




Article

A Multiscale Parallel Pedestrian Recognition Algorithm Based on YOLOv5

Qi Song¹, ZongHe Zhou¹, ShuDe Ji^{1,*}, Tong Cui^{2,*}, BuDan Yao³ and ZeQi Liu⁴

¹ College of Aerospace Engineering, Shenyang Aerospace University, Shenyang 110136, China; songqi@stu.sau.edu.cn (Q.S.); zhouzonghe@stu.asu.edu.cn (Z.Z.)

² College of Artificial Intelligence, Shenyang Aerospace University, Shenyang 110136, China

³ College of Automation, Shenyang Aerospace University, Shenyang 110136, China; yaobodan@stu.sau.edu.cn

⁴ College of Electrical Engineering, Shanghai Dianji University, Shanghai 201306, China; 226001010412@st.sdju.edu.cn

* Correspondence: jishude@sau.edu.cn (S.J.); cuitong@stu.asu.edu.cn (T.C.)

Abstract: Mainstream pedestrian recognition algorithms have problems such as low accuracy and insufficient real-time performance. In this study, we developed an improved pedestrian recognition algorithm named YOLO-MSP (multiscale parallel) based on residual network ideas, and we improved the network architecture based on YOLOv5s. Three pooling layers were used in parallel in the MSP module to output multiscale features and improve the accuracy of the model while ensuring real-time performance. The Swin Transformer module was also introduced into the network, which improved the efficiency of the model in image processing by avoiding global calculations. The CBAM (Convolutional Block Attention Module) attention mechanism was added to the C3 module, and this new module was named the CBAMC3 module, which improved model efficiency while ensuring the model was lightweight. The WMD-IOU (weighted multidimensional IOU) loss function proposed in this study used the shape change between the recognition frame and the real frame as a parameter to calculate the loss of the recognition frame shape, which could guide the model to better learn the shape and size of the target and optimize recognition performance. Comparative experiments using the INRIA public data set showed that the proposed YOLO-MSP algorithm outperformed state-of-the-art pedestrian recognition methods in accuracy and speed.

Keywords: MSP; Swin Transformer; object recognition; YOLOv5 improvement



Citation: Song, Q.; Zhou, Z.; Ji, S.; Cui, T.; Yao, B.; Liu, Z. A Multiscale Parallel Pedestrian Recognition Algorithm Based on YOLOv5. *Electronics* **2024**, *13*, 1989. <https://doi.org/10.3390/electronics13101989>

Academic Editor: Daniel Riccio

Received: 4 March 2024

Revised: 18 April 2024

Accepted: 21 April 2024

Published: 20 May 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Pedestrian recognition is an important research direction in computer vision and has received widespread attention from researchers. However, the main challenges in this field are insufficient accuracy and real-time performance. Pedestrian recognition has two main methods: traditional methods and deep learning technology. Early traditional methods, such as the HOG+SVM proposed by Dalal et al. [1], focused on extracting pedestrian features through shallow learning techniques such as integrating and aggregating channel features [2,3]. However, traditional methods rely on manually designed feature extractors and have difficulty covering all useful information in the image. They have poor robustness and adaptability in the face of complex and changeable real-world environments. Moreover, the level of feature extraction is relatively simple, usually only involving low-level features of images, such as edges, corners, textures, etc., and lacks the ability to extract higher-level semantic information, which limits their performance in complex scenes. Deep learning technology automatically learns complex feature expressions through multilayer nonlinear transformation without manual intervention and automatically extracts rich feature information from large amounts of data. Deep networks can extract rich features from edges to high-level semantics through multilevel nonlinear transformations, greatly improving recognition accuracy and generalization capabilities, and they have attracted widespread attention.

Deep learning algorithms for object recognition are generally divided into two categories: two-stage networks and one-stage networks. Examples of two-stage networks are R-CNN [4], Fast R-CNN [5], and Faster R-CNN [6]. They first generate candidate region proposals and then classify these region proposals. This operation greatly improves the accuracy of recognition, but because of this, the computational load of the secondary network is often very large, resulting in poor real-time performance, especially in scenarios where high-resolution images are processed. This shortcoming led to the development and use of single-stage networks, which aimed to simplify the recognition process by eliminating a separate proposal generation step, predicting object classes, and using bounding boxes directly in a single pass of the network, thus significantly speeding up recognition. One-stage networks like YOLO [7–10] and SSD [11–15] directly predict object categories and locations, reducing some accuracy to speed up processing, making them more suitable for real-time applications. However, SSD uses multiscale feature maps to recognize objects of different sizes, which makes it difficult to see smaller objects on lower-resolution feature maps. In addition, SSD's method of object recognition, which relies on different layers of feature maps, is not as effective as methods such as the feature pyramid network (FPN) used in subsequent versions of YOLO [16], which has led to YOLO entering the mainstream of current single-target recognition algorithms [17–21].

With the development of deep learning, new target recognition algorithms based on self-attention mechanisms that are different from the two-stage region proposal and one-stage predefined anchor frame recognition methods have emerged. The introduction of the Transformer architecture in 2017 revolutionized the field of natural language processing (NLP) [22]. Inspired by this, the FAIR team used Transformer for object recognition, resulting in the Detr [23] algorithm. Released in 2020, Detr demonstrated competitive performance on the COCO data set. Detr uses the self-attention mechanism of the Transformer architecture to capture the global dependencies between different regions in the image, which enables the output of each position to take into account information from all locations in the image, which helps to locate and identify target objects more accurately. However, since the self-attention mechanism of the Transformer architecture essentially involves the self-attention mechanism of all elements in the sequence, this results in a large amount of calculation for attention, especially when processing high-resolution images and a large number of images, and the Transformer architecture capturing global context by processing the entire image. At the same time, this is effective for understanding the entire scene and reduces the focus on smaller objects in the image space. Therefore, Detr still faces some problems in terms of small targets and recognition speed [24–26].

YOLOv8 is the latest work in the YOLO series. It improves on its predecessor YOLOv5, integrating new features and improvements such as the addition of a decoupled head, classification loss as a VFL loss function, and a combination of regression losses DFL loss and CIOU loss. The emergence of pure self-attention deep networks [27] marks major progress in the field of object recognition, providing improvements in speed, performance, and versatility, but there are still some shortcomings in the missed recognition of targets and real-time performance. Therefore, this paper proposes a targeted model based on the YOLOv5 model; the above two aspects are improved, and the YOLO-MSP algorithm is proposed. Our contribution can be summarized as follows:

1. Fusing MSP modules, which, based on residual networks and multiscale parallel concepts in the neck regions, enhances the model's feature extraction, computational efficiency, and accuracy.
2. Fusing Swin Transformer modules in the backbone and neck regions enhances the model's feature extraction, computational efficiency, and accuracy.
3. The CBAM and C3 modules are integrated into the CBAMC3 module to replace the C3 module in the backbone to improve the recognition accuracy of the model.
4. The WMD-IOU loss function is introduced, designed to address the loss effects due to shape dimensions and the equalizing impacts of similar weights.

