 https://doi.org/10.24191/jeesr.v26i1.005

Enhanced Slider CAPTCHA Recognition for Vulnerability Assessment

Wan Xing, Juliana Johari, and Fazlina Ahmat Ruslan*

***Abstract*—This paper addresses the limitations of existing slider CAPTCHA recognition methods by proposing an enhanced approach based on the YOLOv5 model and introduces a novel evaluation metric, mean Relative Offset (mRO), which more accurately assesses the x-coordinate of gap location prediction compared to traditional mean Average Precision (mAP). Furthermore, a novel Offset-based Intersection over Union (OIoU) loss function is proposed, specifically designed to prioritize the accuracy of horizontal displacement, crucial for slider CAPTCHA detection. The study also presents Fixed Quantity Prediction-based Non-Maximum Suppression (FQP-NMS), a modified NMS algorithm ensuring a fixed number of predicted bounding boxes, addressing the variability inherent in standard NMS. Experiments demonstrate that the proposed OIoU and FQP-NMS, when integrated with YOLOv5, significantly improve both mAP and mRO, particularly on challenging SliderCAPTCHA datasets. The incorporation of lightweight attention mechanisms, such as ECA, further enhances the model's performance and robustness, especially when combined with VGG19 and other efficient architectures. The results indicate that the proposed system provides a more accurate and robust solution for slider CAPTCHA recognition.**

***Index Terms*—IoU, NMS, Object Detection, Recognition, Slider CAPTCHA.**

I. INTRODUCTION

In the realm of online security, slider CAPTCHA as a form of CAPTCHA has emerged as a user-friendly alternative to traditional methods. By requiring users to slide a puzzle piece to a designated gap, this approach not only enhances the user experience but also serves as a robust mechanism for confirming human identity. Despite its importance, research on slider verification remains limited. Recent advancements in computer vision, particularly with YOLO (You Only Look Once) frameworks, such as YOLOv5, YOLOv6, and YOLOv7, have spurred interest in using object detection techniques for slider verification [1].

GitHub projects illustrate a trend of employing YOLO to detect and analyze the slider CAPTCHAs. Its capability for real-time object detection makes YOLO an attractive option for improving the effect of slider puzzles. Improving and intensifying research on slider CAPTCHAs is one of the essential means for assessing network security and identifying vulnerabilities.

There is a slider puzzle and a corresponding gap with the same shape in the slider CAPTCHA, and users may move the slider to the gap position accurately [2]. The server judges whether the user is a robot according to the moving track and the final position of the slider. The process of slider CAPTCHAs is shown in Fig. 1.

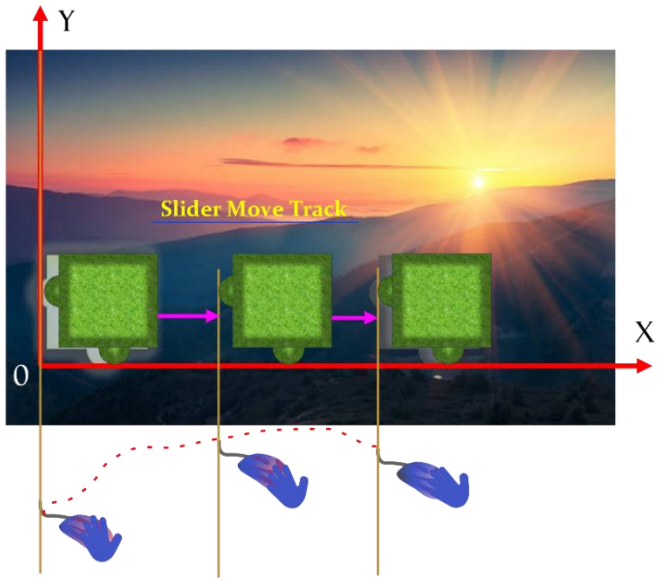

Fig. 1. Login Process based on the Slider CAPTCHA

Slider CAPTCHA usually adopts three types of security mechanisms. The first is to use a very complex background whose color is close to the slider puzzle, making it difficult for attackers to identify the slider. The second is the design of the slider itself, which will put some shadows and fake puzzles to confuse the recognizer and make the attacker misjudge. The third is trajectory analysis based on machine learning methods. The method analyzes many human and machine sliding trajectory data and can determine whether it is human behavior, not just to consider whether the sliding position makes it more difficult for the attacker.

Some examples of slider CAPTCHAs are shown in Fig. 2. Usually, such slider CAPTCHAs have complex backgrounds to interfere with the user's judgment on the gap position. The key to cracking the slider CAPTCHA is to detect the position of the gap, and due to the varying criteria for trajectory recognition across different systems, this study focuses exclusively on

This manuscript is submitted on 14 Feb 2025, revised on 18 February 2025 and accepted on 28 March 2025. Xing Wan, Juliana Johari, and Fazlina Ahmat Ruslan* are from School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia.

*Corresponding author
Email address: fazlina419@uitm.edu.my

identifying and localizing the gap within CAPTCHA images.

Fig. 2. Some Examples of Slider CAPTCHAs

Nowadays, there are mainly two kinds of research frameworks for CAPTCHA recognition: segmentation plus recognition and end-to-end recognition without segmentation. The former divides a picture into gap and background in the pre-processing stage and applies them directly to CAPTCHA recognition by designing the network structure. The latter does not perform segmentation in the preprocessing stage and directly recognizes text or sliders through end-to-end learning. The key to slider CAPTCHA detection is accurately locating where the gap is located. The category of the gap is relatively unimportant, as the sliding CAPTCHA only focuses on whether the mouse was moved accurately. Often sliding CAPTCHA is cracked using an object detection network, so such a network can give an optimal estimate of the target location. This article conducts an in-depth investigation into the task of slider CAPTCHA recognition and proposes an improved scheme based on the YOLOv5 model, offering new perspectives for related research on slider CAPTCHAs.

II. RELATED WORKS

YOLOv5 is an object detection model developed by Ultralytics, which builds on the success of its predecessors in the YOLO (You Only Look Once) series. It is designed to achieve real-time detection speed while maintaining high accuracy, which employs a simple and efficient architecture that divides the detection pipeline into three two components: a backbone and a head. The backbone, typically based on Cross Stage Partial Networks (CSPNet), extracts rich feature representations from input images. The head using features such as Path Aggregation Network (PANet), facilitates multi-scale feature fusion, enhancing the model's ability to detect objects at various sizes, which finally output the head outputs the predicted bounding boxes, class probabilities, and confidence scores for detected objects.

A. Lightweight Attention Mechanisms

In this study, we aim to enhance the performance of slider CAPTCHA recognition by incorporating several lightweight attention mechanisms. Efficient Channel Attention (ECA) mechanism is a refined method in deep learning that enhances convolutional neural networks by improving channel-wise feature recalibration without excessive computational cost [3]. Unlike traditional methods like Squeeze-and-Excitation networks, ECA utilizes a 1D convolution with an optimal kernel size to capture local cross-channel interactions efficiently, as shown in Fig. 3. This approach maintains simplicity and speed, making it ideal for tasks with limited computational resources.

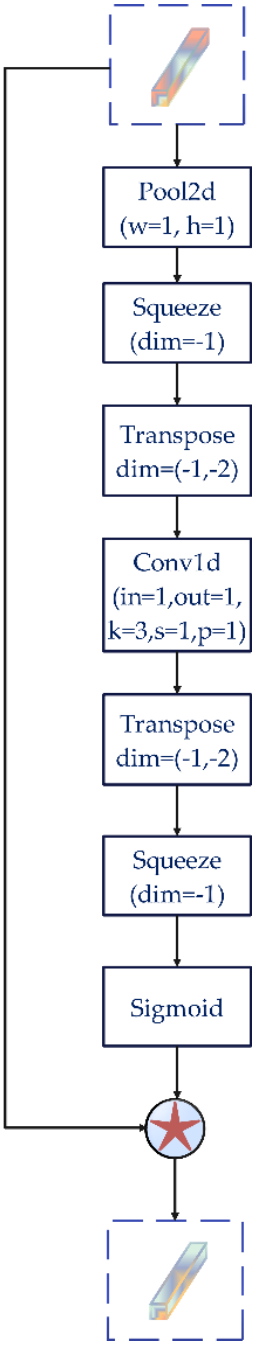

Fig. 3. The Structures of ECA

Convolutional Block Attention Module (CBAM) is an advanced attention mechanism designed to improve networks, which applies both channel and spatial attention modules, enhancing feature refinement by focusing on significant regions and channels of an input feature map [4]. The channel attention module emphasizes meaningful features by leveraging inter-channel relationships, while the spatial attention module highlights important spatial locations. This dual attention

strategy ensures a more comprehensive representation, which has been shown to improve performance across various computer vision tasks such as image classification and object detection, as shown in Fig. 4.

