO better in detecting defects on steel surfaces. These datasets are NEU-DET, DAGM 2007, and SEVERSTAL Steel Defect Detection. Below is a brief introduction to these datasets:

1) NEU-DET (NEU Surface Defect Database): The NEU-DET dataset was collected by Northeastern University and is intended for surface defect detection, particularly in industrial metal surfaces. It comprises six typical surface defect categories: Roll Marks, Patches, Scratches, Inclusion, Cracking, and Pitted Surface. Each defect category contains 300 images, totalling 1800 images. All images are grayscale with dimensions of 200x200 pixels. This dataset is primarily used to research and evaluate the performance of various surface defect detection algorithms.
2) DAGM 2007 (Deutsche Arbeitsgemeinschaft für Mustererkennung): The German Association for Pattern Recognition organized the DAGM 2007 dataset as part of a defect detection competition. The purpose of this dataset is to evaluate the performance of automated

visual inspection systems in detecting different industrial defects. It includes synthetically generated texture images with predefined defect areas and various industrial scene defect images. Each dataset category includes multiple subsets containing defect and non-defect images. These images are used to train and test pattern recognition algorithms, assessing their ability to detect and locate defects.

3) SEVERSTAL Steel Defect Detection: The SEVERSTAL Steel Defect Detection dataset, provided by the Severstal company, is used for detecting defects on steel surfaces. This dataset includes images taken during the steel manufacturing process, annotated with different types of defects such as Scratches, Dents, Linear, and Point defects. The images vary in resolution and size and come with detailed annotations used for training and evaluating surface defect detection models. The SEVERSTAL dataset is widely utilized in deep learning and computer vision research to enhance quality control and automated detection in steel manufacturing.

Each of these datasets presents unique challenges and complexities. They play a significant role in industrial surface defect detection research, contributing to advancing related algorithms and technologies.

B. Training Environment and Detail

The experimental hardware setup in this paper consisted of an Intel(R) Xeon(R) Bronze 3104 CPU @ 1.70GHz processor, 128GB of memory, and two NVIDIA GeForce GTX TITAN XP graphics processors. The operating system used was Ubuntu 22.04. The experiments used the detection 2.28 deep learning framework based on PyTorch 1.3.

During the data preprocessing stage, the following techniques were applied: RandomFlip, Normalize, Random Crop, and Simple Copy Paste. The Adam optimizer was utilized for the training stage with a learning rate 0.0002, a batch size of 16, and a weight decay 0.05. The model was trained for 100 epochs, incorporating the EMA training technique. The FP16 precision format accelerated the training process, and the input image size was set to 224 $\times$ 224 pixels.

C. Evaluation Metrics

In the experimental section evaluating the Hyper-YOLO model, metrics such as Precision, Recall, F1 Score, and mAP are employed to assess the model's classification performance alongside FLOPs and FPS to gauge its practicality. Before formally describing the metrics mentioned above, it is essential to introduce some preliminary concepts, as depicted in Table 1, pertinent to classification tasks, which include:

1) True Positives (TP): The number of positive samples correctly predicted by the model.
2) True Negatives (TN): The number of negative samples correctly predicted by the model.
3) False Positives (FP): The number of negative samples incorrectly predicted as positive by the model.
4) False Negatives (FN): The number of positive samples incorrectly predicted as negative by the model.

By elucidating these preliminary concepts, understanding the evaluation metrics becomes more accessible. The following elucidation provides an introduction to the evaluation mentioned above metrics:

1) Precision: Precision represents the proportion of samples predicted by the classifier as positive instances that genuinely belong to the positive class. It is calculated as TP divided by the sum of TP and FP. The calculation process is shown in Equation (11).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{11}$$

2) Recall: Recall represents the proportion of actual positive samples correctly identified as positive by the classifier. It is calculated as TP divided by the sum of TP and FN. The calculation process is shown in Equation (12).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{12}$$

3) F1 Score: F1 Score represents a metric that combines Precision and Recall, representing their harmonic mean, thereby serving as an evaluation measure of the classifier's performance. The calculation process is shown in Equation (13).

