IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Lightweight Barcode Positioning Algorithm Based on YOLO Model

CHAOCHAO LI[1], QINGTAO ZENG[1], and LIKUN LU[1]

[1]School of Information Engineering, Beijing Institute of Graphic Communication, Beijing 102600, China

Corresponding author: Qingtao Zeng (zengqingtao@bigc.edu.cn)

This work was supported in part by Classification Development of Beijing Municipal Universities - Construction Project of Emerging Interdisciplinary Platform for Publishing at Beijing Institute of Graphic Communication - Key Technology Research and Development Platform for Digital Inkjet Printing Technology and Multifunctional Rotary Offset Press under Grant 04190123001/003; in part by the Operation and Maintenance Agreement for Pediatric Teaching Audio-Video Database at Peking Union Medical College Hospital under Grant 11000301281; in part by the PhD Research Startup Fund - Research on Concept Learning Methods for Symbolic Block Data under Grant 27170124012.

ABSTRACT With the advancement of Internet of Things (IoT) technology, barcode automatic identification systems have played a crucial role. An improved YOLO-MCG barcode localization algorithm was proposed to address the problems of interference, inefficiency, and poor real-time performance encountered by traditional barcode detection methods in complex backgrounds and field environments. First, to reduce the number of parameters and computational complexity, the MobileNetv3-small network is used to replace the backbone network. Second, a Convolutional Attention Mechanism Module (CBAM) is introduced to enhance the perception ability of target information and improve the detection performance of the model. In addition, Generalized Intersection over Union (GIOU) and Focal Loss are used as loss functions to enhance the localization precision of the model. The experiments show that the improved model achieves an mAP@0.5 of 97.8%, Precision of 96.4%, and Recall of 93.9%. The amounts of parameters are reduced to 35% of Yolov8, the amount of computation load is reduced to 25% of Yolov8, and the FPS is 105. The improved model can be used on resource-constrained mobile devices while meeting real-time requirements.

INDEX TERMS Barcode, positioning, YOLO, lightweight

I. INTRODUCTION

The development of the Internet of Things (IoT) [1] technology has enabled the real-time collection of vast amounts of data. However, the effective management and utilization of these data presents a challenge. In this context, the application of barcode detection technology has a significant potential. By identifying barcodes on items, IoT systems can benefit from convenient and reliable means of identification, enabling rapid recognition, localization, and tracking of items from production to sales to use, facilitating automated data recording and management. Furthermore, barcode detection contributes to reducing the time required for data collection in various sectors such as commerce, healthcare, transportation, and manufacturing, enabling fast barcode recognition and localization on embedded devices and mobile platforms. Moreover, the advent of QR codes [2,3] has been inspired by the development of barcodes. Both barcodes and QR codes now coexist in various aspects of daily life. Currently, in China, over 500,000 enterprises and more than 50 million products are equipped with barcode labels, with a barcode adoption rate exceeding 95% [4]. Therefore, ensuring that automatic identification systems can quickly and accurately read barcode information relies crucially on precise and real-time barcode localization algorithms.

Camera-based barcode detection systems are primarily applied in the IoT, and thus, they must be efficient and accurate under all challenging environmental conditions and complex scenarios. In complex environments, the visibility of barcodes is significantly affected by different lighting conditions such as strong light, shadows, and reflections, as well as the varying materials carrying the

barcodes [5]. Moreover, during transmission, barcodes may experience geometric distortion, which can further affect detection. For instance, when scanning barcodes printed with ink on copper plate paper, bright light and the paper's texture may render parts of the barcode invisible, resulting in incomplete or failed localization; in the automated manufacturing of lithium battery electrodes, barcodes are often printed on damaged or scratched electrodes to record errors for subsequent processes. However, when the barcode and the background color are the same, the localization becomes difficult, making decoding impossible. Therefore, an accurate barcode localization solution is essential to address these issues.

In addition, another major challenge faced by barcode localization is how to achieve fast and real-time localization. Barcode localization algorithms are typically applied in scenarios that require quick responses, such as in automation systems on production lines, logistics management, retail checkout, and more. In these scenarios, real-time performance is crucial. If the algorithm is too complex, the consumption of computational resources can lead to processing delays, affecting system efficiency and user experience. Additionally, many barcode detection systems now run on embedded or mobile devices, which typically have very limited computational resources. Using traditional deep learning models may consume too many computational resources, causing the device to run slowly or even crash. Lightweight designs can enable efficient barcode recognition on limited hardware resources. Many existing barcode localization algorithms use deep learning or traditional computer vision techniques, which often require complex computational models and substantial computational resources. This leads to bottlenecks in processing speed and real-time performance, especially on resource-constrained embedded devices such as mobile platforms and industrial equipment.

In conclusion, this paper chooses the YOLO algorithm as our baseline and proposes an improved lightweight YOLO barcode localization model. The main contributions of this algorithm are as follows:

1. We adopts the YOLO model as the basic framework and replaces the backbone network with the MobileNetv3 network to reduce computational complexity and shorten detection time.

2. The introduction of the Convolutional Attention Mechanism Module (CBAM) improves the precision of barcode recognition.

3. Using Generalized Intersection over Union (GIOU) and Focal Loss as loss functions enhances the model's localization precision.

The rest of the paper is organized as follows: Section 2 describes related work, Section 3 introduces the basic theoretical model and improved model of this study, Section 4 introduces experimental parameters and evaluation metrics, Section 5 presents the experimental results verifying the performance of the proposed algorithm. Section 6 provides a discussion, and Section 7 concludes the paper.

II. Related work

Scholars, both domestically and internationally, have proposed various solutions to the barcode localization problem. The current barcode localization methods are divided into those based on traditional digital image processing and those based on deep learning. Image-processing-based barcode localization algorithms typically involve preprocessing the image using operations such as grayscale conversion, binarization, and filtering to reduce irrelevant information interference. Subsequently, morphological operations are applied to enhance barcode region features for coarse localization and obtain the barcode's position. Finally, fine localization is achieved using edge detection [6], Hough transform [7], or affine transformation algorithms [8]. For example, Creusot et al. [9] proposed a Maximum Stable Extremal Region (MSER) [10] algorithm, which initially extracts stable extremal regions using MSER technology from the image as candidate barcodes. Then, these candidate regions are clustered in the Hough space for barcode detection and recognition. However, the algorithm's performance is poor when the image is blurry. Creusot et al. [11] addressed the blurring problem of the MSER algorithm by using the Line Segment Detector (LSD) algorithm [12], thus improving detection precision and real-time performance. Katona et al. [13] proposed two methods for barcode localization: one involves binary thresholding followed by template matching, and the other uses edge detection followed by distance transformation. However, these two methods can only be used in specific scenarios. Yi *et al.* [14] proposed a barcode localization method suitable for high-resolution images with multiple barcodes. Firstly, edge features of the barcodes are extracted, then the barcode regions are annotated using bidirectional contour labeling. Finally, barcode regions are extracted through affine transformation, effectively improving the localization precision and speed. However, the complex preprocessing, edge detection algorithms, and feature matching processes make digital image processing-based methods time-consuming and less robust to changes in image background.

In recent years, neural networks have made remarkable achievements in the field of computer vision, and corresponding research has been conducted on barcode detection algorithms based on deep learning [15]. Barcode localization is performed within the scope of object detection. Object detection methods can be divided into two major categories: region proposal-based object detection algorithms, such as faster region-based convolutional neural network(Faster R-CNN)[16], which first generate candidate regions or bounding boxes, and

then classify these candidate regions and perform bounding box regression, and single-stage object detection algorithms, such as you only look once(YOLO) [17] and single shot detection(SSD)[18], which directly complete object detection within a single stage without explicitly generating candidate regions. Convolutional neural networks after being trained on large datasets can extract robust features and have been applied in many important fields. Guo *et al.* [19] proposed an algorithm based on convolutional neural networks, which innovatively introduced the bottleneck residual block (BRB), achieving a higher recognition accuracy. Zhang *et al.* [20] used the SSD detection framework to locate barcodes, performed rotation correction, and then decoded them using a decoder, solving the positioning difficulties caused by distortion, dirt, and obstruction under harsh conditions. The above work was conducted in a simple background without considering background duplication. Li *et al.* [21] proposed a method to train a barcode detection model using Faster R-CNN framework in complex backgrounds and used an adaptive manifold (AM) filter for deblurring, finally, MSERs were used for barcode orientation detection. Qiao *et al.* [22] proposed a one-dimensional barcode localization method based on two deep learning models. Firstly, the Faster R-CNN model was used to automatically detect barcode regions, followed by using the ResNet-34 [23] model for barcode orientation calibration. Wan *et al.* [24] proposed a lightweight CenterNet network for 2D barcode localization, which enables faster and more accurate localization.

Compared to other models, the YOLO model offers exceptionally high real-time performance, a simple structure, and fast detection speed. Xiao *et al.* [25] proposed a method that combines the YOLO object detection algorithm with the LSD image processing algorithm. They used LSD to precisely locate the barcodes outlined by YOLO, removing complex backgrounds, and achieved significant improvements in precision and speed. Yue *et al.* [26] proposed a YOLO-SM algorithm, which primarily addresses the barcode localization and detection problem under single-class and multiple-deformation conditions, achieving relatively good results. Do *et al.* [27] introduced a computer vision-based supermarket product management system using the YOLOv3[28] algorithm, effectively alleviating the inefficiency of barcode management in supermarkets. Robert *et al.* [29] combined the YOLOv3-tiny3l algorithm with the U-Net network to tackle issues with noisy, poorly exposed, and blurred barcode images encountered during truck loading in real-world scenarios.

