electronics

MDPI

Article

SP-YOLO-Lite: A Lightweight Violation Detection Algorithm Based on SP Attention Mechanism

Zhihao Huang [1,†], Jiajun Wu [1,†], Lumei Su [1,2,*], Yitao Xie [1], Tianyou Li [1] and Xinyu Huang [3]

[1] School of Electrical Engineering and Automation, Xiamen University of Technology, Xiamen 361024, China; ryouta2021@163.com (Z.H.); wujiajun@stu.xmut.edu.cn (J.W.); xie_yitao@163.com (Y.X.); ltyxm@163.net (T.L.)
[2] Xiamen Key Laboratory of Frontier Electric Power Equipment and Intelligent Control, Xiamen 361024, China
[3] School of Information Engineering, Xiamen Ocean Vocational College, Xiamen 361100, China; huangxinyu@xmoc.edu.cn
* Correspondence: sulumei@163.com
† These authors contributed equally to this work.

Abstract: In the operation site of power grid construction, it is crucial to comprehensively and efficiently detect violations of regulations for the personal safety of the workers with a safety monitoring system based on object detection technology. However, common general-purpose object detection algorithms are difficult to deploy on low-computational-power embedded platforms situated at the edge due to their high model complexity. These algorithms suffer from drawbacks such as low operational efficiency, slow detection speed, and high energy consumption. To address this issue, a lightweight violation detection algorithm based on the SP (Segmentation-and-Product) attention mechanism, named SP-YOLO-Lite, is proposed to improve the YOLOv5s detection algorithm and achieve low-cost deployment and efficient operation of object detection algorithms on low-computational-power monitoring platforms. First, to address the issue of excessive complexity in backbone networks built with conventional convolutional modules, a Lightweight Convolutional Block was employed to construct the backbone network, significantly reducing computational and parameter costs while maintaining high detection model accuracy. Second, in response to the problem of existing attention mechanisms overlooking spatial local information, we introduced an image segmentation operation and proposed a novel attention mechanism called Segmentation-and-Product (SP) attention. It enables the model to effectively capture local informative features of the image, thereby enhancing model accuracy. Furthermore, a Neck network that is both lightweight and feature-rich is proposed by introducing Depthwise Separable Convolution and Segmentation-and-Product attention module to Path Aggregation Network, thus addressing the issue of high computation and parameter volume in the Neck network of YOLOv5s. Experimental results show that compared with the baseline network YOLOv5s, the proposed SP-YOLO-Lite model reduces the computation and parameter volume by approximately 70%, achieving similar detection accuracy on both the VOC dataset and our self-built SMPC dataset.

Keywords: violation detection; deep learning; lightweight object detection algorithm; attention mechanism

Citation: Huang, Z.; Wu, J.; Su, L.; Xie, Y.; Li, T.; Huang, X. SP-YOLO-Lite: A Lightweight Violation Detection Algorithm Based on SP Attention Mechanism. *Electronics* **2023**, *12*, 3176. https://doi.org/10.3390/electronics12143176

Academic Editor: George A. Tsihrintzis

Received: 19 June 2023
Revised: 15 July 2023
Accepted: 19 July 2023
Published: 21 July 2023

1. Introduction

In recent years, with the rapid development of the power grid industry, the task of power grid construction has become increasingly heavy, and there has been an increase in the number of on-site work points, leading to frequent accidents at the work site. The direct causes of these accidents are mostly due to workers not wearing complete safety equipment, entering restricted areas, and other habitual violations [1]. In response to this issue, an increasing number of power grid construction sites are deploying on-site safety monitoring systems based on object detection technology to automatically supervise and control workers' violations. Furthermore, in order to meet the real-time requirements of detecting violations, these systems have started to decentralize the automatic detection

process to embedded platforms situated at the edge. This involves deploying object detection models in close proximity to the sensing devices. However, it is challenging to directly deploy common general-purpose object detection models on the embedded platforms of safety monitoring systems.

