Article

Edge-YOLO: Lightweight Infrared Object Detection Method Deployed on Edge Devices

Junqing Li and Jiongyao Ye *

School of Information Science and Engineering, East China University of Science and Technology, Shanghai 200237, China; lijq2020@outlook.com
* Correspondence: yejy@ecust.edu.cn

Abstract: Existing target detection algorithms for infrared road scenes are often computationally intensive and require large models, which makes them unsuitable for deployment on edge devices. In this paper, we propose a lightweight infrared target detection method, called Edge-YOLO, to address these challenges. Our approach replaces the backbone network of the YOLOv5m model with a lightweight ShuffleBlock and a strip depthwise convolutional attention module. We also applied CAU-Lite as the up-sampling operator and EX-IoU as the bounding box loss function. Our experiments demonstrate that, compared with YOLOv5m, Edge-YOLO is 70.3% less computationally intensive, 71.6% smaller in model size, and 44.4% faster in detection speed, while maintaining the same level of detection accuracy. As a result, our method is better suited for deployment on embedded platforms, making effective infrared target detection in real-world scenarios possible.

Keywords: infrared object detection; lightweight network; convolutional attention; YOLOv5; RK3588

1. Introduction

Visible light images are commonly used in target detection due to their high resolution, definition, and detailed visual information that is easily interpreted by the human eye. However, these images are sensitive to external factors such as weather and lighting conditions, which can reduce image quality and negatively impact target detection accuracy. This is where infrared imaging technology plays a crucial role. By overcoming these limitations, infrared imaging allows for image acquisition under diverse lighting and weather conditions, including foggy days and nighttime scenarios. As a result, this technology has been widely adopted in various fields, such as autonomous driving, security monitoring, and remote sensing, and offers a broader range of use cases than visible images. Therefore, the importance of infrared imaging technology in enabling efficient and reliable target detection cannot be overstated.

Traditional infrared target detection techniques [1–3] are mainly model-based methods, such as template matching, threshold segmentation, and the Hausdorff metric. However, with the development of deep learning, target detection techniques based on convolutional neural networks have emerged in recent years. These methods are primarily divided into two-stage algorithms (e.g., Faster-RCNN [4]) and single-stage algorithms (e.g., SSD [5] and YOLO [6]). The single-stage algorithm is designed to achieve a balance between detection speed and accuracy, resulting in a significant improvement in detection speed while maintaining accuracy compared with the two-stage algorithm. Consequently, the YOLO series has become a widely used representative of the single-stage algorithm. Among the mainstream YOLO algorithms, YOLOv5 (version 6.2) has achieved significant improvement in both detection accuracy and speed compared with its predecessor by using Mosaic data enhancement, C3 modules, and an improved SPPF module. Although YOLOv5 performs well on visible images, it encounters several challenges when applied to infrared detection. The first major issue is that infrared images suffer from poor contrast, high noise, and blurred imaging, leading to the loss of crucial target features during deep convolutional network
processing. Moreover, since infrared images lack color information, and the difference between target and background features is minimal, it becomes challenging for deep convolutional neural networks to distinguish useful information from irrelevant data, reducing detection accuracy. Another significant challenge is that embedded devices commonly used in autonomous driving and security monitoring fields have limited computing power, storage space, and power consumption, which makes deploying large target detection models such as YOLOv5 difficult. Additionally, these fields require real-time detection, and collecting data on edge devices and sending them to the server for detection and analysis can lead to network latency and communication congestion problems in widely distributed areas.

The challenges mentioned above necessitate lightweight infrared target detection on edge devices. In summary, this paper proposes a solution to these issues by introducing Edge-YOLO, an infrared target detection algorithm that utilizes lightweight networks and attention mechanisms specifically designed for edge devices. The primary enhancements of this algorithm can be summarized as follows:

1. The bounding box loss function was redesigned, and a loss function with a power hyperparameter \( \alpha \) was made to accelerate the convergence of the loss function and solve the uncertainty of the aspect ratio in CIoU;
2. A lightweight content-aware up-sampling operator was adopted, which can obtain a larger perceptual field than the original nearest-neighbor up-sampling method, while only introducing a small number of parameters and computational cost;
3. The feature extraction network was reconstructed based on the improved ShuffleNetv2, which enhances the extraction ability of strip features in IR scenes and the perception ability of salient features in IR images by embedding a newly designed strip depthwise convolutional attention module in ShuffleBlock while significantly reducing the computational power of the network.