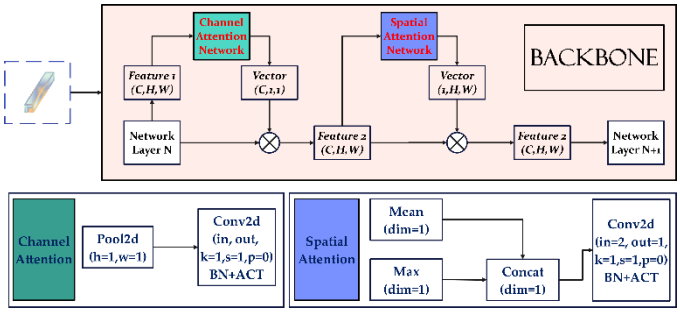

Fig. 4. The Structures of CBAM

Squeeze-and-Excitation (SE) networks introduce a mechanism that adaptively recalibrates channel-wise feature responses [5]. The SE block functions by first squeezing the global spatial information into a channel descriptor, then excites each channel according to its importance. This adaptive process enhances the representational power of a network significantly, allowing it to focus on informative features more effectively. Both SE and ECA are based on channel attention mechanisms; however, they differ in their approaches to generating weight. SE utilizes a fully connected network to generate weights, while ECA employs 1D convolution for weight generation. The Lightweight Separable Kernel Attention (LSKA), builds upon Visual Attention Networks (VANs) by leveraging separable convolution kernels [6]. This allows LSKA to enhance attention efficiency by focusing computational resources on pivotal tasks, thereby maintaining high performance even with reduced computational overhead.

B. Lightweight Backbones

MobileNet uses Deeply separable convolutions architecture to create compact deep neural networks [7]. To efficiently balance the delay and precision, it proposes two straightforward global hyperparameters. ShuffleNet introduces group convolution, which greatly reduces the calculation of the model [8]. It shuffles all channels of feature maps belonging to different groups evenly. This technology makes the recombined group feature map contain one channel in each previous group of feature maps, solving the problem of channel communication blocking caused by group convolution. ShuffleNetv2 is a highly efficient convolutional neural network designed for mobile and edge computing [9]. It is characterized by its use of a channel shuffle operation that enhances feature combination while maintaining low computational cost. The architecture incorporates lightweight building blocks, allowing for a balanced trade-off between accuracy and efficiency. By optimizing the design for various dimensions of performance, ShuffleNetv2 achieves superior accuracy with significantly fewer parameters compared to its predecessors.

C. IoU Loss Functions

The IoU was initially proposed for use in face detection within the UnitBox model, whereby a loss function is constructed based on the intersection over union of the predicted and actual frames [10] , as shown in (1).

$$IoU = \frac{|P \cap G|}{|P \cup G|} \tag{1}$$

If a positive sample satisfies $P(IoU = 1) = 1$, the corresponding IoU loss function is shown in (2).

$$l_{IoU} = 1 - log(IoU) \tag{2}$$

Assume that the predicted bounding box has the top-left corner at the point (x_1^p, y_1^p) and the bottom-right corner at the point (x_2^p, y_2^p). The Ground Truth (GT) bounding box has the top-left corner at the point (x_1^g, y_1^g) and the bottom-right corner at the point (x_2^g, y_2^g). The IoU algorithm is shown in Fig. 5.

Algorithm Calculate IoU

Input: Predicted bounding box $P = (x_1^p, y_1^p, x_2^p, y_2^p)$, Ground truth bounding box $G = (x_1^g, y_1^g, x_2^g, y_2^g)$
Output: Intersection over Union (IoU)
$x_{\text{inter\_min}} \leftarrow \max(x_1^p, x_1^g)$
$y_{\text{inter\_min}} \leftarrow \max(y_1^p, y_1^g)$
$x_{\text{inter\_max}} \leftarrow \min(x_2^p, x_2^g)$
$y_{\text{inter\_max}} \leftarrow \min(y_2^p, y_2^g)$
inter_width $\leftarrow \max(0, x_{\text{inter\_max}} - x_{\text{inter\_min}})$
inter_height $\leftarrow \max(0, y_{\text{inter\_max}} - y_{\text{inter\_min}})$
inter_area $\leftarrow$ inter_width $\times$ inter_height
predicted_area $\leftarrow (x_2^p - x_1^p) \times (y_2^p - y_1^p)$
gt_area $\leftarrow (x_2^g - x_1^g) \times (y_2^g - y_1^g)$
union_area $\leftarrow$ predicted_area + gt_area $-$ inter_area
if union_area <= 0 **then**
  return 0
else
  return IoU $\leftarrow \frac{\text{inter\_area}}{\text{union\_area}}$
end if

Fig. 5. IoU Computing Process

However, when the predicted bounding box and ground truth do not intersect, IoU cannot reflect the distance between them. In addition, it cannot calculate angle, direction and aspect ratio. In this case, the loss function cannot be differentiable and optimized. Other improved algorithms are shown in Table I.

TABLE I. OTHER IOU ALGORITHMS

IoU	Method Description
GIoU [11]	Introduce a penalty when there is no overlap.
DIoU [12]	Add optimization of the distance.
CIoU [12]	Add parameter of aspect ratio.
EIoU [13]	Calculate length and width separately.
SIoU [14]	Add the angle between two boxes.

However, the problem with the IoU loss is that its value is always one when there is no overlapping region between the two boxes, which is not favorable for gradient propagation. To solve this problem, GIoU adds a penalty term to the loss

function in the form of a closure consisting of a prediction frame and a true frame, as shown in **Error! Reference source not found.**.

$$GIoU = IoU - \frac{|C-(P\cup G)|}{|C|} = IoU - 1 - \frac{|U|}{|C|} \quad (3)$$

According to the definition of the GIoU, if the closure of the prediction bounding box P and the real frame G is C, and the union of P and G is U and satisfies inequality (4).

$$GIoU \le IoU, -1 < GIoU \le 1 \quad (4)$$

The GIoU process is just one more closure computation on top of the IoU computation, as shown in Fig. 6.

Algorithm Calculate GIoU

Input: Predicted box $P = (x_1^p, y_1^p, x_2^p, y_2^p)$, Ground truth box $G = (x_1^g, y_1^g, x_2^g, y_2^g)$
Output: Generalized Intersection over Union (GIoU)
$x_{\text{inter\_min}} \leftarrow \max(x_1^p, x_1^g)$
$y_{\text{inter\_min}} \leftarrow \max(y_1^p, y_1^g)$
$x_{\text{inter\_max}} \leftarrow \min(x_2^p, x_2^g)$
$y_{\text{inter\_max}} \leftarrow \min(y_2^p, y_2^g)$
inter_width $\leftarrow \max(0, x_{\text{inter\_max}} - x_{\text{inter\_min}})$
inter_height $\leftarrow \max(0, y_{\text{inter\_max}} - y_{\text{inter\_min}})$
inter_area $\leftarrow$ inter_width $\times$ inter_height
predicted_area $\leftarrow (x_2^p - x_1^p) \times (y_2^p - y_1^p)$
gt_area $\leftarrow (x_2^g - x_1^g) \times (y_2^g - y_1^g)$
union_area $\leftarrow$ predicted_area + gt_area $-$ inter_area
$x_{\text{enclosing\_min}} \leftarrow \min(x_1^p, x_1^g)$
$y_{\text{enclosing\_min}} \leftarrow \min(y_1^p, y_1^g)$
$x_{\text{enclosing\_max}} \leftarrow \max(x_2^p, x_2^g)$
$y_{\text{enclosing\_max}} \leftarrow \max(y_2^p, y_2^g)$
enclosing_width $\leftarrow x_{\text{enclosing\_max}} - x_{\text{enclosing\_min}}$
enclosing_height $\leftarrow y_{\text{enclosing\_max}} - y_{\text{enclosing\_min}}$
enclosing_area $\leftarrow$ enclosing_width $\times$ enclosing_height
GIoU $\leftarrow$ IoU $- \frac{\text{enclosing\_area} - \text{union\_area}}{\text{enclosing\_area}}$
return GIoU

