electronics

MDPI

Article

An Improved Safety Belt Detection Algorithm for High-Altitude Work Based on YOLOv8

Tingyao Jiang, Zhao Li *, Jian Zhao, Chaoguang An, Hao Tan and Chunliang Wang

College of Computer and Information, China Three Gorges University, Yichang 443002, China
* Correspondence: 202108540021079@ctgu.edu.cn

Abstract: High-altitude work poses significant safety risks, and wearing safety belts is crucial to prevent falls and ensure worker safety. However, manual monitoring of safety belt usage is time consuming and prone to errors. In this paper, we propose an improved high-altitude safety belt detection algorithm based on the YOLOv8 model to address these challenges. Our paper introduces several improvements to enhance its performance in detecting safety belts. First, to enhance the feature extraction capability, we introduce a BiFormer attention mechanism. Moreover, we used a lightweight upsampling operator instead of the original upsampling layer to better preserve and recover detailed information without adding an excessive computational burden. Meanwhile, Slim-neck was introduced into the neck layer. Additionally, extra auxiliary training heads were incorporated into the head layer to enhance the detection capability. Lastly, to optimize the prediction of bounding box position and size, we replaced the original loss function with MPDIOU. We evaluated our algorithm using a dataset collected from high-altitude work scenarios and demonstrated its effectiveness in detecting safety belts with high accuracy. Compared to the original YOLOv8 model, the improved model achieves P (precision), R (recall), and mAP (mean average precision) values of 98%, 91.4%, and 97.3%, respectively. These values represent an improvement of 5.1%, 0.5%, and 1.2%, respectively, compared to the original model. The proposed algorithm has the potential to improve workplace safety and reduce the risk of accidents in high-altitude work environments.

Keywords: high-altitude work; safety belt detection; yolov8

Citation: Jiang, T.; Li, Z.; Zhao, J.; An, C.; Tan, H.; Wang, C. An Improved Safety Belt Detection Algorithm for High-Altitude Work Based on YOLOv8. *Electronics* **2024**, *13*, 850. https://doi.org/10.3390/electronics13050850

Academic Editor: George A. Papakostas

Received: 23 December 2023
Revised: 2 February 2024
Accepted: 15 February 2024
Published: 23 February 2024

1. Introduction

The rapid advancement in artificial intelligence and big data technology has led to the adoption of intelligent and digital approaches in numerous industries. In high-risk work environments, it is paramount to enhance safety behaviors and supervision systems to prevent accidents and personnel casualties. In real-life scenarios, electricians face numerous unpredictable factors during high-altitude work, such as the risks of electric shock and falling from heights. It is noteworthy to mention that falls from heights are a major concern in electrical work. According to official statistics from China's National Energy Administration, there has been an increase in the number of electric power accidents in recent years, with high-altitude fall accidents accounting for approximately 26% of these incidents. The working environment at electric power operation sites is characterized by complexity and variability, which significantly impacts the safety of electricians.

Investigations have revealed that the primary cause of personal accidents at high-altitude electrical work sites is inadequate safety measures and behavioral standards among electricians, coupled with a lack of strict adherence to operational procedures and systems for on-site supervision [1]. As a result, many power companies have implemented a requirement for on-site guardians for power staff working at high-altitude power operation sites. This paper aims to enhance the target detection algorithm used for monitoring the safety behavior of staff in modern construction sites, as well as electric power production and maintenance sites. The objective is to achieve accurate and real-time detection and

identification of the field environment of electricians in real-life scenarios. Once abnormal behavior is detected, an alert will be promptly issued to notify and remind the staff.

2. Related Work

Object detection is one of the fundamental challenges in the domain of computer vision. Its main task is to identify all the targets of interest in the image and to determine their categories and locations [2]. Object detection has always been a challenging problem due to the different types of objects with different appearances, postures, and degrees of occlusion, as well as light-intensity interference. Object detection algorithms based on deep learning can be roughly divided into one-stage and two-stage algorithms [3,4]. The fundamental distinction stems from the disparities in the candidate regions; the two-stage algorithm requires the generation of region proposals before predicting the classification and location of the target object using convolutional neural networks. The distinguishing feature of the two-stage algorithms is their comparatively slower processing speed, despite their high level of accuracy. Examples of such algorithms include the RCNN series (R-CNN [5], Fast R-CNN [6], Faster R-CNN [7], and Mask R-CNN [8]). While the one-stage algorithm directly extracts features in the network to predict the classification and location of the target object, this type of algorithm is characterized by its speed, but its accuracy is not as high as that of a two-stage algorithm [9]. Typical examples of the one-stage algorithm include the SSD [10] series and the YOLO [11–18] series. While the SSD series has relatively few applications, the algorithms of the YOLO series have developed rapidly. The original YOLO object detector was first released in 2016 by Joseph Redmon, Ali Farhadi, and Santosh Divvaia [11]. At the time of its release, this architecture was significantly faster than other object detectors, making it the state-of-the-art technology for real-time computer vision applications. It remains a popular choice for real-time object detection today, with an ongoing trend of continuous updates and iterations in the future. In July 2022, YOLOv7 has been released [17], followed by YOLOv8 in January 2023 [18]. In less than eight years since the initial release of YOLOv1, many other versions of the YOLO algorithm have been developed during this period [19].

The focus of this study is safe belt detection for high-altitude work. Currently, research on object detection in high-altitude work mainly focuses on the recognition and detection of safety belts and helmets. For example, Feng Zhizhen and colleagues used a two-stage Mask R-CNN-based approach for high-altitude operation safety belt detection [20], whereas Zhang Meng and colleagues employed an improved one-stage YOLOv4 object detection algorithm [21]. In addition, to address the issues of low timeliness and lack of scene specificity in existing safety-belt-wearing detection methods, Cao Jie and colleagues introduced the YOLOX target detection model [22], which enables real-time detection with high accuracy. These studies suggest that advancements have been made in target detection technology for high-altitude power safety operations. Although there are some existing studies on high-altitude safety operations, there is still room for improvement.

3. Methods

3.1. YOLOv8 Model

As one of the most representative examples of one-stage target detection algorithms, the YOLOv8 algorithm employs a deep neural network to recognize and locate objects. It is known for its speed and can be effectively utilized in real-time systems. The YOLOv8 network architecture is depicted in Figure 1.

The YOLOv8 algorithm is primarily composed of three components: Backbone, Neck, and Head.

- Backbone: The Backbone layer is a network responsible for feature extraction. Its main role is to extract relevant information from images, which can then be utilized by subsequent networks or modules for further processing and analysis.
- Neck: The Neck layer is positioned between the Backbone and the Head to optimize the utilization of features extracted by the backbone. It plays a crucial role in feature

fusion, enabling the Neck layer to effectively combine and integrate the extracted features.

- Head: The Head layer utilizes the previously extracted features to perform recognition.