$$\text{F}_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{13}$$

4) mAP (mean Average Precision): mAP represents the evaluation of performance metrics for tasks such as retrieval or object detection; it represents the average precision under the Precision-Recall curve. In classification tasks, each class is typically regarded as a binary classification problem, where Precision and Recall for each class are computed and then averaged. A higher mAP value indicates better performance of the classifier across various classes. Different IoU thresholds are employed when calculating mAP to assess the accuracy of detection results. IoU refers to overlap between the detection box and the ground truth annotation box. The suffix digits of mAP, such as mAP50 and mAP75, indicate the IoU thresholds utilized during mAP calculation. The calculation process is shown in Equation (14).

$$\text{mAP} = \frac{1}{n} \sum_{i-1}^{n} \int_0^1 P(R) dR \tag{14}$$

5) FLOPs (Floating Point Operations Per Second): FLOPs represent a metric used to measure the computational complexity of deep learning models, representing the number of floating-point operations required by the model during inference or training. FLOPs are commonly utilized to describe models' computational resource requirements, aiding in assessing model complexity, which is crucial for model selection and optimization in resource-constrained environments such as mobile devices and embedded systems.
6) FPS (Frames Per Second): FPS represents a metric used to measure the inference speed of a model, representing the number of frames processed by the model per second when handling images or videos. In practical applications, FPS is often employed to evaluate the performance of models in real-time or near real-time scenarios, such as real-time video analysis and autonomous driving applications.

D. Baseline

In the performance comparison section, the experiments in this paper are divided into two parts: a comparison of the Backbone and a comparison of the overall model. For the Backbone comparison, the current mainstream Backbone networks were selected as the baseline models, specifically:

1) FPN(Feature Pyramid Network)[25]: FPN is a backbone network used for object detection tasks, capable of extracting information from feature maps at different scales. It constructs a feature pyramid through bottom-up and top-down pathways, effectively combining multiscale information and enhancing the detection performance of small objects.
2) PAFPN(Path Aggregation Feature Pyramid Network)[26]: PAFPN is an improvement over FPN, further enhancing the information flow between features. It introduces additional path aggregation modules and increased lateral connections, improving feature representation capabilities and suitability for complex object detection tasks.
3) BiFPN(Bidirectional Feature Pyramid Network): BiFPN is an optimization of the feature pyramid network, focusing on the efficiency and performance of feature fusion. It employs a learnable bidirectional feature fusion approach and reduces computational complexity through simplified network design, making it suitable for real-time object detection tasks.

Besides the Backbone comparison, this paper compares Hyper-YOLO with several mainstream object detection models. The following are brief introductions of these models:

1) YOLO V3 (You Only Look Once Version 3): YOLOv3 is the third version of the YOLO series, a real-time object detection model. It uses a deeper Darknet as the backbone network, adopts multiscale predictions, and introduces residual connections, improving detection accuracy and speed.
2) YOLO V4: YOLOv4 is the fourth version of the YOLO series, a high-performance real-time object detection model. It introduces CSPDarknet53 as the backbone network, uses the Mish activation function, applies data augmentation strategies, and significantly enhances detection accuracy and speed.
3) YOLO V5: YOLOv5 is a new version of the YOLO series Ultralytics developed. YOLOv5 improves performance and speed through streamlined network architecture and optimized training strategies. Its simplicity and lightweight nature characterize it, making it suitable for deployment on mobile and embedded devices.
4) PPYOLO (PaddlePaddle YOLO): PPYOLO is a high-performance object detection model based on the PaddlePaddle framework. PPYOLO incorporates innovative designs, such as IoU Loss and Matrix NMS, to achieve high inference speed while maintaining high accuracy.

V. Result and Analysis

Three sets of experiments were conducted to validate the performance of the Hyper YOLO in SSDD tasks. The experimental results are presented in Tables I to III and Fig. 5 to Fig. 7. All values in the tables are the averages of five independent experiments, with the best results highlighted in bold and the second-best results underlined.