Fig. 6. GIoU Computing Process

Based on the expression of the GIoU, the loss expression and its range of values can be obtained, as shown in **Error! Reference source not found.**.

$$l_{GIoU} = 1 - GIoU = 2 - IoU - \frac{|U|}{|C|}, 0 \le l_{GIoU} < 2 \quad (5)$$

GIoU constructs the gradient of the loss through the closure loss term when the prediction box and GT do not overlap. However, the disadvantage of closure loss is that even if there is no overlap, once the prediction box is enlarged, the closure loss can be reduced. A further issue with GIoU is that when the prediction box is situated entirely within the GT, the penalty associated with GIoU is zero.The difference from GIoU is that the goal of DIoU is to directly reduce the distance between the center points of two rectangular boxes, as shown in Equation (6). Here, c^P and c^G represent the coordinates of the center points of the predicted bounding box and the GT, respectively.

$$DIoU = IoU - \frac{\rho^2(c^P, c^G)}{d^2}, -1 < DIoU \le 1 \quad (6)$$

The symbol ρ represents the Euclidean distance between the points. The symbol d represents the distance between the diagonals of the closure area of the two bounding boxes, whose function is to normalize the distance loss. The loss function of DIoU is shown in (7). Given that the DIoU range is from -1 to 1, the resulting loss range is from 0 to 2. The issue with DIoU is that the two boxes have the potential to increase the diagonal length of the closure while maintaining the center point distance unaltered, which subsequently results in a reduction of the loss.

$$l_{DIoU} = 1 - IoU + \frac{\rho^2(c^P, c^G)}{d^2}, 0 \le l_{DIoU} < 2 \quad (7)$$

The algorithm of DIoU is illustrated in **Error! Reference source not found.** 7, which requires additional computation of the distance between the center points of the predicted and GT bounding boxes.

Algorithm Calculate DIoU

Input: Predicted box $P = (x_1^p, y_1^p, x_2^p, y_2^p)$, Ground truth box $G = (x_1^g, y_1^g, x_2^g, y_2^g)$
Output: Distance Intersection over Union (DIoU)
$x_{\text{inter\_min}} \leftarrow \max(x_1^p, x_1^g)$
$y_{\text{inter\_min}} \leftarrow \max(y_1^p, y_1^g)$
$x_{\text{inter\_max}} \leftarrow \min(x_2^p, x_2^g)$
$y_{\text{inter\_max}} \leftarrow \min(y_2^p, y_2^g)$
inter_width $\leftarrow \max(0, x_{\text{inter\_max}} - x_{\text{inter\_min}})$
inter_height $\leftarrow \max(0, y_{\text{inter\_max}} - y_{\text{inter\_min}})$
inter_area $\leftarrow$ inter_width $\times$ inter_height
predicted_area $\leftarrow (x_2^p - x_1^p) \times (y_2^p - y_1^p)$
gt_area $\leftarrow (x_2^g - x_1^g) \times (y_2^g - y_1^g)$
union_area $\leftarrow$ predicted_area + gt_area $-$ inter_area
$x_{\text{center}^p} \leftarrow \frac{x_1^p + x_2^p}{2}$
$y_{\text{center}^p} \leftarrow \frac{y_1^p + y_2^p}{2}$
$x_{\text{center}^g} \leftarrow \frac{x_1^g + x_2^g}{2}$
$y_{\text{center}^g} \leftarrow \frac{y_1^g + y_2^g}{2}$
$d \leftarrow (x_{\text{center}^p} - x_{\text{center}^g})^2 + (y_{\text{center}^p} - y_{\text{center}^g})^2$
$x_{\text{enclosing\_min}} \leftarrow \min(x_1^p, x_1^g)$
$y_{\text{enclosing\_min}} \leftarrow \min(y_1^p, y_1^g)$
$x_{\text{enclosing\_max}} \leftarrow \max(x_2^p, x_2^g)$
$y_{\text{enclosing\_max}} \leftarrow \max(y_2^p, y_2^g)$
$c \leftarrow (x_{\text{enclosing\_max}} - x_{\text{enclosing\_min}})^2 + (y_{\text{enclosing\_max}} - y_{\text{enclosing\_min}})^2$
DIoU $\leftarrow$ IoU $- \frac{d}{c}$
return DIoU

Fig. 7. The Process of DIoU

Building upon this, Complete Intersection over Union (CIoU) enhances DIoU by further incorporating aspect ratio consistency, thus offering a more comprehensive metric tailored for precise object localization. Recognizing the importance of the aspect ratio in object detection tasks, CIoU extends DIoU by introducing a penalty term for aspect ratio differences, thereby improving the bounding box regression further, as shown in (8). It incorporates three crucial components: IoU, central point distance, and aspect ratio consistency. α is the trade-off parameter for the aspect ratio term, enhancing stability in aspect ratio convergence, as shown in (9). The parameter v is a measure of the aspect ratio discrepancy between the predicted and ground truth boxes, as shown in (10). CIoU incorporates an aspect ratio penalty term to improve detection accuracy.

$$l_{GIoU} = 1 - GIoU = 2 - IoU - \frac{|U|}{|C|}, 0 \le l_{GIoU} < 2 \quad (8)$$

$$\alpha = \frac{v}{(1-IoU)+v} \tag{9}$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w_g}{h_g} - arctan\frac{w_p}{h_p}\right)^2 \tag{10}$$

To address the issues associated with CIoU, researchers have proposed EIoU Loss, which separates the aspect ratio from CIoU. The penalty term of EIoU is derived from the penalty terms of CIoU by individually calculating the length and width of both the ground truth box and the predicted box. This loss function comprises three components: overlapping loss, center distance loss, and width-height loss. The first two components follow the methods used in CIoU, while the width-height loss directly minimizes the difference between the widths and heights of the ground truth and predicted boxes, resulting in a faster convergence rate, as shown in **Error! Reference source not found.**. Here, w_c and h_c represent the width and height of the closure, respectively.

$$EIoU = IoU - \left(\frac{\rho^2(c^P, c^G)}{(d^c)^2} + \frac{(w^P - w^G)^2}{(w^c)^2} + \frac{(h^P - h^G)^2}{(h^c)^2}\right) \tag{11}$$

The algorithms of CIoU and EIoU are illustrated in Fig. 8 and Fig. 9, respectively.