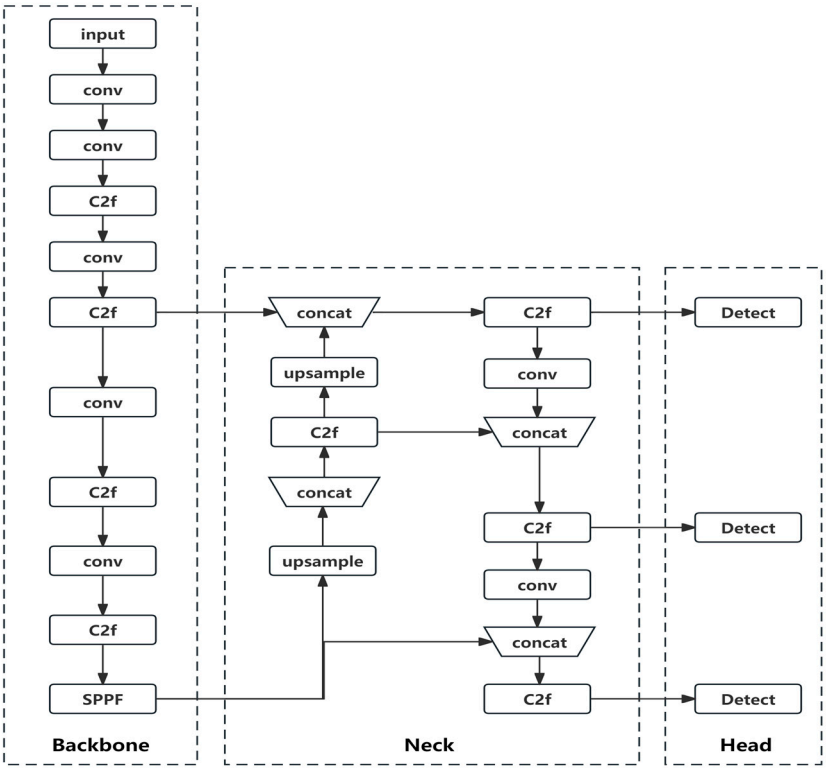


Figure 1. YOLOv8 network model architecture.

3.2. Improvement Measures

The challenges in detecting safety belts during high-altitude operations are as follows:

- Occlusion by Other Objects: During high-altitude electrical work, there can be other objects that occlude the visibility of safety belts worn by electricians.
- Misidentification of Cables: Aerial cables and similar structures in high-altitude electrical work can be mistakenly identified as safety belts, leading to inaccurate detection results.
- Variations in Lighting Conditions and Electrician Movements: The dynamic changes in lighting conditions and the movement of electricians during high-altitude operations introduce complexities in accurately detecting safety belts.

These challenges can result in false positives or false negatives, ultimately affecting the overall detection performance. To overcome these obstacles, this paper proposes an improved YOLOv8 model that effectively addresses these challenges.

The primary focus is to introduce improvements in the Backbone, Neck, and Head layers based on the YOLOv8 architecture. The improvement strategy involves the following key points:

- Firstly, an attention mechanism is integrated into the Backbone layer to enhance the capability of feature extraction. Through multiple experimental trials, attention mechanisms, known as Biformer, are introduced at various locations within the Backbone layer. After careful evaluation, it is determined that adding Biformer attention at the end of the Backbone layer yields the most favorable results. This choice is made considering that incorporating attention mechanisms in the shallower layers of the Backbone layer would lead to increased computational complexity.
- Secondly, in the Neck layer, the original upsampling operations are completely replaced with the CARAFE lightweight upsampling operator. Additionally, a lightweight

network called Slim-neck is employed as the neck structure to maintain the network's performance while reducing the model complexity and making it more lightweight. The C2f module is replaced with the VoVGSCSP module, and all 3×3 convolutions are substituted with GSConv.

- Thirdly, additional auxiliary heads are incorporated into the Head layer to facilitate training and enable the intermediate layers of the network to learn more information.
- Lastly, the original loss function is replaced with the MPDIoU loss function, which optimizes the regression of bounding boxes. This replacement aims to improve the accuracy and precision of object detection.

The improved network model structure is shown in Figure 2 as follows:

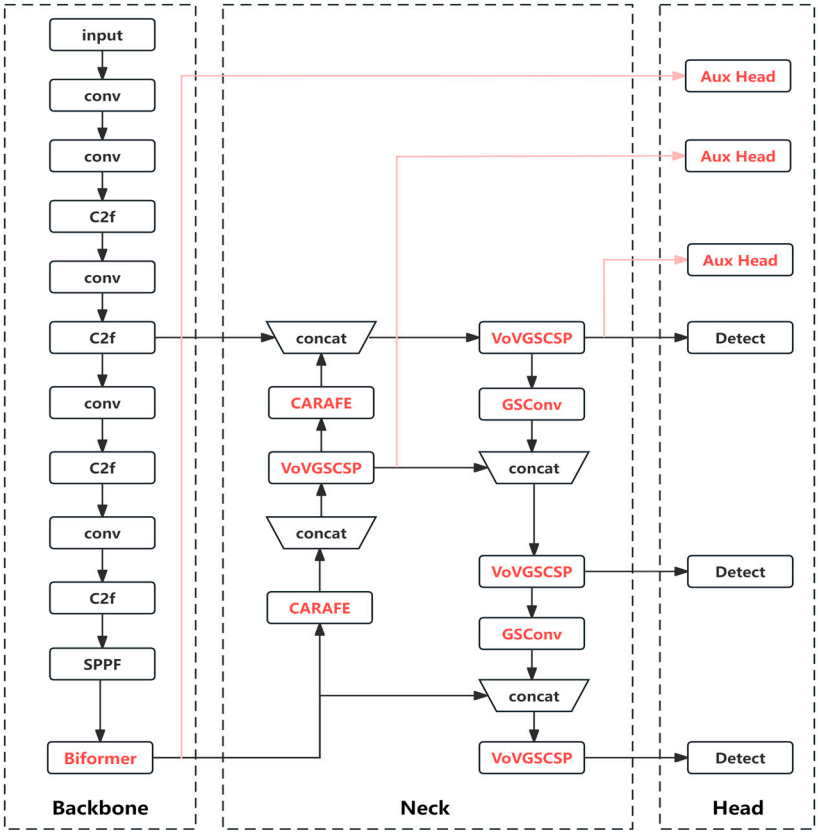


Figure 2. Improved YOLOv8 network model architecture.

3.2.1. Biformer Attention

For small target detection, if sufficient context can be used as additional information to help detect small targets, the detection efficiency will be higher because computers do not directly identify objects visually like human eyes; instead, they abstract the real world into an information world and then digitize the information world into a computer world. The objects in target detection exist digitally in the computer; each pixel in a picture is composed of numbers in the range of 0–255 in the computer, and a three-channel picture is composed of three digital matrices. However, no pixel in the image is isolated; a pixel must have a certain connection with its surrounding pixels, and a large number of pixels are mutually connected to produce various objects in the image. Therefore, to determine which category a pixel at a certain position in the whole picture belongs to, not only the grayscale value of the pixel should be considered but the adjacent pixels should also be fully considered. Attention mechanisms are powerful tools for capturing long-distance contextual dependencies, but traditional attention mechanisms often cause two common problems: high memory consumption and high computational cost.