2. Materials and Methods

2.1. Network Architecture

The architecture of YOLOv5 adopts the CSPNet (cross-stage partial network) [28] architecture to improve computing efficiency. CSPNet is a neural network structure that reduces computational effort by dividing the network into two distinct parts (the backbone and the head). Therefore, the computational complexity is effectively reduced. The YOLOv5s network structure mainly includes four elements: input end, trunk, neck, and prediction end, as shown in Figure 1.

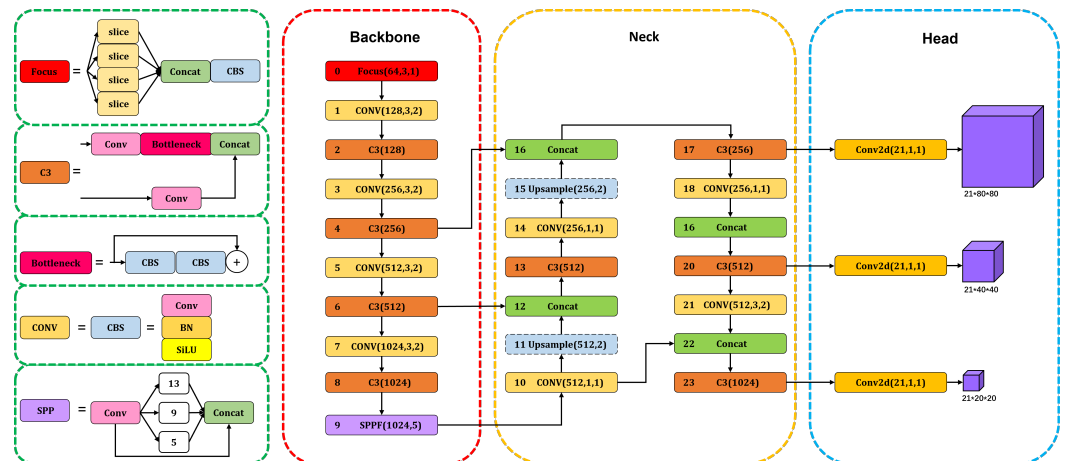


Figure 1. The original structure diagram of the YOLOv5s model.

We incorporated many improvements into the YOLOv5 model architecture. Inspired by the concept of residual networks, the MSP module is introduced in the model neck to increase extractable features and reduce training errors, thereby improving accuracy. The Swin Transformer modules have been integrated into the backbone and neck of the model. In addition, the CBAM attention mechanism and the C3 module are merged to form the CBAMC3 module, which replaces the original C3 module in the backbone, retains the lightweight design of the model, and improves model recognition accuracy. Figure 2 shows the updated YOLO-MSP model architecture.

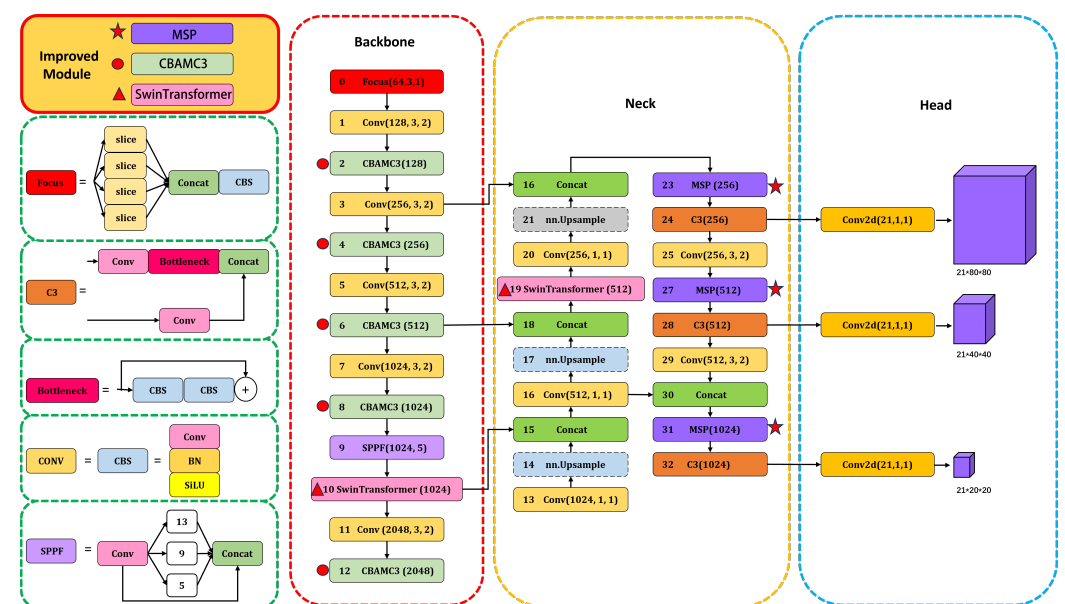


Figure 2. YOLO-MSP model structure diagram.

2.2. Method

In this section, we introduce the key improvements of the YOLO-MSP model to enhance its performance in object recognition tasks. Firstly, the MSP (multiscale parallel) module is integrated into the neck structure of the network to promote efficient information flow and improve multiscale feature extraction. Secondly, the Swin Transformer module is applied to the backbone and neck parts of the network, and a local window hierarchical attention mechanism is used to improve the accuracy and generalization of the model. Thirdly, the CBAM attention mechanism is integrated with the C3 module in the backbone structure to improve the recognition accuracy and generalization ability of the model. Finally, the WMD-IOU loss function is used to account for the shape proportion loss to achieve more accurate target positioning.

2.2.1. MSP Module

The MSP module is applied to the neck part of the network structure. The specific locations are the dark-purple modules numbered 23, 27, and 31 in Figure 2. These modules are marked with red stars.

The MSP module is designed based on residual network ideas and multiscale parallelism. The module uses the connections between layers to help information move quickly and efficiently in the network. As shown in Figure 3, the CM3 component of the MSP module contains three pooling layers, each with a different output channel. It can enhance the extraction capabilities of multiscale features while ensuring real-time performance, thereby processing multiscale data features and improving the accuracy of the model. The final adaptive pooling layer helps reduce the loss of input data features, enhances the model's ability to calculate complex data, and enhances the model's robustness.