Algorithm Calculate CIoU

Input: Predicted box $P = (x_1^p, y_1^p, x_2^p, y_2^p)$, Ground truth box $G = (x_1^g, y_1^g, x_2^g, y_2^g)$
Output: Complete Intersection over Union (CIoU)
$x_{\text{inter\_min}} \leftarrow \max(x_1^p, x_1^g)$
$y_{\text{inter\_min}} \leftarrow \max(y_1^p, y_1^g)$
$x_{\text{inter\_max}} \leftarrow \min(x_2^p, x_2^g)$
$y_{\text{inter\_max}} \leftarrow \min(y_2^p, y_2^g)$
$\text{inter\_width} \leftarrow \max(0, x_{\text{inter\_max}} - x_{\text{inter\_min}})$
$\text{inter\_height} \leftarrow \max(0, y_{\text{inter\_max}} - y_{\text{inter\_min}})$
$\text{inter\_area} \leftarrow \text{inter\_width} \times \text{inter\_height}$
$\text{predicted\_area} \leftarrow (x_2^p - x_1^p) \times (y_2^p - y_1^p)$
$\text{gt\_area} \leftarrow (x_2^g - x_1^g) \times (y_2^g - y_1^g)$
$\text{union\_area} \leftarrow \text{predicted\_area} + \text{gt\_area} - \text{inter\_area}$
$x_{\text{center}^p} \leftarrow \frac{x_1^p + x_2^p}{2}$
$y_{\text{center}^p} \leftarrow \frac{y_1^p + y_2^p}{2}$
$x_{\text{center}^g} \leftarrow \frac{x_1^g + x_2^g}{2}$
$y_{\text{center}^g} \leftarrow \frac{y_1^g + y_2^g}{2}$
$d \leftarrow (x_{\text{center}^p} - x_{\text{center}^g})^2 + (y_{\text{center}^p} - y_{\text{center}^g})^2$
$x_{\text{enclosing\_min}} \leftarrow \min(x_1^p, x_1^g)$
$y_{\text{enclosing\_min}} \leftarrow \min(y_1^p, y_1^g)$
$x_{\text{enclosing\_max}} \leftarrow \max(x_2^p, x_2^g)$
$y_{\text{enclosing\_max}} \leftarrow \max(y_2^p, y_2^g)$
$c \leftarrow (x_{\text{enclosing\_max}} - x_{\text{enclosing\_min}})^2 + (y_{\text{enclosing\_max}} - y_{\text{enclosing\_min}})^2$
$\text{aspect\_ratio} \leftarrow \frac{4}{\pi^2} \cdot \left(\arctan\left(\frac{x_2^p - x_1^p}{y_2^p - y_1^p}\right) - \arctan\left(\frac{x_2^g - x_1^g}{y_2^g - y_1^g}\right)\right)^2$
$\alpha \leftarrow \frac{\text{aspect\_ratio}}{(1-\text{IoU})+\text{aspect\_ratio}}$
$\text{CIoU} \leftarrow \text{IoU} - \frac{d}{c} - \alpha \cdot \text{aspect\_ratio}$
return CIoU

Fig. 8. The Process of CIoU

Currently, there is limited dedicated research on slider CAPTCHA recognition. The only notable study found so far is by Danni Wu et al., published in 2000, which used a YOLOv3-based model to recognize CAPTCHAs [15].

III. METHODOLOGY

Fig. 10 and Fig.11 display samples chosen from Geetest (https://universe.roboflow.com/project-lnr1p/geetest-wqsht) and SliderCAPTCHA (https://universe.roboflow.com/captcha-lwpyk/slide_captcha), respectively. It is obvious that SliderCAPTCHA exhibits greater diversity in both background and color variations, making it recognition more challenging to recognition.

Algorithm Calculate EIoU

Input: Predicted box $P = (x_1^p, y_1^p, x_2^p, y_2^p)$, Ground truth box $G = (x_1^g, y_1^g, x_2^g, y_2^g)$
Output: Efficient Intersection over Union (EIoU)
$x_{\text{inter\_min}} \leftarrow \max(x_1^p, x_1^g)$
$y_{\text{inter\_min}} \leftarrow \max(y_1^p, y_1^g)$
$x_{\text{inter\_max}} \leftarrow \min(x_2^p, x_2^g)$
$y_{\text{inter\_max}} \leftarrow \min(y_2^p, y_2^g)$
$\text{inter\_width} \leftarrow \max(0, x_{\text{inter\_max}} - x_{\text{inter\_min}})$
$\text{inter\_height} \leftarrow \max(0, y_{\text{inter\_max}} - y_{\text{inter\_min}})$
$\text{inter\_area} \leftarrow \text{inter\_width} \times \text{inter\_height}$
$\text{predicted\_area} \leftarrow (x_2^p - x_1^p) \times (y_2^p - y_1^p)$
$\text{gt\_area} \leftarrow (x_2^g - x_1^g) \times (y_2^g - y_1^g)$
$\text{union\_area} \leftarrow \text{predicted\_area} + \text{gt\_area} - \text{inter\_area}$
$x_{\text{center}^p} \leftarrow \frac{x_1^p + x_2^p}{2}$
$y_{\text{center}^p} \leftarrow \frac{y_1^p + y_2^p}{2}$
$x_{\text{center}^g} \leftarrow \frac{x_1^g + x_2^g}{2}$
$y_{\text{center}^g} \leftarrow \frac{y_1^g + y_2^g}{2}$
$d \leftarrow (x_{\text{center}^p} - x_{\text{center}^g})^2 + (y_{\text{center}^p} - y_{\text{center}^g})^2$
$x_{\text{enclosing\_min}} \leftarrow \min(x_1^p, x_1^g)$
$y_{\text{enclosing\_min}} \leftarrow \min(y_1^p, y_1^g)$
$x_{\text{enclosing\_max}} \leftarrow \max(x_2^p, x_2^g)$
$y_{\text{enclosing\_max}} \leftarrow \max(y_2^p, y_2^g)$
$c \leftarrow (x_{\text{enclosing\_max}} - x_{\text{enclosing\_min}})^2 + (y_{\text{enclosing\_max}} - y_{\text{enclosing\_min}})^2$
$\text{width\_diff} \leftarrow (x_2^p - x_1^p) - (x_2^g - x_1^g)^2$
$\text{height\_diff} \leftarrow (y_2^p - y_1^p) - (y_2^g - y_1^g)^2$
$\text{EIoU} \leftarrow \text{IoU} - \frac{d}{c} - \frac{\text{width\_diff}^2}{(x_{\text{enclosing\_max}} - x_{\text{enclosing\_min}})^2} - \frac{\text{height\_diff}^2}{(y_{\text{enclosing\_max}} - y_{\text{enclosing\_min}})^2}$
return EIoU

Fig. 9. The Process of E IoU

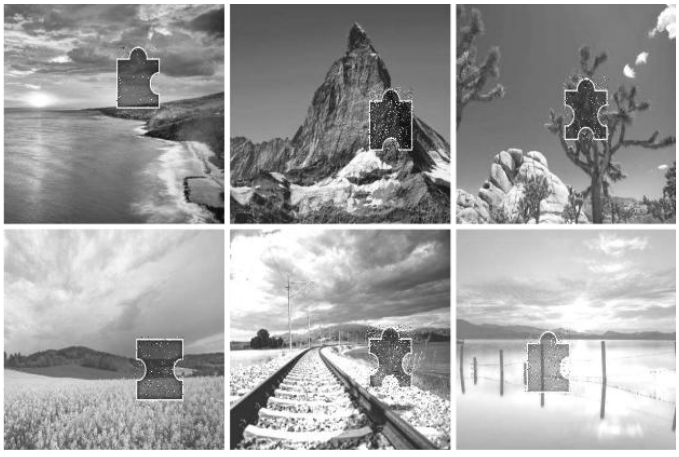

Fig. 10. Samples of Geetest Dataset

For object detection, mAP is an indicator used to evaluate accuracy, measuring how many targets are correctly identified. However, for slider CAPTCHA tasks, it is more crucial to assess whether the horizontal offset position of the target gap is accurately recognized. Therefore, mAP cannot fully reflect the precision of the recognition position. Based on these reasons, a new evaluation metric was proposed called mean Relative Offset (mRO), as shown in (12).

Fig. 11. Samples of SliderCAPTCHA Dataset

$$mRO = \sum_{i=1\to n} \frac{\left|x_1^{t_i} - x_1^{p_i}\right|}{x_2^{t_i} - x_1^{t_i}} \tag{12}$$

Among them, x_1 and x_2 represent the x-coordinates of the top-left corner and the bottom-right corner, respectively. The subscripts p_i and $t_i$represent the ith predicted bounding and the corresponding GT boxes, respectively.

A. Offset-based IoU Loss Functions

This section introduces Offset-based IoU (OIoU), a novel metric explicitly designed to cater to the distinct requirements of slider CAPTCHA tasks, with a focus on horizontal displacement accuracy over traditional overlap-centric approaches. The conventional IoU metric is widely used in object detection tasks due to its simplicity and effectiveness in measuring the overlap between predicted and ground truth bounding boxes. IoU calculates the ratio of the area of overlap to the area of union of two bounding boxes, providing a straightforward metric for object detection performance.

GIoU extends IoU by incorporating the distance between the smallest enclosing box and the union of the boxes, thus addressing localization errors to some extent. DIoU further improves this by considering the central point distance between the bounding boxes, enhancing convergence speed in model training. Similarly, CIoU attempts to integrate aspect ratio considerations. Despite its advantages, IoU and its derivatives, such as GIoU, DIoU, and CIoU, have inherent limitations when applied to sliding CAPTCHA tasks. These metrics emphasize the overlap and central alignment of boxes, which are not critical for sliding CAPTCHAs. In slider tasks, the primary objective is accurately determining the horizontal displacement of the slider bar to match the target location. Traditional IoU-based metrics, which prioritize maximizing overlap and consider vertical alignment, become less relevant.