To address the aforementioned issues, Lei Zhu et al. [23] proposed a novel dynamic sparse attention: the Bi-Level Routing Attention, whose workflow is shown in Figure 3a. The BiFormer block is designed based on the Bi-Level Routing Attention, as shown in Figure 3b.

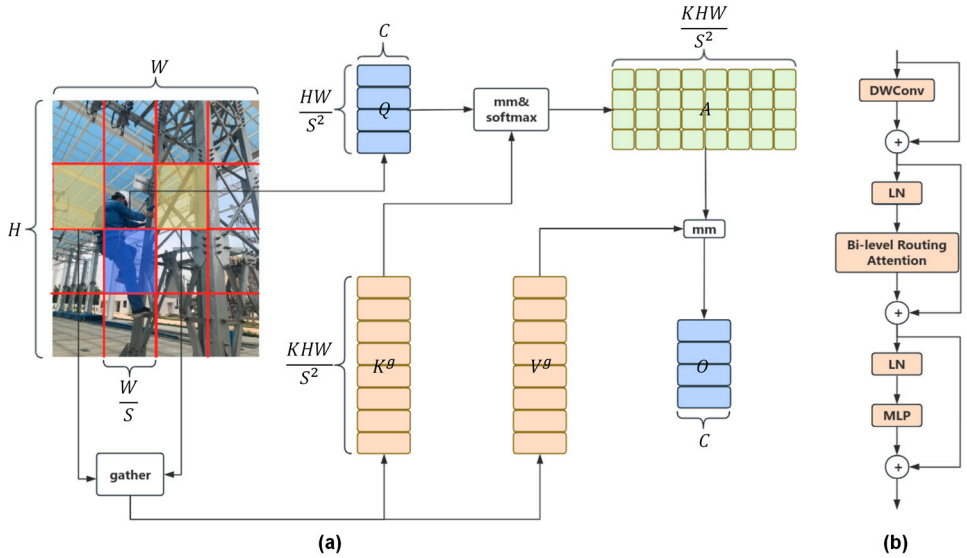


Figure 3. (**a**) Biformer attention mechanism workflow diagram. (**b**) Structure of the Biformer block.

The workflow of the Biformer attention mechanism primarily consists of the following three steps:

1. Partitioning Regions and Linear Mapping:

For a given feature map $X \in R^{H\times W\times C}$, the initial step entails dividing it into $S \times S$ non-overlapping regions, ensuring that each region contains $\frac{HW}{S^2}$ feature vectors. Subsequently, a reshaping operation is performed to transform X into $X^r \in R^{S^2\times \frac{HW}{S^2}\times C}$, facilitating a linear mapping denoted as $Q, K, V \in R^{H\times W\times C}$. The computational equations are as follows, represented by Equations (1)–(3):

$$Q = X^r W^q \tag{1}$$

$$K = X^r W^k \tag{2}$$

$$V = X^r W^v \tag{3}$$

2. Implementing Inter-Region Routing via Directed Graphs:

Firstly, the average value of each region is calculated individually for Q and K, resulting in $Q^r, K^r \in R^{S^2\times C}$. Then, by performing matrix multiplication on these two average values, the affinity adjacency matrix between the regions is obtained. This process can be represented by the following Equation (4):

$$A^r = Q^r (K^r)^T \tag{4}$$

Subsequently, a pruning operation is performed on matrix A^r, eliminating the least relevant token within A^r at the coarse-grained level. The top k regions with the highest relevance scores in A^r are selectively retained, leading to the generation of the routing index matrix denoted as $I^r \in N^{s^2\times k}$. The calculation formula for this procedure is represented by Equation (5):

$$I^r = topkIndex(A^r) \tag{5}$$

3. Token-to-token Attention:

For each query token in region i, it focuses on the key-value pairs belonging to the union of k routing regions indicated by the indices $I^r(i,1)$,$I^r(i,2)$,.....,$I^r(i,k)$. These

regions are distributed throughout the entire feature map, and modern GPUs rely on memory coalescence to load blocks of contiguous bytes efficiently. Therefore, it is necessary to first collect K and V. The calculation formula for this procedure is represented by Equations (6) and (7):

$$K^g = gather(K, I^r) \tag{6}$$

$$V^g = gather(V, I^r) \tag{7}$$

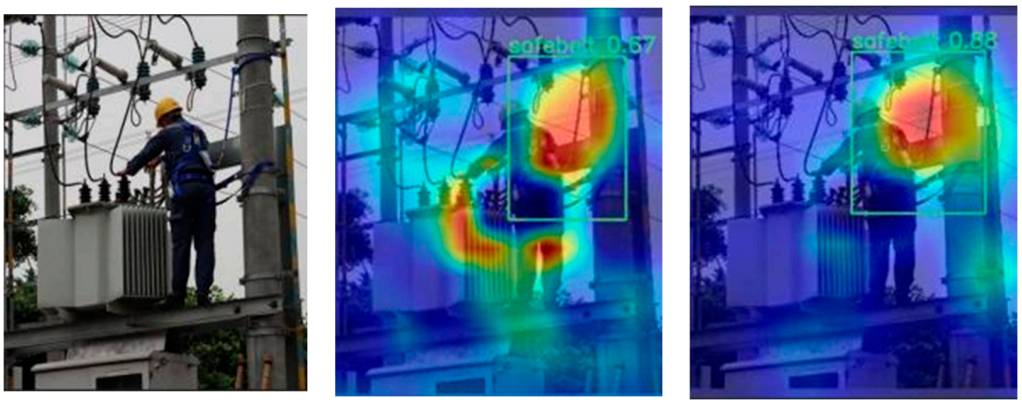
In the end, attention is applied to the collected key-value pairs. The calculation formula for this procedure is represented by Equation (8):

$$\mathrm{O} = Attention(Q, K^g, V^g) + LCE(V) \tag{8}$$

The key idea is to filter out unimportant key-value pairs to achieve fine-grained and sparse attention [24]. It introduces a new two-layer routing attention mechanism, realizing content-aware sparse patterns using an adaptive query, and dynamic and query-aware ways to achieve an efficient allocation of computation, so the Biformer attention mechanism has better performance and lower computational cost.

In addition, introducing the Biformer attention mechanism can help the algorithm better distinguish between targets and backgrounds, thereby reducing the number of false positives and false negatives and improving the accuracy of object detection. By focusing on the key areas of the target, the algorithm can cope better with occlusions, changes in lighting, and other interference factors, thereby increasing the robustness of the object detection algorithm.