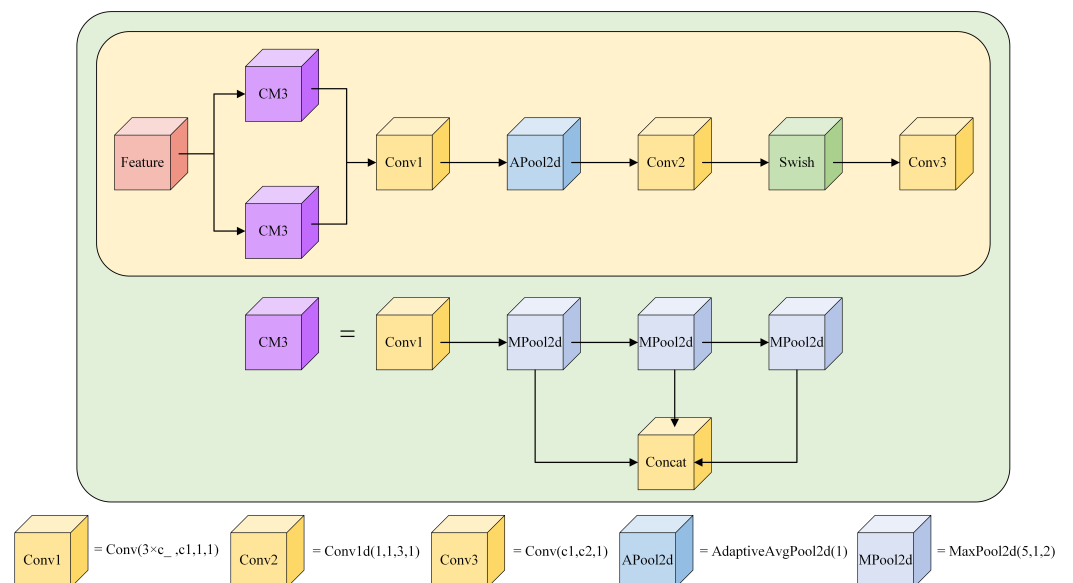


Figure 3. MSP module structure diagram.

The MSP module uses the Swish [29] activation function, which is continuously differentiable and smoother than the ReLU activation function. It helps to improve the smoothness of network model training. The smoother the activation function, the faster the convergence speed and the higher the learning rate. The formula of the Swish activation function is shown in Equation (1):

$$f(x) = x \times \text{sigmoid}(\beta x) \quad (1)$$

The graph of the Swish activation function is shown in Figure 4.

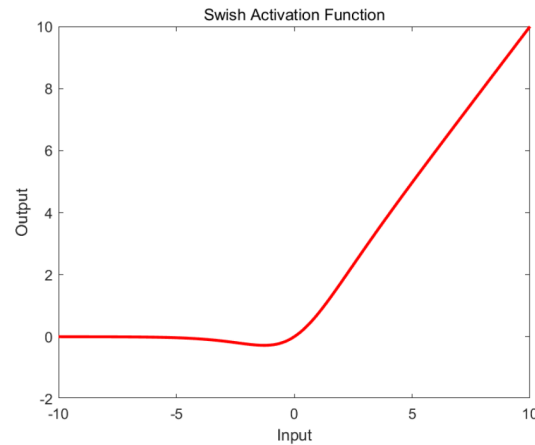


Figure 4. Swish activation function image.

2.2.2. Swin Transformer

The Swin Transformer module is applied to the backbone and neck parts of the network structure. The specific locations are the pink modules numbered 10 and 19 in Figure 2, which are marked with a red triangle in front of the modules. Compared with the Transformer structure, the Swin Transformer has a local window hierarchical attention mechanism and uses grouped convolution and cross-layer connections to enhance model performance, accuracy, and generalization capabilities. The Swin Transformer structure is shown in Figure 5.

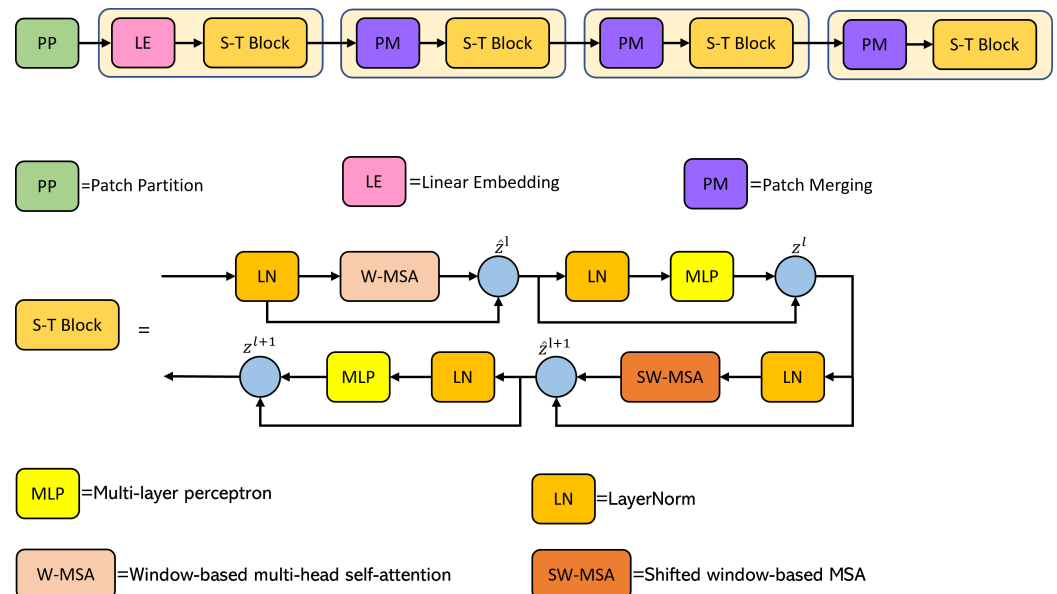


Figure 5. Swin Transformer structure diagram.

The input image is initially divided into patches by the patch partition module, with each patch containing 4×4 adjacent pixels, then flattened along the channel dimension, altering the image shape from $[H, W, 3]$ to $[H/4, W/4, 48]$. A linear embedding layer is applied to modify the channel dimension from 48 to C , transforming the image shape to $[H/4, W/4, C]$. Subsequently, four stages with varying feature map sizes are created.

The Swin Transformer module replaces the conventional multihead self-attention (MSA) [27] module, which utilizes a ‘window’ (W-MSA)- and ‘shifted window’ (SW-MSA)-based MSA module. This is followed by a two-layer perceptron (MLP) with ReLU [30] as the nonlinear function between layers. Layer normalization (LayerNorm) and residual connections are included before and after each MSA module and MLP layer.