To overcome these limitations, this work proposes OIoU, a metric specifically designed for slider CAPTCHA tasks where the horizontal accuracy of displacement is paramount. The OIoU derived from IoU focuses on the precise calculation of horizontal offsets while disregarding vertical misalignment. The OIoU involves quantifying the horizontal displacement error between the predicted bounding boxes and the GT. This is achieved by calculating the squared difference between the x-coordinates of the predicted box and GT. Here, we propose four approaches. The first approach involves directly using the difference in the x-coordinates of the top-left corners of the predicted box and the ground truth box as a penalty term, as shown in (13).

$$OIoU1 = IoU - \left(x_1^p - x_1^t\right)^2, -1 < OIoU1 \leq 1 \tag{13}$$

The advantage of this algorithm lies in its direct calculation of the offset, making it simple and efficient. However, the drawback is that it lacks normalization. The second approach draws inspiration from EIoU by incorporating the normalized center point distance, which also accelerates the model's convergence, as illustrated in (14).

$$OIoU2 = IoU - \left(x_1^p - x_1^t\right)^2 - \frac{\rho^2(c^P, c^G)}{d_c^2} \in (-2,1 \tag{14}$$

In the third approach, we normalize the offset penalty term using the width of the ground truth box, as illustrated in (15).

$$OIoU3 = IoU - \frac{\left(x_1^p - x_1^t\right)^2}{w_g^2} \in (-1,1 \tag{15}$$

The fourth approach employs the width of the bounding box enclosing both the predicted box and the GT to normalize the offset error, as illustrated in (16).

$$OIoU4 = IoU - \frac{\left(x_1^p - x_1^t\right)^2}{w_c^2} \in (-1,1] \tag{16}$$

By designing and comparing four different penalty terms, we can identify the most suitable OIoU configurations for various datasets.

B. Fixed Quantity prediction-based NMS

Due to the fixed number of targets in the slider CAPTCHA recognition task, the standard NMS in YOLOv5 cannot guarantee the output of a fixed number of predicted bounding boxes. To address this issue, we propose the Fixed Quantity Prediction-based Non-Maximum Suppression (FQP-NMS), which addresses the issue of variable numbers of prediction boxes resulting from standard NMS by ensuring a fixed output of prediction boxes, as shown in Fig. 12.

Initially, the algorithm takes the predicted bounding boxes with associated confidence scores and a predefined confidence threshold $conf$. The target is to maintain exactly Mprediction boxes as the output, which is a hyperparameter. The process begins with an empty list $Predictions$ to store the final boxes and initializes $K_{previous}$ to 0 to keep track of previously processed boxes, and $K_{current}$ to 0 for the number of candidate boxes in the current iteration. All N predicted boxes are sorted based on their confidence scores in descending order.

Depending on whether $K_{current}$ is non-zero, the algorithm selects the top $K_{current}$ boxes. Otherwise, it applies the confidence threshold to obtain an initial $K_{current}$, ensuring it is at least M by selecting the top M boxes if necessary. The standard NMS is applied on the newly selected ($K_{current} - K_{previous}$) candidate boxes. The algorithm then calculates C, representing the combined number of boxes after NMS and those already in $Predictions$. If C reaches or exceeds M, it sorts the list by confidence and outputs the top M boxes. If not, it updates $K_{previous}$ to $K_{current}$, appends the current boxes to $Predictions$, and adjusts $K_{current}$ by adding ($C - M$), repeating the process from candidate selection with the updated $K_{current}$. Through this iterative adjustment based on initial confidence filtering and subsequent NMS, the algorithm ensures that the output number of prediction boxes remains fixed at M, making it optimal for applications requiring a constant number of bounding boxes.

Algorithm Fixed Quantity Prediction NMS (FQP-NMS)

Input: Predicted boxes with confidence scores, Confidence threshold $conf$, Required number of boxes M
Output: Exactly M boxes in Predictions list.
Initialize: Predictions = [ ], $K_{\text{previous}} = 0$, $K_{\text{current}} = 0$
$N \leftarrow$ Total number of predicted boxes
while True **do**
  Sort all N boxes by descending confidence scores
  if $K_{\text{current}} \neq 0$ **then**
    Select top K_{current} boxes from sorted list
  else
    Filter boxes with confidence $> conf$ to get initial K_{current} boxes
    if $K_{\text{current}} < M$ **then**
      $K_{\text{current}} \leftarrow M$
      Select top M boxes based on confidence
    end if
  end if
  Apply standard NMS on the last $K_{\text{current}} - K_{\text{previous}}$ boxes
  $C \leftarrow$ Number of boxes after NMS + size of Predictions
  if $C \geq M$ **then**
    return top M boxes based on confidence in Predictions
  else
    Update $K_{\text{previous}} \leftarrow K_{\text{current}}$
    Append all current boxes to Predictions
    $K_{\text{current}} \leftarrow K_{\text{current}} + (C - M)$
    Continue
  end if
end while

Fig. 12. Dataset The Algorithm of FQP-NMS

The essence of FQP-NMS is to produce a fixed number of predictions by continuously adjusting the number of candidate boxes, which is effectively equivalent to dynamically modifying the preset confidence threshold to regulate the output. In the original YOLOv5, the confidence threshold is generally determined based on the optimal F1 Score or is set manually. Therefore, FQP-NMS can be regarded as a Non-Maximum Suppression mechanism based on a dynamic confidence threshold.

C.F Integrating lightweight Attentions and Backbones

In this section, a lightweight attention mechanism will be introduced into YOLOv5 to enhance YOLOv5n. The SE, ECA, CBAM, and LSKA modules will be integrated into the initial layers of the head.

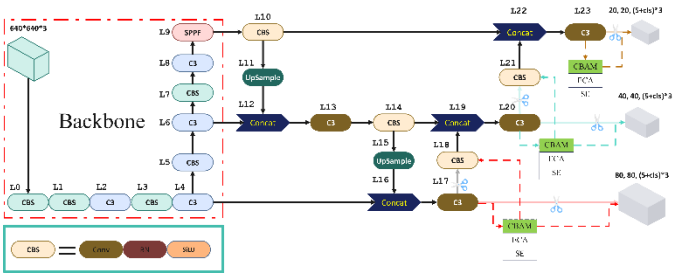

Fig. 13. Integrating CBAM, ECA, and SE into YOLOv5n

Fig. 13 illustrates the schematic diagram of CBAM, ECA, and SE attention mechanisms being added into YOLOv5. Since the three output heads of YOLOv5, P3, P4, and P5 correspond to layers L17, L20, and L23, each attention mechanism only needs to be integrated after the respective C3 module at these three layers. In the proposed improvements, for each of the three attention mechanisms, in addition to being added individually to each position, they will also be simultaneously integrated into all three positions.

Fig. 14 shows that VGG, MobileNetV3, and ShuffleNetV3 are utilized to replace the backbone of YOLOv5n. The outputs of specific feature layers from each of the three lightweight networks were aligned with the feature sizes specified for P3 (80×80), P4 (40×40), and P5 (20×20). It should be noted that MobileNet and ShuffleNet employ Grouped Convolution (GConv), Depthwise Separable Convolution (DWConv), and channel shuffle techniques. Since MobileNet and ShuffleNet have already undergone special processing and optimization of their channels, the incorporation of attention mechanisms may not yield ideal results. Further analysis will be provided in the subsequent results discussion.

