IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.Doi Number

Enhancing Bounding Box Regression for Object Detection: Dimensional Angle Precision IoU-Loss

Hilmy Aliy Andra Putra[1], Aniati Murni[2], Dina Chahyati[3]
Department of Computer Science, Universitas Indonesia, Depok 16424, Indonesia

Corresponding author: Hilmy Aliy Andra Putra (hilmy.aliy11@ui.ac.id).

This work was supported by Tokopedia-Universitas Indonesia Artificial Intelligence (UI AI) Center of Excellence, Universitas Indonesia for Graphics Processing Unit (GPU) computing server facilities.

ABSTRACT Bounding Box Regression (BBR) plays a critical role in object detection by refining the predicted location and size of objects to enhance model accuracy. This process involves adjusting the coordinates of the proposed bounding boxes to enhance their precision. The Intersection over Union (IoU) loss metric was introduced to improve the IoU metric for integration into the model training process, measure discrepancies between the model's predictions and ground truth, and ensures meaningful gradient updates during training. In practice, IoU loss has demonstrated improvements in object detection performance by enhancing the localization accuracy of bounding boxes. Despite significant technological advancements and the various advantages and disadvantages of IoU loss, improving the accuracy and efficiency of BBR remains an active research area in computer vision. Various IoU loss variations have evolved with new formulations and methods to improve accuracy and convergence speed. A new loss function, Dimensional Angle Precision IoU (DAPIoU) loss, is introduced in this research to enhance BBR and serve as a new object detection loss function to address the limitations in previous loss function research results. This study conducts two types of experiments: a simulation experiment on synthetic data and an experiment on real-world datasets. The datasets used are MS-COCO and PASCAL VOC datasets. The object detection models used are YOLOv7, YOLOv9, and Faster R-CNN. The results from the real-world datasets experiments are evaluated using the mean Average Precision (mAP) method, including object size metrics, comparing several previous loss functions based on IoU.

INDEX TERMS Bounding box regression, loss function, intersection over union

I. INTRODUCTION

Object detection is one of the most vital research areas in computer vision, with its success being essential for applications ranging from video surveillance to autonomous navigation systems. Deep learning methods are representation learning methods with several levels of representation, obtained by composing simple but non-linear modules that each transform representations at one level (starting with raw input) into representations at a higher level, slightly more abstract [1]. An object detection model's effectiveness depends on accurately localizing object by refining bounding box coordinates. In the field of object detection, the main target is to determine the location and recognize objects in an image. Localization means estimating the bounding box that surrounds the object. The bounding box is formed based on the coordinates of the pixel in the upper left corner and the pixel in the lower right corner, which are used as a standard method in object localization. The pixel in the upper left corner has coordinates (x_{min}, y_{min}) while the pixel in the lower right corner has coordinates (x_{max}, y_{max}), the pixel coordinate system has an origin $(0,0)$ at the upper left corner pixel of the entire image [2].

Achieving high precision in object detection is challenging due to variations in object scale, occlusion, and background clutter. Modern object detection frameworks, such as YOLO [3], Faster R-CNN [4], and SSD [5], employ Convolutional Neural Networks (CNNs) to extract features and predict bounding boxes and class probabilities. These methods have significantly advanced the state of the art by improving both speed and accuracy. However, the choice of loss function used

to train these models plays a critical role in their performance. Intersection over Union (IoU) based loss functions are widely used to improve the accuracy of bounding box predictions by measuring the overlap between the predicted and ground truth boxes [6]. IoU loss is designed to optimize IoU directly, providing a way to measure the discrepancy between the model predictions and ground truth and providing sufficient gradients for model learning [7]. In practice, IoU loss has shown improvements in object detection performance, providing enhancements in terms of bounding box localization accuracy [8].

Despite significant technological advances, including various advantages and disadvantages of IoU loss, improving the accuracy and efficiency of Bounding Box Regression (BBR) remains an active research area in computer vision. Several variations of IoU loss have developed over time with various new formulas and methods to enhance accuracy and convergence speed. Recent advancements have introduced variations of IoU, such as GIoU [6], CIoU and DIoU [9], which aim to address the limitations of the traditional IoU metric by incorporating additional geometric factors. Currently, numerous IoU-based loss functions have demonstrated improved accuracy and faster convergence rates, making them essential components in developing high-performance object detection models [10]. These advanced loss functions enhance the model's ability to differentiate between high-quality and low-quality bounding box predictions, leading to better localization accuracy [11]. Furthermore, the integration of multi-scale feature representations and anchor box refinements has enabled models to detect objects of varying sizes more effectively. The evaluation of object detection models typically involves metrics such as mean Average Precision (mAP), which provides a comprehensive measure of precision and recall across different intersection thresholds [12]. Continuous research and development in this domain are focused on overcoming existing challenges and pushing the boundaries of object detection capabilities, ensuring robust performance across diverse and complex real-world scenarios [13].

IoU-based loss function have significantly contributed to advancements in 2D object detection, but they still face critical limitations that hinder precise localization. Standard IoU struggles to provide meaningful gradient signals when there is no overlap between the predicted and ground truth bounding boxes, leading to slower convergence and suboptimal alignment. Variants such as GIoU, DIoU, and CIoU have introduces improvements, yet they fall short in addresing fine-grained localization challenges, particularly for small or irregularly shaped objects. Furthermore, these methods neglect angular relationships and corner discrepancies, which can significantly impact the accuracy of BBR. The proposed DAPIoU loss addresses these gaps by incorporating a 3D spatial perspective to enhance 2D localization accuracy. By introducing angular penalties and corner distance metrics, DAPIoU improves gradient consistency, even for non-overlapping boxes, and achieves superior alignment of bounding boxes. This innovation leads to faster convergence and higher precision, especially for small, large, and complex objects. The DAPIoU loss is particularly beneficial in application requiring precise 2D localization, such as autonomous driving, where accurate bounding box predictions are critical for detecting road signs and pedestrians, and medical imaging, where fine-grainned localization is essential fot identifying abnormalities in scans. As illustrated in the Fig. 1, variation in loss values across different IoU-based metrics such as IoU loss, GIoU loss, and CIoU demonstrate their unique sensitivities to bounding box misalignment. DAPIoU loss, with $k = 2$ and $z = 10{,}000$, assigns higher loss values in cases of sever misalignment due to corner distance and angular penalty components, which are designed to encourage more precise localization. To analyze how different loss function behave, Fig. 1 evaluates four key bounding box misalignment scenarios:

1. Center-alignment but not fully overlapping – The predicted bounding box is centered on the ground truth but does not fully align, creating a small misalignment gap. Traditional IoU-based loss functions assign similar penalties in this case, while DAPIoU imposes a slightly stronger penalty to refine alignment.
2. Partial intersection – The predicted box partially overlaps the ground truth. IoU and GIoU assign relatively low penalties, while DAPIoU and CIoU impose stricter losses to encourage more precise localization.
3. Half-size ground truth relative to prediction – The ground truth box is exactly half size of the predicted box, leading to misalignment in both scale and position. DAPIoU produces higher loss in this scenario, penalizing excessive scalse differences and misaligned corner positioning.
4. Extreme misalignment – The predicted bounding box is completely off-target with no meaningful overlap with ground truth. DAPIoU assigns the highest loss in this scenario due to its distance penalty, reinforcing a stronger correction for severe misalignment.

The scenarios where DAPIoU loss values exceed 1 highlight the sticker penalties imposed on highly misaligned predictions, which can drive the model toward improved bounding box alignment during training. However, it is important to note that a higher loss value does not necessarily indicate superiority. Rather, it reflects how aggressively the loss function penalizes localization errors, which can help refine model training. DAPIoU's ability to penalize misalignment more effectively can be particularly beneficial for object detection tasks that require high localization precision, such as small object detection, autonomous driving, and medical imaging. While raw loss values alone do not determine the effectiveness of a loss function, DAPIoU's improvements in mAP and bounding box IoU scores, as demonstrated in the experimental results, confirm its advantages in practical applications.

This capability makes DAPIoU particularly suitable for advanced object detection tasks that require high-fidelity localization. Its ability to impose stricter penalties for misalignment enhances model learning, leading to significant improvements in localization accuracy and convergence speed. These characteristics make DAPIoU a valuable contribution to ongoing advancements in computer vision, particularly in applications demanding high-precision object detection.

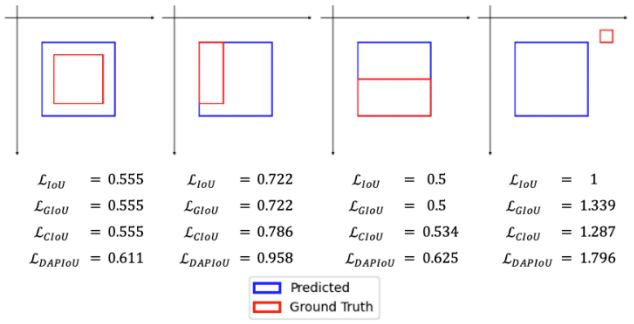

FIGURE 1. Comparative analysis of IoU-Based Loss Function for BBR.

This research focuses on developing a new loss function, DAPIoU loss to improve BBR. By incorporating penalties for the distances between bounding box corners and the precision measurement of both bounding boxes in a 3D space, DAPIoU aims to enhance convergence stability and accuracy for large, medium and small objects. The study involves both synthetic simulations and real-world experiments using benchmark datasets and various detection models, with evaluations based on mAP. This approach aims to contribute to the ongoing research in computer vision, offering new insights and methodologies for improving object detection performance. Although this study primarily focuses on enhancing BBR for object detection, the proposed DAPIoU loss has potential application beyond this domain. Its ability to address fine-grainned localization challenges through angular penalties and corner distance metrics makes it a promising candidate for tasks such as instance segmentation, object tracking, and keypoint detection. These tasks share a reliance on precise localization, and future work will explore the extenstion of DAPIoU to these broader applications.

This study conducts three types of experiments: the first is a single-group BBR simulation experiment on synthetic data, the second is simulation experiment on synthetic data, and the third is an experiment on real-world datasets. The use of synthetic data allows us to assess loss function behavior in controlled environment, ensuring a comprehensive analysis of its impact on BBR. The datasets used are PASCAL VOC [13] and MS-COCO [14] datasets. The detection models used include YOLOv7, YOLOv9, and Faster R-CNN. The results of experiments on real-world datasets employ the evaluation method of mAP, comparing several previous loss functions based on IoU. To address the challenges in BBR and improce object detection performance, this paper makes the following key contributions:

- **Proposal of a Novel Loss function**: We introduce the DAPIoU loss, which integrate angular penalties and corner distance metrics to enhance bounding box alignment precision in 2D object detection tasks.
- **Improved Localization Accuracy**: DAPIoU addresses limitation in existing IoU-based losses by providing meaningful gradient signals, even for non-overlapping bounding boxes, and significantly improving localization accuract for small, large, and irregularly shaped objects.
- **Extensice Benchmark Evaluation**: We conduct comprehensive experiments on benchmark datasets, including PASCAL VOC 2007+2012 and MS COCO 2017, using state-of-the-art detection models such as YOLOv7, YOLOv9, and Faster R-CNN. The results demonstrate that DAPIoU outperforms existing IoU-based losses in terms of mAP and convergence speed.
- **Hyperparameter Insights**: We analyze the sensitivity of hyperparameter k and z, providing practical guidance for optimizing DAPIoU in diverse object detection scenarios.

II. RELATED WORK

A. MODEL DETECTION FOR BOUNDING BOX

In recent times, the landscape of object detection has been transformed by deep learning algorithms, addressing key challenges in classification and localization. Among these advancements, the YOLO series [3], [15], [16], [17], [18] has evolved to include various versions tailored for real-time object detection. YOLO stands out for its rapid and precise real-time detection capabilities. The latest iteration, YOLOv9 represents a significant advancement in the field of object detection, offering a powerful combination of speed, accuracy, and efficiency. It continues the YOLO tradition of providing state-of-the-art real-time detection capabilities, further pushing the boundaries of what is possible in computer vision. Typically, YOLO models use IoU-based box regression and ground truth as soft object labels for detection task with the continuous development of these models, they have become essential tools in numerous real-world applications, pushing the boundaries of what is possible in object detection technology.

Faster R-CNN [19] significantly improves the speed and accuracy of object detection by integrating region proposal generation and object detection into a single, unified framework. Faster R-CNN builds upon the success of its predecessors, and Fast R-CNN [20] and R-CNN [21]. The workflow of Faster R-CNN consists of two main stages: the Region Proposal Network (RPN), which scans the input image and proposes candidate object regions, and the Object Detection Network, which refines these proposals through object classification and BBR.