To facilitate self-attention, the feature map X , represented as $X \in \mathbb{R}^{H \times W \times C}$, undergoes linear projection and reshaping to produce three arrays, Q , K , and V , each in $\mathbb{R}^{N \times C'}$, with N being the product of H and W . In this context, $\mathbb{R}^{N \times C'}$ denotes the feature space after an inevitable transformation, where N represents the total number of elements in the feature map, which is the product of the height H and the width W , while C' refers to the number of features or channels after transformation. These arrays, as transformed versions of X , are used as inputs to the self-attention mechanism, the results of which are calculated using Equations (2) and (3) [31–34].

$$\text{Attention} = TV \quad (2)$$

$$T = \text{SoftMax} \left(\frac{QK^T}{\sqrt{d}} + B \right) \quad (3)$$

Attention matrix $T \in \mathbb{R}^{N \times N}$ is used to characterize the interrelationships among feature map elements, accumulating global information in attention outputs. Absolute position encoding involves adding learnable parameters to each module before self-attention calculation. In contrast, relative position encoding adds parameters that are relative to the computation. The relative position offset, $B \in \mathbb{R}^{M^2 \times M^2}$, varies between $[-M + 1, M - 1]$ for each axis. Optimization is achieved using the W-MSA module. In Vision Transformers, while MSA allows self-attention calculations across all pixels, W-MSA restricts these calculations to adjacent pixels within 7×7 windows. The computational complexity Ω for the mean calculation in a local window of size $m \times m$ in the feature map $X \in \mathbb{R}^{H \times W \times C}$ is determined by Equations (4) and (5).

$$\Omega(\text{MSA}) = 4hwc^2 + 2(hw)^2C \quad (4)$$

$$\Omega(W - \text{MSA}) = 4hwc^2 + 2M^2hwC \quad (5)$$

2.2.3. CBAMC3

The CABMC3 module is applied to the backbone part of the network structure. This location is denoted by the light-green modules numbered 2, 4, 6, 8, and 12 in Figure 2. The front of these modules is marked with a red circle.

The CBAM attention mechanism consists of two submodules: the channel attention module focuses on extracting important information and effectively reducing the space complexity of the feature map while keeping the number of channels consistent. This enables the model to prioritize the most informative features, effectively identifying parts of objects that are still visible despite occlusions. At the same time, the spatial attention module specializes in pinpointing the precise location of objects by analyzing the spatial distribution of features without changing the number of channels. This targeted approach helps the model focus on areas where occluding objects may be present, thereby improving recognition accuracy. Integrating CBAM with the C3 module in the model architecture significantly improves the model's ability to discern important features under different conditions without increasing the total parameters. The cooperation between channel and spatial attention enables the model to retain critical information, thereby improving its robustness and accuracy in scenes where objects are partially hidden. The CBAM module is shown in Figure 6, and the CBAMC3 module is shown in Figure 7.

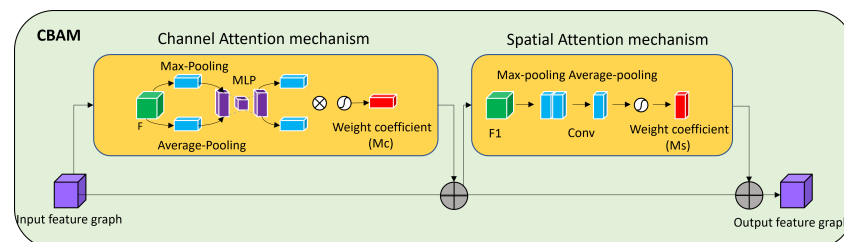


Figure 6. CBAM module structure diagram.

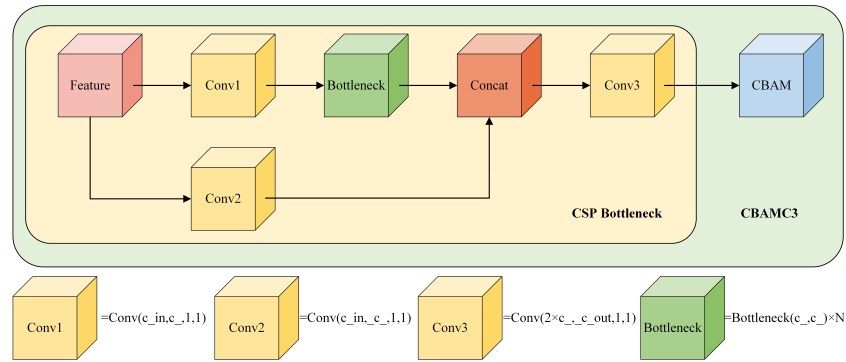


Figure 7. CBAMC3 module structure diagram.

The CBAM module takes the input feature map F with a size of $H \times W \times C$ and consists of two parts: the channel attention mechanism and the spatial attention mechanism. In the channel attention mechanism, the input feature map F is processed with maximum and average pooling to produce two feature maps of $1 \times 1 \times C$ size. These two feature maps are passed through an MLP and a sigmoid activation function to obtain a one-dimensional channel attention map M_c . Multiplying the channel attention map with the original feature map F yields the channel-attention-adjusted feature map F_1 . In the spatial attention mechanism, the feature map F_1 is subjected to maximum and average pooling again, resulting in two feature maps of size $H \times W \times 1$. The two feature maps are concatenated and then convolved to generate a two-dimensional spatial attention map M_s . Finally, multiplying the spatial attention map M_s with the feature map F_1 yields the output feature map of the CBAM module. This design can simultaneously focus on features in different channels and locations, thus improving the model's accuracy.

2.2.4. WMD-IOU Loss Function

The initial loss function used by the YOLOv5 model is the CIOU loss function. The CIOU loss for measuring similarity between two arbitrary shapes $C, C_i \in S \in \mathbb{R}^n$ is calculated as follows:

$$L_{CIOU} = 1 - \frac{|C \cap C_i|}{|C \cup C_i|} + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha \nu \quad (6)$$

$$\alpha = \frac{\nu}{1 - IOU + \nu} \quad (7)$$

$$\nu = \frac{4}{\pi^2} \left(\arctan\left(\frac{w^{gl}}{h^{gl}}\right) - \arctan\left(\frac{w}{h}\right) \right)^2 \quad (8)$$