In this article, we use the Class Activation Mapping (CAM) method to visualize the features of the model's detection results, allowing us to observe which areas of the image the network has paid attention to after going through the Backbone and Head to recognize and locate the target. It also allows us to analyze and compare the YOLOv8 model with and without attention mechanisms to determine the degree of attention paid to the same target. The original image is shown in Figure 4a, Figure 4b is the heat map without adding attention mechanism and Figure 4c is the heat map with adding Biformer attention mechanism. The comparison between Figure 4b,c shows that adding the attention mechanism will pay more attention to the seat belt area.



(**a**) (**b**) (**c**)

Figure 4. (**a**) The original image, (**b**) the heat map of the attention paid to the seat belt without adding the attention mechanism, and (**c**) the heat map of the attention paid to the seat belt with the added attention mechanism.

3.2.2. Lightweight Upsampling Operator CARAFE

The original YOLOv8 employs the nearest-neighbor interpolation method for upsampling in its feature fusion network. This approach solely relies on the spatial position of the pixel point to determine the upsampling kernel, neglecting the semantic information in the feature map. The method can be perceived as a uniform upsampling process that disregards

the potential impact of adjacent feature points, possessing a limited receptive field of 1 × 1. To overcome the limitations of the nearest neighbor interpolation upsampling method, this paper employs the CARAFE (Content-Aware ReAssembly of FEatures) operator [25] as a lightweight universal upsampling alternative. By replacing nearest-neighbor interpolation with CARAFE, the aim is to generate feature maps that contain more extensive semantic information. CARAFE offers improved performance in terms of preserving spatial details and capturing richer contextual information during the upsampling process. The specific structure of CARAFE is shown in Figure 5.

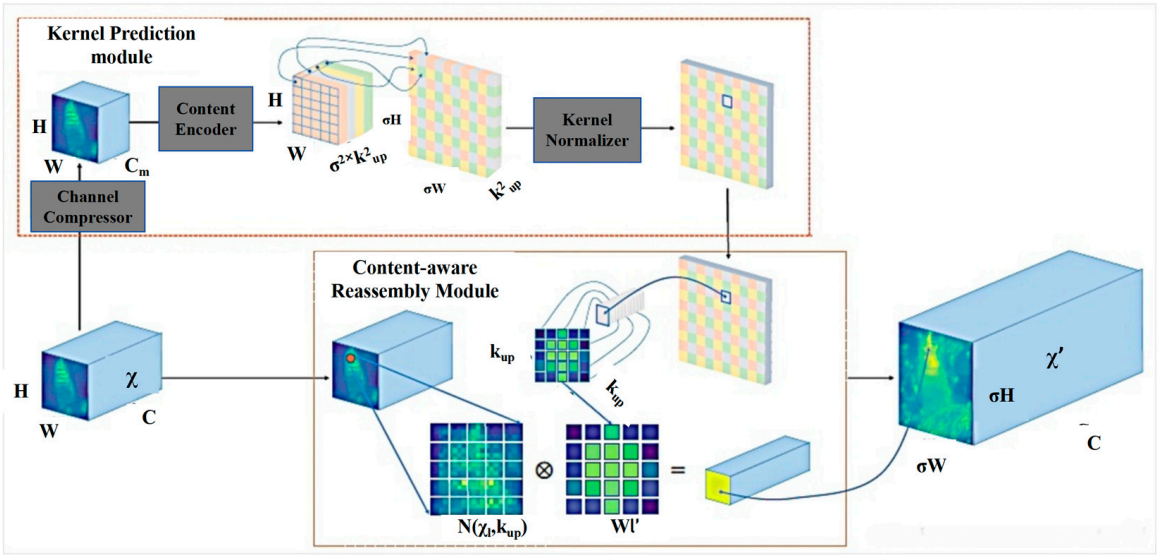


Figure 5. The specific structure of CARAFE.

CARAFE is divided into two main modules: the upsampling kernel prediction module and the content-aware reassembly module. Assuming an upsampling factor of σ, given an input feature map with a shape of $H \times W \times C$, CARAFE first uses the upsampling kernel prediction module to predict the upsampling kernel, and then uses the content-aware reassembly module to complete the upsampling, resulting in an output feature map with a shape of $\sigma H \times \sigma W \times C$. Below are the upsampling steps of CARAFE.

(1) The upsampling kernel prediction module:

The module is mainly responsible for generating upsampling recombination kernels in a content-aware manner so that the content-aware recombination module can complete the upsampling task.

Step 1: Feature Map Channel Compression

For an input feature map with the shape $H \times W \times C$, we first use a 1×1 convolution to compress its number of channels to $H \times W \times C_m$, and the main purpose of this step is to reduce the computational cost of subsequent steps.

Step 2: Content Coding and Upsampling Kernel Prediction

Assuming an upsampling kernel size of $k_{up} \times k_{up}$ (where a larger kernel size implies a wider receptive field and increased computational complexity), if we aim to utilize distinct upsampling kernels for each position in the output feature map, the predicted shape of the upsampling kernel would be denoted as $\sigma H \times \sigma W \times k_{up} \times k_{up}$.

In the initial step, the compressed input feature map is utilized, and a convolutional layer with a kernel size of $k_{encoder} \times k_{encode}$ is employed to predict the upsampling kernel. The input channel is set to C_m, and the output channel is $\sigma^2 k^2_{\text{up}}$. Subsequently, the channel dimension is expanded in the spatial dimension, resulting in an upsampling kernel with the shape $\sigma H \times \sigma W \times k^2_{up}$.

By following this process, CARAFE enables the use of diverse upsampling kernels tailored to different positions in the output feature map. This approach enhances the capacity to capture detailed features and improves the overall quality of the upsampling process.

Step 3: Upsampling Kernel Normalization

The upsampling kernels obtained in the second step using softmax should be normalized, ensuring that the weights of the convolutional kernels sum up to 1.

(2) The content-aware reassembly module:

The module primarily focuses on reassembling the features within local regions. Each position in the output feature map is mapped back to the input feature map, and a $k_{up} \times k_{up}$ region centered around that position is extracted. The dot product is then taken between this region and the predicted upsampling kernel for that specific point to obtain the output value. It is important to note that different channels at the same position share the same upsampling kernel.

3.2.3. Slim-Neck

To alleviate model complexity while maintaining accuracy, Li H, Li J, and Wei H et al. [26] proposed the GSConv module and designed the Slim-neck feature fusion network. In this paper, we replace the original Conv module in the neck layer of YOLOv8 with the GSConv module and substitute the C2f module with the VoVGSCSP module. We make these modifications because using the GSConv module in the backbone may lead to excessive computational complexity. However, in the neck layer, the feature maps have already become slender, and there is no longer a need for transformation. Therefore, by introducing the GSConv module in the neck layer, we strike a balance between computational efficiency and preserving the necessary feature transformations.