B. LOSS FUNCTION FOR BOUNDING BOX REGRESSION

- ℓ_1 Loss

ℓ_1 loss is a common loss function used in regression tasks to measure the difference between predicted values and actual values. ℓ_1 loss takes the absolute value of the differences. This makes ℓ_1 loss less sensitive to outliers compared to ℓ_2 loss. Mathematically, for a set of predictions $\hat{x}_i$ and actual values x_i with n data points, the ℓ_1 loss is defined as:

$$\ell_1\ loss = \sum_{i=1}^{n} |x_i - \hat{x}_i| \tag{1}$$

ℓ_1 loss also known as Mean Absolute Error (MAE), is beneficial in scenarios where robustness to outlier desired. The absolute differences ensure that large deviations do not disproportionately affect the loss, making it suitable for datasets with outlier or noisy data. By focusing on the median of the errors, ℓ_1 loss can providen a more accurate representation of the typical prediction error in such context [22].

- ℓ_2 Loss

ℓ_2 loss is similar to ℓ_1 loss but with squared penalty on errors. It calculates the average of the squared differences between the predicted values $\hat{x}_i$ and the actual values x_i. Mathematically, for a set of predictions and actual values with n data points, the ℓ_2 loss is defined as:

$$\ell_2\ loss = \sum_{i=1}^{n} (x_i - \hat{x}_i)^2 \tag{2}$$

This loss function is widely used because it penalizes larger errors more than smaller ones due to the squaring operation, which can be beneficial in many regression scenarios. Minimizing the ℓ_2 loss during training helps the model learn to produce predictions that are closer to the actual values. The ℓ_2 loss also known as Mean Squared Error (MSE), that is particularly effective scenarios where it is essential to heavilt penalize larger errors, ensuring the model prioritizes accuracy for all prediction [22].

- Smooth ℓ_1 Loss

Smooth ℓ_1 loss also known as Huber Loss, is a loss function commonly used in regression tasks, particularly in the context of BBR in object detection models. It combines the advantages of both ℓ_1 and ℓ_2 loss, making it robust to outliers while providing a smooth transition for small error values. The function is defined piecewise: it behaves like ℓ_2 loss (quadratic) when the absolute error is small, and like ℓ_1 loss (linear) when the absolute error is large. This characteristic helps in stabilizing the training process by providing a smooth gradient for small errors, which aids in faster and more stable convergence, while also being less sensitive to outliers due to the linear behavior for large errors. Smooth ℓ_1 loss is widely used in object detection algorithms, such as Faster R-CNN, for refining bounding box predictions, thus improving the accuracy and reliability of object localization. Mathematically, Smooth ℓ_1 loss defined as:

$$smooth\ \ell_1\ loss = \begin{cases} 0.5(x_i - \hat{x}_i)^2 & if\ |x_i - \hat{x}_i| < 1 \\ |x_i - \hat{x}_i| & otherwise \end{cases} \tag{3}$$

This piecewise definition ensures a balance between penalizing large errors and maintaining stability for small errors, making it an effective loss function for various regression tasks in machine learning [23].

- IoU loss

IoU [7], [24] encodes the shape properties of objects being compared, such as the width, height, and location of two bounding boxes into area properties, then calculates a normalized measure to focus on their area or volume. IoU is commonly used to evaluate the accuracy of object detection and segmentation algorithms, with higher IoU scores indicating better localization performance. While IoU evaluates detection accuracy, it lacks gradient information when no overlap exists. To address this, IoU loss $\mathcal{L}_{IoU}$ optimizes bounding box alignment by directly maximizing the IoU score. The IoU and $\mathcal{L}_{IoU}$ formulas are defined as:

$$IoU = \frac{|B^p \cap B^{gt}|}{|B^p \cup B^{gt}|} \tag{4}$$

$$\mathcal{L}_{IoU} = 1 - IoU \tag{5}$$

The threshold for determining bounding box overlap varies depending on the specific task and dataset used. In the context of BBR, we define two variables $B^p, B^{gt} \epsilon \mathbb{R}^n$, where $B^{gt} = (x^{gt}, y^{gt}, w^{gt}, h^{gt})$ represents the ground truth bounding box, and and $B^p = (x^p, y^p, w^p, h^p)$ represents the predicted bounding box. Here, (x, y) corresponds to the top-left coordinate of the bounding box, while (w, h) define its width and height. The degree of alignment between B^p and B^{gt} is typically measured using the IoU, which is computed as the ratio of the intersection area between the two bounding boxes to their union. This calculation provides a quantitatives measure of overlap, ensuring that the predicted bounding box accurately aligns with the ground truth while minimizing localization errors.

- GIoU loss

Generalized Intersection over Union (GIoU) loss [6] is an extension of the traditional IoU metric that addresses its key limitation: vanishing gradients when bounding boxes do not overlap. Unlike standart IoU, which only measures the overlap ratio between the predicted and ground truth bounding boxes, GIoU introduces an additional term that accounts for the spatial relationship between two boxes. GIoU computers the smallest convex shape $C \subseteq (B^p \cup B^{gt}) \epsilon\ \mathbb{R}^n$ that fully encloses both the predicted B^p and

ground truth B^{gt} bounding boxes. It then penalizes cases where the convex hull area is significantly larger than the union of the two boxes, encouraging the model to adjust the bounding box even when there is no direct overlap. The GIoU loss is expressed in (6), which extends the tradisional IoU loss by incorporating and additional regularization term. This penalty term, formulated in (7), quantifies the discrepancy between the convex hull area and the union of the predicted and ground truth boxes, ensuring that even when IoU is zero, the loss function still provides meaningful gradients to guide the predicted bounding box toward the ground truth.

$$\mathcal{L}_{GIoU} = \mathcal{L}_{IoU} + \mathcal{R}_{GIoU} \tag{6}$$

$$\mathcal{R}_{GIoU} = \frac{|C - (B^p \cup B^{gt})|}{|C|} \tag{7}$$

- DIoU loss

Distance-IoU (DIoU) loss [9] improves traditional IoU-based loss functions by incorporating a distance term that accounts for the Euclidean distance between the center points of the predicted bounding box and the ground truth bounding box. This additional term helps guide the predicted box toward the ground truth more effectively, especially in cases where the overlap is minimal or nonexistent. As shown in (8), the DIoU loss extends the standard IoU loss by introducing a distance penalty term. This penalty, defined in (9), minimizes the Euclidean distance between the center points of the two bounding boxes, encouraging the predicted box to move closer to the ground truth. In this context, b^p and b^{gt} represent the center points of the predicted and ground truth bounding boxes, respectively, while $\rho(\cdot)$ denotes the Euclidean distance function. The term l^c refers to the diagonal length of the smallest enclosing box that contains both B^p and B^{gt}.

$$\mathcal{L}_{DIoU} = \mathcal{L}_{IoU} + \mathcal{R}_{DIoU} \tag{8}$$

$$\mathcal{R}_{DIoU} = \frac{\rho^2(b^p, b^{gt})}{(l^c)^2} \tag{9}$$

- CIoU loss

Complete Intersection over Union (CIoU) loss [9] extends both IoU and DIoU by considering not only the overlapping area and the distance between bounding box centers, but also the aspect ratio consistency between the predicted and ground truth bounding boxes. While DIoU improves localization by minimizing the center distance, CIoU further refines this by ensuring the predicted bounding box shape closely matches the ground truth. As formulated in (10) and further detailed in (11) and (12), CIoU introduces an additional penalty term that accounts for aspect ratio differences. In this formulation, (w^{gt}, h^{gt}) represent the width and height of the ground truth bounding box, while (w^p, h^p) represent the width and height of the predicted bounding box. The aspect ratio penalty term αv in (11) helps correct bounding boxes that exhibit large variations in aspect ratios α, ensuring that the width-to-height ratio of the predicted bounding box aligns more closely with that of the ground truth.

$$\mathcal{L}_{CIoU} = \mathcal{L}_{IoU} + \mathcal{R}_{CIoU} \tag{10}$$

$$\mathcal{R}_{CIoU} = \frac{\rho^2(b^p, b^{gt})}{(l^c)^2} + \alpha v \tag{11}$$

$$v = \frac{4}{\pi^2}\left(arctan\left(\frac{w^{gt}}{h^{gt}}\right) - arctan\left(\frac{w^p}{h^p}\right)\right) \tag{12}$$

$$\alpha = \frac{v}{(1 - IoU) + v} \tag{13}$$

- CFIoU loss

Corner-Point and Foreground-Area IoU (CFIoU) loss [25] in (14) is a specialized loss function designed to enhance object detection performance, particularly for small objects. It improves BBR by introducing new penalty as shown in (15) and (16) as the corner-point distance penalty and the foreground-area penalty. The corner-point distance penalty measures the distances between the corner points of the predicted and ground truth bounding boxes. By incorporating the distances of these corner points, CFIoU provides more precise localization, ensuring that even when the boxes' center points are close, the predicted box still aligns more accurately with the object boundaries. The foreground-area penalty, on the other hand, incorporates information about the target object's foreground area. This addition is particularly important for detecting small objects, as it prevents the predicted bounding box from deviating significantly from the actual object's shape and size. By combining these two penalties, CFIoU ensures better localization, making it especially useful in applications where the accurate detection of small or irregularly shaped objects is crucial.

$$\mathcal{L}_{ICIoU} = \mathcal{L}_{IoU} + \mathcal{R}_{CFIoU} \tag{14}$$

$$\mathcal{R}_{CFIoU} = \frac{\sum_{i=1}^{4} \rho^2\left(b_i^p, b_i^{gt}\right)}{4 \cdot (l^c)^2} + \left[(1-\mu)\frac{|B^p - B^{gt}|^2}{|C|^2} + \mu\frac{|C - B^{gt}|^2}{|C|^2}\right] \tag{15}$$

$$\mu = \begin{cases} 0 & if \quad \rho\left(b_i^p, b_i^{gt}\right) \neq 0 \\ 1 & if \quad \rho\left(b_i^p, b_i^{gt}\right) = 0 \end{cases} \tag{16}$$

- Fused IoU loss

Fused-IoU (FIoU) loss [26] addresses the limitations of traditional IoU-based loss functions by incorporating the benefits of both IoU and ℓ_2 norm. As illustrated in formula (17), FIoU loss measure with a penalty term based on the ℓ_2

distance normalized in (18) by the square of the diagonal of the smallest enclosing box. This combination ensures that FIoU retains the scale-invariant properties of IoU and introduces additional robustness to variations in bounding box size and position. FIoU maintains all the desirable properties of a metric, such as non-negativity, identity of indiscernibles, symmetry, and triangle inequality. It ensures that the loss value is more reflective of the actual spatial relationship between the predicted and ground truth bounding boxes. This makes it particularly effective in scenarios where the boxes do not overlap, where traditional IoU would fail to provide meaningful gradients for optimization.

$$\mathcal{L}_{FIoU} = \mathcal{L}_{IoU} + \mathcal{R}_{FIoU}$$
$$\mathcal{R}_{FIoU} = \frac{l_2}{(l^c)^2} \tag{17}$$

$$l_2 = \left(x_1^p - x_1^{gt}\right)^2 + \left(y_1^p - y_2^{gt}\right)^2 + \left(x_4^p - x_4^{gt}\right)^2 + \left(y_4^p - y_4^{gt}\right)^2 \tag{18}$$

Several studies have explored the effectiveness of IoU-based loss functions across different detection architectures. Alshubbak & Görges [27] investigated the impact of various IoU-based losses in anchor-free object detection models, such as FSAF [28] and FCOS [8], demonstrating that certain loss functions improve mAP scores while maintaining computational efficiency. Similarly, N-IoU [29] loss has been proposed as an enhancement over traditional IoU-based losses, refining BBR through improved optimization properties. While these studies focus on anchor-free models, our work evaluates IoU, GIoU, DIoU, CIoU, CFIoU, FIoU and the proposed DAPIoU loss within anchor-based object detection architectures (YOLOv7, YOLOv9, and Faster R-CNN). Future research could explore how DAPIoU performs in anchor-free models, aligning with recent advancements in loss function optimization.