The remaining sections of this paper are organized as follows: Section 2 provides an overview of related works on target detection with neural networks, including YOLOv5, and other algorithms in infrared target detection. Section 3 provides a detailed introduction to the Edge-YOLO algorithm proposed in this paper. Section 4 presents the results of various experiments conducted to evaluate the performance of Edge-YOLO. Finally, Section 5 summarizes the main contributions of this paper.

2. Related Works

The YOLO family of algorithms, known for their efficiency and simplicity, was first introduced by Redmon et al. in 2015. In the years that followed, Redmon et al. released YOLOv2 and YOLOv3 algorithms, which further reduced network complexity and improved detection speed compared with two-stage algorithms [7]. After Redmon withdrew from the field of computer vision, Glenn Jocher released YOLOv5 in 2020, which has since been updated to version 6.2. YOLOv5 is composed of a backbone feature extraction module, a neck feature fusion module, and a head detection module, as illustrated in Figure 1. The algorithm incorporates five different scales of n, s, m, l, and x, with larger scales delivering higher detection accuracy but slower real-time performance. However, the network structure of the models of different scales remains consistent, differing only in the number of partial layers, and is represented uniformly as “\( \times n \)” in the figure. YOLOv5 uses CSPDarknet53 as its backbone network, which includes the Cross Stage Partial (CSP) structure [8]. The CSP structure integrates gradient changes in the feature map, reducing the problem of repeating gradient information in the backbone network. Moreover, YOLOv5 utilizes a bottleneck structure with residual connections in the backbone network to prevent network degradation due to gradient disappearance, and a bottleneck structure without residual connections in the feature fusion layer to reduce computational effort. Additionally, Jocher employs a modified Spatial Pooling Pyramid Fast (SPPF) structure in place of the original SPP [9]. The modified SPPF achieves the same computational results as the original parallel...
MaxPool layers of three different sizes by serializing multiple MaxPool layers of the same size, significantly reducing computational time.

**Figure 1.** The network structure of YOLOv5 series (including n, s, m, l, and x, depending on the number of duplicates of module C3).

In addition to YOLOv5, various target detection methods have been proposed for infrared scenes by researchers. Li et al. [10] proposed the YOLO-FIRI model, an infrared image area-free target detector based on YOLOv5. They achieved good infrared target detection performance by improving the CSP structure and introducing multiple detection heads. Fan et al. [11] improved the feature extraction capability by using dense connection blocks based on YOLOv5, and improved the detection accuracy by adding a channel focus mechanism and modifying the loss function. Dai et al. [12] proposed TIRNet, which adopted VGG as the feature extractor and used a continuous information fusion strategy to obtain more accurate and smoother detection results. Li et al. [13] designed a dense nested interactive module to achieve progressive interaction among high-level and low-level features. You et al. [14] utilized multiscale mosaic data augmentation to enhance the diversity of objects and proposed a parameter-free attention mechanism to enhance features. Although these methods can be applied to IR target detection, they have some drawbacks. For instance, striped targets in IR road scenes require a more reasonable combination of striped convolution and traditional convolution to extract features. The bounding box loss function of the algorithm needs to be more accurately adapted to the boundary regression of targets in IR images, and the model needs to be more lightweight to be suitable for practical edge devices. Therefore, the method in this paper focuses on improving the above shortcomings.

3. Methods

The structure of our Edge-YOLO model is shown in Figure 2 below.