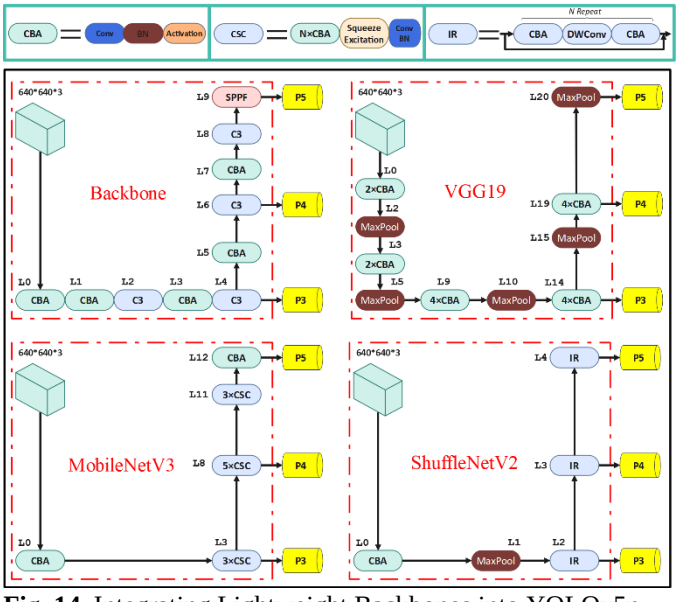

Fig. 14. Integrating Lightweight Backbones into YOLOv5n

IV. RESULTS AND DISCUSSION

The results of the proposed methods are analyzed in this section, by comparison with baseline and benchmark methods.

A. Analysis of OIoU

Table II presents validation results that compare the proposed OIoU metrics with CIoU, DIoU, EIoU, GIoU, and SIoU, on the SliderCAPTCHA dataset. Performance is assessed using mAP and mRO at an IoU threshold of 0.5. Importantly, lower mRO values indicate better performance, reflecting a smaller deviation between predicted and GT bounding boxes. Among the traditional IoU metrics, DIoU yields the highest mAP at 0.960; however, it has a relatively high mRO of 1.86%. In contrast, CIoU, EIoU, GIoU, and SIoU achieve mAP values from 0.942, to 0.955, respectively, with correspondingly higher mROs ranging from 1.72% to 1.80%. In comparison, the OIoU metrics demonstrate enhanced performance, particularly in terms of reducing mRO. OIoU1, which incorporates a penalty based on the L2 distance between the top-left corners of the predicted and GT boxes, achieves a mAP of 0.954 with a significantly lower mRO of 1.32%. The OIoU2 adds a normalized center distance based on OIoU1, leading to a slight improvement in both performance metrics. The main difference between OIoU3 and OIoU1 lies in the normalization of offset using GT width. This adjustment resulted in an increase in mAP to 0.975; however, mRO, increased to 1.39%. In comparison to OIoU3, OIoU4 normalizes using the diagonal length of the bounding box, which resulted in a degradation of both mAP and mRO. This may be attributed to the inherent contradiction between the diagonal length and these metrics, where an increase in one often leads to a decrease in the other. Compared to traditional IoU, OIoU demonstrates improvements in both metrics, with a particularly significant increase observed in mRO.

TABLE II VALIDATION RESULTS WITH DIFFERENT IOUS ON SLIDERCAPTCHA

Type	mAP@0.5	mRO@0.5
YOLOv5n + CIoU	0.942	1.77%
YOLOv5n + DIoU	0.96	1.86%
YOLOv5n + EIoU	0.955	1.72%
YOLOv5n + GIoU	0.944	1.80%
YOLOv5n + SIoU	0.95	1.72%
YOLOv5n + OIoU1	0.954	1.32%
YOLOv5n + OIoU2	0.958	1.28%
YOLOv5n + OIoU3	0.975	1.39%
YOLOv5n + OIoU4	0.97	1.43%

Table III displays the test results of different IoU metrics on the Geetest dataset. It is observed that the majority of IoU metrics outperform their counterparts tested on the SliderCAPTCHA, indicating that Geetest is relatively easier for recognition tasks. Furthermore, the three OIoU variations generally perform better than traditional IoU on the Geetest dataset. Notably, both performance metrics of OIoU2 are lower than those of OIoU1, suggesting that the normalized center distance may be counterproductive for directly optimizing mRO on simpler datasets. The mRO for IoU3 is the same as that of OIoU1, indicating that whether offset normalization is applied has little impact on mRO in simpler datasets; however, it does benefit mAP, a trend also confirmed by results on the SliderCAPTCHA dataset. Finally, OIoU4's with still perform poorly due to the conflicting optimization paths.

Table IV illustrates the performance on the SliderCAPTCHA dataset after applying L1 distance and Focal-OIoU. The table indicates that all four OIoU losses experience a decline in performance with L1 distance. This decline may be due to the slower convergence speed of L1 distance compared to L2 distance and its lesser stability in high-dimensional spaces. Focal-OIoU is designed to further optimize predictions with higher IoU while minimizing the impact of predictions with lower IoU. As a result, the table shows an overall increase in mAP with the use of Focal, while mRO decreases.

TABLE III VALIDATION RESULTS WITH DIFFERENT IOUS ON GEETEST

Type	mAP@0.5	mRO@0.5
YOLOv5n + CIoU	0.993	1.41%
YOLOv5n + DIoU	0.983	1.41%
YOLOv5n + EIoU	0.994	1.40%
YOLOv5n + GIoU	0.988	1.29%
YOLOv5n + SIoU	0.99	1.46%
YOLOv5n + OIoU1	0.985	1.27%
YOLOv5n + OIoU2	0.978	1.38%
YOLOv5n + OIoU3	0.993	1.27%
YOLOv5n + OIoU4	0.984	1.62%

TABLE IV VALIDATION RESULTS WITH L1 DISTANCE AND FOCAL-OIOUS ON SLIDERCAPTCHA

Type	mAP@0.5	mRO@0.5
YOLOv5n + OIoU1	0.954	1.32%
YOLOv5n + OIoU1 (L1)	0.957	1.46%
YOLOv5n + Focal-OIoU1	0.967	1.33%
YOLOv5n + OIoU2	0.958	1.28%
YOLOv5n + OIoU2 (L1)	0.943	1.29%
YOLOv5n + Focal-OIoU2	0.919	1.53%
YOLOv5n + OIoU3	0.975	1.39%
YOLOv5n + OIoU3 (L1)	0.953	1.64%
YOLOv5n + Focal-OIoU3	0.919	1.75%
YOLOv5n + OIoU4	0.97	1.43%
YOLOv5n + OIoU4 (L1)	0.957	1.57%
YOLOv5n + Focal-OIoU4	0.967	1.44%

The performance on Geetest is generally consistent with that on SliderCAPTCHA, except for OIoU4, as shown in Table V. L2 distance outperforms L1 distance for most OIoU metrics, and while Focal-OIoU enhances mAP, it concurrently leads to a decline in mRO performance. In summary, based on the previous experimental analyses, using L2 distance in OIoU (without focal loss) demonstrates superior overall performance,

particularly in the mRO metric. Among the metrics, OIoU1 and OIoU2 exhibit the best performance; however, OIoU2 involves additional computation due to the normalization of the center distance. Therefore, OIoU1 offers a favorable balance between performance and computational efficiency.

TABLE V VALIDATION RESULTS WITH L1 DISTANCE AND FOCAL-OIOUS ON GEETEST

Type	mAP@0.5	mRO@0.5
YOLOv5n + OIoU1	0.985	1.27%
YOLOv5n + OIoU1 (L1)	0.977	1.36%
YOLOv5n + Focal-OIoU1	0.987	1.45%
YOLOv5n + OIoU2	0.978	1.38%
YOLOv5n + OIoU2 (L1)	0.99	1.33%
YOLOv5n + Focal-OIoU2	0.986	1.36%
YOLOv5n + OIoU3	0.993	1.27%
YOLOv5n + OIoU3 (L1)	0.984	1.38%
YOLOv5n + Focal-OIoU3	0.994	1.51%
YOLOv5n + OIoU4	0.984	1.62%
YOLOv5n + OIoU4 (L1)	0.991	1.31%
YOLOv5n + Focal-OIoU4	0.993	1.35%

B. Analysis of FQP-NMS

Table VI presents the validation results of the proposed FQP-NMS method in conjunction with OIoU1 to OIoU4 on the SliderCAPTCHA. The performance metrics are measured using mAP and mRO. FQP-NMS is a novel approach designed to maintain a consistent number of predicted bounding boxes during the detection process, which is crucial for slider CAPTCHA tasks where typically only one valid output exists. Traditional NMS can yield a variable number of prediction boxes, leading to scenarios with too few or no predictions. By ensuring a fixed number of predictions, FQP-NMS effectively addresses this variability. The results indicate that integrating FQP-NMS with the OIoU algorithms generally improves performance across both mAP and mRO metrics. reaffirming that improvements in precision coincide with lower relative offset values, signifying more accurate predictions.