The GSConv module enhances the non-linear expressive power by introducing the DWConv layer and Shuffle operation. Introducing the GSConv module in the Backbone layer may lead to an increase in the number of network layers, resulting in a more complex model and a substantial increase in the inference computation time. However, when adopting the GSConv module in the neck layer, the feature map's channel dimension C has already reached its maximum value, and the height H and width W have reached their minimum values. Consequently, there is minimal redundant information and no need for compression. The structural diagram of the GSConv module is illustrated in Figure 6.

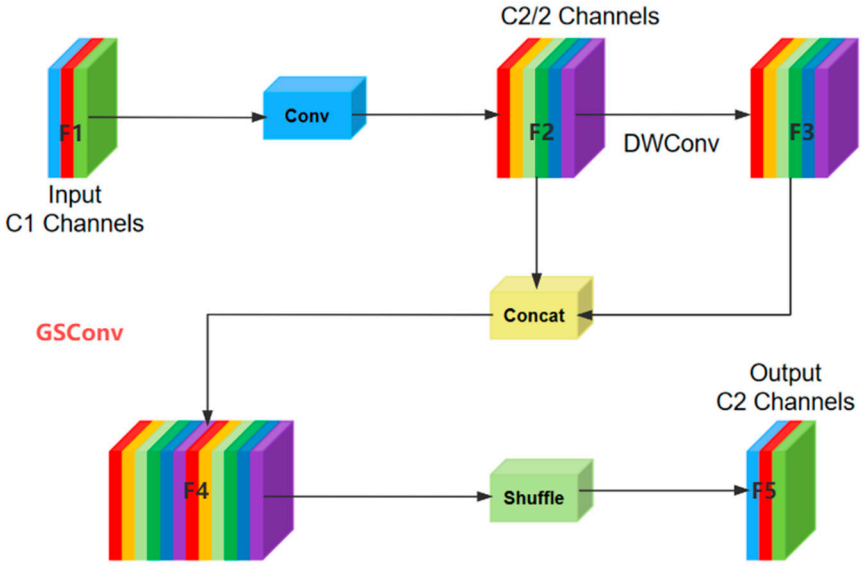


Figure 6. The structure of GSConv.

In the GSConv structure, the input feature map F_1 undergoes a downsampling operation using a 3×3 convolutional layer, resulting in the feature map F_2. Next, DWConv

is applied to F_2, generating the feature map F_3. F_2 and F_3 are then concatenated along the channel dimension, creating a new feature map F_4. Finally, the Shuffle operation is employed to shuffle the feature channels, resulting in the output feature map F_5. The computation process is illustrated in Equation (9):

$$F_{GSC} = Shuffle(Cat(\alpha(F_1)_{C_2/2}, \delta(\alpha(F_1)_{C_2/2})))_{C_2} \tag{9}$$

In this equation, F_1 represents the input feature map with a channel number of C_1. α represents the Conv operation, and δ represents the DWConv operation. F_{GSC} represents the output feature map obtained after the GSConv operation with a channel number of C_2.

To further reduce model complexity, a cross-stage partial network module called VoVGSCSP is designed utilizing the aggregation method inspired by ResNet's concept. The structure of VoVGSCSP is illustrated in Figure 7.

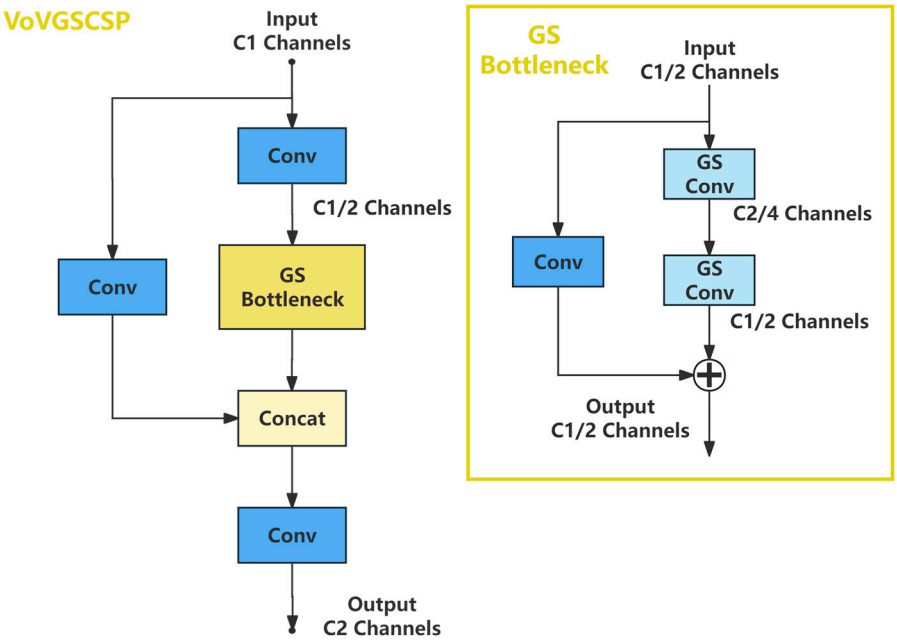


Figure 7. The structure of VoVGSCSP.

The VoVGSCSP module is designed as follows. Firstly, an initial feature extraction is performed on the input using a 1×1 convolutional layer, reducing the channel dimension to half of the original input. The resulting feature map is then fed into the GS Bottleneck. Within the GS Bottleneck, the residual concept is employed, where the input feature map undergoes two GSConv convolutions. The output of these convolutions is then added to a feature map obtained through a 1×1 convolutional layer, resulting in the module's output. At this stage, the channel dimension is $C1/2$. Subsequently, the input to the VoVGSCSP module is subjected to a 1×1 convolutional operation and concatenated with the output of the GS Bottleneck. Finally, the output is obtained by passing through a 1×1 convolutional layer, resulting in a channel dimension of $C2$. The formulas are shown as Equations (10) and (11):

$$GSB_{out} = F_{GSC}(F_{GSC}(\alpha(F_1)_{C_2})) + \alpha(F_1)_{C_1/2} \tag{10}$$

$$VoVGSCSP_{out} = \alpha(Concat(GSB_{out}, \alpha(F_1))) \tag{11}$$

In the equations, GSB_{out} represents the output of the GS Bottleneck, and $VoVGSCSP_{out}$ represents the final output of this module.

3.2.4. Auxiliary Training Heads

The main reason for adding the Aux head in YOLOv8 is to allow the intermediate layers of the network to learn more information and have rich gradient information to aid in training. Good gradient information can help a network with the same parameter count perform better. In YOLOv8, the method of adding the auxiliary head involves extracting some shallow-level features such as the Aux head, whereas the deep-level features, which are the final output of the network, serve as the Lead head, as shown in Figure 8a. During the loss calculation, the Lead head independently calculates its own loss, whereas the Aux head uses the positive samples obtained by matching with the Lead head (here, it involves coarse matching, selecting the neighboring grids of the central point of the ground truth box as the positive sample selection region) as its own positive samples, and calculates the loss accordingly. Finally, the losses are added together with the different weights, as shown in Figure 8b.