Despite the significant performance of these methods, their model size and inference speed limit their practical application in the real world. In contrast, YOLO-based models are faster, have smaller model sizes, and are suitable for real-time inference. However, barcode detection performance remains relatively low, which is still an issue to be addressed. Therefore, we recreated a dataset that includes potential problematic cases mentioned above and modified the YOLO model to achieve a good balance in model performance.

III. METHODOLOGY

A. YOLO MODEL

YOLO, short for "You Only Look Once," is an object detection algorithm. It employs a single-stage detection method by treating the entire detection process as a single neural network inference, enabling real-time object detection. Compared to two-stage detection methods, YOLO offers faster speed and higher efficiency because it does not require additional complex processing steps, making it more suitable for barcode detection and recognition scenarios. YOLO incorporates rotation, translation, concatenation, and other data augmentation techniques in data processing, effectively enhancing object detection performance. Compared to YOLOv3 and YOLOv4[30], YOLO's structure is further optimized for more accurate object detection. The YOLO network achieves higher detection precision and faster inference speed.

YOLO can be divided into four parts: Input, Backbone, Neck, and Head.

In the Input part, YOLO employs Mosaic data augmentation, adaptive anchor boxes, and adaptive image scaling methods. Mosaic randomly selects four images, chooses one of them as the background image, and then stitches the four augmented images together based on random scaling, cropping, and arrangement. Therefore, through Mosaic data augmentation, YOLO can increase the diversity of data during training, improve the model's robustness and generalization ability, thus enhancing the performance of object detection.

The backbone part is primarily responsible for extracting features from input images. It utilizes the Focus module, Cross-Stage Partial Network (CSPNet), and Spatial Pyramid Pooling module (SPP). Before inputting the image into the backbone, the Focus module slices and merges the image to improve processing speed. CSPNet integrates gradient information into feature maps, alleviating the gradient vanishing problem. The SPP module addresses image distortion in cropping and scaling operations by fusing information from feature maps of different sizes.

The neck part is responsible for multi-scale feature fusion of feature maps and passing these features to the prediction layer. YOLO uses feature pyramid network (FPN) and path aggregation network (PANet) for feature fusion. FPN merges feature information through top-down up-sampling, while PANet passes features from bottom to top. Through the feature fusion of FPN and PANet, the representation capability of features and the detection performance for objects of different sizes are further improved. The network

architecture of YOLO is shown in Figure 1. The three boxes at the bottom of the figure depict the details of the backbone, neck, and detection head, representing the overall architecture of the YOLO model. The remaining four images show the details of the different colored blocks from the three sub-figures below.

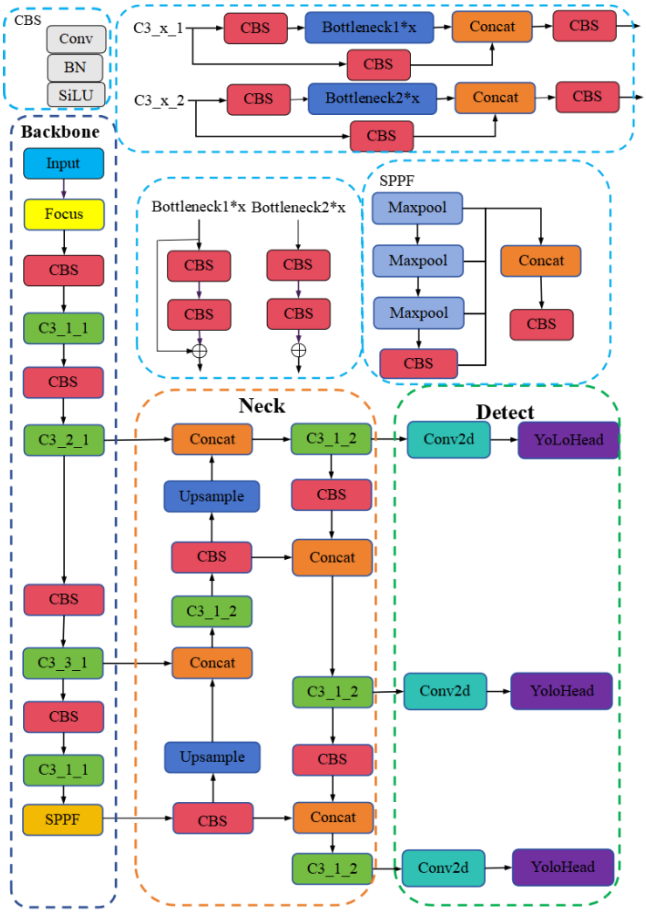


FIGURE 1. **YOLO architecture diagram.**

B. IMPROVEMENTS IN THE YOLO NETWORK STRUCTURE

1) MOBILENETV3-SMALL

In order to achieve fast and accurate localization with limited computational resources, and to meet the application requirements in mobile devices and complex environments, we replace the backbone network of YOLO with the MobileNetv3-Small network [31]. MobileNetv3-Small, released by Google in 2019, is a lightweight convolutional neural network architecture. Through a more efficient network structure and lightweight design, it significantly reduces the computational load of the model, thereby enhancing real-time detection speed. This improvement is necessary for applications that require barcode recognition on embedded or mobile devices. For instance, in warehouses, robots can efficiently scan and identify goods without consuming excessive computational resources.

The main architecture of MobileNetv3 consists of a series of bneck blocks, which include depth-wise separable convolution [32], inverted residual connection and channel attention mechanism (SE) [33].

The depth-wise separable convolution consists of two processes: depth-wise convolution, where each channel corresponds to a convolutional kernel in the channel direction, and pointwise convolution, where a normal 1x1 convolution outputs the specified number of channels. Depth-wise convolution can more effectively extract features, and pointwise convolution can combine features from different channels, enhancing feature representation. At the same time, it reduces the number of parameters, lowering the risk of overfitting. The implementation process of depth-wise separable convolution is shown in Figure 2.

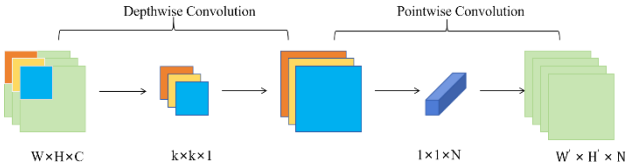


FIGURE 2. **Implementation process of depth-wise separable convolution. The left half of the figure represents the depthwise convolution process, while the right half represents the pointwise convolution process.**

For the l-th layer of a network with a three-dimensional input tensor x^l, the input to depth-wise separable convolution is denoted as $x^l \in R^{H^l \times W^l \times D^l}$. Here, H^l, W^l, D^l represent the height, width, and depth of the input, respectively. Selecting an element (i^l, j^l, d^l) from the input x^l, where (i^l, j^l, d^l) specifies any specific triplet element indicating it resides in the d-th channel at position (i^l, j^l). The convolution is performed with D filters $f, f \in R^{H \times W \times D^l \times D}$, each of size $H \times W$. According to reference [34], the definition of depth-wise separable convolution is:

$$y_{i^{l+1}, j^{l+1}, d} = \sum_{i=0}^{H} \sum_{j=0}^{W} \sum_{d=0}^{D} f_{i,j,d} \times x^l_{i^{l+1}+i, j^{l+1}+j, d} \qquad (1)$$

where $x^l_{i^{l+1}+i, j^{l+1}+j, d}$ refers to the elements of the input $(i^{l+1}+i, j^{l+1}+j, d)$.

SE mainly consists of two parts: Squeeze and Excitation, which are used to enhance the model's focus on input features for better differentiation between different targets and backgrounds.

The features first undergo squeeze, which is a global average pooling process, aggregating feature maps of the input feature map $U, U \in R^{H \times W \times C}$ across spatial dimensions

$H \times W$ to generate channel-wise statistics $z, z \in R^C$. After compression, the feature map is reduced to a $1 \times 1 \times C$ vector. Essentially, the statistics z_c are generated by reducing the spatial size of U along the channel dimension. According to reference [33], The definition of squeezing is:

$$z_c = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j) \quad (2)$$

where z_c refers to c-th statistic, u_c refers to feature map of the c-th channel, with a size of $H \times W$, and (i, j) refers to value at that position on the feature map.

The Excitation operation models the global descriptor vector through two fully connected layers to learn the weights for each channel. These weights are normalized by an activation function to ensure their sum is 1. Then, they are multiplied element-wise with the original feature map to enhance the relevant features of each channel. According to reference [33], the definition of Excitation is:

$$s = F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_2 \delta(W_1 z)) \quad (3)$$

where δ refers to ReLU activation function, and σ refers to sigmoid activation function. $W_1 \in R^{\frac{C}{r} \times C}, W_2 \in R^{C \times \frac{C}{r}}$,these are the weight matrices of two fully connected layers. r is the dimensionality reduction factor. The final output of the block is obtained by rescaling the transformation output U with the activations, the implementation process of SE is as shown in Figure 3. According to reference [33], the definition of Scale is:

$$x_c = F_{scale}(u_c, s_c) = s_c \bullet u_c \quad (4)$$

where $x = | x_1, x_2, \ldots, x_c |$, x_c is the feature map of a certain feature channel in x , $F_{scale}(u_c, s_c)$ refers to channel-wise multiplication between the feature map $u_c \in R^{H \times W}$ and the scalar s_c .

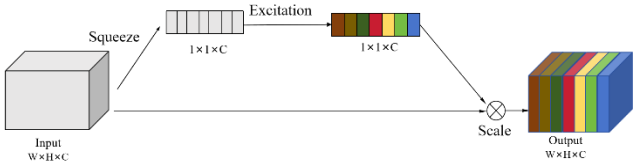


FIGURE 3. SE conceptual diagram

The bneck block in MobileNet uses an inverted residual structure, which means it first upsamples the dimensions using a 1×1 convolution, then performs feature extraction using 3×3 depth-wise separable convolution, and finally downsamples the dimensions using a 1×1 convolution. This structure is opposite to the residual structure of Residual Network (ResNet), hence it is called an inverted residual structure. The specific implementation process of the MobileNetv3 bneck is shown in Figure 4.