Firstly, the backbone of the model uses the improved ShuffleBlock to replace the C3 module in YOLOv5 to enhance the feature extraction capability for the characteristics of IR images of road scenes while reducing the complexity of the model.
Secondly, in the feature up-sampling structure, the original nearest-neighbor up-sampling operator is replaced by the improved CAU-Lite module. Thirdly, although not shown in the figure, we utilize the recently proposed EX-IoU instead of CIoU as the bounding box loss function of our model. This new loss function provides better and more accurate convergence during training, thus leading to improved detection performance.

Figure 2. The network structure of Edge-YOLO.

3.1. Improved Bounding Box Loss Function EX-IoU

The latest version of the YOLOv5 algorithm (version 6.2) uses the Complete-IoU (CIoU) as its bounding box loss function, as proposed by Zheng [15]. The CIoU integrates three aspects of the intersection–union ratio (IoU) between the predicted box and the ground truth box: the ratio of the distance between their centroids to the length of the diagonal of the minimum outer rectangle, and the similarity of their aspect ratios. The equation for the CIoU is as follows:

\[
L_{\text{CIoU}} = 1 - \text{IoU} + \rho \frac{b - b'}{c} + \alpha v \\
\alpha = \frac{v}{(1 - \text{IoU}) + v} \\
v = 4 \left( \arctan \frac{w'}{w} - \arctan \frac{w'}{w'} \right)^2
\]

where \(\alpha\) is the weight coefficient, \(\rho(b, b')\) is the distance between the center point of the predicted box and the ground truth box, \(c\) is the diagonal length of the minimum outer rectangle, \(v\) indicates the difference between the aspect ratio of the predicted box and the ground truth box, and \(v\) is 0 if they have the same aspect ratio.
The CIoU metric used in the current version of the YOLOv5 algorithm is designed to integrate three aspects of the intersection–union ratio IoU between the prediction box and the ground truth box. However, the aspect ratio used in CIoU is a relative value, which can introduce uncertainty during calculation and potentially hinder the optimization of the model. To address this issue, we propose the use of Efficient-IoU (EIoU) as the bounding box loss function, as proposed by Zhang et al. [16]. EIoU splits the aspect ratio based on CIoU and replaces the original aspect ratio difference between the predicted box and ground truth box with the ratio of the width difference between the predicted box and the ground truth box to the width of the minimum circumscribed rectangle, and the ratio of the height difference to the height of the minimum outer rectangle. This approach leads to a more accurate bounding box loss function and facilitates better model optimization. The formula is as follows:

\[
L_{EIoU} = 1 - IoU + \frac{\rho^2 (h, h^{gt})}{C^2_h} + \frac{\rho^2 (w, w^{gt})}{C^2_w} + \frac{\rho^2 (h, h^{gt})}{C^2_h}
\] (2)

where \(C_w\) and \(C_h\) denote the width and height of the minimum outer rectangle, respectively. The literature [17] proposes to use the hyperparameter \(\alpha\) as a power on each term in the loss function IoU, with the following simplified formula.

\[
L_{\alpha-IoU} = 1 - IoU^\alpha
\] (3)

The parameter \(\alpha\) is crucial in emphasizing the importance of the loss and gradient of objects with high IoU, thereby enhancing the accuracy of the bounding box regression. To improve the bounding box loss function, this paper incorporates the power \(\alpha\) into the EIoU equation, resulting in a new function known as EX-IoU. This function exponentially magnifies the importance of the IoU value, centroid distance, width difference, or height difference between any predicted box and the ground truth box, leading to an exponential reduction effect on losses and an improvement in the accuracy of the bounding box regression. The optimal value of \(\alpha\) is determined through experiments discussed in Section 4.

\[
L_{EX-IoU} = 1 - IoU^\alpha + \frac{\rho^2 (h, h^{gt})}{C^2_h} + \frac{\rho^2 (w, w^{gt})}{C^2_w} + \frac{\rho^2 (h, h^{gt})}{C^2_h}
\] (4)