III. DIMENSIONAL ANGLE PRECISION IOU LOSS

A. DAPIOU LOSS AS BOUNDING BOX REGRESSION

DAPIoU loss, or $\mathcal{L}_{DAPIoU}$, is designed by measuring the positions of bounding boxes in both ground truth and predictions with different levels of precision. While traditional bounding boxes in images are in 2D, the calculation of DAPIoU loss simulates a 3D condition. This approach introduces an additional dimension z in the traditional IoU calculation to impose a penalty based on the angle θ formed between the two bounding boxes. This method accounts for positional differences in 3D space, even if the original data is in 2D. The primary goal of this approach is to impose a larger penalty when the difference between the predicted bounding box and the ground truth is more significant in the additional dimension. This can enhance the precision of models in detecting objects in images, especially in scenarios where the position and orientation differences of objects are critical. The visual implementation is depicted in Fig. 2, where $\mathcal{L}_{DAPIoU}$ yields a result of 0 if the generated θ is 90°.

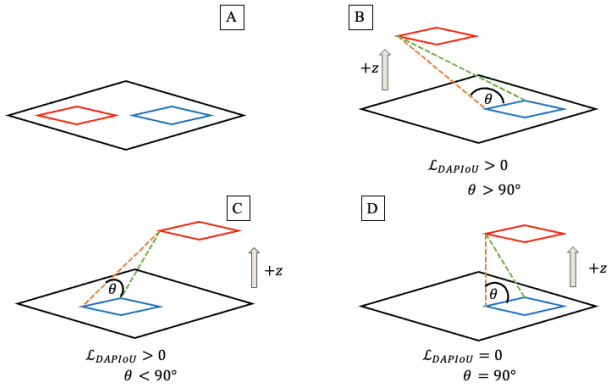

FIGURE 2. (A) Both bounding boxes are in 2D space (B) 3D coordinate position where the position of one of the bounding boxes is augmented with the z-coordinate, resulting $\mathcal{L}_{DAPIoU} > 0$ in $\theta > 90°$ (C) The same condition but different position, resulting $\mathcal{L}_{DAPIoU} > 0$ in $\theta < 90°$ (D) The same condition and precision of both bounding boxes, resulting $\mathcal{L}_{DAPIoU} = 0$ in $\theta = 90°$.

DAPIoU not only considers the differences in overlap area and distance but also considers the differences in angles and positions in 3D space, thereby providing a more accurate penalty aligned with the actual positional differences of the objects. This approach is expected to significantly improve the performance of object detection models, especially in complex and varied cases. DAPIoU loss builds on existing IoU-based loss function by introducing two key components: angular penalty and corner distance penalty. These additions address specific challenges in BBR, such as misalignment of bounding boxes and poor gradient signals for non-ocerlapping cases. The calculation of $\mathcal{L}_{DAPIoU}$ in (19) is divided into three parts: $\mathcal{L}_{IoU}$, which is the loss function based on IoU; $\mathcal{R}_{CD}(B^p, B^{gt})$, which is an additional penalty for corner distance, measuring the distance of each corner between the two bounding boxes; and $\mathcal{R}_{3DAP}(B^p, B^{gt})$, which is an additional penalty measured based on the precision angle of the two bounding boxes in 3D space.

$$\mathcal{L}_{DAPIoU} = \mathcal{L}_{IoU} + \mathcal{R}_{CD}(B^p, B^{gt}) + \mathcal{R}_{3DAP}(B^p, B^{gt}) \tag{19}$$

$\mathcal{R}_{CD}(B^p, B^{gt})$ is a penalty calculated as the Euclidean Distance $\rho(\cdot)$ for each corner between the two bounding boxes, divided into four quadrants: $\left(b_i^p, b_i^{gt}\right)$ for $i = 1,2, \ldots 4$, calculated successively from the top-left, top-right, bottom-left, and bottom-right corners for both the predicted bounding box and the ground truth, as visualized in Fig. 3. The difference between (15) and $\mathcal{R}_{CD}(B^p, B^{gt})$ that corner distance penalty includes the addition of a hyperparameter k that controls the number of l^c, which represents the diagonal length of the box surrounding both bounding boxes, with $1 \leq k \leq 4$. This is formulated as:

$$\mathcal{R}_{CD}(B^p, B^{gt}) = \frac{\sum_{i=1}^{4} \rho^2\left(b_i^p, b_i^{gt}\right)}{k \cdot (l^c)^2} \tag{20}$$

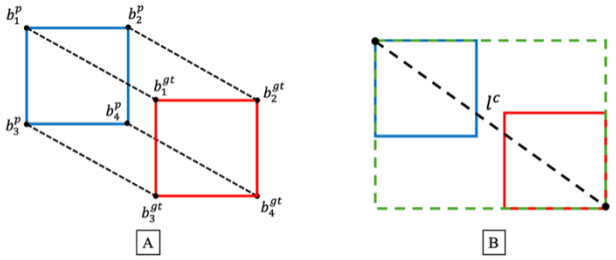

FIGURE 3. Part (A) shows the visualization of the corner positions of both bounding boxes and the distances measured at each corner. Part (B) shows the visualization of the diagonal distance of the box (green color) that encompasses both bounding boxes.

Furthermore, $\mathcal{R}_{3DAP}(B^p, B^{gt})$ represents the 3 Dimensional Angle Precision (3DAP) penalty, which slightly follows the CIoU concept from [9] in (12) for the variable v and (13) for the variable α. The difference is that the variable v is calculated as the angle θ_i of the triangle determined with $i = 1,2,\ldots 4$. (21) shows the formula for $\mathcal{R}_{3DAP}(B^p, B^{gt})$, (22) shows α, which is the weighting factor of the aspect ratio v, and (23) measures the level of inaccuracy or imprecision of the aspect ratio between the sides of the predicted bounding box and the corner parts of the ground truth, forming a triangle that is compared to $\pi/2$ as a measure of precision.

$$\mathcal{R}_{3DAP}(B^p, B^{gt}) = \alpha \cdot v \tag{21}$$

$$\alpha = \frac{v}{(1 - IoU) + v} \tag{22}$$

$$v = \frac{4}{\pi^2}\left(\sum_{i=1}^{4}\left(\theta_i - \frac{\pi}{2}\right)^2\right) \tag{23}$$

$$\theta_i = \cos^{-1}(I_i)$$

$$I_{1,2} = \frac{(w^p)^2 + \left(b_{1,2}\right)^2 - \left(c_{1,2}\right)^2}{2 \cdot w^p \cdot b_{1,2}} \tag{24}$$

$$I_{3,4} = \frac{(h^p)^2 + \left(b_{3,4}\right)^2 - \left(c_{3,4}\right)^2}{2 \cdot h^p \cdot b_{3,4}}$$

Whereas (24) for the variable I_i is calculated from several defined sides, namely w^p, h^p, b_i, and c_i. The side w and h are calculated as the width and length of the predicted bounding box, as shown in (25) and (26). This is necessary as the calculation of the base of the triangle.

$$w^p = x_2^p - x_1^p \tag{25}$$

$$h^p = y_2^p - y_1^p \tag{26}$$

Next, (27) show b_i represents the height of each defined triangle and c_i represents the slope side of each defined triangle. Both are added with the variable z where $z \geq 1$, which is a hyperparameter to control the z-coordinate in 3D space as the distance between the predicted bounding box and the ground truth. The visualization of variables a_j, b_i, and c_i along with θ_i is shown in Fig. 4.

$$\begin{aligned}
b_1 &= \sqrt{\left(x_1^p - x_1^{gt}\right)^2 + \left(y_1^p - y_1^{gt}\right)^2 + z^2} \\
b_2 &= \sqrt{\left(x_2^p - x_2^{gt}\right)^2 + \left(y_1^p - y_1^{gt}\right)^2 + z^2} \\
b_3 &= \sqrt{\left(x_1^p - x_1^{gt}\right)^2 + \left(y_2^p - y_2^{gt}\right)^2 + z^2} \\
b_4 &= \sqrt{\left(x_2^p - x_2^{gt}\right)^2 + \left(y_2^p - y_2^{gt}\right)^2 + z^2} \\
c_1 &= \sqrt{\left(x_2^p - x_1^{gt}\right)^2 + \left(y_1^p - y_1^{gt}\right)^2 + z^2} \\
c_2 &= \sqrt{\left(x_1^p - x_2^{gt}\right)^2 + \left(y_1^p - y_1^{gt}\right)^2 + z^2} \\
c_3 &= \sqrt{\left(x_1^p - x_1^{gt}\right)^2 + \left(y_1^p - y_2^{gt}\right)^2 + z^2} \\
c_4 &= \sqrt{\left(x_2^p - x_2^{gt}\right)^2 + \left(y_1^p - y_2^{gt}\right)^2 + z^2}
\end{aligned} \tag{27}$$

The calculation of DAPIoU loss, which utilizes angles in 3D space, is expected to provide significant improvements in the performance of object detection models. This approach considers the positional and angular differences between the two bounding boxes more accurately. By integrating the z-coordinate, this calculation not only penalizes the differences in overlap area but also in angles and positions within the 3D space. The computation of I_i results in values constrained within the range $-1 < I_i < 1$. This is due to the formulation of I_i in (24), which follows a modified cosine rule. Since I_i serves as the input to the inverse cosine function, θ_i must be within the valid range of the arccos function. Consequently, the values of θ_i are restricted to $0 < \theta_i < \pi$, ensuring that the computed angles remain within the feasible range for further calculations in (23). This contraint is crucial for maintaining numerical stability and ensuring the correctness of the angular penalty of $\mathcal{R}_{3DAP}$. The implementation of DAPIoU has the potential to enhance the precision and accuracy of object detection models, particularly in scenarios involving complex variations in object positions and orientations.

B. GRADIENT ANALYSIS OF DAPIOU LOSS

In the development of object detection models, the optimization of loss functions plays a crucial role in enhancing prediction accuracy and convergence speed. One

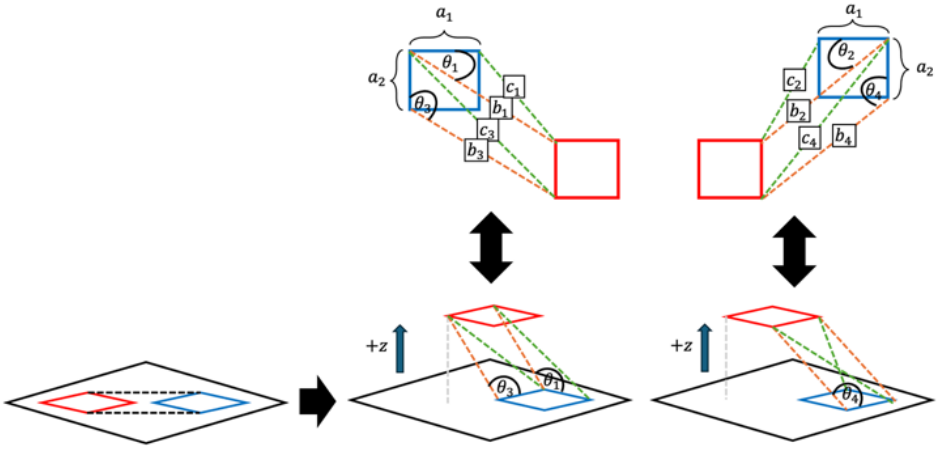

FIGURE 4. **The angles ($\theta_1, \theta_2, \ldots, \theta_4$) determined from the two bounding boxes between the ground truth (red) and the prediction (blue) are adjusted by adding a coordinate value z.**

of the key aspects of this optimization is the calculation of gradients, which determines the direction and magnitude of changes to model parameters to minimize the loss function. This subsection discusses the importance of gradients in the optimization process and how gradient calculation is performed in the context of various loss function, specifically in object detection models. Gradient provide guidance on how model parameters should be adjusted to reduce prediction errors. Without gradients, the optimization process would lack a clear direction to achieve an optimal solution. Gradients are vectors that contain the partial derivatives of the loss function with respect to each model parameter. They indicate the fastest direction in which the value of the loss function increases. In the context of optimization, this information is used to update model parameters in the direction opposite to the gradient, thereby minimizing the loss function. By incorporating additional dimensions and angles into the loss calculation, the gradient provides a more detailed direction for adjustments, especially in complex scenarios involving 3D spatial relationships. This enhanced gradient direction ensures that the bounding boxes are iteratively refined to better align with the ground truth, thereby improving the model's precision in detecting and localizing objects. Consequently, while addressing potential issues such as gradient explosion by modifying the step size, the critical optimization pathways are maintained, ensuring the effectiveness and stability of the training process. This approach results in higher accuracy and robustness in object detection tasks, particularly in challenging environments with significant variations in object position and orientation. The gradient of the DAPIoU loss with respect to the bounding box parameters can be expressed as:

$$\Delta\mathcal{L}_{DAPIoU} = \left(\frac{\partial\mathcal{L}_{DAPIoU}}{\partial(x^p, y^p, w^p, h^p)}\right) \tag{28}$$