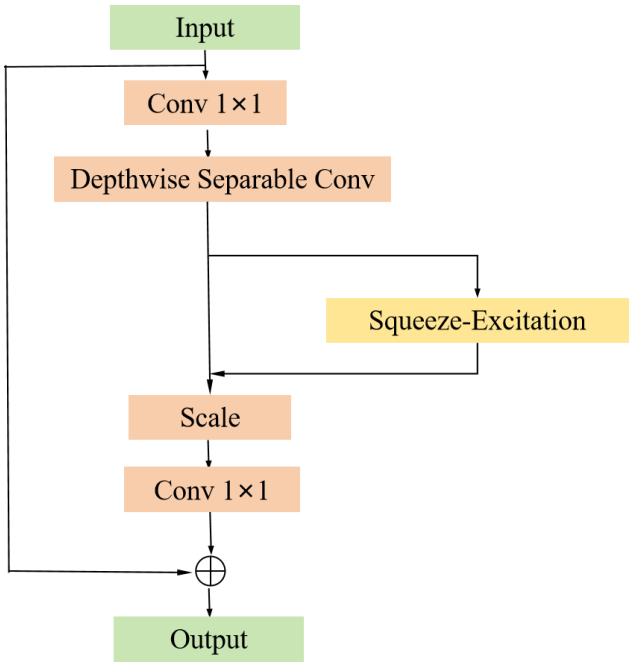


FIGURE 4. The bneck block in MobileNetv3-small.

2) CBAM

There are many irrelevant features in barcode images that can affect the robustness of detection models. To help the model focus better on the useful features in the image, we attempt to apply weighting to the input feature map to enhance the important regions. Therefore, we introduced the CBAM [35]. CBAM enhances the key information in the barcode image (such as the black and white stripes of the barcode) by weighting both the channels and the spatial dimensions of the feature map, thereby improving the accuracy of barcode recognition and preventing missed detections. CBAM is a simple yet effective attention module, which consists of a channel attention module (CAM) and a spatial attention module (SAM). It can be seamlessly integrated into CNN architectures and trained end-to-end. The structure of the convolutional attention mechanism is shown in Figure 5.

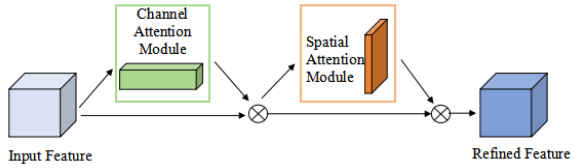


FIGURE 5. CBAM structure diagram, consisting of the channel attention (CA) module and spatial attention (SA) module.

The CAM mainly extracts features from the channels. The channel attention module performs global average pooling and global max pooling on the input features, reducing the dimensionality across channels. Then, a shared multi-layer

perceptron generates the channel attention weights. This helps the network focus more on important channels (such as barcode lines, background noise, etc.), thereby extracting more meaningful features. According to reference [35], the specific calculation formula for the Channel Attention module is:

$$M_c(F) = \sigma(MLP(AvgPool(F) + MLP(MaxPool(F)))) = \sigma(W_1(W_0(F^c_{avg})) + W_1(W_0(F^c_{\max}))) \quad (5)$$

where the input is a feature $F \in R^{H\times W\times C}$, σ refers to sigmoid activation function, and MLP refers to a multi-layer perceptron, W_0, W_1 are two weight matrices, $W_0 \in R^{C/r\times C}, W_1 \in R^{C\times C/r}$.

The SAM primarily performs feature extraction in the spatial domain. The spatial attention module takes the results from the channel attention module, applies max pooling and average pooling to generate two channel descriptions of size 1×H×W, and then stacks the tensors together via a fully connected operation. Finally, a convolution operation and sigmoid activation function are used to obtain the weight coefficients. This fusion of channel and spatial dimensions helps better capture the position and spatial relationships of the targets (focus on the area where the barcode is located and suppress background noise). According to reference [35], the specific calculation formula for the Spatial Attention module is:

$$M_s(F) = \sigma(f^{7\times7}([AvgPool(F); MaxPool(F)])) = \sigma(f^{7\times7}[F^s_{avg}; F^s_{\max}]) \quad (6)$$

where $f^{7\times7}$ represents a convolution operation with a filter size of 7x7, and σ denotes the sigmoid function.

In the Neck layer of the YOLO model, the CBS module downsamples the feature map by changing the stride of the convolution operation, capturing low-level local features. Additionally, during the FPN (top-down) upsampling process, the resolution of the feature map is gradually restored, maintaining a high resolution. In the PAN (bottom-up) stage, after adding CBAM, the feature maps at each layer are enhanced with high-level semantic information, improving multi-scale object detection. By utilizing the adaptive channel and spatial attention mechanism, CBAM enhances the network's ability to express these low-level features. Therefore, adding CBAM after CBS effectively improves the model's understanding and representation of high-level semantic information, which can reduce the missed detection rate for barcodes that are farther from the camera. This article adds CBAM modules to the four branches of the neck network to help the model better capture target features, thereby improving the model's performance and robustness. The improved model is called YOLO-MCG, and a simplified overview is shown in Figure 6.

3) IMPROVEMENT OF LOSS FUNCTION

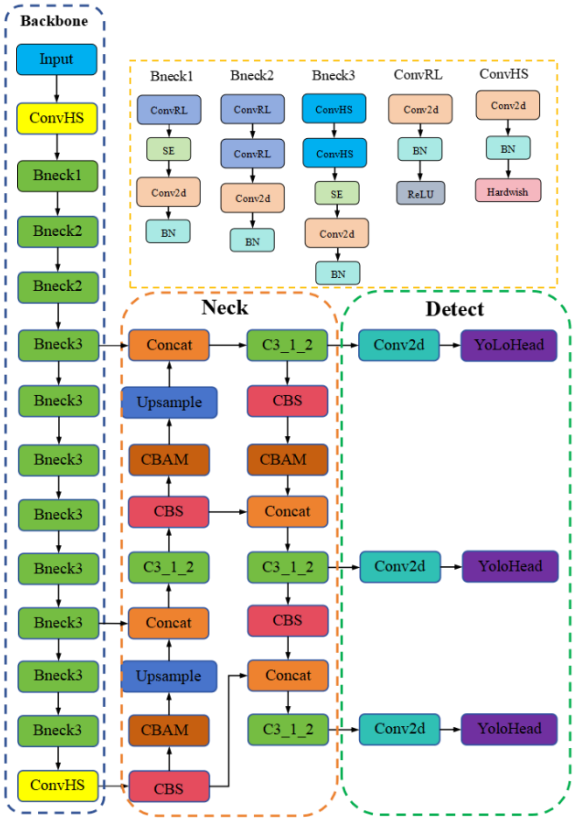


FIGURE 6. YOLO-MCG architecture diagram (omitting description of modules shared with YOLO model).

In the YOLO model, we use the GIOU loss function to replace the original CIOU loss. Compared to CIOU, the GIOU function solves the issue where IOU fails to correctly reflect the intersection when the predicted box and the ground truth box do not overlap. GIOU not only focuses on the overlapping area but also considers other non-overlapping regions. Therefore, GIOU can better reflect the degree of overlap between the two, improving both the training speed and inference accuracy. In this paper, we propose a method that combines the GIOU loss function [36] with focal loss [37] to replace the original CIoU loss function, thus optimizing the bounding box loss.

According to reference [36], the definition of the GIOU loss function as follows:

$$IoU = \frac{|B \cap B^{gt}|}{|B \cup B^{gt}|} \quad (8)$$

$$GIOU = IOU - \frac{|C \setminus (B \cup B^{gt}|}{|C|} \quad (9)$$

$$L_{GIoU} = 1 - GIoU = 1 - IOU + \frac{|C \setminus B \cup B^{gt}|}{|C|} \quad (10)$$

where $B^{gt} = (x^{gt}, y^{gt}, w^{gt}, h^{gt})$ is the ground-truth, $B = (x, y, w, h)$ is the predicted box, and C is the minimum enclosing box of B and B^{gt}.

The use of the GIOU loss function enhances the accuracy and robustness of the barcode localization algorithm. By optimizing the relationship between the shape, size, and position of the bounding boxes, GIOU improves the localization accuracy, reduces interference from occlusion and background, and also enhances the stability of training and the model's convergence speed. For barcodes that may be occluded or distant, GIOU can better adapt to these characteristics, ensuring efficient and accurate localization in various complex scenarios.

Focal Loss is a loss function designed to address the problem of class imbalance, particularly in object detection tasks. It aims to tackle the imbalance between a large number of background class samples and a small number of target class samples in object detection.

In the standard cross-entropy loss function, all samples are weighted equally, which can cause the model to focus excessively on easily classifiable background samples when they are abundant, while neglecting the fewer target class samples. Focal Loss adjusts the weights of the loss function to prioritize difficult-to-classify samples, effectively addressing the problem of class imbalance.

According to reference [37], the classic computation formula for Focal Loss is:

$$FL(p) = -(1-p_t)^{\gamma}\log(p_t) \tag{11}$$

$$p_t = \begin{cases} p, y=1 \\ 1-p, y=0 \end{cases} \tag{12}$$

where $y \in \{1,0\}$ specifies the ground-truth class and $p \in [0,1]$ denotes the estimated probability for the class with label y = 1. γ is the tunable focusing parameter.