3.2. Content-Aware Lightweight Up-Sampling Operator CAU-Lite

Up-sampling is a crucial operation in the Feature Pyramid Network (FPN [18]), and the two commonly used methods for up-sampling are linear interpolation and deconvolution. Linear interpolation methods, such as nearest neighbor interpolation and bilinear interpolation, only take into account the sub-pixel neighborhood of the current pixel, which results in insufficient semantic information and limited receptive fields. On the other hand, the deconvolution method expands the dimensionality of the feature map through convolution, but it applies the same convolution kernel across the entire feature map, which makes it challenging to capture local changes and variations in the feature map. Furthermore, this method introduces a large number of parameters and computational overhead into the network. The CARAFE proposed by Wang et al. [19] compensates the shortcomings of the above two types of methods to some extent: CARAFE perceives and aggregates contextual information within a larger reception field, and instead of applying a fixed convolution kernel to all features, it dynamically generates adaptive up-sampling kernels, and then reorganizes the features based on the predicted up-sampling kernels. The up-sampling kernel prediction module of CARAFE changes the dimensionality of the input feature map by convolution layer to generate a feature map with the channel number of \(\sigma^2 k_{up}^2\), where \(\sigma\) indicates the up-sampling rate (generally 2) and \(k_{up}\) indicates the size of the up-sampling kernel (the value in this paper is 5), and then uses the pixel shuffle method to expand the channel dimension in the spatial dimension to obtain an up-sampling kernel map with the shape of \(\sigma H \times \sigma W \times k_{up}^2\), which contains \(\sigma H \times \sigma W\) up-sampling kernels. Each position
of the output feature map is then mapped back to the input feature map in the feature reassembly module. The region of $k_{up} \times k_{up}$, the center of the map, is taken out and dotted with the predicted up-sampling kernels at that point to obtain the result. Each channel at the same spatial location on the feature map uses the same up-sampling kernel.

The analysis reveals that CARAFE uses a zero-padding strategy on the feature map edge positions in the feature reorganization stage, which leads to imperfect edge information of the up-sampled images and it is difficult to correctly upsample the target features at the edge of the feature map. Based on this, this paper proposes an improved Content-Aware Up-Sampling-Lite (CAU-Lite) method to replace the nearest neighbor up-sampling method in YOLOv5. Before finding the neighborhood of $k_{up} \times k_{up}$, the nearest neighbor interpolation is used to upsample the input feature map, so that the spatial dimension of the feature map is the same as the spatial dimension of the up-sampled kernel map. Then, at each spatial position in the feature map, the element with the size $1 \times 1 \times k_{up}^2 \times C$ is taken out and reshaped to the size of $k_{up} \times k_{up} \times C$. At the same time, the upper sampling core of $1 \times 1 \times k_{up}^2$ is reshaped at the corresponding position in the upper sampling core map to the size of $k_{up} \times k_{up} \times 1$. The product of each channel of the feature map with the upper sampling core is dotted to obtain a result with the size $1 \times 1 \times C$, and the result obtained for all channels is the result of the corresponding position in the output feature map. The improved CAU-Lite structure and calculation process are shown in Figure 3.

**Figure 3.** Structure of CAU-Lite.

### 3.3. YOLOv5 Network Model Improvement

The YOLOv5 algorithm is mainly designed for the visible light domain and is better suited for deployment on GPUs due to its high number of model parameters, large computational requirements, and large model size. However, the objective of this paper is to create lightweight target detection networks for edge-embedded devices, making it reasonable to employ a lightweight network structure instead of the heavy CSPDarknet backbone network of YOLOv5. One promising candidate for such a lightweight network model is ShuffleNetv2, proposed by Ma et al. of Megvii’s team [20], which achieves a good balance between model accuracy and running speed. By using lightweight structures...
such as grouped convolution and depthwise convolution, ShuffleNetv2 is optimized for computational complexity, storage access cost, and parallelism, resulting in a noticeable improvement in actual running speed. In light of this, we chose to use an improved version of ShuffleNetv2 as the backbone network structure of the Edge-YOLO algorithm. The following problems usually exist in IR road scenes:

1. Compared with visible light images, original infrared images lack detailed texture and feature complexity, making them less susceptible to network perception, especially in deeper and complex networks where features are lost and additional noise is introduced.