The DAPIoU loss introduces two penalty terms, $\mathcal{R}_{CD}(B^p, B^{gt})$ and $\mathcal{R}_{3DAP}(B^p, B^{gt})$, whose gradient analysis is crucial for understanding their optimization influence on bounding box adjustments. The gradient analysis of $\mathcal{R}_{CD}(B^p, B^{gt})$ and $\mathcal{R}_{3DAP}(B^p, B^{gt})$ must be conducted to gain deeper insights into the effectiveness of DAPIoU loss. By computing $\partial\mathcal{R}_{CD} / \partial(x^p, y^p, w^p, h^p)$ and $\partial\mathcal{R}_{3DAP} / \partial(x^p, y^p, w^p, h^p)$, the influence of each bounding box parameter on localization accuracy can be evaluated, ensuring precise and stable bounding box predictions.

$$\frac{\partial\mathcal{R}_{CD}}{\partial x^p} = \frac{4\left(x_1^p - x_1^{gt}\right) + 4\left(x_2^p - x_2^{gt}\right)}{k \cdot (l^c)^2} \tag{29}$$

$$\frac{\partial\mathcal{R}_{CD}}{\partial y^p} = \frac{4\left(y_1^p - y_1^{gt}\right) + 4\left(y_2^p - y_2^{gt}\right)}{k \cdot (l^c)^2} \tag{30}$$

$$\frac{\partial\mathcal{R}_{CD}}{\partial w^p} = 0 \tag{31}$$

$$\frac{\partial\mathcal{R}_{CD}}{\partial h^p} = 0 \tag{32}$$

The partial derivative of corner distance penalty with respect to the predicted bounding box parameters provide insight into how DAPIoU adjusts localization errors during training. The gradients with respect to the horizontal (x^p) and vertical (y^p) coordinate are defined as (29) and (30). These expression indicate that when the predicted bounding box deviates significantly from the ground truth along the horizontal or vertical axis, the model produces larger gradients to guide the bounding box toward the correct position. The correction sensitivity is scaled by the factor k, ensuring that adjustments are proportional to the relative positioning of the bounding boxes. As the predicted box moves closer to the ground truth, the gradient magnitude decreases, signaling that only minor refinements are required to optimize localization accuracy. Meanwhile, (31) and (32) show that the partial

derivatives with respect to the predicted width (w^p) and height (h^p) are both zero. This means that the corner distance penalty exclusively influences positional adjustments (x^p, y^p) but does not affect the width and height (w^p, h^p) of the bounding box. The gradient formulation of $\mathcal{R}_{CD}$ ensures that the primary focus remains on reducing localization errors while relying on other loss components to regulate bounding box size.

Furthermore, for $R_{3DAP}(B^p, B^{gt})$, the chain rule must be applied due to the partial differentiation across multiple interrelated equations, specifically (23), (24), and (27). This gradient computation is closely related to the approach used in CIoU as described in [9], which focuses on differentiating with respect to v. To facilitate clearer elaboration, the variable U_i where $i = 1,2, \dots 4$ is defined in (34), allowing the partial derivatives of the bounding box parameters with respect to v to be explicitly formulated in (35) and (36).

$$v = \frac{4}{\pi^2}\left(\sum_{i=1}^{4}(U_i)^2\right) \tag{33}$$

$$U_i = \theta_i - \frac{\pi}{2}$$
$$U_i = \cos^{-1}(I_i) - \frac{\pi}{2} \tag{34}$$

$$\frac{\partial v}{\partial(x^p, y^p, w^p, h^p)} = \frac{4}{\pi^2}\sum_{i=1}^{4}\left(2\cdot(U_i)\cdot\frac{\partial U_i}{(x^p, y^p, w^p, h^p)}\right) \tag{35}$$

$$\frac{\partial U_i}{\partial(x^p, y^p, w^p, h^p)} = -\frac{1}{\sqrt{1-\left(I_i - \frac{\pi}{2}\right)^2}} \cdot \frac{\partial I_i}{\partial(x^p, y^p, w^p, h^p)} \tag{36}$$

The gradient of I_i with respect to x^p, as shown in (37), indicates that when the predicted bounding box and the ground truth are close to each other, I_i and $\left(x_1^p - x_1^{gt}\right)$ approach zero. This results in a gradient of $-1 / b_1$ and $1 / b_2$, demonstrating that the gradient remains stable between -1 and 1 and does not experience gradient explosion. The same principle applies to (38), (41), and (42). Similarly, in (39) and (40), when the predicted bounding box and the ground truth are close, the gradient value approaches zero. The gradient calculations from (37) to (42) are performed relative to their respective positions but consistently maintain values within the range of -1 to 1, ensuring that the gradient of U_i remains stable. For (43), when w^p is small, the denominator $2(w^p)^2 \cdot b_{1,2}$ shrinks, leading to a potential gradient explosion and unstable updates. However, the presence of $b_{1,2}$, one side of the triangle, prevents this issue by appropriately scaling the gradient. Since $b_{1,2}$ does not decrease as drastically as w^p, it ensures that the calculation remains focused on positional differences rather than box size variations. Similarly, when w^p is sufficiently large, the numerator and denominator maintain a balanced ratio, keeping the gradient values within the safe range of -1 to 1 to prevent gradient explosion. The same principle applies to (44) with h^p and $b_{3,4}$.

$$\frac{\partial I_1}{\partial x^p} = -\frac{1 + I_1\left(x_1^p - x_1^{gt}\right)}{b_1} \tag{37}$$

$$\frac{\partial I_2}{\partial x^p} = \frac{1 - I_2\left(x_2^p - x_2^{gt}\right)}{b_2} \tag{38}$$

$$\frac{\partial I_{3,4}}{\partial x^p} = -\frac{I_{3,4}\left(x_{1,2}^p - x_{1,2}^{gt}\right)}{b_{3,4}} \tag{39}$$

$$\frac{\partial I_{1,2}}{\partial y^p} = -\frac{I_{1,2}\left(y_{1,2}^p - y_{1,2}^{gt}\right)}{b_{1,2}} \tag{40}$$

$$\frac{\partial I_3}{\partial y^p} = \frac{1 - I_3\left(y_1^p - y_1^{gt}\right)}{b_3} \tag{41}$$

$$\frac{\partial I_4}{\partial y^p} = -\frac{1 + I_4\left(y_2^p - y_2^{gt}\right)}{b_{3,4}} \tag{42}$$

$$\frac{\partial I_{1,2,3,4}}{\partial w^p} = \frac{(w^p)^2 - \left(b_{1,2}\right)^2 + \left(c_{1,2}\right)^2}{2(w^p)^2 \cdot b_{1,2}} \tag{43}$$

$$\frac{\partial I_{1,2,3,4}}{\partial h^p} = \frac{(h^p)^2 - \left(b_{3,4}\right)^2 + \left(c_{3,4}\right)^2}{2(h^p)^2 \cdot b_{3,4}} \tag{44}$$

The gradient formulation across these equations ensures numerical stability and prevents extreme gradient updates, which could otherwise lead to instability in the optimization process. By maintaining the gradient values within a controlled range, the model can achieve more stable convergence, allowing for precise bounding box localization without being overly sensitive to variations in box size or positional offsets. This structured gradient behavior is crucial in ensuring the robustness and efficiency of the DAPIoU loss in object detection tasks. For detailed algorithm of DAPIoU loss, please refer to Appendix A.

IV. EXPERIMENTAL RESULTS

A. SINGLE-GROUP BBR SIMULATION EXPERIMENT

In this study, we employ the single-group BBR simulation Experiment to evaluate the effectiveness of BBR in a controlled environment. This approach, originally introduced in prior research, focuses on optimizing a single predicted bounding box toward a fixed ground truth target using a continuous loss function. The predicted bounding box is initialized at a predefined position with specific width and height, while the target bounding box remains constant. Through an iterative regression process governed by a learning rate , the bounding box updates its position and dimensions based on gradient adjustments. This simulation allows us to analyze how different loss functions influence

convergence behavior, stability, and optimization dynamics before applying them to more complex real-world datasets. By isolating individual bounding box transformations, we ensure that the impact of hyperparameter tuning and gradient adjustments can be studied without interference from external dataset variability. This experiment is specifically designed for DAPIoU loss, as the selection of the hyperparameters k (ranging from 1 to 4) and z (where $z \geq 1$) needs to be carefully determined. Ensuring the appropriate tuning of these hyperparameters is crucial for maximizing the accuracy of DAPIoU loss when applied to real-world datasets.

In this experiment, a single predicted bounding box is iteratively adjusted to align with a fixed ground truth box. The initial predicted bounding box is set at (3,3,1,0.5), while the target ground truth box is positioned at (1,1,0.5,0.5). To ensure stability during training and prevent gradient explosion, a relatively high learning rate of 0.25 is used and runs for 1500 iterations. This choice allows for observing the convergence behavior of DAPIoU loss efficiently while ensuring the learning dynamics remain stable across different hyperparameter values. Since the hyperparameter z is continuous and constrained to $z \geq 1$, multiple values are selected for evaluation, specifically z are 1, 10, 100, 1000, 10,000, and 100,000. This approach aims to investigate whether increasing or decreasing the value of z results in a lower final loss value. Naturally, this evaluation is conducted over a sufficiently large number of iterations to observe how DAPIoU loss converges. Meanwhile, the hyperparameter k is tested with values of $k = 1,2,3,4$. Fig. 5 provides a representative visualization of all tested hyperparameter values. Given the difficulty in distinguishing differences between the plotted results for each selected hyperparameter, the visualization is simplified to show only iteration 1 and iteration 1500. This summary highlights that, while all hyperparameter settings produce visually similar results in terms of bounding box adjustments, they do not necessarily yield the same final loss values at each iteration.

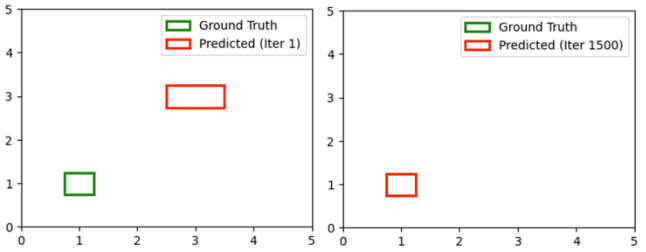

FIGURE 5. Visualization of the single-group BBR simulation for DAPIoU loss at the first iteration and the 1500th iteration. Since the plots for different hyperpatameter value of *k* and *z* exhibit similar patterns, only one representative plot is shown.

Tab. 1 presents the final DAPIoU loss values at the 1500th iteration across different hyperparameter settings for k and z in the single-group BBR simulation. The values of z range from 1 to 100,000, while k varies from 1 to 4, allowing an analysis of how these parameters influence the loss function. The results indicate that smaller values of z, such as 1 and 10, tend to produce lower loss values in most cases, whereas extremely large values, such as 100,000, do not necessarily yield further improvements. The lowest DAPIoU loss, highlighted in bold, is observed at $z = 10{,}000$ and $k = 2$, suggesting that this setting is more effective in minimizing loss.

TABLE 1
FINAL DAPIOU LOSS VALUES FOR DIFFERENT K AND Z SETTINGS AT THE 1500TH ITERATION IN THE SINGLE-GROUP BBR SIMULATION

k	$z = 1$	$z = 10$	$z = 100$	$z = 1000$	$z = 10{,}000$	$z = 100{,}000$
1	0.0074	0.0055	0.0108	0.0151	0.0039	0.0036
2	0.0057	0.0047	0.0285	0.0026	**0.0014**	0.0055
3	0.0073	0.0138	0.0094	0.0076	0.0175	0.0166
4	0.0247	0.0070	0.0122	0.0171	0.0049	0.0350

TABLE 2
FINAL DAPIOU LOSS VALUES AT THE 1500TH ITERATION FOR K AND Z AROUND 10,000 IN THE SINGLE-GROUP BBR SIMULATION

k	$z = 9000$	$z = 9500$	$z = 10{,}000$	$z = 10{,}500$	$z = 11{,}000$	$z = 11{,}500$
2	0.0067	0.0164	0.0014	0.0005	0.0010	0.0419

Tab. 2 further refines the exploration of z around 10,000, with values ranging from 9000 to 11,500, while keeping $k = 2$, as it demonstrated promising results in Tab. 1. The findings confirm that tuning z within this range significantly impacts the final loss value, with $z = 10{,}500$ achieving the lowest loss. These results emphasize that selecting an optimal z value is crucial for improving DAPIoU loss performance, as overly large or small values may lead to suboptimal loss minimization. Additionally, the results indicate that increasing or decreasing z arbitrarily does not consistently lead to lower loss values, and the same applies to k. Therefore, careful selection of these hyperparameters is necessary to achieve optimal performance on real-world datasets. However, when applying DAPIoU loss in detection

models and datasets, further experiments with different z values are still required to determine the best setting for maximizing mAP. The determination of z and k can actually be explored further, as there is a possibility of finding even more optimal values for future research. For detailed algorithm of single-group BBR simulation experiment, please refer to Appendix B.