TABLE VI VALIDATION RESULTS WITH FQP-NMS ON SLIDERCAPTCHA

Type	mAP@0.5	mRO@0.5
YOLOv5n + OIoU1	0.954	1.32%
YOLOv5n + OIoU1 + FQP-NMS	0.97	1.29%
YOLOv5n + OIoU2	0.958	1.28%
YOLOv5n + OIoU2 + FQP-NMS	0.972	1.25%
YOLOv5n + OIoU3	0.975	1.39%
YOLOv5n + OIoU3 + FQP-NMS	0.99	1.38%
YOLOv5n + OIoU4	0.97	1.43%
YOLOv5n + OIoU4 + FQP-NMS	0.981	1.32%

Overall, the results underscore the efficacy of FQP-NMS in stabilizing prediction counts while enhancing overall performance in slider CAPTCHA detection. The combination of FQP-NMS with OIoU not only demonstrates significant improvements in mAP but also ensures that the mRO remains low, emphasizing the suitability for tasks with specific detection requirements, such as sliding CAPTCHAs.

TABLE VII VALIDATION RESULTS WITH FQP-NMS ON GEETEST

Type	mAP@0.5	mRO@0.5
YOLOv5n + OIoU1	0.985	1.27%
YOLOv5n + OIoU1 + FQP-NMS	0.994	1.25%
YOLOv5n + OIoU2	0.978	1.38%
YOLOv5n + OIoU2 + FQP-NMS	0.987	1.32%
YOLOv5n + OIoU3	0.993	1.27%
YOLOv5n + OIoU3 + FQP-NMS	0.995	1.24%
YOLOv5n + OIoU4	0.984	1.62%
YOLOv5n + OIoU4 + FQP-NMS	0.991	1.55%

Table VII illustrates the performance of FQP-NMS on the Geetest dataset. Like the results on the SliderCAPTCHA dataset, OIoU1, OIoU2, OIoU3, and OIoU4 all show improvements in both mAP and mRO, further demonstrating the exceptional performance and robustness of FQP-NMS across different datasets. In summary, FQP-NMS is particularly well-suited for detection tasks where the number of objects in the images is relatively fixed.

C. Analysis of Lightweight Attentions on SliderCAPTCHA

TABLE VIII VALIDATION RESULTS WITH DIFFERENT ATTENTIONS ON SLIDERCAPTCHA

Type	Layer	mAP@0.5	mRO@0.5
YOLOv5n (Baseline)	\	0.942	1.77%
Baseline + CBAM	+24	0.904	2.41%
Baseline + CBAM	+21	0.97	1.60%
Baseline + CBAM	+18	0.96	1.62%
Baseline + CBAM	+18,22,26	0.957	1.95%
Baseline + ECA	+24	0.976	1.49%
Baseline + ECA	+21	0.967	1.75%
Baseline + ECA	+18	0.973	1.6%
Baseline + ECA	+18,22,26	0.944	1.59%
Baseline + SE	+24	0.945	1.73%
Baseline + SE	+21	0.954	1.48%
Baseline + SE	+18	0.959	1.85%
Baseline + SE	+18,22,26	0.858	2.36%
Baseline + C3-LSKA	=17,20,23	0.927	2.05%
Baseline + C3-LSKA	=13,17,20,23	0.963	1.84%

In Table VIII, it is observed that integrating CBAM into L21

and L18 of YOLOv5n results in performance improvements, while other placements show a decline, leading to an overall modest enhancement. This may be because attention mechanisms added to the deeper layers of the network focus more on global perspectives rather than local bounding boxes, thus showing limited effectiveness. Unlike CBAM, SE employs only channel attention and uses fully connected networks to generate channel weights, like CBAM, but it achieves a more noticeable improvement in mRO. ECA also utilizes channel attention; however, all four integration methods enhance both mAP and mRO. Unlike SE, which uses fully connected layers to generate weights, ECA employs 1D convolution, considering only the influence of adjacent channels, which improves the model's generalization ability. Additionally, the use of C3-LSKA results in decreased performance, further proving that spatial attention, especially with large receptive fields, is less effective for tasks like slider CAPTCHA recognition.

TABLE IX VALIDATION RESULTS WITH DIFFERENT ARCHITECTURES ON GEETEST

Type	Layer	mAP@0.5	mRO@0.5
YOLOv5n (Baseline)	\	0.993	1.41%
Baseline + CBAM	+24	0.993	1.40%
Baseline + CBAM	+21	0.984	1.69%
Baseline + CBAM	+18	0.984	1.64%
Baseline + CBAM	+18,22,26	0.989	1.49%
Baseline + ECA	+24	0.99	1.41%
Baseline + ECA	+21	0.986	1.71%
Baseline + ECA	+18	0.985	1.38%
Baseline + ECA	+18,22,26	0.994	1.56%
Baseline + SE	+24	0.983	1.56%
Baseline + SE	+21	0.993	1.44%
Baseline + SE	+18	0.994	1.50%
Baseline + SE	+18,22,26	0.991	1.62%
Baseline + C3-LSKA	=17,20,23	0.982	1.61%
Baseline + C3-LSKA	=13,17,20,23	0.994	1.61%

Table IX displays the results on the Geetest dataset, indicating that all attention mechanisms, except for ECA (+18) and ECA (+24), experience a decline in mRO. This suggests that adding attention mechanisms in the head of the network is not effective in reducing offset errors for simpler datasets. Combining the results from both datasets, it is observed that the ECA attention mechanism demonstrates relatively better robustness, with ECA (+24) exhibiting the most stable performance.

The validation results on SliderCAPTCHA presented in Table X highlight the performance of various architectures on the SliderCAPTCHA dataset, with mAP and mRO indicating the effectiveness of different combinations of models and attention mechanisms. The notable performance enhancements observed with VGG19, YOLOv5n (baseline), and YOLOv3 (benchmark) when combined with ECA can be attributed to these architectures' robustness and their ability to leverage the enhanced feature extraction capabilities provided by ECA. The integration of ECA allows these models to assign different weights to various channels, thereby improving sensitivity to critical features that contribute to accurate object detection. Consequently, this results in higher mAP values and lower mRO values, indicating better alignment with ground truth. In contrast, the performance of MobileNetV3 and ShuffleNetV2 diminishes when ECA is applied. This may be due to the inherent design philosophy of these architectures, which prioritize efficiency and are optimized for lightweight applications. The addition of ECA to these models could introduce complexity that undermines the benefits of enhanced feature extraction, especially where computational constraints are crucial. Additionally, MobileNetV3 and ShuffleNetV2 already incorporate their own attention mechanisms optimized for resource efficiency, and adding ECA might disrupt their balanced architecture, leading to poorer performance.

TABLE X VALIDATION RESULTS WITH THE BENCHMARK ON SLIDERCAPTCHA

Type	Layer	mAP@0.5	mRO@0.5
YOLOv5n (Baseline)	\	0.942	1.77%
Baseline + ECA + OIoU1 + FQP-NMS	+24	0.978	1.18%
Baseline + VGG19 + OIoU1 + FQP-NMS	\	0.992	0.97%
Baseline + VGG19 + ECA + OIoU1 + FQP-NMS	+35	0.994	0.88%
Baseline + ShuffleNetV2 + OIoU1 + FQP-NMS	\	0.995	1.08%
Baseline + ShuffleNetV2 + ECA + OIoU1 + FQP-NMS		0.988	1.32%
Baseline + MobileNetV3 + OIoU1 + FQP-NMS	\	0.987	1.43%
Baseline + MobileNetV3 + ECA + OIoU1 + FQP-NMS		0.963	1.55%
YOLOv3 (Benchmark)	\	0.995	1.09%
Benchmark + OIoU1 + FQP-NMS	\	0.995	0.93%

In conclusion, the combination of Baseline + VGG19 + ECA + OIoU1 + FQP-NMS achieves the best performance, followed closely by YOLOv3 + OIoU1 + FQP-NMS. The Baseline + ShuffleNetV2 + OIoU1 + FQP-NMS also demonstrates commendable performance, confirming that the right combinations of architectures and attention mechanisms can significantly enhance detection capabilities on the SliderCAPTCHA dataset.