2. In road scenes, pedestrians and bicycles are typically present in slender strips rather than regular shapes. However, traditional convolution layers use regular convolution kernels (N × N), which may result in a loss of feature information due to their inability to adapt to target shape changes.

To enhance the extraction capability of strip-shaped features in IR scenes and the perception of salient features in IR images without increasing computational effort, this paper proposes embedding the strip depthwise convolutional attention module (SDCA) into ShuffleBlock. As shown in Figure 4, SDCA takes an input and generates a shortcut branch. The local information is aggregated by a 5 × 5 depthwise convolution, and the output is convolved by three branches: a pair of 1 × 7 and 7 × 1 depthwise strip convolutions, a pair of 1 × 11 and 11 × 1 depthwise strip convolutions, and a shortcut branch. These branches capture multi-scale contextual information and strip features, and their results are summed and passed through a 1 × 1 normal convolutional layer to model the relationship between different channels. The output of this layer is then used as the attention weights to weigh the input of SDCA by multiplying it with the generated shortcut branch.

![Figure 4. Strip depthwise convolutional attention module.](image)

The improved structure obtained by embedding the strip depthwise convolutional attention module into ShuffleBlock, as well as the addition of SENet [21] as the channel attention mechanism in the right branch, are shown in Figure 5. Typically, 1 × 1 convolutions are used before and after depthwise convolutions to fuse information between channels and to increase or decrease the number of channels. However, the original ShuffleBlock uses 1 × 1 convolution layers before and after the depthwise convolution layer in...
its right branch, which generates redundancy. To reduce the number of parameters and computational demand, this paper removes the $1 \times 1$ convolution layer after the depthwise convolution.

![Diagram of Improved ShuffleBlock](image_url)

**Figure 5.** Improved ShuffleBlock. (a): the base unit with SDCA applied; (b): the base unit for down-sampling ($2 \times$).

4. Experiments

4.1. Experimental Environment and Dataset

The experiments in this paper were conducted using an Intel Xeon Platinum 8255C CPU and an NVIDIA RTX 3090 GPU with CUDA version 11.7. To evaluate the detection performance of Edge-YOLO, we used the publicly available FLIR dataset, which is an infrared dataset released by FLIR in 2018. The dataset consists of more than 10,000 images classified into four categories: Person, Bicycle, Car, and Dog. However, since there are only a few Dog images in the dataset, this paper only evaluated the detection performance of Edge-YOLO for the remaining three categories.

4.2. Bounding Box Hyperparameter Study

In the improved EX-IoU bounding box loss function of this paper, there is a hyperparameter $\alpha$ that affects the model’s accuracy performance. To determine the optimal value of $\alpha$ for the Edge-YOLO algorithm, we conducted multiple training and testing experiments using different values of $\alpha$. The accuracy results obtained are shown in Figure 6. From the results, it can be observed that the highest mAP value of 78.8% is achieved when the value of $\alpha$ is set to 3, while the mAP value of the model decreases to 76.9% when the value of $\alpha$ is set to 8. This indicates that the model’s detection accuracy improves by 2.47% when using the optimal value of $\alpha$, and the model achieves its best performance in terms of detection performance. As a result, this paper selects 3 as the power of each term in EX-IoU to obtain the best accuracy performance.
Figure 6. Variation in accuracy for different values of $\alpha$ on the algorithm.

4.3. Model Lightweighting Experiment

By replacing the backbone feature extraction network of YOLOv5 with the improved ShuffleBlock in this paper, i.e., Edge-YOLO shown in Figure 2, the overall number of parameters, computation, and model size of the algorithm model can be effectively reduced, and Table 1 below shows the comparison of each parameter after the model is lightened and improved.