B. SIMULATION EXPERIMENT

In addition to real-world datasets, we employ synthetic data in our experiments to better analyze the behavior of DAPIoU loss under controlled conditions. Unlike real-world datasets, which contain inheret biases such as occlusion, background clutter, and class imbalance, synthetic data allows us to isolate specific bounding box properties and precisely evaluate how the loss function reponds to various localization challenges. Using synthetic data enables us to systematically vary bounding box characteristics, including aspect ratios, overlap levels, and positioning, providing deeper insights into effectiveness of DAPIoU. In particular, this approach helps analyze how the loss function mitigates gradient inconsistencies for non-overlapping and misaligned bounding boxes, which are difficult to control in real-world datasets. Furthermore, synthetic data ensures a consistent and unbiased evaluation environment, eliminating dataset-specific biases that may affect real-world dataset benchmarks. Additionally, it offers a computationally efficient approach way to test and refine the loss function before apply it to large datasets, thereby reducing the computational cost and training time. By first validating DAPIoU on synthetic data, we ensure that improvements are attributable to the loss function itself rather than external dataset factors, allowing afor a far comparison across different IoU-based loss function. To conduct synthetic experiments, we generated 10,000 random points within a circular area with a radius of 3, where each point represents a bounding box center. Each point is associated with 49 anchor boxes of size 7×7, with 7 different aspect ratios ranging from 1:4 (vertical) to 4:1 (horizontal). Additionally, the dataset includes 7 different size scales ranging from 50 to 200 pixels. Some of the anchor box centers are highlighted in orange, while the red boxes indicate different anchor box scales centered around a blue point, as shown in Fig. 6. Moreover, the dataset includes seven fixed target boxes positioned at coordinates (10,10), each with a constant size but varying aspect ratios, ensuring a structured experimental setup for performance evaluation. This synthetic dataset allows us to analyze DAPIoU's gradient behavior, convergence stability, and robustness across varying bounding box conditions, providing crucial insights into its ability to enhance object detection accuracy.

Based on Fig. 7 in the total ℓ_1 error graph, DAPIoU demonstrates a similar reduction in Total ℓ_1 error compared to IoU loss, GIoU loss, DIoU loss, and CIoU loss. Specifically, the DAPIoU loss shows a significantly faster convergence rate, achieving a lower ℓ_1 error in fewer epochs compared to the other loss functions. The IoU loss, represented by the blue line, shows the slowest convergence and highest ℓ_1 error. GIoU, DIoU, and CIoU losses, represented by orange, red, and green lines respectively, show improved performance over IoU but still lag behind DAPIoU in terms of convergence speed and final error values. The DAPIoU loss, represented by the purple line, reaches an ℓ_1 error close to zero significantly faster, indicating superior efficiency and accuracy in BBR tasks.

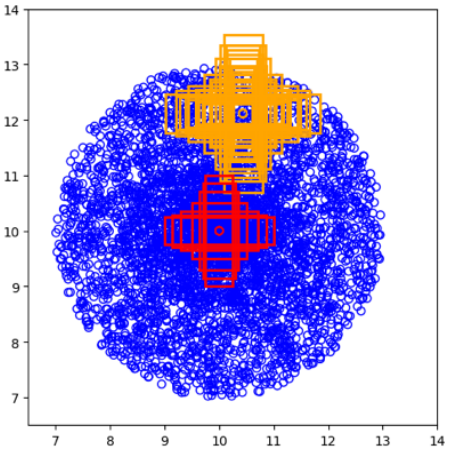

FIGURE 6. Visualisasion of simulation experiment used to evaluate the performance of IoU-based loss function. Orange boxes represent anchor boxes, while red boxes indicate target bounding boxes at different scales and aspect ratios. These variations allow a systematic analysis of the impact of IoU-based loss function on localization errors.

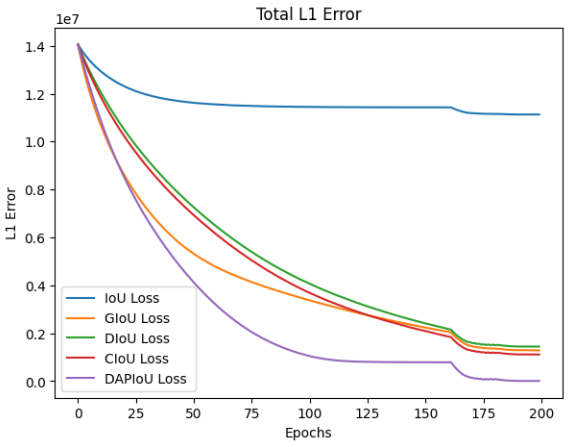

FIGURE 7. Total ℓ_1 error curves of BBR IoU-based loss function in simulation experiments.

This result highlights the effectiveness of DAPIoU in reducing the error more quickly and thoroughly than traditional IoU-based loss functions. The ability of DAPIoU to adaptively penalize misalignment and discrepancies in aspect ratios between predicted and ground truth boxes contributes to its superior performance, making it a promising choice for object detection applications requiring

precise localization. For detailed algorithm of simulation experiment, please refer to Appendix C.

C. EXPERIMENT ON THE REAL-WORLD DATASETS

In this section, we evaluate the proposed DAPIoU loss function by comparing its detection performance with some widely-used object detectors, such as YOLOv7 [18], YOLOv9, and Faster R-CNN [19], on the challenging object detection benchmarks PASCAL VOC [13] and MS COCO [14], were conducted using various loss functions such as IoU, GIoU, DIoU, CIoU, CFIoU, FIoU, and DAPIoU. These experiments were implemented on an NVIDIA DGX A100 system provided by the Tokopedia-UI AI Center of Excellence. For the PASCAL VOC 2007+2012 dataset, all models were trained for 300 epochs with a batch size of 16, ensuring thorough training and convergence. For the MS COCO 2017 dataset, all models were trained for 100 epochs with a batch size of 16, focusing on evaluating the convergence speed of the models under the proposed DAPIoU loss and other IoU-based loss functions. The consistent use of a batch size of 16 across both datasets was strategically chosen to prevent out-of-memory errors on the CUDA-enabled server, ensuring stable and efficient training processes. Several IoU-based loss functions, including IoU, GIoU, DIoU, CIoU, CFIoU, FIoU, and the proposed DAPIoU, were implemented and evaluated in each of the three models to assess their effectiveness and efficiency in enhancing object detection performance.

This evaluation aims to validate the effectiveness of the proposed DAPIoU loss and analyze the impact of each of its components. First, we conduct a hyperparameter analysis of the DAPIoU loss by varying the hyperparameters k and z coordinate. This analysis helps us understand the sensitivity and robustness of the DAPIoU loss to different settings of these hyperparameters. For the PASCAL VOC 2007+2012 datasets, we evaluate the models using the following metrics: $AP_{50:95}$, AP_{50}, AP_{75}, AP_{85}, and AP_{95}. These metrics provide a comprehensive assessment of the model's performance across different IoU thresholds. For the MS COCO 2017 dataset, we evaluate the models using the following metrics: $AP_{50:95}$, AP_{50}, AP_{75}, AP_S, AP_M, and AP_L. All the metrics represents mAP averaged over IoU thresholds. AP_S, AP_M, and AP_L correspond to the mAP for small, medium, and large objects, respectively. Specifically, AP_S is calculated for objects with an area less than 32^2, AP_M for objects with an area between 32^2 and 96^2 pixels, and AP_L for objects with an area greater than 96^2 pixels. Fig. 8 illustrate the impact of the hyperparameters k on $AP_{50:95}$ for the DAPIoU loss. Fig. 8 shows that $k = 2$ achieves the highest mAP of 50.6%, outperforming other values of k. This indicates that the model's performance is optimal when k is set to 2.

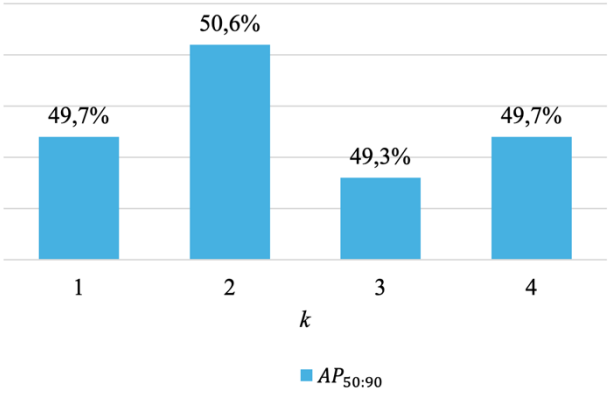

FIGURE 8. mAP comparison using DAPIoU loss (in %) with hyperparameter *k* determined on YOLOv7 for PASCAL VOC 2007+2012 dataset.

TABLE 3
PERFORMANCE COMPARISON OF DIFFERENT IOU-BASED LOSS FUNCTION (IN %) USING YOLOV7 ON PASCAL VOC 2007+2012 DATASET

Loss Function	$AP_{50:95}$	AP_{50}	AP_{75}	AP_{85}	AP_{95}
$\mathcal{L}_{IoU}$	49.4	73.0	58.8	35.2	4.77
$\mathcal{L}_{GIoU}$	50.3	73.8	60.1	36.5	5.34
$\mathcal{L}_{DIoU}$	49.2	72.9	58.7	35.1	4.65
$\mathcal{L}_{CIoU}$	49.5	73.3	59.0	35.0	4.93
$\mathcal{L}_{CFIoU}$	49.3	72.7	59.0	35.4	4.44
$\mathcal{L}_{FIoU}$	50.0	73.5	59.9	36.2	5.30
$\mathcal{L}_{DAPIoU}$ $k = 2, z = 9000$	50.0	73.4	59.5	36.0	5.27
$\mathcal{L}_{DAPIoU}$ $k = 2, z = 9500$	50.1	73.5	59.6	36.2	5.34
$\mathcal{L}_{DAPIoU}$ $k = 2, z = 10{,}000$	**50.6**	**73.8**	**60.3**	**36.5**	**5.62**
$\mathcal{L}_{DAPIoU}$ $k = 2, z = 10{,}500$	50.1	73.6	59.9	36.5	**5.64**
$\mathcal{L}_{DAPIoU}$ $k = 2, z = 11{,}000$	50.0	73.4	5.98	3.58	**5.62**

TABLE 4
PERFORMANCE COMPARISON OF DIFFERENT IOU-BASED LOSS FUNCTION (IN %) USING YOLOV7 ON MS COCO 2017 DATASET

Loss Function	$AP_{50:95}$	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
$\mathcal{L}_{CFIoU}$	48.8	66.8	53.0	32.0	53.0	63.1
$\mathcal{L}_{FIoU}$	48.7	67.0	53.0	32.1	53.0	63.5
$\mathcal{L}_{DAPIoU}$ $k=2, z=10{,}000$	**48.9**	**67.0**	**53.2**	**32.5**	**53.6**	**64.1**
$\mathcal{L}_{DAPIoU}$ $k=2, z=10{,}500$	**49.0**	**67.2**	**53.2**	**32.6**	**53.7**	**63.6**
$\mathcal{L}_{DAPIoU}$ $k=2, z=11{,}000$	**49.0**	**67.3**	**53.6**	**32.9**	**53.7**	**63.6**
$\mathcal{L}_{DAPIoU}$ $k=2, z=11{,}500$	**49.0**	**67.2**	**53.4**	**32.8**	**53.8**	**63.8**
$\mathcal{L}_{DAPIoU}$ $k=2, z=12{,}000$	48.3	**67.5**	52.0	**32.4**	52.7	63.5