A. Comparative Performance Analysis of Hyper FPN

In Experiment 1, the primary focus was on comparing the performance of different backbones in detection models, with YOLO V5 being consistently used as the detector. Specific experimental data are presented in Tables I, demonstrating that Hyper FPN achieved either the best or second-best results across almost all metrics and datasets. BiFPN exhibited the best performance among all the baseline models and is therefore referred to as the optimal baseline model. To visually present the performance of Hyper FPN, various metrics were plotted proportionally to the optimal baseline (set as 100), as illustrated in Fig. 5.

Specifically, in the NEU-DET dataset, Hyper FPN achieved the best results in Precision, Recall, F1, mAP, mAP50, and mAP75, improving over the optimal baseline model by 2.50%, 3.53%, 3.07%, 5.73%, 6.58%, and 7.94%, respectively. For the FPS metric, Hyper FPN obtained the second-best result, showing a 2.43% deficit compared to the

TABLE I
Performance Comparison of Different Detection Models' Backbone Components

Datasets	Backbone	Metrics							
		Precision	Recall	F1	mAP	mAP50	mAP75	FPS	FLOPs
NEU-DET	FPN	0.754	0.753	0.753	0.572	0.742	0.402	**48.00**	**3.5**
	PAFPN	0.795	0.755	0.785	0.620	0.783	0.456	40.00	3.8
	BiFPN	0.801	0.765	0.782	0.628	0.790	0.466	41.00	4.0
	Hyper FPN	**0.821**	**0.792**	**0.806**	**0.664**	**0.842**	**0.503**	40.00	4.3
DAGM 2007	FPN	0.844	0.839	0.843	0.641	0.831	0.45	**48.00**	**3.9**
	PAFPN	0.890	0.845	0.879	0.694	0.877	0.511	39.00	4.2
	BiFPN	0.897	0.856	0.876	**0.745**	0.885	0.522	36.00	4.5
	Hyper FPN	**0.920**	**0.887**	**0.903**	0.739	**0.943**	**0.563**	35.00	4.8
SEVERSTAL	FPN	0.658	0.657	0.657	0.499	0.648	0.351	**41.00**	**3.1**
	PAFPN	0.694	0.659	0.685	0.541	0.684	0.398	35.00	3.3
	BiFPN	0.700	**0.697**	0.683	0.548	0.690	0.407	36.00	3.5
	Hyper FPN	**0.717**	0.688	**0.696**	**0.580**	**0.735**	**0.439**	35.00	3.8

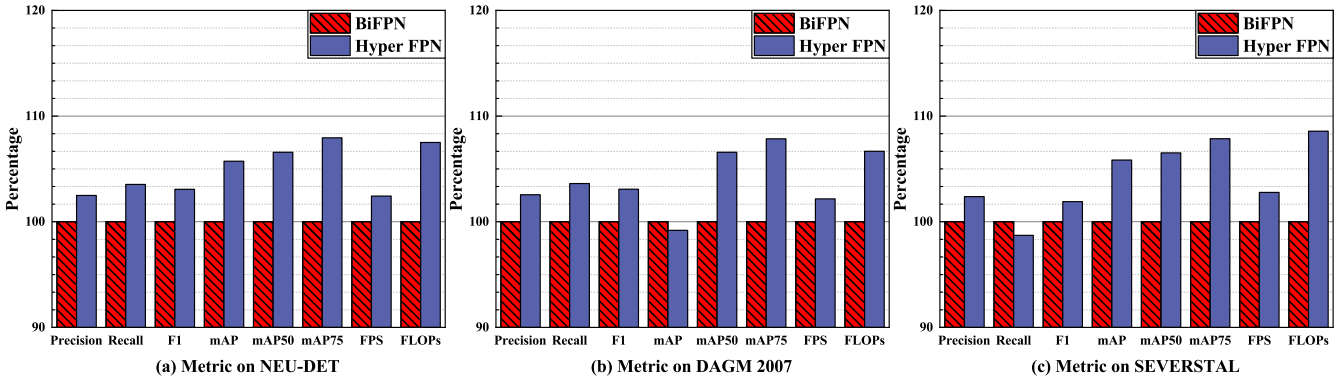

Fig. 5. Proportional Evaluation Metrics of HyperFPN Based on the Optimal Baseline Model