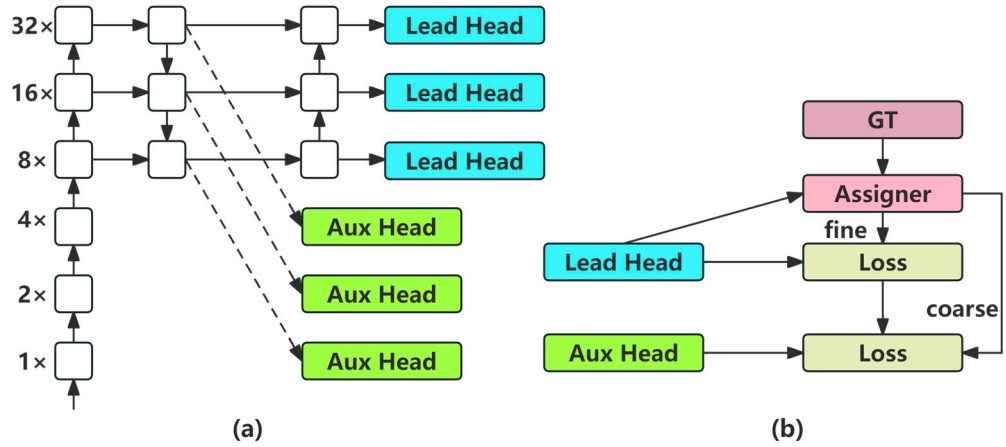


Figure 8. (**a**) Model with auxiliary head; (**b**) coarse-to-fine lead guided assigner.

3.2.5. Improvement of the Loss Function

In object detection, the accuracy of bounding boxes is of great importance as it directly affects the performance of detection algorithms. The traditional Intersection over Union (IoU) metric measures the overlap between two bounding boxes, but it may not be accurate in certain cases, especially when there is an overlap between objects or significant differences in bounding box sizes. In previous research, YOLOv8 employed the CIOU loss as a measure for bounding box evaluation to address this issue. The CIOU loss calculation takes into account the overlap area, center distance, and aspect ratio of the bounding boxes. However, the aspect ratio description in CIOU is a relative value, which introduces a certain level of ambiguity.

Taking inspiration from the collective features of horizontal rectangles, Siliang M et al. [27] designed a loss function for bounding box regression. It combines the concepts of Maximum Precision Distance (MPD) and Intersection over Union (IoU) to address the issue that most existing bounding box regression loss functions fail to optimize when the predicted box has the same aspect ratio as the ground truth box but significantly different width and height values. Moreover, it incorporates the relevant factors considered in existing loss functions, such as overlapping or non-overlapping regions, center point distance, and deviations in width and height. It simplifies the computation process by calculating the IoU by minimizing the point distance between the predicted bounding box and the ground truth bounding box.

This innovative loss function has garnered significant attention since its introduction due to its outstanding performance across multiple datasets. It considers the size, position, and degree of overlap of the objects, contributing to its remarkable performance. In comparison to the traditional Intersection over Union (IoU), MPDIOU provides a more accurate reflection of the relative positions and sizes of the objects, thereby enhancing

the accuracy of object detection. The calculation process of *MPDIOU* is illustrated in Equations (12)–(14):

$$d_1^2 = (x_1^B - x_1^A)^2 + (y_1^B - y_1^A)^2 \tag{12}$$

$$d_2^2 = (x_2^B - x_2^A)^2 + (y_2^B - y_2^A)^2 \tag{13}$$

$$MPDIOU = \frac{A \cap B}{A \cup B} - \frac{d_1^2}{w^2 + h^2} - \frac{d_2^2}{w^2 + h^2} \tag{14}$$

In Equations (9)–(11), A and B represent the predicted box and the ground truth box, respectively. The variables w and h represent the width and height of the input image. (x_1^A, y_1^A) and (x_2^A, y_2^A) represent the coordinates of the top-left and bottom-right points of the predicted bounding box, respectively. (x_1^B, y_1^B) and (x_2^B, y_2^B) represent the coordinates of the top-left and bottom-right points of the ground truth box, respectively. The calculated distances between the top-left points and bottom-right points of the predicted bounding box and the ground truth bounding box are denoted as d_1^2 and d_2^2, respectively. Finally, the optimization process aims to minimize the distances between the top-left and bottom-right points of the predicted bounding box and the ground truth bounding box. The final loss function for bounding box regression is represented by Equation (15):

$$L_{MPDIOU} = 1 - MPDIOU \tag{15}$$

MPDIOU simplifies the similarity comparison between two bounding boxes, aiding the algorithm in selecting the most suitable bounding box to accurately localize the target. By utilizing the *MPDIOU* loss function, the issues of overlapping anchor boxes, occlusion, and the removal of partially overlapping boxes during non-maximum suppression can be effectively addressed in object detection. This leads to a reduction in false negatives and effectively lowers the instances of missed detections.

4. Experiments and Analysis

4.1. Experimental Setting

In this study, we strictly controlled or fixed the experimental environment and its parameters. Other than the differences in methods, all other environments were consistent, and random seeds were controlled to make the results under different experimental conditions comparable. This allowed for the validation and replication of the experimental results, ensuring the credibility and reliability of the experimental outcomes. The configuration environment during the experiment is shown in Table 1, and the experimental parameters during the experiment are shown in Table 2.

Table 1. Experimental configuration.

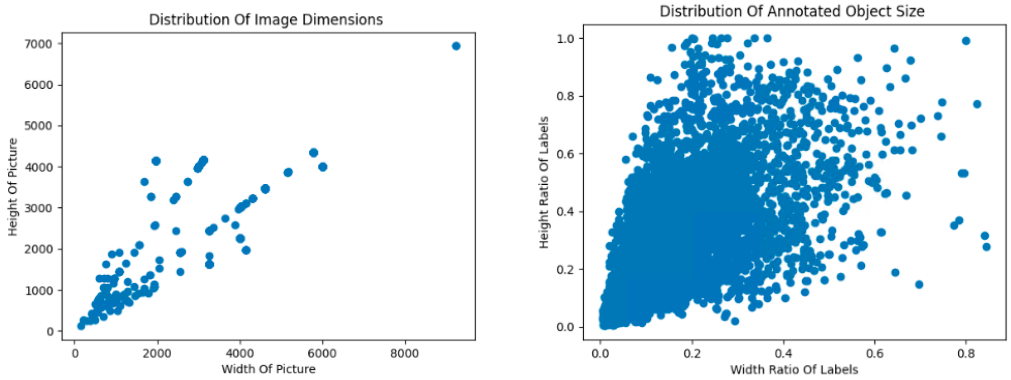
Name	Configure
Operating system	Windows 10
Processor	Intel(R) Xeon(R) W-2255 CPU @ 3.70GHz
Video card	NVIDIA GeForce RTX 3080Ti
Run memory	64GB
GPU internal storage	12GB
Programming tools	Pycharm
Programming language	python
Deep learning framework	Pytorch2.0.0

Table 2. Experimental parameters.