The aspect ratio measurement in CIOU [35] is complex, leading to its slow convergence. Yi-Fan Zhang and colleagues consider decomposing the aspect ratio loss into the difference between the predicted width and height and the minimum bounding box, thereby speeding up convergence and improving regression accuracy. In addition, they introduced focal loss to solve the sample imbalance problem in the bounding box regression task. This approach reduces the optimization contribution of low-quality anchor boxes that least overlap with the target box, focusing the regression process on other high-quality anchor boxes. The formula of the EIOU [36] loss function is as follows:

$$L_{EIOU} = 1 - \frac{|C \cap C_i|}{|C \cup C_i|} + \frac{\rho^2(b, b^{gt})}{(h^c)^2 + (w^c)^2} + \frac{\rho^2(h, h^{gt})}{h^c} + \frac{\rho^2(w, w^{gt})}{w^c}. \quad (9)$$

Although EIOU considers the overlap loss, center distance loss, and aspect ratio loss between frames, it does not fully account for changes in the target box shape ratio. To overcome this problem, YOLO-MSP introduces WMD-IOU (weighted multidimensional IOU loss function) with the shape ratio of the target and anchor boxes as the weighting

factor to enhance the bounding box regression in object recognition, which makes the difference in shape ratio directly affect the size of the loss value and guides the model to better match the shape proportions. Additionally, it provides a personalized approach to the impact of loss factors, allowing for customized adjustments to different data sets. The formula of the WMD-IOU loss function is as follows:

$$L_{WMD-IOU} = 1 + (\alpha IOU(A, B) + \beta \frac{\rho^2(b, b^{gt})}{(h^c)^2 + (w^c)^2} + \delta \frac{\rho^2(h, h^{gt})}{h^c} + \lambda \frac{\rho^2(w, w^{gt})}{w^c} + \mu \sum_{\omega t = \Delta w, \Delta h} (1 - e^{\omega t})) \quad (10)$$

$$\Delta w, \Delta h = \frac{w}{w^{gt}}, \frac{h}{h^{gt}} \quad (11)$$

$$\mathcal{L}_{alliou} = IOU^\gamma L_{iou} \quad (12)$$

where $\alpha, \beta, \Delta, \lambda, \mu$ are weight influencing factors, tailored according to the characteristics of the data set, with a weight range of $(0, 1)$. $\Delta w, \Delta h$ represent the shape ratios between the predicted box and the anchor box, addressing losses attributed to shape differences and enhancing loss diversity.

In this formula, $\mathcal{L}_{alliou}, L_{iou}$ represent the final and optimized loss functions, respectively. The additional γ parameter is introduced to address the issue of sample imbalance. By adjusting γ , loss contributions from specific samples can be amplified or reduced. This allows γ to be customized based on sample characteristics, making the loss function target key samples more effectively. After integration with focal loss, the final framework thoroughly addresses the impacts of sample imbalance by considering the loss contributions of challenging samples. The summation $\sum_{\omega t = \Delta w, \Delta h} (1 - e^{\omega t})$ in the formula ensures careful consideration of differences in width and height. The use of the exponential function is because, in $1 - e^{\omega t}$, as ωt increases, $e^{\omega t}$ rapidly approaches or exceeds 1, which results in a decrease in the value of the entire expression, thus reducing the contribution of this part of the loss. This design means that when the difference in shape between the predicted box and the initial box is small, the loss increases, thereby encouraging the model to better learn to minimize such discrepancies. Additionally, it allows for a smoother adjustment of the loss function's sensitivity to shape differences, avoiding excessive perturbation to other aspects. The explanatory diagram for WMD-IOU is illustrated in Figure 8.

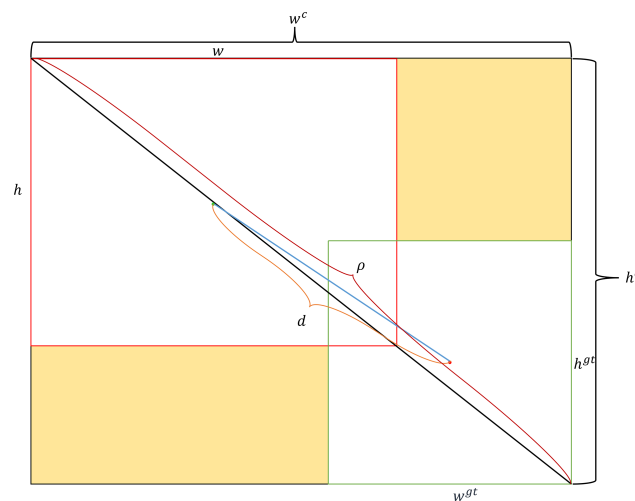


Figure 8. WMD-IOU loss function boundary regression diagram. The yellow area represents the overall image size, the red line area represents the real box size, the green line area represents the predicted box size, and the blue line represents the offset between the predicted box and the real box.

In summary, the WMD-IOU loss function incorporates comprehensive error measurement and enhanced shape sensitivity coupled with adjustable weight coefficients. It can enable more precise target localization and increased adaptability to variations in target size and shape.

3. Experiment

3.1. Experiment Environment

In this experiment, the software environment used was Pytorch 1.12.0, and the hardware included Intel (R) Core(TM) i7-11700K and NVIDIA GeForce RTX 3080Ti, with 16 GB of memory and Ubuntu operating system installed, as shown in Table 1. The parameters used for training included a weight decay coefficient of 0, a batch size of 16, and an epoch of 300. In order to avoid overfitting, a label smoothing method was used to optimize the image classification labels with a smoothing coefficient of 0.01.

3.2. Dataset Details

This study uses the INRIA public data set [37] collected by Navneet Daal. The INRIA data set is a widely used collection of labeled images of standing or walking pedestrians, designed to identify upright pedestrians in pictures and videos. It includes a training set consisting of 614 positive samples (1237 pedestrians) and 1218 negative samples, and a test set consisting of 288 positive samples (589 pedestrians) and 453 negative samples. This data set has the characteristics of complex background, diverse human postures, and diverse environments. In order to improve model performance, this experiment selected 200 simulated pedestrian data from the FOG part of the CrowdSim2 [38] data set and added them to the INRIA data set. The ratio of the experimental and training set, validation set, and test set of this study is 6:2:2.

3.3. Evaluation Metrics

This experiment uses evaluation indicators commonly used in deep learning object recognition, precision (P), recall (R), average precision (AP), and mean average precision (mAP) as evaluation indicators. P and R are defined using true positive (TP), false positive (FP), true negative (TN), and false negative (FN) as

$$P = \frac{TP}{TP + FP} \quad (13)$$

$$R = \frac{TP}{TP + FN} \quad (14)$$

P-R curves are drawn according to precision P and recall R, and AP values are calculated according to the area:

$$AP = \int_0^1 P(R) dR \quad (15)$$

The mAP value is calculated based on the average of all AP in N categories

$$mAP = \frac{1}{N} \sum AP \quad (16)$$

The F1 value can be calculated according to the precision P and the recall R

$$F1 = 2 \frac{P \cdot R}{P + R} \quad (17)$$

3.4. Experimental Results

This study uses the YOLO-MSP model to conduct experiments on the INRIA public data set and compares the results with those of other representative models. As shown in Table 1, YOLO-MSP outperforms its peers with a maximum average precision (mAP)

of 96% and an F1 value of 91%. Compared with YOLOv8s, YOLO-MSP's mAP has increased by 0.8%, its inference speed has increased by 0.4 ms, and its recognition frame rate has increased by 1 FPS. Notably, YOLO-MSP shows superior performance in identifying unlabeled small objects in raw images, with smaller center points and size deviations compared to other models. In the table, best and worst results are displayed in red and green, respectively.