optimal baseline, and in the FLOPs metric, it had a 7.5% deficit. In the DAGM 2007 dataset, Hyper FPN achieved the best results in Precision, Recall, F1, mAP50, and mAP75, with improvements of 2.56%, 3.62%, 3.08%, 6.57%, and 7.85%, respectively, over the optimal baseline. It attained the second-best result in the mAP metric, trailing the optimal baseline by 0.80%. The FPS and FLOPs metrics showed deficits of 2.17% and 6.67%, respectively, compared to the optimal baseline. In the SEVERSTAL dataset, Hyper FPN achieved the best results in Precision, F1, mAP, mAP50, and mAP75, with improvements of 2.39%, 1.90%, 5.84%, 6.52%, and 7.86%, respectively, over the optimal baseline. It achieved the second-best result in the Recall metric, with a 1.30% deficit compared to the optimal baseline. The FPS and FLOPs metrics showed deficits of 2.78% and 8.57%, respectively, compared to the optimal baseline.

Across the three datasets, Hyper FPN achieved the best results in Precision and Recall metrics five times, with average improvements of 2.48% and 1.95%, respectively, over the optimal baseline. Higher Precision indicates that most positive predictions were correct, meaning the model produced fewer FP. Higher Recall indicates that the model identified the most positive samples, meaning fewer FN. Correspondingly, the F1 Score was also high, with an average improvement of 2.68% over the optimal baseline. This suggests that Hyper FPN achieved a good balance in classification, improving one metric without sacrificing another, further validating its overall superior performance. The improvements in these three metrics are mainly attributed to the bidirectional feature propagation mechanism of the GhostC3 module, which effectively integrates different features.

In terms of mAP, Hyper FPN showed a 3.59% improvement over the second-best result. Notably, at IoU thresholds of 0.5 and 0.75, Hyper FPN outperformed the second-best model by 6.56% and 7.88%, respectively, in mAP50 and mAP75. A higher mAP50 indicates the model can accurately detect objects under the relatively lenient condition of IoU $\geq$ 0.5, meaning it can effectively identify most objects while allowing some localization errors. A higher mAP75 indicates the model can accurately detect objects under the stricter condition of IoU $\geq$ 0.75, meaning it not only identifies objects but also locates their boundaries more precisely. High values in both mAP50 and mAP75 reflect the model's overall excellent performance in detection tasks, identifying and detecting most objects. Furthermore, the model's ability to maintain high performance under different IoU thresholds indicates robustness and stability, implying it is less affected by threshold variations. This robustness is mainly attributed to adding the CA Block module after the Backbone, an efficient attention mechanism that better preserves positional information within the Backbone network, enhancing the model's spatial feature perception, which is crucial for image segmentation tasks.

The main drawbacks are in the FPS and FLOPs metrics. Hyper FPN shows an average deficit of 16.67% in FPS compared to the best result; however, since the model with the best FPS performance has significant accuracy deficits, this difference is not meaningful. Compared to the optimal baseline model, Hyper FPN averages only one frame less, indicating similar performance in image processing speed, still achieving real-time detection. Although Hyper FPN has an average FLOPs deficit of 7.82% compared to the optimal baseline, indicating a need for more computational resources, considering its accuracy improvements and the FPS deficit of only one frame, the additional resource consumption is acceptable.

In summary, as the Backbone of the detector, Hyper FPN outperforms mainstream models and lays a solid foundation for the Hyper YOLO model.