Parameter	Value
Learning	0.01
Batch Size	4
Epochs	300
Workers	4

4.2. Dataset

This experiment used a dataset mainly from the Tiachi competition held by Alibaba Cloud in July 2021, as well as relevant images downloaded from the Internet. It contained a total of 2546 images. The dataset's labels were divided into one category: "safety belt". The labeled images were divided into a training set of 2036 images, a validation set of 255 images, and a test set of 255 images. The Distribution of image dimensions as shown in Figure 9a, and distribution of annotated object size as shown in Figure 9b.



(**a**) (**b**)

Figure 9. (**a**) Scatter distribution of the size of the image in the dataset; (**b**) scatter distribution of the size of the object marked by the labels in the dataset.

4.3. Evaluation Metrics

In order to evaluate the performance of the algorithm, the evaluation metrics used in this study were precision (P), recall (R), and mean average precision (mAP).

Precision is defined from the perspective of the predicted results, indicating the proportion of positive samples among the samples with positive predicted results, i.e., the probability of detecting the target correctly among all detected targets. The calculation formula is as follows:

$$Precision = \frac{TP}{TP + FP} \tag{16}$$

Recall is a metric from the perspective of the sample, indicating the proportion of actual positive samples in the predicted positive samples and the entire positive samples, i.e., the probability of being correctly identified among all positive samples. The calculation formula is as follows:

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

Precision reflects the model's ability to distinguish between negative samples. The higher the precision, the stronger the model's ability to distinguish between negative samples. *Recall* reflects the model's ability to identify positive samples. The higher the recall rate, the stronger the model's ability to identify positive samples.

AP is a recognizer for a single category, and *AP* is the area enclosed by the *P*-*R* curve (*P* is the vertical axis and *R* is the horizontal axis). The calculation formula is as follows:

$$AP = \int_0^1 P(R)dR \tag{18}$$

Meanwhile, *mAP* is the average of *AP* from the category dimension, and *mAP* is the mean of the average precision *AP* of all categories, so it can evaluate the performance of multi-classifiers. Typically, *AP* is calculated for all images of each category when *IOU* = 0.5, and then the average is taken for all categories, that is, mAP@0.5. The calculation formula is as follows:

$$mAP = \frac{\sum_{i=1}^{m} AP_i}{m} \tag{19}$$

Among them, *TP* (true positives) refers to the number of positive samples correctly identified as positive by the model, i.e., the amount predicted correctly in the model; *FP* (false positives) refers to the number of negative samples incorrectly identified as positive by the model, i.e., the amount predicted incorrectly in the model; *FN* (false negatives) refers to the number of positive samples incorrectly identified as negative by the model, i.e., the number of positive samples that were misdetected by the model. m refers to the number of detection categories.

4.4. Comparative Analysis and Experimental Results

After conducting numerous experiments, the comparative results between the original YOLOv8 algorithm and the improved YOLOv8 algorithm are presented in Table 3.

Table 3. Comparison of detection performance before and after improvement.

Model	Precision/%	Recall/%	mAP@0.5/%	mAP@0.5:0.95%	Flops/G	Model Size/MB
before	0.929	0.909	0.962	0.765	8.1	5.9
after	0.98	0914	0.973	0.781	7.6	7.8

From Table 3, it is evident that the improved YOLOv8 algorithm outperforms the original YOLOv8 algorithm in terms of precision, recall, mAP@0.5, and mAP@0.5:0.95 for the safety belt dataset. The experimental results indicate that the precision value increased by 5.1%, the recall value increased by 0.5%, the mAP@0.5 value increased by 1.1%, and the mAP@0.5:0.95 value increased by 1.6%. Although the improved YOLOv8 model is slightly larger in size compared to the original YOLOv8 model, it actually reduced the floating-point operations. Overall, this improvement is reasonable as it has resulted in an overall enhancement in the detection performance of YOLOv8. In the future, further research will be conducted on techniques such as pruning and distillation to optimize and refine the model.

4.4.1. The Ablation Experiments

In order to validate the effectiveness of the proposed improved YOLOv8 algorithm and to comprehensively compare the performance of different models, ablation experiments were conducted. The purpose of these experiments was to compare the impact of adding or removing modules in the original model. The results of the ablation experiments are presented in Table 4.

During the experiment, a comparison was made among three of the most classic YOLO algorithms: YOLOv5s, YOLOv7-tiny, and YOLOv8n. These three algorithms represent the smallest models within YOLOv5, YOLOv7, and YOLOv8, with the lowest number of parameters and the smallest model sizes. From Table 3, it can be observed that YOLOv8 performed the best among the three algorithms. It has the highest mAP value in detection, the lowest floating-point operations, and the smallest model size. This indicates that YOLOv8 has an overall superior detection performance compared to YOLOv5 and

YOLOv7. Therefore, various optimization strategies and improvements were carried out using YOLOv8 as the foundation.

Table 4. Comparison of ablation experiments.

Model	Precision/%	Recall/%	mAP@0.5/%	Flops/G	Model Size/MB
yolov7-tiny	0.947	0.845	0.895	13	11.7
yolov5s	0.968	0.938	0.959	16.3	14.4
yolov8	0.929	0.909	0.962	8.1	5.9
yolov8 + Biformer	0.956	0.938	0.965	8.1	6.8
yolov8 + TripleAttention	0.908	0.938	0.957	8.1	6.2
yolov8 + Biformer + CARAFE + slim	0.944	0.93	0.968	7.6	6.4
yolov8 + Biformer + CARAFE + slim + Aux	0.949	0.92	0.969	7.6	7.8
yolov8 + Biformer + CARAFE + slim + Aux + mpdiou	0.98	0.914	0.973	7.6	7.8
yolov8 + Biformer + CARAFE + dyhead	0.964	0.926	0.971	9.8	7.9
yolov8 + Biformer + CARAFE + bifpn + Aux	0.942	0.91	0.969	7.4	6.1
yolov8 + Biformer + CARAFE + bifpn + Aux + mpdiou	0.944	0.908	0.967	7.4	6.1

Firstly, the addition of attention modules to the Backbone layer resulted in a slight improvement in detection performance. Then, in the Neck layer, lightweight modules such as the Slim-neck network and CARAFE were introduced, leading to an mAP value of 0.968. Subsequently, the introduction of the Aux-head auxiliary detection head in the Head layer resulted in a small improvement in detection performance. Many papers have utilized BiFPN feature pyramids in the feature fusion stage of the YOLO series, and experimental comparisons showed similar performance to our method. However, the experimental results demonstrated that the yolov8 + Biformer + CARAFE + slim + Aux + mpdiou optimization strategy achieved the best performance after introducing the MPDIOU loss function; moreover, replacing the loss function had almost no impact on the parameter count and size of the model.