According to reference [38], we rewrite the formula for Focal Loss as follows:

$$L_{FL}(x,y) = -\sum_{i=1}^{K} y_i (1-q_i(x))^{\gamma}\log q_i(x) \tag{13}$$

where x is the input sample, y is the ground truth label, and $q_i(x)$ is the model's predicted probability for the input sample x, representing the probability that sample xxx belongs to class i. $(1-q_i(x))^{\gamma}$ is the scaling factor, where γ is the tunable focusing parameter. $\log q_i(x)$ is the cross-entropy loss for class i. When γ= 0, the focal loss reduces to the traditional cross-entropy loss $L_{CE}(x,y) = -\sum_{i=1}^{K} y_i \log q_i(x)$. As γ increases, the impact of the modulation parameter gradually becomes larger. By combining GIOU with focal loss, the model's detection performance can be more effectively improved, and the accuracy of localization is enhanced, especially in complex environments where barcodes may be partially obscured or distorted due to perspective issues. For example, in warehouse item stacking, barcodes may not be fully visible, but the combination of GIOU and focal loss ensures accurate localization and identification of barcodes even in such situations. This makes the model suitable for applications requiring precise localization and efficient scanning, such as automated warehousing systems, smart logistics, and unmanned retail.

4) OPTIMIZATION ALGORITHMS

The training process of deep learning models is an optimization process aimed at finding optimal parameters that minimize the loss function. Optimization algorithms continuously adjust model parameters to gradually decrease the loss function, thereby improving model performance. Using appropriate optimization algorithms in YOLO can accelerate convergence, reduce training time, and efficiently find global optimal solutions to enhance detection accuracy and training stability.

SGD (Stochastic Gradient Descent) is a gradient-based optimization algorithm that updates parameters by computing gradients for each sample or batch. However, SGD updates using single samples or batches are prone to noise interference and high fluctuations. Its stochastic nature can lead to getting trapped in local minima, limiting its ability to escape them.

To address these problems, AdamW[39] is employed for model optimization in this study. AdamW utilizes separate adaptive learning rates for different parameters based on their first and second moment estimates of gradients. It incorporates momentum to accelerate gradient descent and independently manages weight decay, applying it only to the model's weight parameters. Compared to SGD, AdamW effectively balances the speed and direction of parameter updates during model training, mitigating the risk of local minima and significantly improving convergence speed. Algorithm 1 provides a detailed description of the construction process of the Adam algorithm.

Algorithm 1 AdamW

Input: α(lr), β_1 =0.9, β_2 =0.999(exponential decay rate), ε= 10^{-8} ,λ(weight decay factor)

Initialize: t←0(time step), $\theta_{t=0}$ (parameter vector), $m_{t=0} \leftarrow 0$ (first moment vector), $v_{t=0} \leftarrow 0$ (second moment vector), $\eta_{t=0}$ (schedule multiplier)

Repeat:

t←t+1

$\nabla f_t(\theta_{t-1}) \leftarrow SelectBatch(\theta_{t-1})$

$g_t \leftarrow \nabla f_t(\theta_{t-1})$

$m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_2$

$v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$

$\hat{m}_t \leftarrow m_t / (1-\beta_1^t)$

$\hat{v}_t \leftarrow v_t / (1-\beta_2^t)$

$\eta_t \leftarrow SetScheduleMultiplier(t)$

$\theta_t \leftarrow \theta_{t-1} - \eta_t(\alpha \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon) + \lambda_t \theta_{t-1})$

until *stopping criterion is met*

return optimized parameters θ_t

IV. MATERIALS AND METHODS

A. DATASET AND ENVIRONMENT

The dataset in this study comprises an open-source barcode dataset and a synthetic barcode dataset. The aim is to create a barcode detection dataset under complex backgrounds. The open-source dataset includes Muenster and Coco datasets. Since the open-source Muenster dataset has a limited number of barcode images that are not suitable for network learning, a total of 5327 images were obtained by reorganizing the open-source dataset and combining it with synthetic data. The training set contains 3999 images, while the test set contains 1328 images. Use the labeling tool LabelImg to annotate the dataset in YOLO format. The software and hardware environment and parameters used in this experiment are shown in Table 1.

TABLE I
Experimental platform

	Configuration
Operating System	Windows11
CPU	Intel(R) Core (TM) i5-9700 CPU @3.10GHz *8
Device Memory	16.0GB
Graphics Processor	GeForce GTX 3050
Experimental Language	Python 3.10.13
Accelerated Environment	CUDA 11.8
Deep learning Framework	Torch 2.1.0

B. EXPERIMENTAL PARAMETER SETTINGS AND EVALUATION METRICS

During training, the input images were resized to 640×640, and AdamW was used as the optimization function for model training. The training epochs were set to 300, with a batch size of 16 and an initial learning rate of 0.01. The same data augmentation algorithm as the original algorithm was used in this experiment.

To validate the effectiveness of the improved algorithm, we select several common metrics in object detection as evaluation indicators for the model, according to reference [40]. These include Precision, Recall, mean Average Precision (mAP), and FPS. Precision(P) refers to the proportion of actual positive samples among all samples predicted as positive. It measures the accuracy of the model in barcode detection, with higher precision indicating more correct detections of barcodes. Recall (R) represents the proportion of correctly predicted positive samples out of all actual positive samples. It measures the comprehensiveness of the model in detecting barcodes, ensuring that as many barcodes as possible are correctly detected. mAP is the mean Average Precision across all categories, used to evaluate the performance of deep learning methods in object detection tasks, rather than just a single aspect of performance. FPS (Frames Per Second) measures the real-time processing capability of the model. For barcode localization algorithms, high FPS indicates that the model can locate each barcode at a higher speed. According to reference [41], its calculation formula as follows:

$$P = \frac{TP}{TP + FP} \tag{12}$$

$$R = \frac{TP}{TP + FN} \tag{13}$$

$$mAP = \frac{\sum_{i=1}^{n} A_i P}{n} \tag{14}$$

Where TP (True Positives) refers to the number of barcodes correctly detected by the object detection model, FN (False Negatives) refers the number of actual barcodes that were not detected by the model, FP (False Positives) represents the number of instances where the model incorrectly predicted the presence of a barcode, AP (Average Precision) is the area between the PR curve and the coordinate axis. These metrics are effective for evaluating the accuracy, comprehensiveness, overall performance, and processing speed of the barcode localization algorithm. They provide valuable insights into the performance of the lightweight model studied in this paper.

V. RESULTS

The YOLO model has five different versions ranging from n, s, m, l, to x, with increasing depth and complexity. To investigate the impact of different sizes on barcode detection, P, R, mAP, and FPS are used as indicators for comparison. Due to the excessive computational and parameter requirements of x, it is not considered. The results are shown in Table 2.

TABLE II
Results of Four Different Networks.

Model	P	R	mAP	Params	GFLOPs	FPS
YOLOv5n	96.3%	92%	96%	1.76M	4.1	101
YOLOv5s	95.8%	93.4%	96.7%	7.03M	16.0	98
YOLOv5m	96.8%	93.8%	97%	20.86M	47.9	49.8
YOLOv5l	96.4%	94%	97.2%	46.11M	107.7	36.4

From Table 2, it can be seen that the increase from YOLOv5n to YOLOv5l did not result in a gradual improvement in detection accuracy. This is because larger model complexity may lead to overfitting or insufficient training, and the limited size of the custom dataset restricts the improvement in model performance. Compared to YOLOv5n and YOLOv5s, YOLOv5m and YOLOv5l achieve the highest precision, recall, and mAP, however, their parameter and computational costs are too high, and their FPS is lower, lacking in lightweight and real-time characteristics. Among YOLOv5n and YOLOv5s, YOLOv5s exhibits the highest recall at 93.4% and also achieves the highest mAP at 96.7%. As this study aims to build lightweight models, YOLOv5m and YOLOv5l with their high parameter and computational costs are not considered, and YOLOv5n did not yield the best results. Therefore, YOLOv5s has been chosen as the base model for this research.

Furthermore, to compare the impact of different loss functions on barcode detection, we contrast the prevailing loss functions: GIOU, EIOU, CIOU, and DIOU. Each loss function has distinct effects, as illustrated in Table 3.

From Table 3, it can be observed that GIOU achieves the highest precision at 95.8%. EIOU achieves the highest recall rate and mAP, reaching 93.1% and 97.1% respectively. However, EIOU has the lowest precision among them. GIOU's mAP is close to that of EIOU, outperforming the original loss function CIOU. Although the difference is small, it is believed to enhance detection precision.

Figure 7 shows the precision-recall curve of the improved YOLO model. It can be seen from the graph that the improved model achieved excellent detection results for both classes of objects, with a mAP value reaching a high value of 97.8%.

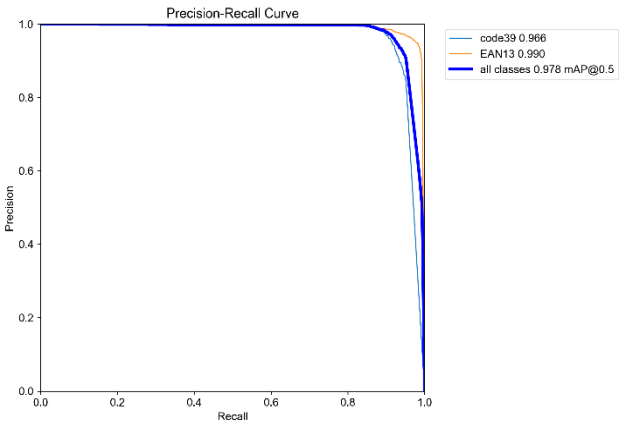


FIGURE 7. **The Precision-Recall curve of the improved YOLO model.**

The baseline network used in this paper is YOLO model, which is improved upon. The baseline model and the improved model are compared in terms of precision, recall, and mAP. The comparison results are shown in Figure 8.