The challenge of deploying general-purpose object detection models on embedded platforms stems from several factors. Firstly, general-purpose object detection models are typically based on deep learning neural networks, which require high computational and memory resources for deployment [2]. This makes them unsuitable for embedded platforms with limited computing power, storage capacity, and memory size. Secondly, in the field of on-site safety monitoring, the rapid identification and processing of violations are of utmost importance. However, general-purpose object detection models have limited detection speed due to their high computation and parameter volume, making it difficult to meet the real-time demands of violation detection tasks [3]. Thirdly, embedded platforms typically possess unique hardware architecture and software frameworks that differ from conventional computing systems. Consequently, general-purpose object detection models may lack compatibility with these platforms, necessitating substantial modifications or adaptations to ensure optimal performance. Such endeavors can consume considerable time and resources, while also introducing additional complexities into the deployment process [4]. Lastly, general-purpose object detection models often consume a significant amount of energy to support their complex computational needs. This poses a challenge for embedded platforms with limited resources, as they require low power consumption. To address these challenges, a lightweight object detection algorithm for embedded platforms that can be tailored to actual application scenarios and balance accuracy, energy consumption, and speed is urgently needed in the on-site safety monitoring field. The deployment of lightweight detection models on embedded platforms can greatly enhance the safety monitoring capabilities at power grid construction sites.

This paper proposes a lightweight violation detection algorithm based on the SP attention mechanism, called SP-YOLO-Lite. It is an improved YOLOV5s algorithm with lower deployment costs and can realize comprehensive and effective violation detection. The main contributions of this paper are as follows:

- Designing a lightweight Backbone network based on the Lightweight Convolutional Block (LC-Block). The network is constructed by directly stacking LC-Blocks based on Depthwise Separable Convolutions, maintaining a single path from input to output. It has lower network fragmentation and no additional computational costs such as kernel startup. Compared to backbone networks built with conventional convolutional modules in general-purpose object detection algorithms, the proposed network significantly reduces computational and parameter requirements while greatly improving inference efficiency.
- Proposing a novel attention mechanism called Segmentation-and-Product (SP) attention mechanism. This mechanism is designed to address the characteristic of violation detection tasks where the detection targets are concentrated in local regions. It innovatively incorporates image segmentation operations to divide the input image into regions and applies attention operations to each region separately. This effectively captures the local features of the image, thereby improving the accuracy of the detection model. Compared to existing attention mechanisms, this attention mechanism focuses more on capturing local spatial feature information, making it more suitable for violation detection tasks.
- Proposing a Neck network that is both lightweight and feature-rich. For this network, we use Depthwise Separable Convolution to replace complex CSP modules and conventional convolutional modules as the upsampling operator, significantly reducing the computational and parameter complexity. Additionally, we insert SP attention modules into the network to effectively enhance the model's ability to extract meaningful features from the targets. Furthermore, we optimize the channel configuration of each layer in the Neck network, reducing the memory access cost. Compared to

the feature fusion networks used in general-purpose object detection algorithms, this Neck network achieves a better balance between accuracy, speed, and complexity.
- Collecting and processing monitoring images from different power operation scenes to construct a Security Monitor for Power Construction (SMPC) dataset that includes multiple detection targets such as a safety belt, fence, and seine, and is suitable for violation detection tasks.

2. Related Work

2.1. Violation Behavior Detection

Currently, research on violation behavior detection mostly focuses on detecting the wearing of safety helmets. The detection methods of safety helmets can be divided into traditional machine learning methods and deep learning methods. Traditional machine learning methods use manually extracted image features of safety helmets to train classifiers for automatic detection. Man-Woo et al. [5] extracted the HOG features of safety helmets and used supervised learning to train support vector machines for the automatic detection of safety helmets in indoor scenes. However, such methods cannot handle high-dimensional data in complex scenes due to the shallow models constructed. In contrast, deep learning methods adopt deep models to overcome the shortcomings of machine learning methods and have a stronger ability to extract image features. Wang et al. [6] proposed an improved YOLOv5-based object detection method for detecting the wearing of safety helmets in complex environments, achieving a mean Average Precision (mAP) of 95.9%. Similarly, Zhang et al. [7] proposed a workshop safety helmet-wearing detection model, SCM-YOLO, to meet the demand for real-time and accurate detection of safety helmets in complex scenarios. The model incorporates the Spatial Pyramid Pooling (SPP) structure and the Convolutional Block Attention Module (CBAM) into the YOLOv4-tiny model, achieving a mAP of 93.19%, which is 4.76% higher than the YOLOv4-tiny algorithm. The inference speed of the model reaches 22.9 FPS. In 2023, Wang et al. [8] proposed a helmet-wearing detection model called YOLO-M. The model employed MobileNetv3 as the backbone network, effectively reducing model complexity. Additionally, residual edges were introduced in feature fusion, enhancing the detection capability for small targets. Experimental results demonstrated that YOLO-M achieved a 2.22% improvement in detection accuracy compared to the baseline network YOLOv5s while using only three-quarters of its parameter count. However, existing approaches have difficulty meeting the practical requirements of violation detection tasks. Firstly, they are limited in the types of detectable targets, mostly applicable only for detecting the wearing of safety helmets. Secondly, they lack targeted design and optimization for deployment on embedded platforms. In contrast, the proposed SP-YOLO-Lite in this paper effectively addresses these issues. It not only has a wider range of applications, capable of detecting multiple target types such as safety belts, fences, and seines but also incorporates an extensive lightweight design and optimization specifically for deployment on embedded platforms. As a result, it exhibits better practical performance on embedded platforms.