4.4.2. Test the Effect of The Experimental Pictures

Based on the aforementioned experiments, we have obtained an optimized YOLOv8 model. Finally, we conducted tests on four identical images to observe the detection results of the original YOLOv8 model and the improved YOLOv8 model. Figure 10a represents the labeled image. Figure 10b shows the results predicted by the original model, and Figure 10c displays the results predicted by the improved model.

(**a**) Labeled of picture

Figure 10. *Cont.*

(**b**) The results of YOLOv8

(**c**) The results of improved YOLOv8

Figure 10. (**a**) The labeled picture, where the red rectangles represent the ground truth boxes. (**b**) The results predicted by the original model, where the red rectangles represent the predicted boxes and the numbers indicate the predicted probabilities. (**c**) The results predicted by the improved model.

From Figure 10, it is clear that the improved YOLOv8 model performs better in terms of detection compared to the original YOLOv8 model.

5. Conclusions

This paper proposes an improved YOLOv8 algorithm to enhance its detection performance, particularly for small objects. To achieve this, the algorithm introduces a BiFormer attention mechanism based on the Transformer architecture, which strengthens the feature extraction capability and improves the detection performance for seat belts. Additionally, the algorithm incorporates Slim-neck by replacing the traditional convolution modules (SC) with lightweight GSConv modules and replacing the C2f module with the VoV-GSCSP module based on the GSConv design. This not only reduces the model's parameter count

effectively but also mitigates the increased computational complexity caused by the BiFormer attention mechanism. Furthermore, it enhances the feature fusion ability of the neck layer, resulting in an overall performance improvement. The algorithm also introduces auxiliary detection heads, enabling the Head layer to learn more semantic information from the intermediate layers. Finally, the algorithm optimizes candidate boxes using the MPDIOU loss function. Experimental results on a seat belt detection dataset demonstrate that compared to the original YOLOv8n, the improved algorithm achieves better accuracy in seat belt detection while also showing overall performance improvements.

Further work will focus on lightweight optimization techniques such as model pruning and distillation. The goal is to minimize the model size and improve inference efficiency while maintaining high performance levels. The aim is to enhance the deployment efficiency and user experience on edge devices, mobile devices, and embedded systems.

Author Contributions: T.J.: Paper direction and suggestions. Z.L.: conceptualization, data curation, methodology, software, and supervision. Z.L., H.T., C.A., J.Z. and C.W.: writing–original draft, validation, and test. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Sun, S.; Zhao, J.; Fu, G. Study on behavioral causes of falling accidents based on "2-4" model. *China Saf. Sci. J.* **2019**, *29*, 23–28.
2. Diwan, T.; Anirudh, G.; Tembhurne, J.V. Object detection using YOLO: Challenges, architectural successors, datasets and applications. *Multimed. Tools Appl.* **2023**, *82*, 9243–9275. [CrossRef] [PubMed]
3. Deng, J.; Xuan, X.; Wang, W.; Li, Z.; Yao, H.; Wang, Z. A review of research on object detection based on deep learning. *J. Phys. Conf. Ser.* **2020**, *1684*, 012028. [CrossRef]
4. Hebbache, L.; Amirkhani, D.; Allili, M.S.; Hammouche, N.; Lapointe, J.-F. Leveraging Saliency in Single-Stage Multi-Label Concrete Defect Detection Using Unmanned Aerial Vehicle Imagery. *Remote Sens.* **2023**, *15*, 1218. [CrossRef]
5. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
6. Girshick, R. Fast R-CNN. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
7. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 91–99. [CrossRef] [PubMed]
8. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 386–397. [CrossRef] [PubMed]
9. Wang, T.; Anwer, R.M.; Cholakkal, H.; Khan, F.S.; Pang, Y.; Shao, L. Learning rich features at high-speed for single-shot object detection. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1971–1980.
10. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the 14th European Conference on Computer Vision, Amsterdam, The Netherland, 11–14 October 2016; pp. 21–37.
11. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
12. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
13. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
14. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
15. Mseddi, W.S.; Ghali, R.; Jmal, M.; Attia, R. Fire detection and segmentation using YOLOv5 and U-net. In Proceedings of the 2021 29th European Signal Processing Conference (EUSIPCO), Dublin, Ireland, 23–27 August 2021; pp. 741–745.
16. Chen, Q.; Wang, Y.; Yang, T.; Zhang, X.; Cheng, J.; Sun, J. You only look one-level feature. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 13034–13043.
17. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 7464–7475.
18. Reis, D.; Kupec, J.; Hong, J.; Daoudi, A. Real-Time Flying Object Detection with YOLOv8. *arXiv* **2023**, arXiv:2305.09972.

19. Terven, J.; Cordova-Esparza, D. A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond. *arXiv* **2023**, arXiv:2304.00501.
20. Feng, Z.; Zhang, W.; Zheng, Z. High-altitude operating seat belt detection based on Mask R-CNN. *Comput. Syst. Appl.* **2021**, *30*, 202–207. (In Chinese)
21. Zhang, M.; Han, Y.; Liu, Z. Detection method of high-altitude safety protective equipment for construction workers based on deep learning. *China Saf. Sci. J. (CSSJ)* **2022**, *32*, 140–146. (In Chinese)
22. Cao, J.; Guo, Z.B.; Pan, L.Z. Detection of Safety Belt Wearing in Aerial Work. *J. Hum. Univ. Sci. (Nat. Sci. Ed.)* **2022**, *37*, 92–99. (In Chinese)
23. Zhu, L.; Wang, X.; Ke, Z.; Zhang, W.; Lau, R.W. BiFormer: Vision Transformer with Bi-Level Routing Attention. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 10323–10333.
24. Gao, X.; Tang, Z.; Deng, Y.; Hu, S.; Zhao, H.; Zhou, G. HSSNet: A end-to-end network for detecting tiny targets of apple leaf diseases in complex backgrounds. *Plants* **2023**, *12*, 2806. [CrossRef] [PubMed]
25. Wang, J.; Chen, K.; Xu, R.; Liu, Z.; Loy, C.C.; Lin, D. Carafe: Content-aware reassembly of features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Long Beach, CA, USA, 16–20 June 2019; pp. 3007–3016.
26. Li, H.; Li, J.; Wei, H.; Liu, Z.; Zhan, Z.; Ren, Q. Slim-neck by GSConv: A better design paradigm of detector architectures for autonomous vehicles. *arXiv* **2022**, arXiv:2206.02424.
27. Ma, S.; Xu, Y. MPDIoU: A Loss for Efficient and Accurate Bounding Box Regression. *arXiv* **2023**, arXiv:2307.07662.