<table>
<thead>
<tr>
<th>Model</th>
<th>Params/M</th>
<th>Flops/G</th>
<th>Size/MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>YOLOv5m</td>
<td>20.9</td>
<td>47.9</td>
<td>42.2</td>
</tr>
<tr>
<td>Edge-YOLO</td>
<td>5.8</td>
<td>14.2</td>
<td>12.0</td>
</tr>
</tbody>
</table>

The table above shows that by replacing the backbone network with the improved ShuffleBlock, Edge-YOLO reduces the number of network parameters by 72.2%, the amount of computation by 70.3%, and the model size by 71.6% compared with YOLOv5m. This demonstrates the significant lightweight effect of the proposed method, which helps to reduce the storage and computation resources required by the model and is more suitable for deployment on edge-embedded devices.

4.4. Ablation Experiments

In this part, the original ShuffleNetv2 is firstly used as the backbone network of YOLOv5m, based on which the ablation experiments of several improvement strategies proposed in this paper are conducted to better understand the effects of different improvement strategies on the detection effect in Edge-YOLO, and the results are shown in Table 2 below.

<table>
<thead>
<tr>
<th>+EX-IoU</th>
<th>+CAU-Lite</th>
<th>+SDCA</th>
<th>mAP@0.5/%</th>
<th>FPS</th>
<th>Params/M</th>
<th>Flops/G</th>
<th>Size/MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>√</td>
<td></td>
<td></td>
<td>73.1</td>
<td>85.3</td>
<td>5.4</td>
<td>12.9</td>
<td>11.3</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td></td>
<td>74.3</td>
<td>85.1</td>
<td>5.4</td>
<td>12.9</td>
<td>11.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>√</td>
<td>74.7</td>
<td>84.0</td>
<td>5.4</td>
<td>13.1</td>
<td>11.4</td>
</tr>
<tr>
<td>√</td>
<td>√</td>
<td>√</td>
<td>76.2</td>
<td>81.5</td>
<td>5.8</td>
<td>14.1</td>
<td>12.0</td>
</tr>
<tr>
<td>√</td>
<td>√</td>
<td>√</td>
<td>78.8</td>
<td>80.7</td>
<td>5.8</td>
<td>14.2</td>
<td>12.0</td>
</tr>
</tbody>
</table>

As can be seen from Table 2, compared with the first group of experiments using only the basic model, the second group of experiments with the addition of EX-IoU solves the
problem of uncertainty in the aspect ratio of CIoU by improving the loss function of the bounding box and accelerating the convergence of the loss function, and the detection accuracy is improved by 1.2% from the results, while the remaining parameters remain unchanged. The third group of experiments replaces the original nearest neighbor interpolation up-sampling with the CAU-Lite up-sampling operator proposed in this paper, which senses and aggregates contextual information within a larger reception field, dynamically generates adaptive up-sampling kernels, and performs feature reorganization based on the generated up-sampling kernels. It can be seen that with CAU-Lite, the detection accuracy of the model is improved by 1.6%, but the FPS is also slightly reduced. The fourth group of experiments applies the strip depthwise convolutional attention module proposed in this paper, which replaces the original ShuffleNetv2 network structure with an improved ShuffleBlock, enhancing the feature extraction capability for strip-shaped targets and the perception of the saliency of infrared targets. As seen in the table, the detection performance of the model is significantly improved by 3.1% compared with the original model, but the number of parameters, computation, and size of the model increased due to the addition of the new module. The final fifth set of experiments uses a combination of the three improvement points proposed in this paper, and from the results, a larger performance improvement is obtained at the cost of fewer computational and storage resources compared with the original model.

Figure 7 below shows the P–R curves of each class of different improvement strategies applied to the base model and the complete Edge-YOLO. The figure shows that compared with the base model, the APs of all three target categories are improved with different improvement strategies, and the AP of the bicycle category is improved most significantly.