2.2. Lightweight Object Detection Algorithms

In order to meet the requirements of resource-limited platforms such as embedded systems [9], researchers have proposed a series of lightweight object detection algorithms that balance detection accuracy and real-time performance by lightening the original algorithms such as YOLO [10–14] and SSD [15]. The most widely used lightweight object detection algorithm is the YOLO series. In 2015, Redmon et al. [10] proposed a lightweight version of YOLOv1, Tiny-YOLOv1, which simplified the original 24-layer convolutional structure of YOLOv1 to nine layers. Although its mAP on the VOC2007 dataset is lower than that of the original YOLOv1, its detection speed is improved by 3.4 times. Based on this, Redmon et al. also released corresponding lightweight versions of YOLOv2 [11] and YOLOv3 [12], namely Tiny-YOLOv2 and Tiny-YOLOv3, respectively. They have reduced the size of the model while achieving higher mAP. In 2021, Chen et al. [16] proposed

the YOLOv5-Lite series of lightweight object detection algorithms based on YOLOv5, which are lighter, faster, and easier to deploy. This series of algorithms has smaller Flops (Floating Point of Operations), lower memory, and fewer parameters, and due to the introduction of lightweight network modules, it has a faster inference speed. In 2023, Li et al. [17] proposed a lightweight infrared object detection method called Edge-YOLO. The method constructs the backbone network by stacking lightweight ShuffleBlocks and a strip depthwise convolutional attention module. Additionally, CAU-Lite was applied as the upsampling operator, and EX-IoU was employed as the bounding box loss function. Experimental results demonstrate that compared to YOLOv5m, Edge-YOLO achieves a reduction in model size by 71.6% while maintaining the same level of detection accuracy. Existing lightweight object detection algorithms have achieved a high level of lightweight optimization compared to general-purpose object detection algorithms. However, most algorithms fail to incorporate task-specific network design and optimization targeted towards the characteristics of the applied tasks and hardware platforms, resulting in their inadequate practical performance that fails to meet practical requirements. In this paper, we have conducted targeted lightweight design and optimization based on the characteristics of the violation detection task and the deployed hardware platform. As a result, we propose a lightweight object detection algorithm that better meets practical requirements. Compared to existing lightweight object detection algorithms, it achieves a better balance between detection accuracy and model complexity, leading to improved practical performance.

2.3. Attention Mechanism

The attention mechanism enables neural networks to focus on important features of the input images while ignoring unimportant ones during feature extraction and has shown great potential in improving the performance of convolutional neural networks. Hu et al. [18] first proposed an effective attention mechanism, called the Squeeze-and-Excitation (SE) attention mechanism, which effectively improved the detection accuracy of models by utilizing 2D global pooling and fully connected structures. However, the SE attention mechanism ignored the position information in images that is equally important to channel information, limiting its ability to improve model accuracy. To address this issue, Woo et al. [19] proposed the Convolutional Block Attention Module (CBAM), which aggregated features using both average and max pooling, and fused channel and spatial information, leading to further improvement in model accuracy. Both methods aimed to develop more complex attention modules for better performance, inevitably increasing the complexity of the models. Therefore, Wang et al. [20] proposed a more efficient attention module, the Efficient Channel Attention (ECA) module, to balance model performance and complexity, which contained only a small number of parameters but still resulted in a significant performance improvement. However, in violation detection tasks, the target objects to be detected are often concentrated in a certain region of the images, and most attention mechanisms tend to ignore the local spatial information of each feature channel, limiting their effectiveness in violation detection tasks. To address this issue, this paper proposes a lightweight attention mechanism, called the Segmentation-and-Product (SP) Attention Mechanism, which effectively captures local spatial information for better performance in violation detection tasks while balancing model accuracy and complexity.

3. SP-YOLO-Lite Network Model

As shown in Figure 1, the SP-YOLO-Lite network consists of three parts: Backbone, Neck, and Head. The Backbone is primarily composed of multiple Lightweight Convolutional Blocks (LC-Blocks) [21], which extract features of different scales from the input image through a series of convolution operations. The Neck is built upon the PANet and incorporates the Depthwise Separable Convolution (DSConv) [22] and Segmentation-and-Product (SP) attention module, enabling the aggregation and fusion of features from various layers of the Backbone. The Head consists of multiple detection heads composed of convolutional layers, pooling layers, and fully connected layers. Each detection head

receives feature maps of different scales from the Neck network and finally outputs the position, category, and confidence of different scale targets.