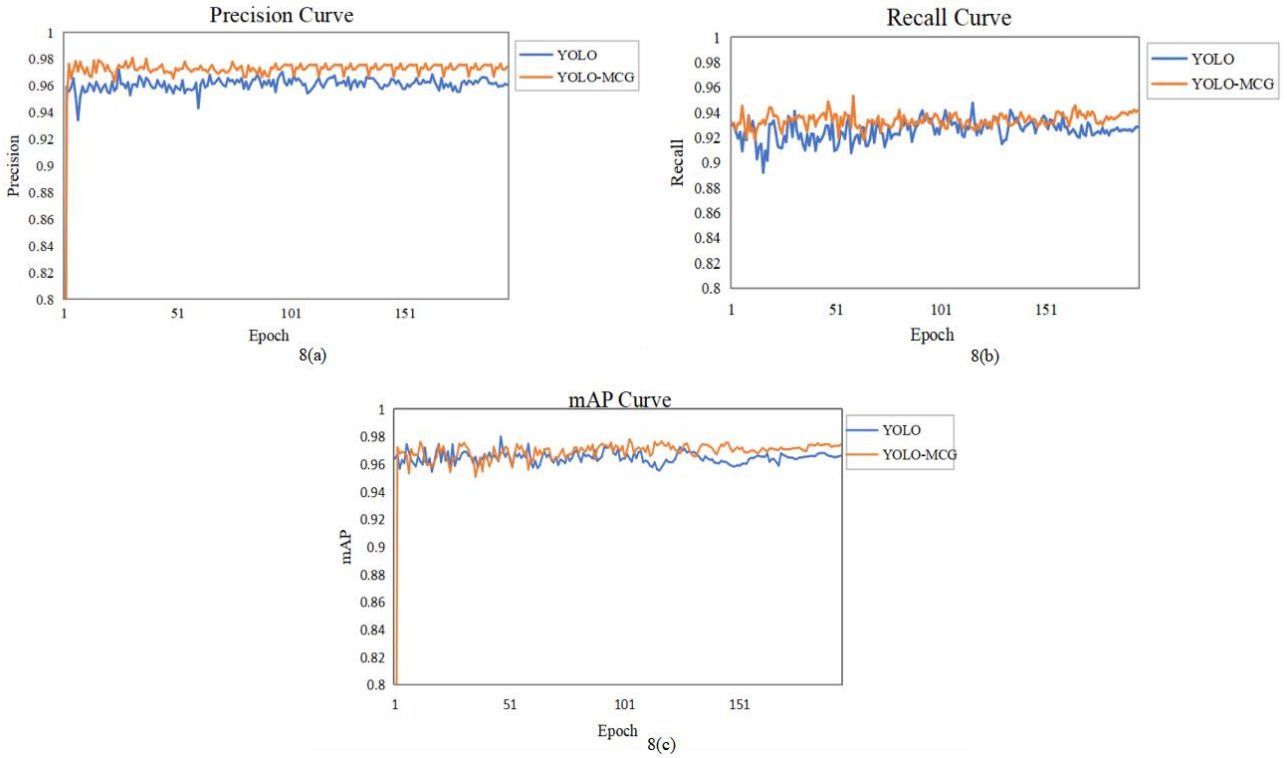


FIGURE 8. **Comparison between the baseline YOLO model and the improved YOLO-MCG. (a), (b), and(c) depict comparisons in terms of precision, recall, and mAP respectively.**

From Figure 8, it can be seen that the improved model outperforms the original model in terms of precision, recall, and mAP, demonstrating the effectiveness of the improvement.

To verify the performance of the improved model, a series of ablation experiments were designed based on the dataset, and comparisons were made with the original YOLO model as the baseline. To ensure lightweight network localization performance, the following improvements were made: 1) Original YOLO baseline model. 2) Replacing the YOLO backbone network with MobileNetv3. 3) Adding CBAM to the Neck of the YOLO model. 4) Replacing CIOU with GIOU and Focal Loss. 5) After replacing the backbone network with MobileNetv3, adding CBAM to the Neck network. 6) After replacing MobileNetv3 and adding CBAM, using GIOU and Focal Loss functions instead of the original loss function. The detection results are shown in Table 4.

TABLE IV
Ablation experiment

No.	YOLO	MobileNet	CBAM	GIOU+Focal Loss	P	R	mAP	Params	GFLOPs	FPS
1	√				95.8%	93.4%	96.7%	7.03M	16.0	98
2	√	√			95.4%	92.6%	96.5%	3.9M	7.0	95.2
3	√		√		96.0%	94.0%	97.8%	7.03M	15.8	120
4	√			√	95.9%	94.5%	97.5%	7.03M	15.8	88
5	√	√	√		96.2%	93.1%	97.5%	3.9M	7.0	90
6	√	√	√	√	96.4%	93.9%	97.8%	3.9M	7.0	105

As shown in Table 4, firstly, after using CBAM, there is a certain improvement in precision, recall, and mAP. The precision increased by 0.2% compared to the baseline network, recall increased by 0.6%, and mAP increased by 1.1%. Secondly, after introducing GIOU and Focal Loss, the precision increased by 0.1%, recall improved by 1.1%, and mAP increased by 0.8% compared to the baseline network. The improved network structure incorporates the lightweight MobileNetv3 network, which is designed to reduce model size and computational complexity. Lightweight networks often sacrifice some performance for these benefits. From the ablation experiments, it can be observed that the performance metrics indeed decrease after using the MobileNet network. However, through all the improvement methods applied, the accuracy and recall of the lightweight network are maintained with minimal loss, while mAP is improved. Moreover, achieving an FPS of 105 meets real-time requirements. The experiments indicate that the improvements made in this paper can enhance the barcode detection results.

To validate the effectiveness of the optimization algorithms used in this experiment, SGD, Adam, RMSprop, and AdamW were compared in terms of loss function, with the results shown in Figure 9.

Due to RMSprop's slow convergence and high loss function values, it was not plotted in Figure 9. From Figure 9, it can be observed that AdamW converges the fastest and achieves the best performance. To further validate the impact of the improved optimization algorithm AdamW on other evaluation metrics, it was compared with other optimization algorithms in terms of accuracy, recall, and mAP, as shown in Table 5.

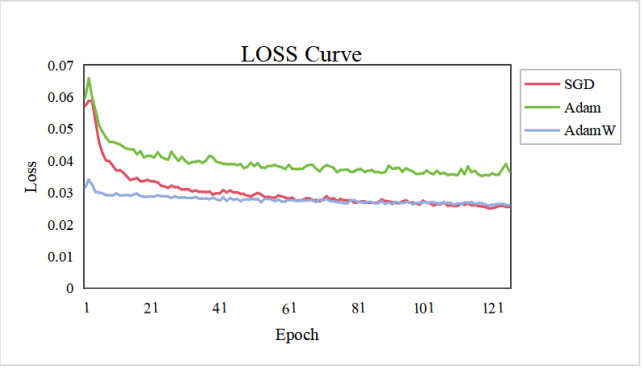


FIGURE 9. **Comparison of optimization algorithms in loss functions.**

TABLE V
Comparison of optimization algorithms

Model	P	R	mAP
RMSprop	50.3%	26.1%	35.3%
SGD	96.2%	92.8%	97.6%
Adam	96.8%	92%	96.6%
AdamW	96.4%	93.9%	97.8%

From Table 5, it is evident that AdamW performs the best. Compared to the RMSprop optimization algorithm, precision improved by 46.1%, recall improved by 67.8%, and mAP increased by 62.5%. Compared to SGD (the optimization algorithm used in the baseline model), precision increased by 0.2%, recall increased by 1.1%, and mAP improved by 0.2%. Compared to Adam, precision did not improve, but recall increased by 1.9% and mAP improved by 1.2%. This is because AdamW demonstrates significant effectiveness in training on large-scale datasets. Additionally, AdamW handles weight decay separately,

effectively preventing overfitting. By correctly controlling the model's complexity, AdamW ensures better generalization to unseen data, thereby enhancing P, R, and mAP metrics.

To further validate the superiority of the proposed algorithm in performance, the improved algorithm was compared with YOLOv5n, YOLOv5m, YOLOv5l, YOLOv8s, and SSD (where SSD uses VGG16 as the backbone network) on the same dataset. Figures 10(a), 10(b), and 10(c) compare precision, recall, and mAP respectively. The comparison results are shown in Figure 10.