Table 1. Comparison results for YOLOv5s, YOLOv7-tiny, Detr, YOLOv8s, and YOLO-MSP.

Network	Precision (%)	Recall (%)	mAP@IoU = 0.5 (%)	F1 Score	Inference Speed per Image (ms)	FPS
YOLOv5s	0.875	0.930	0.921	0.90	28.1	35
YOLOv7-tiny	0.918	0.875	0.937	0.89	27.8	36
Detr	0.921	0.883	0.939	0.90	26.9	37
YOLOv8s	0.925	0.909	0.952	0.92	26.5	37
YOLO-MSP	0.936	0.889	0.960	0.91	25.3	39

We plotted the precision–recall (P–R) of YOLOV8S and YOLO-MSP and the comparison curves of map@0.5 and mAP@0.5:0.95. Through these images, we can intuitively observe the process in which various indicators of the model gradually stabilize and improve as training proceeds. Among them, the YOLO-MSP precision–recall (P–R) improvement is more significant, and the map value is higher than that of YOLOV8S. The left side of the picture is the YOLOv8S model, and the right is the YOLO-MSP model. The images are shown in Figure 9 and Figure 10, respectively.

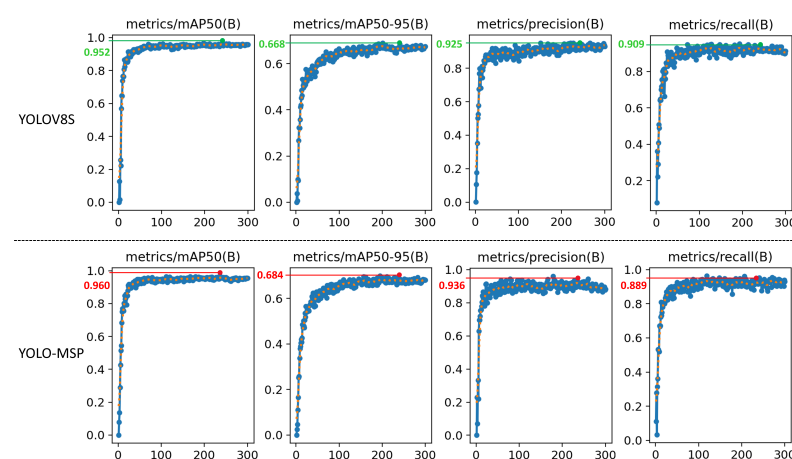


Figure 9. Training indicator trend chart of YOLOV8S and YOLO-MSP. The horizontal axis represents the number of training iterations, and the vertical axis represents the corresponding indicator value. The figure from left to right shows average precision (mAP@0.5 and mAP@0.5:0.95) and recall rate (precision/recall).

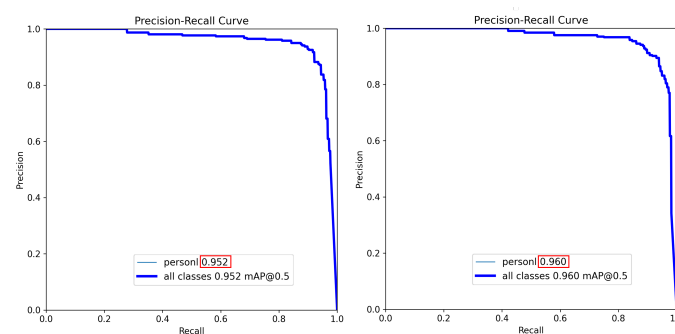


Figure 10. P–R curves of YOLOV8S and YOLO-MSP algorithms.

We spliced the recognition frame results of different algorithms on the same picture in the INRIA public data, as shown in Figure 11. It can be seen that compared with other algorithms, YOLO-MSP not only has a high intersection over union (IoU) but also can recognize small targets that are not labeled in the original data set. The overall map@0.5 value of the recognition frame and the inference speed per image (ms) value of different algorithms on the same image are shown in Table 2. In the table, best and worst results are displayed in red and green, respectively.



Figure 11. The recognition frame results of different algorithms on the INRIA public data set, from left to right, are the real frame, YOLOv5s, YOLOv7-tiny, Detr, YOLOv8S, and YOLO-MSP.

Table 2. Comparative analysis of object recognition algorithms.

	YOLOv5s		YOLOv7-tiny		Detr		YOLOv8S		YOLO-MSP	
	mAP	Time (ms)	mAP	Time (ms)	mAP	Time (ms)	mAP	Time (ms)	mAP	Time (ms)
Figure 1	0.902	31.3	0.915	28.7	0.913	27.2	0.937	27.0	0.909	26.3
Figure 2	0.911	30.8	0.929	31.2	0.928	26.3	0.955	25.8	0.974	25.1
Figure 3	0.919	31.6	0.924	30.5	0.923	26.9	0.949	26.4	0.963	25.3
Figure 4	0.807	32.5	0.811	31.8	0.820	28.7	0.820	27.8	0.730	27.0

3.5. Low-Brightness Experiments

The YOLO algorithm has a poor target recognition rate under low-brightness conditions. Therefore, this study used some data from the KAIST [39] data set to conduct low-brightness experiments to compare the recognition performance differences between the improved model and other models. The INRIA data set was used in all model training processes, and evaluation was performed every 10 training cycles. A total of 300 epochs were trained in the experiment. According to the recognition results, it can be seen that this model has a better recognition rate than other models under low brightness conditions, and the mAP value can meet the needs of pedestrian recognition at night. The experimental results are shown in Figure 12.