B. Comparative Performance Analysis of Hyper YOLO

Based on Experiment 1, Experiment 2 utilized the best-performing Hyper FPN as the Backbone and employed various YOLO networks as detectors. The specific experimental data are presented in Table II. Hyper YOLO achieved the best or second-best results across nearly all evaluation metrics on three datasets. YOLO V5 demonstrated the best performance among the baseline models and is referred to as the optimal baseline model for comparative purposes. Like the previous experiment, the performance metrics are plotted as percentages relative to the optimal baseline model (set as 100), as illustrated in Fig. 6.

Specifically, in the NEU-DET dataset, Hyper YOLO achieved the best results in Precision, F1, mAP, mAP50, and mAP75, with improvements of 3.27%, 2.06%, 3.87%, 5.23%, and 10.31% over the optimal baseline model, respectively. It obtained the second-best result in the Recall metric, with a 0.89% improvement over the optimal baseline. The FPS and FLOPs metrics showed deficits of 4.44% and 13.16%, respectively, compared to the optimal model. In the DAGM 2007 dataset, Hyper YOLO achieved the best results in Precision, Recall, F1, mAP, mAP50, and mAP75, with improvements of 1.11%, 4.88%, 2.99%, 3.83%, 5.25%, and 10.31% over the optimal baseline model, respectively.

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT YOLO NETWORKS AND HYPER YOLO

Datasets	Detector	Metrics							
		Precision	Recall	F1	mAP	mAP50	mAP75	FPS	FLOPs
NEU-DET	YOLO V3	0.721	0.723	0.721999	0.541	0.682	0.400	**52.00**	**3.2**
	PPYOLO	0.762	0.763	0.762	0.586	0.743	0.429	50.00	3.8
	YOLO V4	0.782	**0.794**	0.788	0.611	0.771	0.450	40.00	4.0
	YOLO V5	0.795	0.785	0.790	0.620	0.783	0.456	45.00	3.8
	Hyper YOLO	**0.821**	0.792	**0.806**	**0.644**	**0.824**	**0.503**	43.00	4.3
DAGM 2007	YOLO V3	0.708	0.709	0.708	0.605	0.763	0.448	**48.00**	**3.6**
	PPYOLO	0.733	0.744	0.738	0.656	0.832	0.480	46.00	4.3
	YOLO V4	0.746	0.753	0.749	0.684	0.864	0.504	42.00	4.5
	YOLO V5	0.794	0.779	0.786	0.694	0.877	0.511	41.00	4.3
	Hyper YOLO	**0.803**	**0.817**	**0.810**	**0.721**	**0.923**	**0.563**	40.00	4.6
SEVERSTAL	YOLO V3	0.727	0.709	0.718	0.503	0.539	0.473	**50.00**	**2.9**
	PPYOLO	0.762	0.733	0.747	0.510	0.546	0.502	47.00	3.3
	YOLO V4	0.786	0.755	0.770	0.532	0.533	0.529	45.00	3.5
	YOLO V5	0.790	0.782	0.786	**0.557**	**0.603**	0.498	46.00	3.4
	Hyper YOLO	**0.828**	**0.830**	**0.829**	0.546	0.593	**0.560**	44.00	3.6

The FPS and FLOPs metrics showed deficits of 2.44% and 6.98%, respectively, compared to the optimal model. In the SEVERSTAL dataset, Hyper YOLO achieved the best results in Precision, Recall, F1, and mAP75, with improvements of 4.81%, 6.14%, 5.43%, and 12.45% over the optimal baseline model, respectively. It obtained the second-best results in the mAP and mAP50 metrics, with deficits of 1.97% and 1.66%, respectively, compared to the optimal baseline, indicating similar performance levels. The FPS and FLOPs metrics showed deficits of 5.88% and 4.35%, respectively, compared to the optimal baseline model.