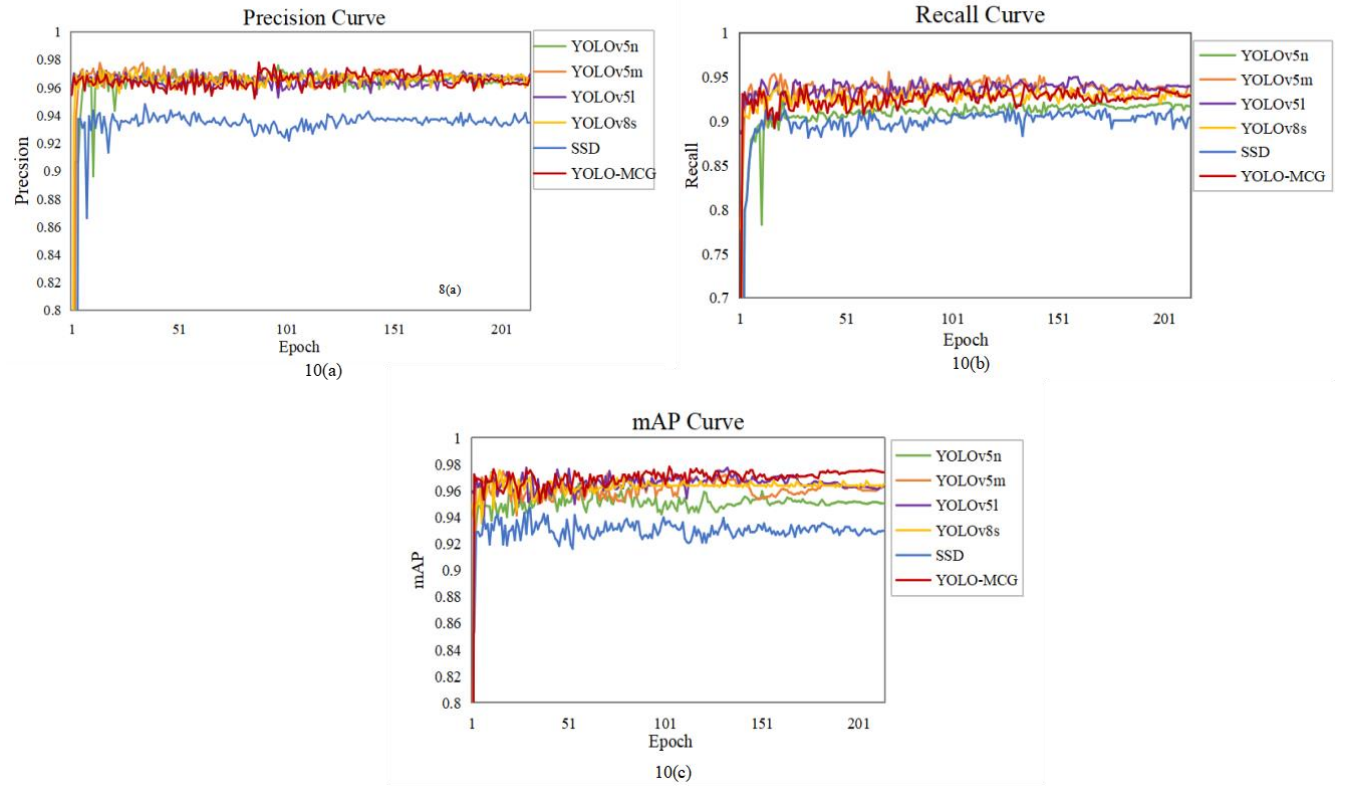


FIGURE 10. **Performance comparison of different models. (a), (b), and (c) depict comparisons in terms of precision, recall, and mAP respectively. The horizontal axis represents epochs, while the vertical axis represents the corresponding values of P, R, and mAP.**

From the figure 10, it can be seen that the convergence speed of different models is fast. The precision and recall of the improved model are similar to YOLOv8, but its mAP value is the highest, which further indicates the excellent improvement effect.

To further demonstrate the superiority of the improvement methods, an evaluation of the model's efficiency and complexity was conducted. The improved model was compared with the YOLO baseline model, SSD, Faster R-CNN(resnet50) and YOLOv8s in comparative experiments. Precision, recall, mAP value, parameter count, and GFLOPs were used as evaluation metrics. The experimental results are presented in Table 6.

TABLE VI
Comparative experiments

Model	mAP	Params	GFLOPs	FPS
SSD	92.8%	23.9M	55	54.3
Faster R-CNN(resnet50)	97.3%	137M	370.2	20
YOLOv5n	96%	1.76M	4.1	101
YOLOv5s	96.7%	7.03M	16.0	98
YOLOv5m	97%	20.8M	47.9	49.8
YOLOv5l	97.2%	46.1M	107.7	36.4
YOLOv8s	97.1%	11.3M	28.4	130
YOLO-MCG	97.6%	3.93M	7.0	105

From Table 6, it can be seen that the improved model has the highest mAP compared to mainstream network models,

with significantly reduced parameters and computational complexity. The improved YOLO-MCG achieves a 4.8% higher mAP than SSD, reduces parameter count to 56% of the original model, and decreases to 35% of YOLOv8. Computational complexity is reduced to 44% of the original model and 25% of YOLOv8. Although Faster R-CNN has a high mAP, its parameter count and computational cost are too large, resulting in a low FPS, which makes it unsuitable for real-time applications. Figure 11 illustrates the superiority of the improved YOLO-MCG over different models in terms of mAP, Params, GFLOPs, and FPS. In scenarios requiring real-time performance, this method is more suitable for fast and precision barcode detection.

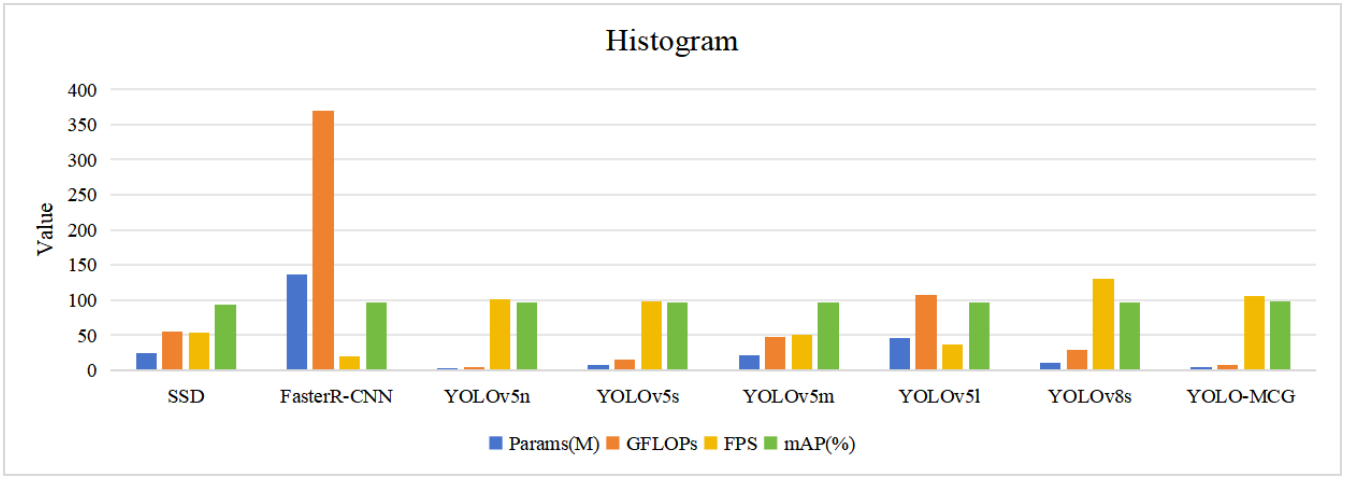
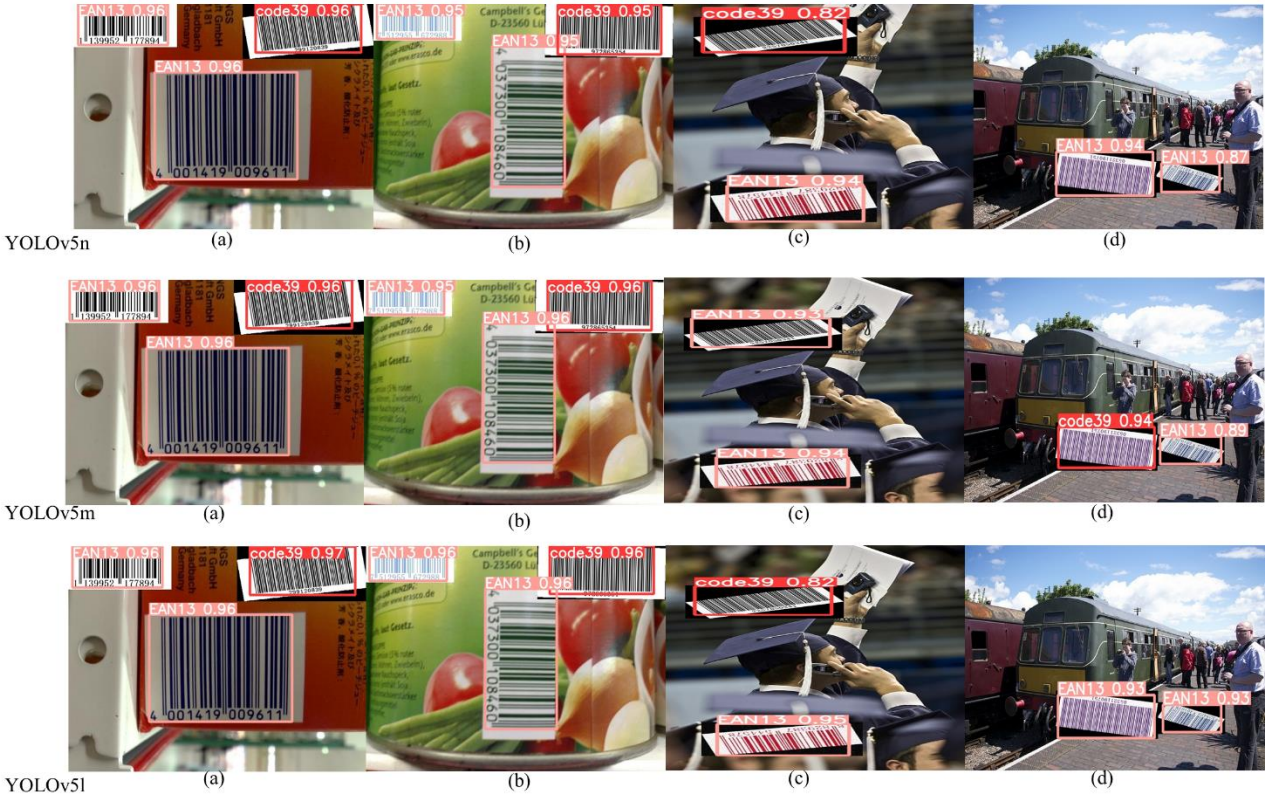


FIGURE 11. Comparison of different models in terms of Params, GFLOPs, FPS and mAP.