4.5. Comparison Experiments

To further verify the detection performance of the Edge-YOLO algorithm, this section compares Edge-YOLO with Faster R-CNN, SSD, YOLOv5m, YOLOv7 [22], and other mainstream target detection algorithms for comparison experiments, and the results are shown in Table 3 below.

<table>
<thead>
<tr>
<th>Model</th>
<th>AP@0.5/%</th>
<th>mAP@0.5/%</th>
<th>FPS</th>
<th>Params/M</th>
<th>Flops/G</th>
<th>Size/MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN</td>
<td>76.6</td>
<td>50.4</td>
<td>85.7</td>
<td>70.9</td>
<td>17.6</td>
<td>66.1</td>
</tr>
<tr>
<td>SSD</td>
<td>63.5</td>
<td>43.5</td>
<td>79.3</td>
<td>62.1</td>
<td>31.9</td>
<td>46.9</td>
</tr>
<tr>
<td>YOLOv5m</td>
<td>83.8</td>
<td>64.1</td>
<td>90.6</td>
<td>79.5</td>
<td>55.9</td>
<td>20.9</td>
</tr>
<tr>
<td>YOLOv7</td>
<td>86.4</td>
<td>67.3</td>
<td>91.4</td>
<td>81.7</td>
<td>27.9</td>
<td>37.2</td>
</tr>
<tr>
<td>YOLO-FIRI</td>
<td>79.5</td>
<td>52.4</td>
<td>88.6</td>
<td>73.5</td>
<td>71.4</td>
<td>7.2</td>
</tr>
<tr>
<td>Edge-YOLO</td>
<td>83.2</td>
<td>63.0</td>
<td>90.2</td>
<td>78.8</td>
<td>80.7</td>
<td>5.8</td>
</tr>
</tbody>
</table>

From the table, we can first see that Faster R-CNN, as a two-stage algorithm, lags far behind the single-stage algorithm in detection speed, and the current detection accuracy does not have an advantage over the single-stage algorithm; while the SSD algorithm in the single-stage algorithm speeds up the detection speed compared with Faster R-CNN, but the detection accuracy is also reduced accordingly. Both algorithms are not comparable to the current YOLO series algorithms. Second, the detection accuracy of the algorithm in this paper is basically the same compared with the YOLOv5m algorithm, but it has obvious advantages in detection speed and consumption of computational and spatial resources. Again, compared with the latest YOLOv7 algorithm, the detection accuracy of the Edge-YOLO algorithm is slightly behind that of the YOLOv7 algorithm, but the resources consumed by the YOLOv7 algorithm and its detection speed are completely inferior to that of this paper. Finally, compared with the YOLO-FIRI target detection
algorithm proposed by Li et al. also for IR scenes, this paper achieves higher detection accuracy with less resource consumption and better results in real-time.

![P-R Curves of Person](image1)

![P-R Curves of Bicycle](image2)

![P-R Curves of Car](image3)

![P-R Curves of All](image4)

**Figure 7.** P–R curves for each class. (a): P-R curves of the class ‘Person’; (b): P-R curves of the class ‘Bicycle’; (c): P-R curves of the class ‘Car’; (d): P-R curves of all classes.

### 4.6. Comparison of Test Results

Figure 8 below shows the detection results of some images in the dataset under YOLO-FIRI, YOLOv5m, YOLOv7, and Edge-YOLO of this paper.

Since the Faster R-CNN and SSD in the previous subsection are lagging in detection accuracy and detection speed, only YOLO-FIRI, YOLOv5m, and YOLOv7 are used in the visualization effect comparison with the algorithms in this paper.

From the figure, it can be seen that compared with YOLO-FIRI, the target detection algorithm for infrared road scenes, the algorithm in this paper has a certain lead in accuracy and a higher confidence level in detecting targets. In addition, observing the fourth figure, we can see that the YOLO-FIRI algorithm misdetects some pedestrian legs as bicycles, which has some defects. After comparing this algorithm with the YOLOv5m algorithm...
and YOLOv7 algorithm, we can see that the three algorithms basically maintain the same
detection results, and they can detect cars, pedestrians, and a small number of bicycles in
road scenes well. Because the algorithm in this paper is a lightweight network model, it is
better than the other two algorithms in terms of the number of parameters, computation,
and model size, so this algorithm has a more practical application value.