Hyper YOLO achieved the best results in the Precision and Recall metrics five times across the three datasets, indicating that most of its optimistic predictions were correct, resulting in fewer false positives. The Hyper YOLO model also identified almost all positive samples, resulting in fewer false negatives. Correspondingly, the F1 Score, as the harmonic mean of Precision and Recall, achieved the best results three times, indicating that the Hyper YOLO model successfully balanced Precision and Recall without sacrificing one for the other. Furthermore, Hyper YOLO achieved the best results multiple times in the mAP, mAP50, and mAP75 metrics, with improvements of 1.91%, 2.94%, and 11.02%, respectively, over the second-best results. The mAP metric comprehensively considers the model's performance across all categories, with a higher mAP indicating that the model can detect objects with high Precision and Recall across different categories. A higher mAP50 suggests the model performs well when the overlap between the bounding box and the target is high. Notably, the more stringent mAP75 metric showed an improvement of over 10% compared to other models, indicating that the Hyper YOLO's predicted bounding boxes had a higher overlap with the actual bounding boxes, reflecting superior performance under high overlap requirements.

Similar to Experiment 1, the FPS value of Hyper YOLO showed a significant deficit compared to the best and second-best results; however, the model with the best FPS performance had a substantial accuracy deficit, rendering it incomparable. Therefore, the optimal baseline model was used as a comparison. The results indicated Hyper YOLO had an average FPS deficit of 3.74% compared to the optimal baseline model, primarily due to the increased computational load of the new IoU Loss function. This was also reflected in the FLOPs metric, with an average deficit of 8.64% compared to the optimal baseline model. However, this trade-off is deemed acceptable, considering the significant improvements in other metrics.

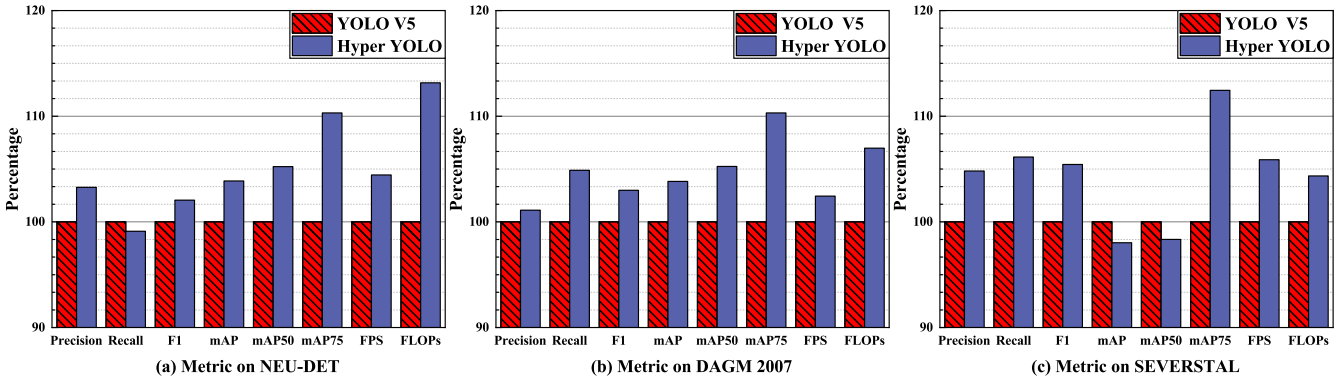

Fig. 6. Proportional Evaluation Metrics of HyperYOLO Based on the Optimal Baseline Model

TABLE III
PERFORMANCE COMPARISON OF LOSS FUNCTIONS USING HYPER YOLO MODEL

Loss Function	Metrics							
	Precision	Recall	F1	mAP	mAP_{50}	mAP_{75}	FPS	FLOPs
CIoU	0.815	0.782	0.798	0.655	0.816	0.495	40.00	4.3
α-IoU	0.821	0.792	0.8062	0.664	0.824	0.430	40.00	4.3

C. Comparative Performance Analysis of Loss Functions

Additionally, this paper conducted a performance comparison of loss functions, with the model uniformly employing Hyper YOLO on NEU-DET dataset. The experimental results are shown in Table III. The results indicate that the proposed α-IoU outperforms the original CIoU, primarily due to CIoU's poor performance in handling small objects. However, SSDD tasks focus on detecting small objects[27]. Thus, the proposed α-IoU can also be considered an optimization method for small object detection.