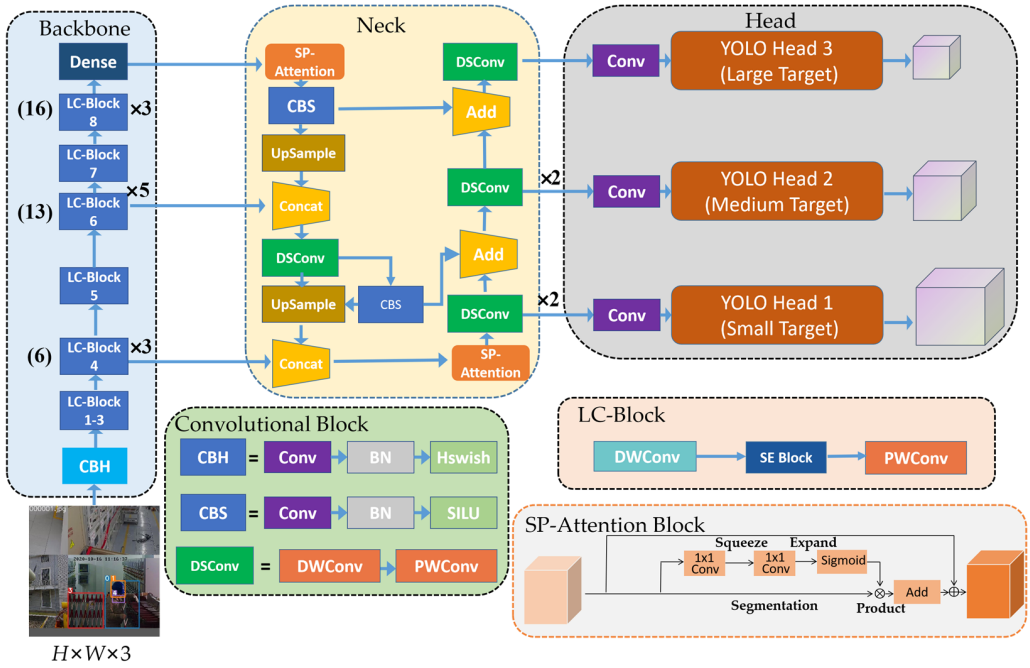


Figure 1. The architecture of SP-YOLO-Lite network. In the Head part, the light purple cubes represent the final output of the network.

3.1. Lightweight Backbone Network Based on LC-Block

In this paper, we propose a lightweight Backbone network based on LC-Block. LC-Block is a lightweight convolutional module that exhibits smaller parameter and computational requirements, as well as higher efficiency in feature extraction compared to conventional convolutional modules commonly used in general-purpose object detection algorithms. The specific structure of the LC-Block is illustrated in Figure 2. It is composed of Depthwise Convolution (DWConv), Pointwise Convolution (PWConv), and a Squeeze-and-Excitation (SE) attention module. DWConv is a lightweight convolutional module that performs convolution operations only on one channel of the input tensor for each convolution kernel. PWConv is a conventional convolution module with a kernel size of 1×1, which is often used in combination with DWConv to linearly combine the feature maps of each channel and obtain more complex feature representations. The SE attention module is inserted between the PWConv and DWConv to enhance the model's ability to extract effective features from the images. The specific structure of the SE attention module used in this paper is shown in Figure 2. The output feature map of size $C \times H \times W$ from DWConv will be fed into the SE module for the following attention operations. Firstly, the feature map is compressed to a $1 \times 1 \times C$ feature vector by pooling operation. Then, two 1×1 convolution modules are used to compress and expand channels, respectively, to generate channel attention score vectors. Finally, the vector will be weighted on each channel of the input feature map to generate a weighted feature map of size $C \times H \times W$.

The specific structure and parameter configuration of the lightweight Backbone network constructed in this paper is shown in Table 1. When building the network, we first directly stack multiple LC-Blocks so that the constructed Backbone network always maintains a single path from input to output, without extra network branches. This results in a low degree of network fragmentation, which is more friendly to low-computing devices compared to multi-branch networks commonly used in general-purpose object detection algorithms, and there are no additional calculation costs such as kernel startup and synchronization. Secondly, we adjusted the number of feature channels in each layer of the

network. On the basis of reducing the channel number of each layer, we kept the input and output channel numbers of LC-Block 4, LC-Block 6, and LC-Block 8 consistent. This design effectively reduces the model's memory consumption and actual operating energy consumption. In addition, the original YOLOv5 uses the Focus layer for downsampling the input feature map, but its "Slice" operation brings an unavoidable computational burden. Therefore, we replace the Focus layer with a 3×3 convolutional layer as the first layer of the network to improve the model's inference efficiency on actual devices.