The validation results on Geetest datasets consistently reveal the effectiveness of specific model combinations, as shown in Table XI. Combining the performance results from both the SliderCAPTCHA and Geetest datasets, we can conclude that the configuration of Baseline + VGG19 + ECA + OIoU1 + FQP-NMS consistently delivers the best performance, achieving high mAP and low mRO in both scenarios. Following this, the YOLOv3 model paired with OIoU1, and FQP-NMS also demonstrates strong results, highlighting its effectiveness in object detection tasks. Additionally, the Baseline +

ShuffleNetV2 + OIoU1 + FQP-NMS configuration shows commendable performance, affirming its capability as a viable alternative.

TABLE XI VALIDATION RESULTS WITH THE BENCHMARK ON GEETEST

Type	Layer	mAP@0.5	mRO@0.5
YOLOv5n (Baseline)	\	0.993	1.41%
Baseline + ECA + OIoU1 + FQP-NMS	+24	0.988	1.20%
Baseline + VGG19 + OIoU1 + FQP-NMS	\	0.995	0.92%
Baseline + VGG19 + ECA + OIoU1 + FQP-NMS	+35	0.995	0.93%
Baseline + ShuffleNetV2 + OIoU1 + FQP-NMS	\	0.995	1.08%
Baseline + ShuffleNetV2 + ECA + OIoU1 + FQP-NMS	+18	0.995	1.17%
Baseline + MobileNetV3 + OIoU1 + FQP-NMS	\	0.995	1.18%
Baseline + MobileNetV3 + ECA + OIoU1 + FQP-NMS	+18	0.995	1.19%
YOLOv3 (Benchmark)	\	0.945	1.27%
Benchmark + OIoU1 + FQP-NMS	+35	0.995	0.92%

TABLE XII PARAMS AND GFLOPS WITH DIFFERENT MODELS

Type	Layer	PARAMs	GFLOPs
YOLOv5n (Baseline)	\	1760518	4.1
Baseline + CBAM	+24	1826408	4.2
Baseline + CBAM	+21	1777128	4.1
Baseline + CBAM	+18	1764776	4.1
Baseline + CBAM	+18,22,26	1847276	4.2
Baseline + ECA	+24	1760521	4.1
Baseline + ECA	+21	1760521	4.1
Baseline + ECA	+18	1760521	4.1
Baseline + ECA	+18,22,26	1760527	4.1
Baseline + SE	+24	1793286	4.2
Baseline + SE	+21	1768710	4.1
Baseline + SE	+18	1762566	4.1
Baseline + SE	=18,22,26	1803526	4.2
Baseline + C3-LSKA	=17,20,23	1515078	3.9
Baseline + C3-LSKA	=13,17,20,23	1503686	3.8
Baseline + VGG19 + ECA	\	1805769	20.3
YOLOv5n (Baseline) + ECA	+24	1760521	4.1
Baseline + ShuffleNetV2	\	1855554	3.8
Baseline + MobileNetV3	\	2092342	3.0
YOLOv3 (Benchmark)	\	61497430	154.5

Table XII presents the PARAMs and computational complexity in terms of GFLOPs for various architectures. Compared to other attention mechanisms, the ECA mechanism demonstrates the smallest parameter count and computational complexity. Notably, while YOLOv3 demonstrates superior performance, it significantly surpasses YOLOv5 in both parameter count and computational complexity, with approximately 35 times more parameters and 37 times greater GFLOPs. The configuration of Baseline + VGG19 + ECA achieves the best performance among the models assessed, with a parameter count and computational complexity that are considerably lower than those of YOLOv3. However, it still exhibits roughly five times the GFLOPs of YOLOv5, indicating a higher computational burden. In contrast, Baseline + ShuffleNetV2 and Baseline + ECA combinations offer good performance while maintaining a lower parameter count and computational complexity. These models demonstrate that it is possible to achieve relatively high accuracy while keeping the resource requirements manageable, making them suitable for deployment in resource-constrained environments. Overall, this analysis highlights the trade-offs between model performance, complexity, and efficiency across different architectures.

Fig. 15 presents the visualization results of YOLOv5 combined with the ShuffleNet backbone on the SliderCAPTCHA dataset. Despite the complex background and the presence of gap artifacts that can interfere with detection, the model demonstrates excellent overall performance in accurately identifying the relevant elements. This indicates that the model can maintain high accuracy and reliability despite the complexities introduced by the background.

Fig. 15. Visualization Results of YOLOv5n with the ShuffleNet on the SliderCAPTCHA

Fig. 16 displays the visualization results of the same model on the Geetest dataset. Despite the images in the Geetest dataset primarily being grayscale with low contrast, the model effectively identifies the position of the gaps, demonstrating a high level of detection accuracy. This showcases the robustness of the YOLOv5n along with the lightweight ShuffleNet backbone in slider detection in challenging visual conditions.

Fig. 16. Visualization Results of YOLOv5n with the ShuffleNet on the Geetest

V.CONCLUSIONS

This study focuses on the recognition of slider CAPTCHAs and introduces a more suitable evaluation metric, namely mRO. Additionally, a novel offset-based IoU and a Non-Maximum Suppression (NMS) mechanism are proposed, and both algorithms significantly enhance the mRO of all evaluated models. Furthermore, the YOLOv5n model is optimized by incorporating lightweight attention mechanisms and enhancing the backbone network. Experimental results demonstrate that the implementation of the proposed algorithms effectively reduces deviations in slider CAPTCHA recognition, providing new insights and references for further CAPTCHA research.

ACKNOWLEDGMENT

The Authors would also like to thank and acknowledge the Faculty of Electrical Engineering Universiti Teknologi MARA Shah Alam for their support.

REFERENCES

[1] W. Xing, M. R. Sultan Mohd, J. Johari, and F. Ahmat Ruslan, "A review on object detection algorithms based deep learning methods / Wan Xing ... [et al.]," *Journal of Electrical and Electronic Systems Research (JEESR)*, vol. 23, no. 1, pp. 1–13, Oct.2023.

[2] G. Chang, H. Gao, G. Pei, S. Luo, Y. Zhang, N. Cheng, Y. Tang, and Q. Guo, "The robustness of behavior-verification-based slider CAPTCHAs," *Journal of Information Security and Applications*, vol. 81, p. 103711, Mar.2024.

[3] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks." arXiv, Apr. 07, 2020.

[4] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," in *Computer Vision – ECCV 2018*, Cham, 2018, pp. 3–19.

[5] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-Excitation Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 2011–2023, 2020.

[6] K. W. Lau, L.-M. Po, and Y. A. U. Rehman, "Large Separable Kernel Attention: Rethinking the Large Kernel Attention design in CNN," *Expert Syst. Appl.*, vol. 236, no. C, 2024

[7] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications." arXiv, Apr. 16, 2017. [Online]. http://arxiv.org/abs/1704.04861.

[8] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 6848–6856.

[9] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design," 2018, pp. 116–131. [Online]. https://openaccess.thecvf.com/content_ECCV_2018/html/Ningning_Light-weight_CNN_Architecture_ECCV_2018_paper.html.

[10] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "UnitBox: An Advanced Object Detection Network," in *Proceedings of the 24th ACM international conference on Multimedia*, New York, NY, USA, 2016, pp. 516–520.

[11] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression." arXiv, Apr. 14, 2019. [Online]. http://arxiv.org/abs/1902.09630.

[12] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression." arXiv, Nov. 19, 2019.

[13] Y.-F. Zhang, W. Ren, Z. Zhang, Z. Jia, L. Wang, and T. Tan, "Focal and Efficient IOU Loss for Accurate Bounding Box Regression." arXiv, Jul. 15, 2022.

[14] Z. Gevorgyan, "SIoU Loss: More Powerful Learning for Bounding Box Regression." arXiv, May 25, 2022.

[15] D. Wu, J. Qiu, H. Huang, L. Yin, Z. Gu, and Z. Tian, "Resnet-based slide puzzle captcha automatic response system," *Communications in Computer and Information Science*, vol. 1254 CCIS, pp. 140–153, 2020.