Furthermore, introducing the CIoU loss function may lead to a more unstable training process. Fig. 7 displays the standard deviations of the five evaluation metrics for both loss functions across five experiments. Since the F1 Score is the harmonic mean of Precision and Recall, it is not presented. It can be observed that the standard deviation of α-IoU is more petite, indicating higher stability during the training process.

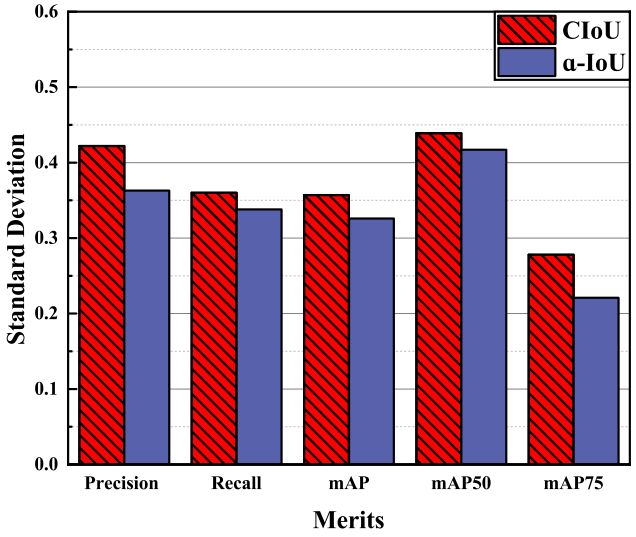

Fig. 7. Comparison of Standard Deviations of Two Loss Functions

D. Summarize

Hyper YOLO demonstrates superior performance across all evaluation metrics on the NEU-DET dataset, surpassing current mainstream visual classification detection models based on YOLO. This highlights its potential application value in surface defect detection. The GhostC3 module in Hyper FPN ensures that the Backbone component can better capture feature information at different scales. In contrast, the CA Block module effectively captures information bi-directionally in the feature maps, mitigating positional information loss caused by traditional 2D global pooling, which is crucial for object detection tasks. The improved IoU loss function makes the model more sensitive to small-sized targets, thus enhancing its performance in defect detection tasks.

VI. CONCLUSION

This paper addresses surface defect detection on steel strips. We have substantially enhanced the conventional YOLO object detection model by replacing the backbone network's CSP module to improve computational accuracy. Additionally, optimizations have been applied to the PAFPN module to heighten the model's sensitivity towards small-sized targets. These refinements collectively result in a notable improvement in the model's detection capabilities across a diverse range of target sizes.

Furthermore, we have refined pertinent loss functions, introducing a novel loss function termed α-CIoU. This new loss function more accurately gauges the precision and positional accuracy of bounding boxes, effectively enhancing the model's detection performance. The proposed detection model in this paper achieves significant accuracy improvements in surface defect detection on steel strips, thereby contributing to the advancement of the steel industry.

Notably, the improvement methodologies and insights proposed in this paper exhibit a high level of generality, allowing for seamless transplantation onto analogous tasks. This adaptability is poised to enhance efficiency in related domains. The contributions made in this paper hold the potential to positively impact industrial applications and foster technological advancements in relevant fields.