TABLE 5
PERFORMANCE COMPARISON OF DIFFERENT IOU-BASED LOSS FUNCTION (IN %) USING YOLOV9 ON PASCAL VOC 2007+2012 DATASET

Loss Function	$AP_{50:95}$	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
$\mathcal{L}_{IoU}$	54.4	74.9	58.4	17.5	39.1	64.3
$\mathcal{L}_{GIoU}$	53.9	74.2	58.1	16.2	39.1	63.7
$\mathcal{L}_{DIoU}$	54.3	75.0	58.2	17.7	39.8	63.7
$\mathcal{L}_{CIoU}$	54.3	75.0	58.9	17.2	39.9	64.0
$\mathcal{L}_{CFIoU}$	54.6	75.0	58.7	16.6	39.4	64.5
$\mathcal{L}_{FIoU}$	54.7	75.4	59.0	17.7	40.0	64.4
$\mathcal{L}_{DAPIoU}$ $k=2, z=9500$	54.7	75.1	58.9	**18.1**	**40.0**	**64.3**
$\mathcal{L}_{DAPIoU}$ $k=2, z=10{,}000$	**54.8**	75.0	**59.3**	**18.1**	39.6	**64.7**
$\mathcal{L}_{DAPIoU}$ $k=2, z=10{,}500$	**54.8**	**75.5**	58.9	**17.9**	39.8	**64.5**
$\mathcal{L}_{DAPIoU}$ $k=2, z=11{,}000$	54.6	74.9	**59.0**	16.9	39.9	64.3

TABLE 6
PERFORMANCE COMPARISON OF DIFFERENT IOU-BASED LOSS FUNCTION (IN %) USING YOLOV9 ON MS COCO 2017 DATASET

Loss Function	$AP_{50:95}$	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
$\mathcal{L}_{CFIoU}$	49.9	66.5	54.5	31.6	55.7	65.7
$\mathcal{L}_{FIoU}$	50.0	66.8	54.3	32.9	55.7	66.0
$\mathcal{L}_{DAPIoU}$ $k=2, z=10{,}500$	**50.3**	**66.8**	**54.7**	**33.2**	**56.0**	**66.1**
$\mathcal{L}_{DAPIoU}$ $k=2, z=11{,}000$	**50.5**	**67.0**	**55.2**	32.1	**55.9**	**66.5**
$\mathcal{L}_{DAPIoU}$ $k=2, z=11{,}500$	**50.3**	**67.2**	**54.9**	32.4	**56.3**	65.9

TABLE 7
PERFORMANCE COMPARISON OF DIFFERENT IOU-BASED LOSS FUNCTION (IN %) USING FASTER-RCNN ON MS COCO 2017 DATASET

Loss Function	$AP_{50:95}$	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
$\mathcal{L}_{CFIoU}$	34.7	50.6	38.3	17.3	37.8	47.9
$\mathcal{L}_{FIoU}$	34.1	49.6	37.8	17.0	37.1	47.5
$\mathcal{L}_{DAPIoU}$ $k=2, z=10{,}000$	34.7	50.5	38.2	**17.6**	37.6	**48.4**
$\mathcal{L}_{DAPIoU}$ $k=2, z=10{,}500$	**34.8**	**51.0**	**38.4**	**17.4**	37.6	**48.4**
$\mathcal{L}_{DAPIoU}$ $k=2, z=11{,}000$	34.5	50.5	38.3	16.9	37.6	**48.2**

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2025.3567767

In the context of hyperparameter tuning, there are no strict limitations on defining the hyperparameter z because it exists on a continuous spectrum. Since z is not restricted to discrete values, it can be adjusted incrementally across a wide range to optimize the model's performance. The performance of the model can vary unpredictably with

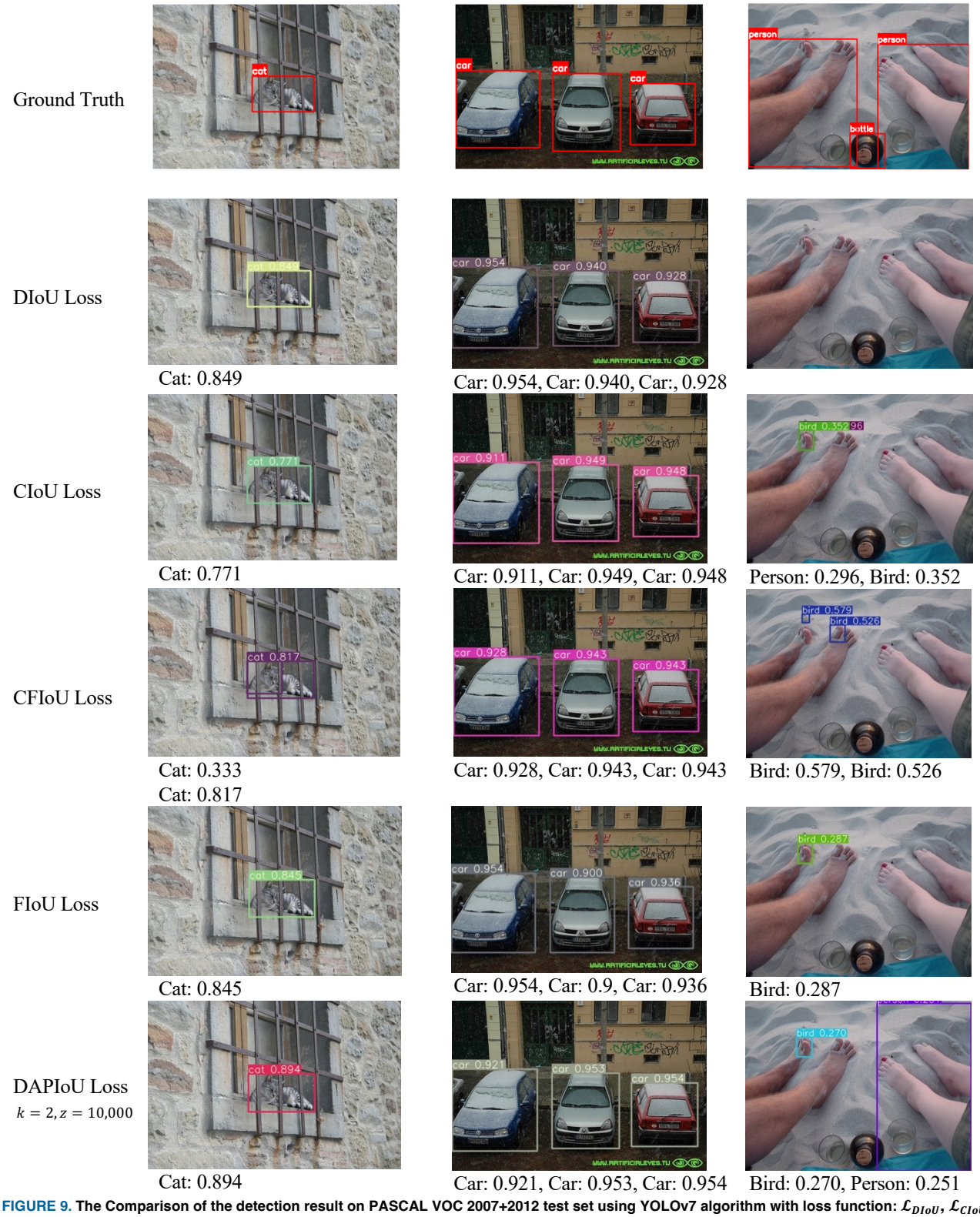

FIGURE 9. The Comparison of the detection result on PASCAL VOC 2007+2012 test set using YOLOv7 algorithm with loss function: $\mathcal{L}_{DIoU}$, $\mathcal{L}_{CIoU}$, $\mathcal{L}_{CFIoU}$, $\mathcal{L}_{FIoU}$, $\mathcal{L}_{DAPIoU}$ with hyperparameter $k = 2$, and $z = 10,000$, and images with ground truth annotation

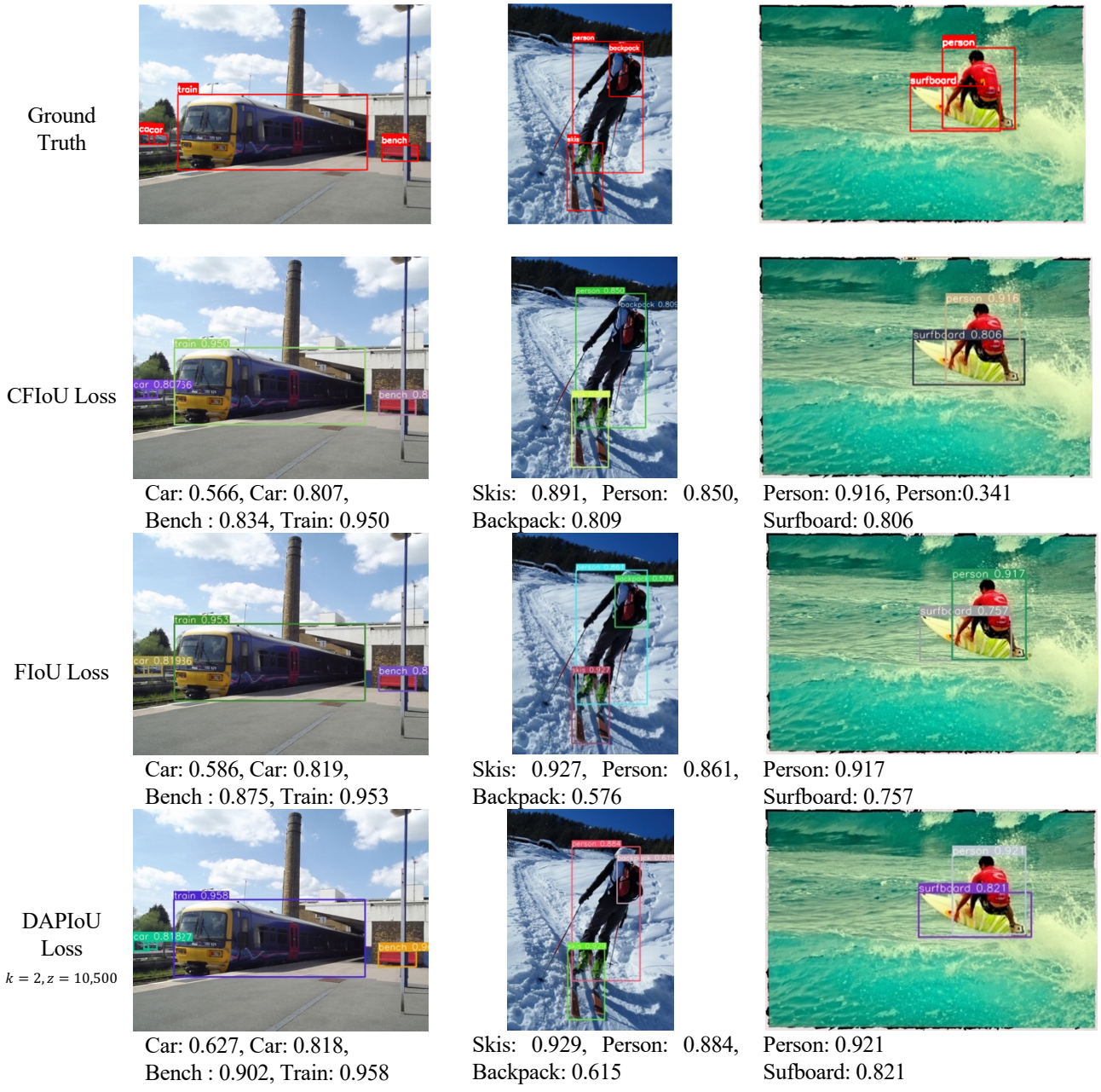

FIGURE 10. **The Comparison of the detection result on MS COCO 2017 validation set using YOLOv7 algorithm with loss function: $\mathcal{L}_{CFIoU}$, $\mathcal{L}_{FIoU}$, $\mathcal{L}_{DAPIoU}$ with hyperparameter $k = 2$, and $z = 10,500$, and images with ground truth annotations.**

changes in the z value, and both smaller and larger values can impact results in different ways depending on the specific data and model architecture. This unpredictability reinforces the importance of conducting thorough hyperparameter tuning across a range of values to identify the optimal setting, rather than assuming that extreme values will necessarily produce superior results. This flexibility allows for precise tuning, as any real number value within the practical limits of the model can be explored. By leveraging this continuity, hyperparameters such as 9500, 10,000, or 11,000 can be fine-tuned to maximize accuracy or efficiency for a particular task. However, there is no guarantee that the smallest or highest values of the z coordinate will yield the best mAP. Results from YOLOv7 trained on the PASCAL VOC dataset 2007+2012 dataset demonstrate that DAPIoU loss achieves superior performance across multiple evaluation metrics including mAP with also hyperparameter z variation. For the higher IoU thresholds, DAPIoU continues to perform

competitively. This demonstrates that DAPIoU not only excels in general object detection accuracy but also maintains robustness in stricter localization scenarios. Tab. 3 and Tab. 5 present a comparison of different IoU-based loss functions assessed using the YOLOv7 and YOLOv9 model on the PASCAL VOC 2007+2012 dataset. In YOLOv7, experiments were conducted by setting the hyperparameter z from 9000 to 11,000 with 500 between them and $k = 2$. DAPIoU loss particularly at $k = 2$, $z = 10{,}000$, outperforms others with mAP of 50.6% in $AP_{50:95}$ indicating its superiority in achieving precise object localization.