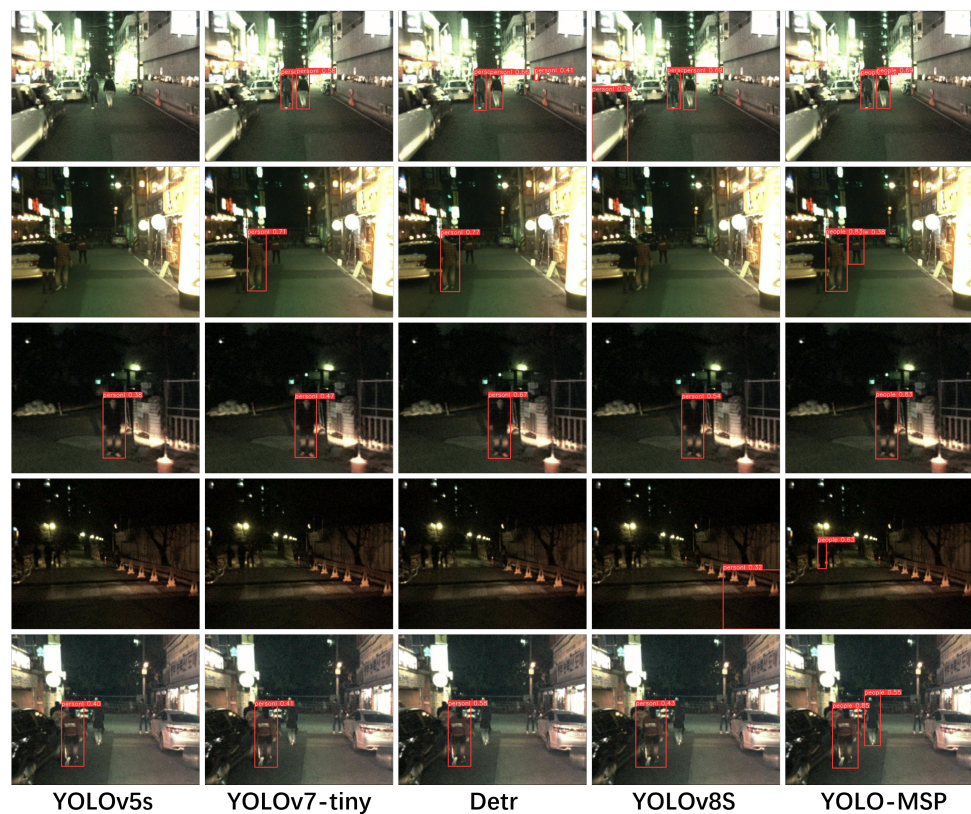


Figure 12. The recognition frame results of different algorithms on the KAIST public data set, from left to right, are the YOLOv5s, YOLOv7-tiny, Detr, YOLOv8S, and YOLO-MSP.

3.6. Ablation Experiments

Ablation experiments were conducted in this study to compare the performance difference between the improved and other models. Label smoothing was used to process images during the training, and evaluation was conducted every ten training epochs. A total of 300 epochs were trained in the experiment, and the results are presented in Table 3. In the table, best and worst results are displayed in red and green, respectively.

Table 3. Experimental platform configuration.

Model	Swin Transformer	MSP	CBAMC3	WMD-IOU	mAP(%)	F1
Model0					0.919	0.88
Model1	✓				0.928	0.89
Model2	✓	✓			0.935	0.89
Model3	✓	✓	✓		0.947	0.90
YOLO-MSP	✓	✓	✓	✓	0.960	0.91

3.7. Hyperparameter Tuning

Manual tuning was performed to obtain the best hyperparameters for the YOLO-MSP model and observe its performance differences on a self-made data set due to the effects of different hyperparameter settings on the recognition performance of YOLO-MSP. The experimental results are shown in the table. The experiments proved that the model achieved the best performance when using a batch size of 16 and an SGD optimizer and when scaling the image size to 1280×1280 . We believe that a scale size of 1280 can better preserve the feature information of small objects in the images, thus improving the recognition accuracy. The experimental results are presented in Table 4. In the table, best and worst results are displayed in red and green, respectively.

Table 4. Experimental results of YOLO-MSP under different hyperparameters.

Methods	Image Size	Batch Size	Optimizer	mAP (%)
YOLO-MSP	1280	16	SGD	0.960
YOLO-MSP	1280	8	SGD	0.943
YOLO-MSP	1280	32	SGD	0.939
YOLO-MSP	1024	16	SGD	0.921
YOLO-MSP	640	16	SGD	0.926

3.8. Crossover Experiments

In this study, to evaluate the model's robustness, we divided the entire data set into ten mutually exclusive subsets. Then, we adopted the five-fold cross-validation method: two subsets were selected as the test set each time, and the remaining eight subsets were used as the training set. In this way, we observed the model's performance changes under different training sets.

Experimental results show that the model's average accuracy (mAP) on different data sets reaches a maximum of 0.960 and a minimum of 0.947; the highest value of the F1 score is 0.91, and the lowest value is 0.89. These results show that despite changes in training data, the model's performance remains at a high level, indicating that the model is robust. In addition, experiments also show that the model's real-time performance remains unchanged on different data sets, further verifying the stability of the model. The experimental results are presented in Table 5.

Table 5. Experimental results of YOLO-MSP under different crossover settings.

Model	Precision (%)	Recall (%)	mAP@IoU = 0.5 (%)	F1-Score	Inference Speed per Image (ms)	FPS
Model1	0.931	0.871	0.958	0.90	25.3	39
Model2	0.939	0.882	0.959	0.91	25.3	39
Model3	0.936	0.867	0.959	0.90	25.2	39
Model4	0.931	0.871	0.947	0.90	25.1	39
Model5	0.936	0.889	0.960	0.91	25.3	39

4. Conclusions

The YOLO-MSP algorithm proposed in this study uses the MSP module based on residual networks and multiscale parallel concepts to enhance pedestrian feature recognition. The local attention mechanism of the Swin Transformer can reduce the amount of calculation and improve the real-time performance of the algorithm. The CABMC3 module is formed by merging the CBAM and C3 modules to improve the recognition accuracy of the algorithm. The WMD-IOU loss function takes into account the shape difference loss, thereby improving the recognition accuracy of the target. However, since the algorithm reduces a certain accuracy rate in order to pursue a higher recall rate, the F1 value cannot fully reflect the overall advantages of the model. The YOLO-MSP algorithm is very suitable for application scenarios with a large amount of pedestrian traffic and requiring powerful real-time detection capabilities. This is of particular value for autonomous driving systems that require accurate and timely recognition of pedestrians. In the future, our research will combine more advanced attention mechanisms and loss functions to further improve detection accuracy without affecting the real-time performance of the algorithm, focusing on fine-tuning the balance between precision and recall to improve the F1 score. In addition, we plan to apply the algorithm to various environments and test its effectiveness under different weather and lighting conditions, which is crucial for its application in real scenarios.