REFERENCES

[1] X. Wen, J. Shan, Y. He, and K. Song, "Steel Surface Defect Recognition: A Survey," *Coatings*, vol. 13, no. 1, p. 17, 2022.
[2] B. Tang, L. Chen, W. Sun, and Z.-k. Lin, "Review of Surface Defect Detection of Steel Products Based on Machine Vision," *IET Image Processing*, vol. 17, no. 2, pp. 303–322, 2023.
[3] X. Tao, D. Zhang, W. Ma, X. Liu, and D. Xu, "Automatic Metallic Surface Defect Detection and Recognition with Convolutional Neural Networks," *Applied Sciences*, vol. 8, no. 9, p. 1575, 2018.
[4] G. Fu, P. Sun, W. Zhu, J. Yang, Y. Cao, M. Y. Yang, and Y. Cao, "A Deep-learning-based Approach for Fast and Robust Steel Surface Defects Classification," *Optics and Lasers in Engineering*, vol. 121, pp. 397–405, 2019.
[5] Y.-F. Zhang, W. Ren, Z. Zhang, Z. Jia, L. Wang, and T. Tan, "Focal and Efficient IOU Loss for Accurate Bounding Box Regression," *Neurocomputing*, vol. 506, pp. 146–157, 2022.
[6] X. Wang and J. Song, "ICIoU: Improved Loss Based on Complete Intersection Over Union for Bounding Box Regression," *IEEE Access*, vol. 9, pp. 105 686–105 695, 2021.
[7] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object Detection Using YOLO: Challenges, Architectural Successors, Datasets and Applications," *Multimedia Tools and Applications*, vol. 82, no. 6, pp. 9243–9275, 2023.
[8] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of YOLO Algorithm Developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022.
[9] J. Du, "Understanding of Object Detection Based on CNN Family and YOLO," in *Journal of Physics: Conference Series*, vol. 1004. IOP Publishing, 2018, pp. 120–129.
[10] W. Zhang, Y. Zhao, Y. Guan, T. Zhang, Q. Liu, and W. Jia, "Green Apple Detection Method Based on Optimized YOLOv5 Under Orchard

Environment," *Engineering Letters*, vol. 31, no. 3, pp. 1104–1113, 2023.
[11] Y. Liu and Y. Tian, "DCMS-YOLOv5: A Dual-Channel and Multi-Scale Vertical Expansion Helmet Detection Model Based on YOLOv5," *Engineering Letters*, vol. 31, no. 1, pp. 373–379, 2023.
[12] X. Zhang and Y. Tia, "Improved YOLOv5s Traffic Sign Detection," *Engineering Letters*, vol. 31, no. 4, pp. 1883–1893, 2023.
[13] Y. Feng, X. Wang, Y. Xin, B. Zhang, J. Liu, M. Mao, S. Xu, B. Zhang, and S. Han, "Effective Feature Enhancement and Model Ensemble Strategies in Tiny Object Detection," in *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*. Springer, 2020, pp. 324–330.
[14] J. Chen, H. Mai, L. Luo, X. Chen, and K. Wu, "Effective Feature Fusion Network in BIFPN for Small Object Detection," in *2021 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2021, pp. 699–703.
[15] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and Efficient Object Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 781–10 790.
[16] S. Naddaf-Sh, M.-M. Naddaf-Sh, A. R. Kashani, and H. Zargarzadeh, "An Efficient and Scalable Deep Learning Approach for Road Damage Detection," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 5602–5608.
[17] M. Tan and Q. Le, "Efficientnet: Rethinking Model Scaling for Convolutional Neural Networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6105–6114.
[18] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of YOLO Algorithm Developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022.
[19] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "Ghostnet: More Features from Cheap Operations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1580–1589.
[20] Y. Chen, X. Dai, D. Chen, M. Liu, X. Dong, L. Yuan, and Z. Liu, "Mobile-former: Bridging Mobilenet and Transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5270–5279.
[21] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei, "Rethinking Bisenet for Real-time Semantic Segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9716–9725.
[22] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
[23] X. Liu, J. Hu, H. Wang, Z. Zhang, X. Lu, C. Sheng, S. Song, and J. Nie, "Gaussian-IoU Loss: Better Learning for Bounding Box Regression on PCB Component Detection," *Expert Systems with Applications*, vol. 190, p. 116178, 2022.
[24] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 12 993–13 000.
[25] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2117–2125.
[26] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path Aggregation Network for Instance Segmentation," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 8759–8768.
[27] S. Li and W. Liu, "Small Target Detection Model in Aerial Images Based on YOLOv7X+," *Engineering Letters*, vol. 32, no. 2, pp. 436–443, 2024.