In YOLOv9, The results indicate that the DAPIoU loss with $k = 2$ and $z = 10{,}000$ and $z = 10{,}500$ achieve the best performance in several categories. Specifically, the configuration with $z = 10{,}000$ has the highest $AP_{50:95}$ at 54.8%, AP_{75} at 59.3% and performs strongly in detecting small objects with AP_S at 18.1%. It also provides strong results for medium and large objects, with AP_M at 39.6% and AP_L at 64.7%. The DAPIoU loss variant with $z = 10{,}500$ also performs well, with the highest AP_{50} at 75.5%, and solid results across other categories. All the mAP results in YOLOv9 are higher than YOLOv7.

These results highlight the potential benefits of refining IoU-based loss functions to enhance object detection models' accuracy and reliability. For the MS COCO 2017 dataset used with the YOLOv7 in Tab. 4, YOLOv9 in Tab. 6, and Faster R-CNN in Tab. 7 models featuring ResNet-50. In YOLOv7, the experiments were conducted by setting the hyperparameter z from 10,000 to 12,000. Among the tested loss functions, DAPIoU loss with $k = 2$ and $z = 11{,}000$ achieves the best overall performance. It has the highest $AP_{50:95}$ at 49.0%, AP_{50} at 67.3% and excels in detecting small objects with AP_S at 32.9%. For medium and large objects, its performance is also strong, with AP_M at 53.7% and AP_L at 63.6%. For YOLOv9 also gives the best overall mAP at DAPIoU loss with $k = 2$ and $z = 10{,}500$. For Faster R-CNN, DAPIoU loss with parameters $k = 2$ and $z = 10{,}500$ achieves the highest scores in most metrics. It outperforms the others with the highest $AP_{50:95}$ of 34.8, AP_{50} of 51.0, and AP_{75} of 38.4. Additionally, it shows superior performance in small object detection with an AP_S of 17.4% and maintains competitive results in detecting medium and large objects, with AP_M at 37.6% and AP_L at 48.4%. In summary, for YOLOv7, YOLOv9, and Faster R-CNN on the PASCAL VOC 2007+2012 and MS COCO 2017 dataset, DAPIoU loss demonstrates the best overall performance, making it the most effective loss function among the compared options.

In the context of hyperparameter tuning, there are no strict limitations on defining the hyperparameter z because it exists on a continuous spectrum. Since z is not restricted to discrete values, it can be adjusted incrementally across a wide range to optimize the model's performance. This flexibility allows for precise tuning, as any real number value within the practical limits of the model can be explored. By leveraging this continuity, hyperparameters such as 9500, 10,000, or 11,000 can be fine-tuned to maximize accuracy or efficiency for a particular task. However, there is no guarantee that the smallest or highest values of the z coordinate will yield the best mAP. The performance of the model can vary unpredictably with changes in the z value, and both smaller and larger values can impact results in different ways depending on the specific data and model architecture. This unpredictability reinforces the importance of conducting thorough hyperparameter tuning across a range of values to identify the optimal setting, rather than if extreme values will necessarily produce superior results.

Tab. 8 and Tab. 9 present both the training process time per epoch (in seconds) and inference time per image for each IoU-based loss function using the YOLOv7 model on two different datasets: PASCAL VOC 2007+2012 and MS COCO 2017. In Tab. 8, FIoU loss exhibits the longest training time at 135.1 sec/epoch, followed by DAPIoU loss at 132.7 sec/epoch, while other loss functions demonstrate relatively lower training durations. Meanwhile, Tab. 9 indicates that DAPIoU loss requires the longest training time on the MS COCO 2017 dataset, reaching 747.1 sec/epoch, surpassing other loss functions. This suggests that DAPIoU loss may introduce additional computational complexity due to more intricate gradient calculations or additional bounding box optimization steps.

Regarding inference time, Tab. 8 shows that IoU loss achieves the fastest inference speed at 2.0 sec/img, while DIoU loss has the slowest at 4.6 sec/img. Notably, DAPIoU loss achieves an inference time of 3.4 sec/img, which is comparable to other advanced IoU-based losses. Similarly, Tab. 9 reveals that FIoU loss exhibits the highest inference time at 15.6 sec/img, while DAPIoU loss maintains competitive performance at 5.3 sec/img, slightly outperforming CFIoU loss at 5.9 sec/img. These results indicate that while DAPIoU may require longer training times, its inference efficiency remains within a reasonable range, making it viable for real-time applications.

Tab. 10 presents the computational complexity of Faster R-CNN, YOLOv7, and YOLOv9 across different datasets, measured in FLOPs (Floating Point Operations) and the number of model parameters (in millions). Faster R-CNN, evaluated on the MS COCO 2017 dataset, has a FLOP count of 85.6 GFLOPs and 41.7 million parameters, reflecting its computational demand. YOLOv7, when tested on the PASCAL VOC 2007+2012 dataset, requires 103.5 GFLOPs with 36.5 million parameters, while on MS COCO 2017, it slightly increases to 104.5 GFLOPs with 36.9 million parameters. Meanwhile, YOLOv9 demonstrates significantly higher computational complexity, requiring 236.9 GFLOPs with 50.7 million parameters for PASCAL VOC 2007+2012 and 237.6 GFLOPs with 50.8 million parameters for MS COCO 2017. These results highlight the increasing computational cost associated with more advanced detection models, where YOLOv9 exhibits a substantial rise in

complexity compared to YOLOv7 and Faster R-CNN, potentially affecting real-time deployment considerations.

TABLE 8
TRAINING PROCESS TIME AND INFERENCE TIME OF DIFFERENT IOU-BASED LOSS FUNCTION USING YOLOV7 ON PASCAL VOC 2007+2012 DATASET

Loss Function	Training Process Time (sec/epoch)	Inference time (sec/img)
$\mathcal{L}_{IoU}$	121.7	2.0
$\mathcal{L}_{GIoU}$	124.4	3.7
$\mathcal{L}_{DIoU}$	122.8	4.6
$\mathcal{L}_{CIoU}$	123.5	3.7
$\mathcal{L}_{CFIoU}$	126.5	4.3
$\mathcal{L}_{FIoU}$	135.1	3.9
$\mathcal{L}_{DAPIoU}$	132.7	3.4

TABLE 9
TRAINING PROCESS TIME AND INFERENCE TIME OF DIFFERENT IOU-BASED LOSS FUNCTION USING YOLOV7 ON MS COCO 2017 DATASET

Loss Function	Training Process Time (sec/epoch)	Inference Time (sec/img)
$\mathcal{L}_{CFIoU}$	708.6	5.9
$\mathcal{L}_{FIoU}$	707.2	15.6
$\mathcal{L}_{DAPIoU}$	747.1	5.3

TABLE 10
COMPUTATIONAL COMPLEXITY OF FASTER R-CNN, YOLOV7, AND YOLOV9 ACROSS DATASETS

Model	Dataset	Complexity	
		FLOP	Parameters (Million)
Faster R-CNN	MS COCO 2017	85.6	41.7
YOLOv7	PASCAL VOC 2007+2012	103.5	36.5
	MS COCO 2017	104.5	36.9
YOLOv9	PASCAL VOC 2007+2012	236.9	50.7
	MS COCO 2017	237.6	50.8

D. DISCUSSION

The introduction of the DAPIoU loss has demonstrated notable improvements in object detection accuracy. The experiments conducted on both synthetic and real-world datasets reveal that DAPIoU offers a significant advantage in bounding box localization precision compared to traditional IoU-based loss functions. The gradient analysis indicates that DAPIoU's structure, which incorporates penalties for misaligned angles and corner distances, results in a more effective convergence during training. This suggests that DAPIoU enables object detection models to better adjust predictions, particularly in scenarios where fine-grained localization is essential, such as detecting small or irregularly shaped objects. Additionally, the performance evaluation using the PASCAL VOC and MS COCO datasets confirmed that the DAPIoU loss outperforms other advanced IoU variants, particularly in handling diverse object sizes and shapes. Our study focused on anchor-based models because DAPIoU is designed to enhance IoU-based BBR, which is a core component of anchor-based detection frameworks. In contrast, anchor-free object detectors such as FCOS, FSAF, FOVEA, and CenterNet predict object locations without predefined anchor boxes. These models follow different localization strategies: some (e.g., CenterNet, CornerNet) use keypoint-based localization, detecting object centers or corners to infer bounding box positions, while others (e.g., FCOS, FOVEA) directly regress bounding box coordinates from feature maps.

To further assess DAPIoU's generalization ability, we evaluated it on various object detection models, including YOLOv7, YOLOv9, and Faster R-CNN. The results confirm that DAPIoU enhances detection performance across different architectures, proving that its impact is not limited to a single

model. These findings reinforce that improving localization accuracy through loss function optimization can have a measurable impact on detection performance, making DAPIoU a valuable contribution to object detection research. Since DAPIoU primarily improves IoU-based BBR, it was more relevant to evaluate its performance in anchor-based architectures where IoU loss directly impacts BBR optimization. However, we acknowledge that anchor-free object detectors represent a promising direction for IoU-based loss function adaptations. Future research could explore whether DAPIoU can be effectively integrated into anchor-free detection models, particularly those that incorporate IoU-based constraints in their regression heads.

The proposed loss function not only accelerates convergence but also maintains robust performance across a wide range of hyperparameter settings, highlighting its adaptability to different datasets and detection models. In addition to its demonstrated effectiveness in object detection, the DAPIoU loss has the potential to impact other computer vision tasks that require precise localization. Its ability to address alignment challenges through angular and corner penalties makes it particularly relevant for applications such as instance segmentation, object tracking, and keypoint detection. Future research will further explore these applications to validate DAPIoU's versatility and extend its benefits to a broader range of vision problems.

While the proposed DAPIoU loss enhances 2D object detection by improving bounding box alignment through corner distance and angular penalties, it is important to note that DAPIoU is limited to 2D BBR. The formulation of DAPIoU is specifically designed for 2D spatial relationships and does not directly extend to 3D BBR used in LiDAR-based or multi-sensor object detection systems. Applying DAPIoU in 3D object detection tasks would require significant modifications, including adjustments to account for depth information and 3D orientation angles (yaw, pitch, roll). Future research could explore how DAPIoU's principles might be adapted for 3D object detection by incorporating higher-dimensional spatial constraints. Additionally, while this study evaluates DAPIoU in anchor-based object detection frameworks (YOLO and Faster R-CNN), future research could explore its applicability to anchor-free detectors such as FCOS, FSAF, FOVEA, and CenterNet.

V. CONCLUSION

DAPIoU loss enhances BBR performance across diverse datasets and detection conditions, ensuring robust object localization. By incorporating angular penalties and corner distance metrics, it addresses challenges such as gradient inconsistency and enhances localization accuracy for both small and large objects. The results on benchmark datasets with state-of-the-art models demonstrate consistent improvements in $AP_{50:95}$. Specifically, DAPIoU shows string performance AP_S for small object and AP_L for large object, underscoring its versatility across diverse object scales. The findings of this study underscore the potential of DAPIoU as a superior alternative for BBR in object detection tasks. By incorporating angle precision and corner penalties, the DAPIoU loss addresses some of the limitations of traditional IoU-based loss functions, offering enhanced accuracy and faster convergence. This makes it particularly suitable for complex object detection challenges involving varied object scales and positions. A primary limitation of DAPIoU is its current focus on 2D BBR, requiring further modifications for effective integration into 3D object detection tasks. Applying DAPIoU in 3D detection frameworks would require modifications to account for depth information and 3D orientation angles. Future work could investigate how DAPIoU's core principles can be adapted for 3D object detection, particularly in applications involving LiDAR, radar, or multi-view depth estimation.