Author Contributions: Conceptualization, Q.S., Z.Z., S.J., T.C. and Z.L.; methodology, Q.S., Z.Z., S.J., T.C. and B.Y.; software, Q.S., Z.Z., T.C. and Z.L.; formal analysis, Q.S., Z.Z., S.J., B.Y. and Z.L.; investigation, Q.S., Z.Z., S.J. and B.Y.; resources, Q.S., Z.Z., S.J. and B.Y.; data curation, Q.S., Z.Z., B.Y. and Z.L.; writing—original draft preparation, Z.Z.; writing—review and editing, Q.S., Z.Z., S.J., T.C. and B.Y.; visualization, Z.Z. and T.C.; supervision, Q.S., S.J. and T.C.; project administration, Q.S. and

S.J.; funding acquisition, Q.S. and S.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by The State Key Laboratory of Robotics grant number 2023-O04.

Data Availability Statement: Data are contained within the article.

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

References

- Dalal, N.; Triggs, B. Histograms of oriented gradients for human recognition. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 886–893.
- Dollár, P.; Tu, Z.; Perona, P.; Belongie, S. Integral channel features. In Proceedings of the British Machine Vision Conference, BMVC 2009, London, UK, 7–10 September 2009.
- Dollár, P.; Appel, R.; Belongie, S.; Perona, P. Fast feature pyramids for object recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 1532–1545. [CrossRef] [PubMed]
- LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
- Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object recognition with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 1–9.
- He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
- Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object recognition. *arXiv* **2020**, arXiv:2004.10934.
- Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
- Ultralytics. Yolov5. Available online: <https://github.com/ultralytics/yolov5> (accessed on 18 October 2022).
- Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. Scaled-yolov4: Scaling cross stage partial network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13029–13038.
- Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox recognizer. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016*; Springer International Publishing: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
- Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. Dssd: Deconvolutional single shot recognizer. *arXiv* **2017**, arXiv:1701.06659.
- Shen, Z.; Liu, Z.; Li, J.; Jiang, Y.G.; Chen, Y.; Xue, X. Dsod: Learning deeply supervised object recognizers from scratch. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1919–1927.
- Jeong, J.; Park, H.; Kwak, N. Enhancement of SSD by concatenating feature maps for object recognition. *arXiv* **2017**, arXiv:1705.09587.
- Li, Z.; Zhou, F. FSSD: Feature fusion single shot multibox recognizer. *arXiv* **2017**, arXiv:1712.00960.
- Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
- Xie, H.; Xiao, Z.; Liu, W.; Ye, Z. PVNet: A Used Vehicle Pedestrian recognition Tracking and Counting Method. *Sustainability* **2023**, *15*, 14326. [CrossRef]
- Lan, W.; Dang, J.; Wang, Y.; Wang, S. Pedestrian recognition based on YOLO network model. In Proceedings of the 2018 IEEE International Conference on Mechatronics and Automation (ICMA), Changchun, China, 5–8 August 2018; pp. 1547–1551.
- Yang, X.; Wang, Y.; Laganieri, R. A scale-aware YOLO model for pedestrian recognition. In *Advances in Visual Computing: 15th International Symposium, ISVC 2020, San Diego, CA, USA, 5–7 October 2020*; Springer International Publishing: Berlin/Heidelberg, Germany, 2020; pp. 15–26.
- Krišto, M.; Ivacic-Kos, M.; Pobar, M. Thermal object recognition in difficult weather conditions using YOLO. *IEEE Access* **2020**, *8*, 125459–125476. [CrossRef]
- Xue, Y.; Ju, Z.; Li, Y.; Zhang, W. MAF-YOLO: Multi-modal attention fusion based YOLO for pedestrian recognition. *Infrared Phys. Technol.* **2021**, *118*, 103906. [CrossRef]
- Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Online, 16–20 November 2020; pp. 38–45.
- Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; Dai, J. Deformable Detr: Deformable transformers for end-to-end object recognition. *arXiv* **2020**, arXiv:2010.04159.
- Lin, M.; Li, C.; Bu, X.; Sun, M.; Lin, C.; Yan, J.; Ouyang, W.; Deng, Z. Detr for crowd pedestrian recognition. *arXiv* **2020**, arXiv:2012.06785.
- Pu, Y.; Liang, W.; Hao, Y.; Yuan, Y.; Yang, Y.; Zhang, C.; Hu, H.; Huang, G. Rank-Detr for high quality object recognition. *Adv. Neural Inf. Process. Syst.* **2024**, *36*, 1–14.
- Srinivasan, A.; Srikanth, A.; Indrajit, H.; Narasimhan, V. A novel approach for road accident recognition using Detr algorithm. In Proceedings of the 2020 International Conference on Intelligent Data Science Technologies and Applications (IDSTA), Valencia, Spain, 19–22 October 2020; pp. 75–80.

27. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
28. Wang, C.Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. Cspnet: A new backbone that can enhance learning capability of CNN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1571–1580.
29. Ramachandran, P.; Zoph, B.; Le, Q.V. Searching for Activation Functions. *arXiv* **2017**, arXiv:1710.05941.
30. Glorot, X.; Bordes, A.; Bengio, Y. Deep sparse rectifier neural networks. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; JMLR Workshop and Conference Proceedings; pp. 315–323.
31. Hu, H.; Gu, J.; Zhang, Z.; Dai, J.; Wei, Y. Relation networks for object recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3588–3597.
32. Hu, H.; Zhang, Z.; Xie, Z.; Lin, S. Local relation networks for image recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3464–3473.
33. Bao, H.; Dong, L.; Wei, F.; Wang, W.; Yang, N.; Liu, X.; Wang, Y.; Gao, J.; Piao, S.; Zhou, M.; et al. Unilmv2: Pseudo-masked language models for unified language model pre-training. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 642–652.
34. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 5485–5551.
35. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12993–13000.
36. Zhang, Y.F.; Ren, W.; Zhang, Z.; Jia, Z.; Wang, L.; Tan, T. Focal and efficient IOU loss for accurate bounding box regression. *Neurocomputing* **2022**, *506*, 146–157. [[CrossRef](#)]
37. Dalal, N.; Triggs, B. INRIA Person Dataset. 2020. Available online: <https://paperswithcode.com/dataset/inria-person> (accessed on 20 April 2024).
38. Foszner, P.; Szczesna, A.; Ciampi, L.; Messina, N.; Cygan, A.; Bizoń, B.; Cogiel, M.; Golba, D.; Macioszek, E.; Staniszewski, M. CrowdSim2: An open synthetic benchmark for object recognizeors. *arXiv* **2023**, arXiv:2304.05090.
39. KAIST Multispectral Pedestrian Detection Benchmark. 2015. Available online: <https://paperswithcode.com/dataset/kaist-multispectral-pedestrian-detection> (accessed on 20 April 2024).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.