Figure 8. Part of the test results.

4.7. Actual Edge Device Deployment Testing

This paper uses the RK3588 embedded development board of Rockchip as the verification
platform, as shown in Figure 9 below. RK3588 platform is equipped with quad-core
A76+ quad-core A55, an octa-core CPU, and NPU with 6TOPs computation power. Its high
computation power NPU supports INT4, INT8, INT16, and FP16 mixed computing, which
can accelerate the inference of network models. The photo of RK3588 is shown below.
The algorithm model in this paper and the comparison algorithm model are first exported to the compatible ONNX format, and then converted to the RKNN model supported by the NPU of the RK3588 platform using the RKNN-Toolkit2 and rknpu2 tools with inference acceleration such as asymmetric hybrid quantization, and these models are used to infer the test set images, and the performance comparison is obtained as shown in the following table. In addition to inference using NPU, the performance of only CPU inference is also tested in this paper and is shown together in Table 4 below.

Table 4. Testing results on the RK3588 platform.

<table>
<thead>
<tr>
<th>Model</th>
<th>mAP@0.5/%</th>
<th>FPS(CPU)</th>
<th>FPS(NPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YOLO-FIRI</td>
<td>73.0</td>
<td>0.8</td>
<td>21.6</td>
</tr>
<tr>
<td>YOLOv5m</td>
<td>79.1</td>
<td>0.5</td>
<td>14.5</td>
</tr>
<tr>
<td>YOLOv7</td>
<td><strong>81.2</strong></td>
<td>0.3</td>
<td>8.8</td>
</tr>
<tr>
<td>Edge-YOLO</td>
<td>78.6</td>
<td><strong>1.1</strong></td>
<td><strong>31.9</strong></td>
</tr>
</tbody>
</table>

As can be seen from the table, the accuracy of all four models on the RK3588 platform has a slight decrease due to model quantization. In addition, if only the ARM CPU is used for inference, the FPS of algorithms such as YOLO-FIRI is less than 1, i.e., the number of images that can be inferred is less than one per second, and the algorithm in this paper only has an FPS of 1.1, which cannot be deployed in practical application scenarios. After using NPU to accelerate, we can see that the inference speed of each algorithm was improved by tens of times. However, the FPS of YOLOv5m and YOLOv7 are only 14.5 and 8.8, respectively, which are more obvious to notice lags in real-world applications, while the algorithm in this paper can achieve 31.9 FPS, which can meet the performance requirements of practical scenarios.

5. Conclusions

The proposed method in this paper, Edge-YOLO, is a lightweight IR target detection approach that aims to ensure good performance in road scenes and is suitable for edge-embedded devices. The algorithm utilizes an optimized bounding box loss function, the improved EX-IoU, to enhance the regression accuracy of the bounding box. Moreover, to improve the up-sampling effect, the algorithm adopts the improved CAU-Lite up-sampling operator, which perceives the contextual content. Lastly, the lightweight ShuffleBlock replaces the backbone feature extraction part of the network, and the strip depthwise convolutional attention module is used to enhance the extraction capability of strip-shaped targets and other salient features present in the IR feature map for the ShuffleBlock, thus further enhancing the detection accuracy of the model. The experimental results on the FLIR dataset demonstrate that Edge-YOLO is essentially equivalent to YOLOv5m in terms
of accuracy, while reducing the number of network parameters, computation, and model size by 72.2%, 70.3%, and 71.6%, respectively. Additionally, the detection speed is increased by 44.4%, making the algorithm more suitable for embedded device applications.

**Author Contributions:** Conceptualization, J.Y.; methodology, J.L.; software, J.L.; validation, J.L.; writing—original draft preparation, J.L.; writing—review and editing, J.L. and J.Y.; supervision, J.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**


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.