Beyond object detection, DAPIoU's potential for precise localization could be explored in other computer vision tasks such as instance segmentation, object tracking, and keypoint detection, These applications share similar challenges in bounding box alignment, and adapting DAPIoU to these domains could further enhance its impact in high-precision vision tasks.

APPENDIX A
DAPIOU LOSS ALGORITHM

In this section, we present the DAPIoU loss algorithm to provide a clearer and more structured representation of its computation process. This algorithm outlines the key steps involved in integrating IoU calculations, corner distance and angular penalties, making it easier to understand how DAPIoU optimizes BBR.

Algorithm 1 DAPIoU Loss Algorithm

Input: Bounding box of Prediction $B^p = (w^p, h^p, x^p, y^p)$
Input: Bounding box of Ground Truth $B^{gt} = (w^{gt}, h^{gt}, x^{gt}, y^{gt})$
Output: $\mathcal{L}_{DAPIoU}$

1: $\mathcal{R}_{CD}(B^p, B^{gt}) = \dfrac{\sum_{i=1}^{4} \rho^2\left(b_i^p, b_i^{gt}\right)}{k \cdot (l^c)^2}$

$$b_1 = \sqrt{\left(x_1^p - x_1^{gt}\right)^2 + \left(y_1^p - y_1^{gt}\right)^2 + z^2}$$

$$b_2 = \sqrt{\left(x_2^p - x_2^{gt}\right)^2 + \left(y_1^p - y_1^{gt}\right)^2 + z^2}$$

$$b_3 = \sqrt{\left(x_1^p - x_1^{gt}\right)^2 + \left(y_2^p - y_2^{gt}\right)^2 + z^2}$$

2:

$$b_4 = \sqrt{\left(x_2^p - x_2^{gt}\right)^2 + \left(y_2^p - y_2^{gt}\right)^2 + z^2}$$

$$c_1 = \sqrt{\left(x_2^p - x_1^{gt}\right)^2 + \left(y_1^p - y_1^{gt}\right)^2 + z^2}$$

$$c_2 = \sqrt{\left(x_1^p - x_2^{gt}\right)^2 + \left(y_1^p - y_1^{gt}\right)^2 + z^2}$$

$$c_3 = \sqrt{\left(x_1^p - x_1^{gt}\right)^2 + \left(y_1^p - y_2^{gt}\right)^2 + z^2}$$

$$c_4 = \sqrt{\left(x_2^p - x_2^{gt}\right)^2 + \left(y_1^p - y_2^{gt}\right)^2 + z^2}$$

3: $I_{1,2} = \frac{(w^p)^2 + b_{1,2}{}^2 - c_{1,2}{}^2}{2 \cdot w^p \cdot b_{1,2}}$ and $I_{3,4} = \frac{(h^p)^2 + b_{3,4}{}^2 - c_{3,4}{}^2}{2 \cdot h^p \cdot b_{3,4}}$

4: $\theta_i = \cos^{-1}(I_i)$

5: $v = \frac{4}{\pi^2}\left(\sum_{i=1}^{4}\left(\theta_i - \frac{\pi}{2}\right)^2\right)$

6: $\alpha = \frac{v}{(1 - IoU) + v}$

7: $\mathcal{R}_{3DAP}(B^p, B^{gt}) = \alpha \cdot v$

8: $\mathcal{L}_{IoU} = 1 - IoU$

9: $\mathcal{L}_{DAPIoU} = \mathcal{L}_{IoU} + \mathcal{R}_{CD}(B^p, B^{gt}) + \mathcal{R}_{3DAP}(B^p, B^{gt})$

APPENDIX B
SINGLE-GROUP BBR SIMULATION EXPERIMENT ALGORITHM

In this section, we represented single-group BBR simulation as an algorithm for DAPIoU loss, specifically designed to determine the optimal values of the hyperparameter k and z. This tuning process directly impacts the effectiveness of DAPIoU loss when applied to real-world datasets.

Algorithm 2 Single-Group BBR Simulation Experiment

Input: The function $\mathcal{L}$ is a continuous bounded loss function defined on $\mathbb{R}_+^4$. The predicted bounding box is initialized with bottom-left coordinate (3,3), a width of 1, and a height of 0.5, represented as $B^{t=0} = (3,3,1,0.5)$. The target bounding box has bottom-left coordinate (1,1), a width of 0.5, and a height of 0.5, represented as $B^{gt} = (1,1,0.5,0.5)$. The regression process is gorverned by a learning rate η and runs for T iterations.

Output: Regression error $\boldsymbol{E}$

1: Initialize $\boldsymbol{E} = \mathbf{0}$ and maximum iteration T
2: Do BBR:
3: **for** $t = 1$ to T **do**
4: $y = \begin{cases} 0.1 & t \le 0.9T \\ 0.01 & t > 0.9T \end{cases}$
5: ∇B^{t-1} is gradient of $\mathcal{L}(B^{t-1}, B^{gt})$ w.r.t. B^{t-1}
6: $B^t = B^{t-1} + \eta \nabla B^{t-1}$
7: **end for**

APPENDIX C
SIMULATION EXPERIMENT ALGORITHM

In this section, we present a simulation experiment as an algorithm used to evaluate the performance of traditional IoU-based loss function and our proposed loss function, including IoU loss, GIoU loss, DIoU loss, CIoU loss, and DAPIoU loss. This algorithm serves as a supplementary explanation to further illustrate the methodology behind the experiment, particularly supporting the analysis presented in Fig. 6 and Fig. 7. By analyzing these loss functions in a controlled setting, we provide deeper insights into their optimization behavior and their impact on improving object detection performance.

Algorithm 3 Simulation Experiment Algorithm

Input: The function $\mathcal{L}$ is a continuous bounded loss function defined on $\mathbb{R}_+^4$. The set $\mathbb{M}$ consists of anchor boxes $\{\{B_{n,s}\}_{s=1}^{S}\}_{n=1}^{N}$ for $N = 5000$ uniformly scattered points within a circular region centered at (10,10) with a radius of 3. Each point has $S = 7 \times 7$ anchor boxes, covering 7 different scales and 7 aspect ratios.

The set $\mathbb{M}^{gt}$ consists of target boxes $\{B_i^{gt}\}_{i=1}^{7}$ that are fixed at (10,10) with area 1 and have 7 aspect ratios.

Output: Regression error $\boldsymbol{E}$

1: Initialize $\boldsymbol{E} = \mathbf{0}$ and maximum iteration T
2: Do BBR:
3: **for** $n = 1$ to N **do**
4: **for** $s = 1$ to S **do**
5: **for** $i = 1$ to 7 **do**
6: **for** $t = 1$ to T **do**
7: $\eta = \begin{cases} 0.1 & t \le 0.8T \\ 0.01 & 0.8T < t \le 0.9T \\ 0.001 & t > 0.9T \end{cases}$
8: $\nabla B_{n,s}^{t-1}$ is gradient of $\mathcal{L}(B_{n,s}^{t-1}, B_i^{gt})$ w.r.t. $B_{n,s}^{t-1}$
9: $B_{n,s}^t = B_{n,s}^{t-1} + \eta\left(2 - IoU_{n,s}^{t-1}\right)\nabla B_{n,s}^{t-1}$
10: $E(t,n) = E(t,n) + \left|B_{n,s}^t - B_i^{gt}\right|$
11: **end for**
12: **end for**
13: **end for**
14: **end for**
15: **return** $\boldsymbol{E}$

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, 2015.
[2] R. Atienza, *Advanced Deep Learning with Keras*. Packt Publishing Ltd, 2019.
[3] 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.*, 2016, pp. 779-788.
[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91-99.
[5] 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*., 2016, pp. 21-37.
[6] 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. IEEE Conf. Comput. Vis. Pattern Recognit*., 2019, pp. 658-666.

[7] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "UnitBox: An Advanced Object Detection Network," in *Proc. 24th ACM Int. Conf. Multimedia*, 2016, pp. 516-520.
[8] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully Convolutional One-Stage Object Detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis*., 2019, pp. 9627-9636.
[9] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression," in *Proc. AAAI Conf. Artif. Intell*., 2020, vol. 34, no. 07, pp. 12993-13000.
[10] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the Gap Between Anchor-based and Anchor-free Detection via Adaptive Training Sample Selection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit*., 2020, pp. 9759-9768.
[11] P. Liu, G. Zhang, B. Wang, H. Xu, X. Liang, Y. Jiang, and Z. Li, "Loss Function Discovery for Object Detection via Convergence-Simulation Driven Search," ArXiv, 2021.
[12] Z. Zheng, P. Wang, D. Ren, W. Liu, R. Ye, Q. Hu, and W. Zuo, "Enhancing Geometric Factors in Model Learning and Inference for Object Detection and Instance Segmentation," IEEE Transactions on Cybernetics, vol. 52, pp. 8574-8586, 2020.
[13] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *Int. J. Comput. Vis*., vol. 88, no. 2, pp. 303-338, 2010.
[14] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014, pp. 740-755, doi: 10.1007/978-3-319-10602-1_48.
[15] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*., 2017, pp. 6517-6525.
[16] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.
[17] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv preprint arXiv:2004.10934*, 2020.
[18] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint arXiv:2207.02696*, 2022.
[19] S. Ren, K. 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, 2015.
[20] R. Girshick, "Fast R-CNN," in Proc. *IEEE Int. Conf. Comput. Vis*., 2015, pp. 1440-1448.
[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*., 2014, pp. 580-587.
[22] A. Géron*, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. O'Reilly Media, 2019.
[23] P. J. Huber, "Robust Estimation of a Location Parameter," *Ann. Math. Stat*., vol. 35, no. 1, pp. 73-101, 1964.
[24] P. Jaccard, "The distribution of the flora in the alpine zone," *New Phytologist*, vol. 11, no. 2, pp. 37-50, 1912.
[25] D. Cai, Z. Zhang, & Z. Zhang, "Corner-Point and Foreground-Area IoU Loss: Better Localization of Small Objects in Bounding Box Regression", Sensors, 23(10)., 2023.
[26] Y. Sun, J. Wang, H. Wang, S. Zhang, Y. You, Z. Yu, and Y. Peng, "Fused-IoU Loss: Efficient Learning for Accurate Bounding Box Regression," IEEE Access, vol. 12, pp. 37363-37375, 2024, doi: 10.1109/ACCESS.2024.3359433.
[27] A. Alshubbak and D. Görges, "Investigation of the Performance of Different Loss Function Types Within Deep Neural Anchor-Free Object Detectors," In Proceedings of the 16th International Conference on Agents and Artificial Intelligence (ICAART 2024), vol. 3, pp. 401-411, 2024, doi: 10.5220/0012354900003636.
[28] C. Zhu, Y. He, and M. Savvides, "Feature Selective Anchor-Free Module for Single-Shot Object Detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit*., 2019.
[29] K. Su, L. Cao, B. Zhao, N. Li, D. Wu, and X. Han, "N-IoU: better IoU-based bounding box regression loss for object detection," Neural Computing and Applications, 2023, doi: 10.1007/s00521-023-09133-4.

HILMY ALIY ANDRA PUTRA earned a master's degree in Computational Science from the Faculty of Mathematics and Natural Science, Bandung Institute of Technology. He is currently a Ph.D candidate in Computer Science at Faculty of Computer Science, Universitas Indonesia. His research focuses on machine learning dan computer vision, with a particular emphasis on object detection.

ANIATI MURNI ARYMURTHY is a Professor at the Faculty of Computer Science, Universitas Indonesia. She was graduated from Department of Electrical Engineering, Universitas Indonesia, Jakarta, Indonesia. She earned her Master of Science from Department of Computer and Information Sciences, The Ohio State University (OSU), Columbus, Ohio, USA. She also holds Doktor from Universitas Indonesia and a sandwich program at the Laboratory for Pattern Recognition and Image Processing (PRIP Lab), Department of Computer Science, Michigan State University (MSU), East Lansing, Michigan, USA. She is the head of Laboratory for Machine Learning and Computer Vision, Faculty of Computer Science, Universitas Indonesia. Her research interest includes the use of Pattern Recognition and Image Processing methods in several applications such as remote sensing, biomedical application, cultural artefak, agriculture and e-livestock.

DINA CHAHYATI received the B.Sc. degree in Computer Science from Universitas Indonesia, Depok, Indonesia. She also earned her M.Sc. and Ph.D. degrees in Computer Science from the same university. Currently, she is a Lecturer in the Faculty of Computer Science at Universitas Indonesia. Her research interests include computer vision and pattern recognition, particularly within the domains of artificial intelligence and data science & analytics.