This study proposes the YOLO-MCG algorithm based on the YOLO model for barcode localization. To investigate the detection performance of the improved model on barcodes, several models were visually compared, and the detection results are shown in Figure 12.



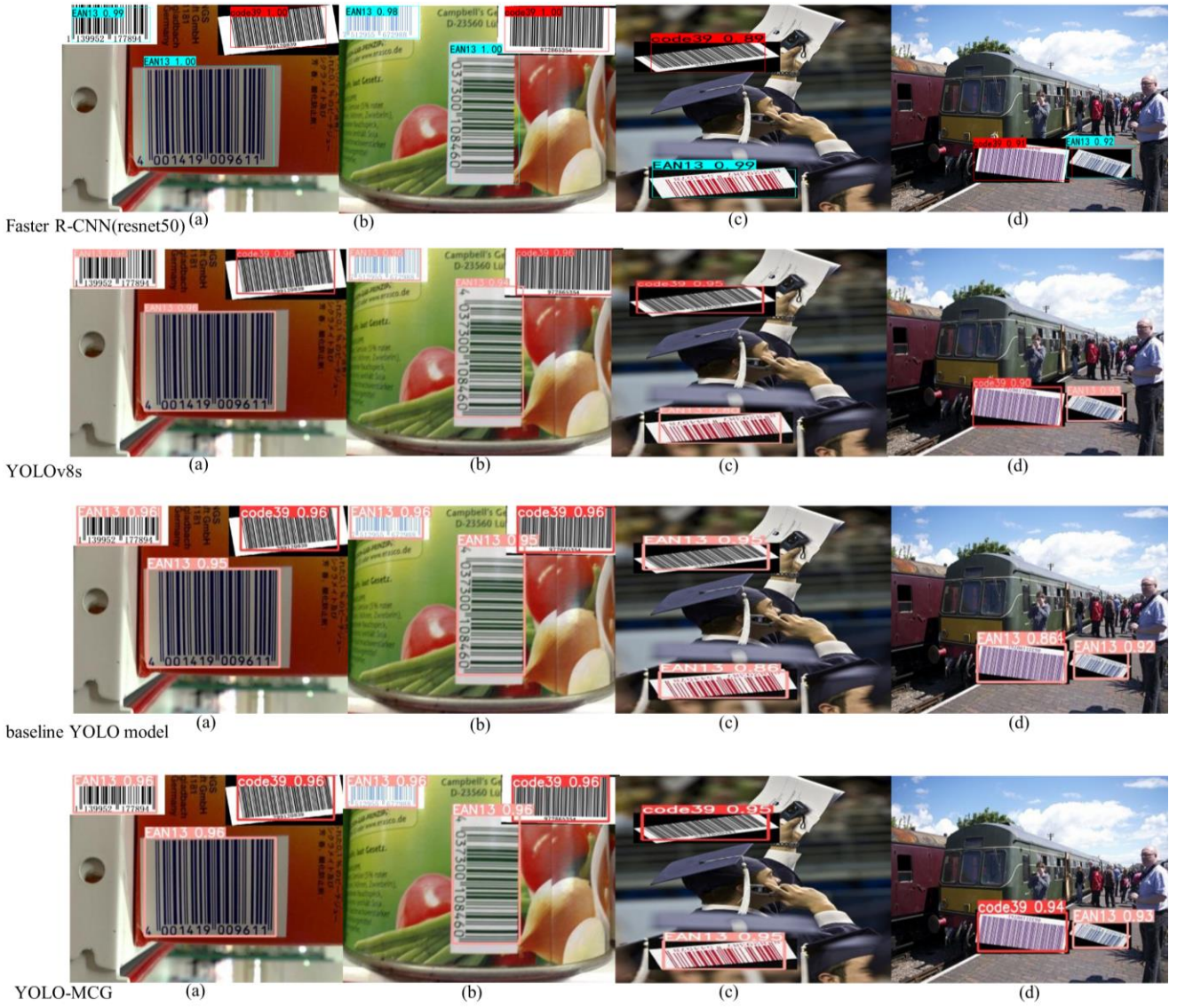


FIGURE 12. **Model detection results visualization. From top to bottom: YOLOv5n, YOLOv5m, YOLOv5l, YOLOv8s, baseline YOLO model, and improved YOLO-MCG. From left to right: (a), (b), (c), and (d) represent four different scenarios.**

In Figure 12, two types of barcodes are present: EAN13 and Code39. Figures 12(a) and 12(b) depict scenarios with clear and large barcodes located close to the camera, while Figures 12(c) and 12(d) show cases with smaller and blurred barcodes located farther away. From the figures, it is observed that each model successfully classifies and detects the clear and large barcodes with high accuracy. However, for the smaller and blurred barcodes (Figure 12(c) and 12(d)), YOLOv5n, YOLOv5l, and the baseline YOLO model (YOLOv5s) fail to correctly identify the Code39 barcodes, misclassifying them as EAN13 barcodes. YOLOv5n and YOLOv5l exhibit lower detection rates in these cases. YOLOv5m, YOLOv8s, Faster R-CNN, and the improved YOLO-MCG successfully identify these barcodes, with Faster R-CNN and YOLO-MCG achieving the highest detection rates. However, as seen from previous experiments, Faster R-CNN has a large number of parameters and high computational complexity, resulting in a very low FPS, which cannot meet real-time requirements. In contrast, the lightweight characteristics of YOLO-MCG further validate the feasibility of the improved YOLO-MCG for use in mobile devices and fast-paced production lines.

VI. Discussion

In this study, we propose a lightweight barcode localization algorithm based on an improved YOLO model. To address the issue of fast and efficient barcode localization, a series of innovative improvements were implemented, resulting in significant performance gains. First, we created

a custom barcode dataset that meets the requirements of deep learning models in complex backgrounds. This dataset consists of one-dimensional barcode images of two types: EAN13 and Code39. These two types of barcodes represent the most commonly used barcodes in daily life. Although barcode localization and recognition have attracted widespread attention from researchers, there are few publicly available barcode datasets, and even fewer that meet the requirements for deep learning, which requires large volumes of images. Furthermore, these datasets need to cater to the challenge of localizing barcodes in complex backgrounds. Therefore, in this situation, we opted to create a custom dataset to meet the research needs. Next, we replaced YOLO's backbone network with MobileNetv3, significantly reducing the mode's parameters and computational load, thus enhancing its practicality for deployment on embedded or mobile devices. We also introduced the CBAM (Convolutional Block Attention Module) attention mechanism, which helps the model focus more on the barcode regions in the feature map, suppressing the interference from irrelevant background information. This effectively improved the model's robustness and localization accuracy, significantly enhancing the distinction between targets. We adopted the GIOU and Focal Loss functions, which, when handling barcode detection localization tasks, not only consider positional accuracy but also increase attention to difficult-to-detect targets (such as small or partially occluded barcodes). Focal Loss addresses the class imbalance problem effectively, especially in complex backgrounds, enabling the model to better handle the contrast between barcodes and backgrounds, further improving localization accuracy and stability.

A series of experiments (such as comparison experiments and ablation studies) were conducted to validate the performance of the improved model. The experimental results showed that although the model size was significantly reduced, it still maintained high localization accuracy. This is crucial for the real-time requirements in barcode detection scenarios. The improved YOLO-MCG model demonstrated better detection performance and faster inference speed in complex environments, making it suitable for deployment on edge devices. This further confirms the feasibility and superiority of the proposed algorithm and verifies the effectiveness and practical application potential of the method presented in this paper.

Future Work: Although this study has yielded good results, there are still some limitations and room for improvement. First, this paper focuses only on the localization of one-dimensional barcodes and does not address other types of barcodes, such as QR codes, which have significant differences in shape and texture from one-dimensional barcodes. Future work will focus on extending the existing methods to develop an algorithm model that can efficiently and simultaneously detect multiple types of barcodes in real time.

Moreover, although GIOU and Focal Loss performed well in this study, they may still be affected by different datasets and environmental factors in practical applications. Future work could involve conducting experiments with a wider variety of datasets, further optimizing the loss functions, or combining them with other advanced optimization methods (such as adaptive loss functions) to enhance model performance. In summary, the improved YOLO-MCG model proposed in this paper provides a new solution for barcode localization, and through further optimization and expansion of the algorithm framework and datasets, it can drive the practical application of barcode localization technology in a broader range of use cases.

VII. Conclusion

This paper presents a barcode localization algorithm based on the YOLO model to address the problem of fast and efficient barcode localization. By replacing YOLO's backbone network with MobileNetv3, the model size is significantly reduced. The introduction of the CBAM attention mechanism enhances the model's focus on the target, improving its performance. At the same time, GIOU and Focal Loss were adopted as loss functions suitable for our dataset, effectively improving the model's localization accuracy. The effectiveness and feasibility of this method were validated through experiments. The experimental results show that the improved YOLO-MCG model achieves a maximum mAP of 97.6%, which is 0.9% higher than the baseline model, with the number of parameters reduced to 56% and the computational load reduced to 44% of the original model. This study focuses only on one-dimensional barcodes and does not include various types of QR codes. Future work will explore an algorithm model capable of efficiently and simultaneously detecting multiple types of barcodes in real time.

REFERENCES

[1] K. Liu, Y. R. Bi and D. Liu, "Internet of Things based acquisition system of industrial intelligent bar code for smart city applications," *Comput. Commun.*, vol. 150, pp. 325-333, 2020, doi: 10.1016/j.comcom.2019.11.044.

[2] S. H. Bach, P. B. Khoi, and S. Y. Yi, "Application of QR code for localization and navigation of indoor mobile robot," *IEEE Access*, vol. 11, pp. 28384-28390, 2023, doi:10.1109/ACCESS.2023.3250253.

[3] H. Zheng, Z. Y. Guo, C. Liu, X Li, T. Y. Wang, C. H. You, "Blind deblurring of QR code using intensity and gradient prior of positioning patterns," *Vis. Comput.*, vol. 40, pp. 441-455, 2024.

[4] H. P. Liu, "Improve the quality of barcode service inspection work," *Bar Code Info*. Sys., vol. 4, pp.36-37, May. 2023.

[5] Z. Y. Guo, S. Y. Wang, Z. H. Zheng, K. Sun, "Printer source identification of quick response codes using residual attention network and smartphones," *Eng. Appl. Artif. Intell.*, vol. 131, pp. 107822, 2024.

[6] H. Yang, L. Z. Chen, Y. F. Chen, Y. Lee, and Z. P. Yin, "Automatic barcode recognition method based on adaptive edge detection and a mapping model," *J. Electron. Imaging*, vol.25, no. 5, Sep. 2016, doi: 10.1117/1.JEI.25.5.053019.

[7] E. I. Ershov, A. P. Terekhin, and D. P. Nikolaev, "Generalization of the fast hough transform for three-dimensional images," *J. Commun. Technol. Electron.*, vol. 63, pp. 626-636, 2018.

[8] Y. H. Xie, J. Shen, and C. D. Wu, "Robust Object Tracking Using Affine Transformation and Convolutional Features," *IEEE Access*, vol.7, pp. 182489-182498, Dec. 2019.

[9] C. Creusot, A. Munawar, "Real-time barcode detection in the wild," in *Proc. IEEE Winter. Conf. Appl. Comput. Vis. (WACV)*, Waikoloa, HI, USA, Jan 2015, pp.239-245, doi:10.1109/WACV.2015.39.

[10] D. Kalpita, S. Ritesh, K. Mahantapas, N. Mita, and D. Nibaran, "Natural scene text localization and detection using MSER and its variants: a comprehensive survey," *Multimed. Tools Appl.* Vol. 83, pp. 55773-55810, 2024.

[11] C. Creusot, A. Munawar, "Low-computation egocentric barcode detector for the blind," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Phoenix, Arizona, USA, Sep 2016, pp.25-28, doi:10.1109/ICIP.2016.7532881.

[12] L. Teplyakov, L. Erlygin, and E. Shvets. "Lsdnet: Trainable modification of lsd algorithm for real-time line segment detection," *IEEE Access*, vol. 10, pp. 45256-45265, 2022, doi:10.1109/ACCESS.2022.3169177.

[13] M. Katona, P. Bodnár, and L. G. Nyúl, "Distance transform and template matching based methods for localization of barcodes and QR codes," *Comput. Sci. Inf. Syst.*, vol. 17, no. 1, pp. 161-179, 2020, doi:10.2298/CSIS181011020K.

[14] J. W. Yi, Y. B. Xiao, "Efficient localization of multitype barcodes in high-resolution images," *Math. Prob. Eng.*, vol. 2022, pp. 1-10, Mar. 2022. doi:10.1155/2022/5256124.

[15] K. Alex, S. Ilya, and E. H. Geoffrey, "Imagenet classification with deep convolutional networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, Granada, Spain, Dec 2011, pp. 1097–1105.

[16] S. Q. Ren, K. M. He, R Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137-1149, Jun. 2016, doi: 10.1109/TPAMI.2016.2577031.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, Nevada, USA, May 2016, pp. 779-788, doi:10.1109/CVPR.2016.91.

[18] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "Ssd: single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Amsterdam, Netherlands, Dec. 2016, pp. 21-37, doi:10.1007/978-3-319-46448-0_2.

[19] Z. Y. Guo, H. Zheng, C. H. You, X. H. Xu, X. B. Wu, Z. H. Zheng, J. P. Ju, "Digital forensics of scanned QR code images for printer source identification using bottleneck residual block," *Sensors*, val. 20, no. 21, 2020.

[20] H. Zhang, G. L. Shi, L. Liu, M. Zhao, and Z. Liang, "Detection and identification method of medical label barcode based on deep learning," in *Proc. 2018 8th Int. Conf. Image Process. Theory Tools Appl. (IPTA)*, Madrid, Italy, Nov. 2018, pp. 1-6, doi:10.1109/IPTA.2018.8608144.

[21] J. J. Li, Q. Zhao, X. Tan, Z. X. Luo, and Z. Tang, "Using deep convnet for robust 1D barcode detection," *Intell. Syst. Interact. Appl.: Proc. 2nd Int. Conf. Intell. Interact. Syst. Appl. (IISA2017)*, Suzhou, China, Nov. 2018, pp. 261-267, doi:10.1007/978-3-319-69096-4_36.

[22] L. C. Qiao, J. L. Wang, B. H. Gao, X. G. Yang, W. T. Feng, Y. X. Zhang, Y. Wang, H. Liu, and W. Liu, "Efficient 1D barcode localization method for imagery shipping label using deep learning models," in *Proc. 2021 12th Int. Symp. Parallel Archit. Algorithms Program. (PAAP)*, Xi'an, China, Dec. 2021, pp. 119-124, doi:10.1109/PAAP54281.2021.9720443.

[23] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, Nevada, USA, Jun. 2016, pp. 770-778, doi:10.1109/CVPR.2016.90.

[24] W. T. Wan, C. F. Li, H. B. Zhu, Y. R. Tao, "Two-dimensional barcode positioning algorithm of lightweight CenterNet network," *J. Electron. Meas. Instrum.*, vol. 36, no. 5, pp. 128-135, 2023.

[25] Y. Z. Xiao, J. X. Jiang, and K. Xu, "An integrated deep-learning and geometric approach to 1D barcode," in *Proc. 4th Int. Workshop Pattern Recognit.*, Nanjing, China, Jul. 2019, pp. 4-9, doi:10.1117/12.2540364.

[26] X. B. Yue, L. Meng, "YOLO-SM: A Lightweight Single-Class Multi-Deformation Object Detection Network," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 8, no. 3, pp. 2467 – 2480, June 2024, doi:10.1109/TETCI.2024.3367821.

[27] H. T. Do, V. C. Pham, "Deep Learning Based Goods Management in Supermarkets," *J. Adv. Inf. Technol.*, vol.12, no. 2, pp. 164-168, May 2021.

[28] J. Redmon, A. Farhadi, "Yolov3: An incremental improvement. Computer Vision and Pattern Recognition," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, Utah, USA, Apr. 2018, pp. 1-6, doi:10.48550/arXiv.1804.02767.

[29] R. Brylka, U. Schwanecke, and B. Bierwirth, "Camera based barcode localization and decoding in real-world applications,"in *Proc. 2020 Int. Conf. Omni-Layer Intell. Syst.*, Barcelona, Spain, Aug. 2020, pp.1-8, doi:10.1109/COINS49042.2020.9191416.

[30] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "Yolov4: Optimal speed and precision of object detection," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, USA, Jun. 2020.

[31] A. Howard, M. Sandler, B. Chen, W. J. Wang, L. C Chen, M. X. Tan, G. Chu, V. Vasudevan, Y. K. Zhu, R. M. Pang, H. Adam, and Q. Le, "Searching for mobilenetv3," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, South Korea, Oct. 2019, pp.1314-1324, doi:10.1109/ICCV.2019.00140.

[32] F. Chollet, "Xception: Deep learning with depthwise separable convolutions." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, July 2017, pp. 1251-1258.

[33] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, USA, June 2018, pp. 7132-7141.

[34] K. KC, Z. D. Yin, M. Y. Wu, and Z. L. Wu, "Depthwise separable convolution architectures for plant disease classification," *Comput. Electron. Agr.*, vol. 165, pp. 104948-104954, Oct. 2019.

[35] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, "Cbam: convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 3-19, doi:10.1007/978-3-030-01234-2_1.

[36] H. Rezatofighi, N. Tsoi; J. Y. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: a metric and a loss for bounding box regression," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Berkeley, California, USA, Jun. 2019, pp. 658-666, doi:10.1109/CVPR.2019.00075.

[37] T. Y. Lin, P. Goyal, R. Girshick, K. M. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Aug. 2017, pp. 2980-2988.

[38] L. W. Tao, M. J. Dong, and C. Xu, "Dual focal loss for calibration," *in Proc. Int. Conf. Mach. Learn. (ICML)*, Hawaii, USA, May 2023, pp. 33833-33849.

[39] I. Loshchilov, F. Hutter, "Decoupled Weight Decay Regularization," in Proc. *Int. Conf. Learn. Represent. (ICLR)*, New Orleans, USA, May 2019.

[40] R. Wudhikarn, P. Charoenkwan, and K. Malang, "Deep learning in barcode recognition: A systematic literature review," *IEEE Access*, vol. 10, pp. 8049-8072, Jan. 2022.

[41] K. Ravpreet, S. Sarbjeet, "A comprehensive review of object detection with deep learning," *Digit. Signal. Process.*, vol. 132, pp. 103812-103829, Jan. 2023.

CHAOCHAO LI received the B.S. degree from Xinzhou Normal University, China, in 2022. She is currently pursuing the master's degree in information and communication engineering at the Beijing Institute of Graphic Communication, Beijing, China. She current research interests include internet of things, image processing.

QINGTAO ZENG received the Ph.D. degree from Beijing University of Posts and Telecommunications, China, in 2015. He is currently a faculty member with the School of Information Engineering, Beijing Institute of Graphic Communication, and serves as the Deputy Director of the Publication Data Asset Evaluation Laboratory. His research interests include variable information digital printing control systems and high-speed printing image processing.

LIKUN LU received the Master's degree from Beijing University of Technology, China, in 2014. He is currently an Associate Professor with the School of Information Engineering, Beijing Institute of Graphic Communication. His research interests include inkjet digital printing technology, information processing technology, and embedded systems.