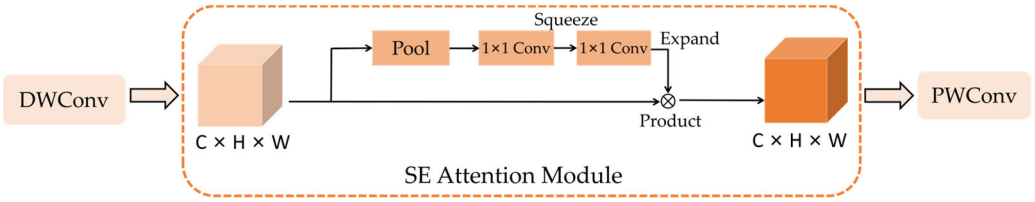


Figure 2. The architecture of LC-block. The light orange and orange cubes represent the input feature map and output feature map of the SE attention module, respectively.

Table 1. Specific Structure and Parameter Configuration of the lightweight Backbone. "$K \times K$" of LC-Block refers to the kernel size of DWConv, and "Utilization of SE Attention Module" indicates whether the corresponding LC_Block is inserted with SE attention module.

Module	Output Feature Map Size ($W \times H \times C$)	Number of Modules	Parameters [$K \times K$, *Stride*]	Utilization of SE Attention Module
Conv + BN + H-Swish	$320 \times 320 \times 32$	1	$[3 \times 3, 2]$	-
LC-Block 1	$160 \times 160 \times 64$	1	$[3 \times 3, 2]$	False
LC-Block 2	$160 \times 160 \times 64$	1	$[3 \times 3, 1]$	False
LC-Block 3	$80 \times 80 \times 128$	1	$[3 \times 3, 2]$	False
LC-Block 4	$80 \times 80 \times 128$	3	$[3 \times 3, 1]$	False
LC-Block 5	$40 \times 40 \times 256$	1	$[3 \times 3, 2]$	False
LC-Block 6	$40 \times 40 \times 256$	5	$[5 \times 5, 1]$	False
LC-Block 7	$20 \times 20 \times 512$	1	$[5 \times 5, 2]$	True
LC-Block 8	$20 \times 20 \times 512$	3	$[5 \times 5, 1]$	True
Conv + H-Swish + Dropout	$20 \times 20 \times 512$	1	$[1 \times 1, 1]$	-

3.2. Segmentation-and-Product Attention Mechanism

The attention mechanism is often used to improve the accuracy of deep neural networks [18–20]. However, most attention mechanisms come with a significant computational cost, making it challenging to apply them to lightweight models. Moreover, in violation detection tasks, the detection targets typically include workers, their safety equipment, or the fence that spans across a specific area. These targets are often concentrated in specific localized regions of surveillance footage, with a small proportion and low resolution within the frame. However, existing attention mechanisms usually directly process the entire surveillance image, attempting to learn local semantic information from the global semantics of the image. This approach might overlook the local spatial information within each feature channel, limiting their effectiveness in violation detection tasks.

To address this issue, we propose a lightweight attention mechanism called the Segmentation-and-Product (SP) Attention Mechanism, which is applicable to violation detection tasks. This mechanism takes into account the fact that the targets of violation detection tasks are concentrated in localized regions. It incorporates image segmentation operations to divide the input feature map into multiple region feature maps. Each segmented feature map is then subject to attention operations. This approach enables the model to directly and efficiently extract spatial local information, significantly improving the model's ability to extract target feature information from each region. Consequently, it enhances the detection accuracy of the target detection model in violation detection tasks. Additionally, in the attention operations of this mechanism, we utilize 1×1 convolutions to

compress the input feature map channels. This effectively reduces the computational and parameter complexity, allowing the model to improve target detection accuracy without imposing excessive computational burden.

As shown in Figure 3, the output tensor of the attention module has different colors in different positions in each channel. The parts of each channel that are closer to blue represent the background feature areas that have a smaller impact on detection accuracy [23], while the parts that are closer to red represent the important feature areas containing the targets such as people and safety helmets, which are assigned higher attention weights to help the model obtain more effective feature information.

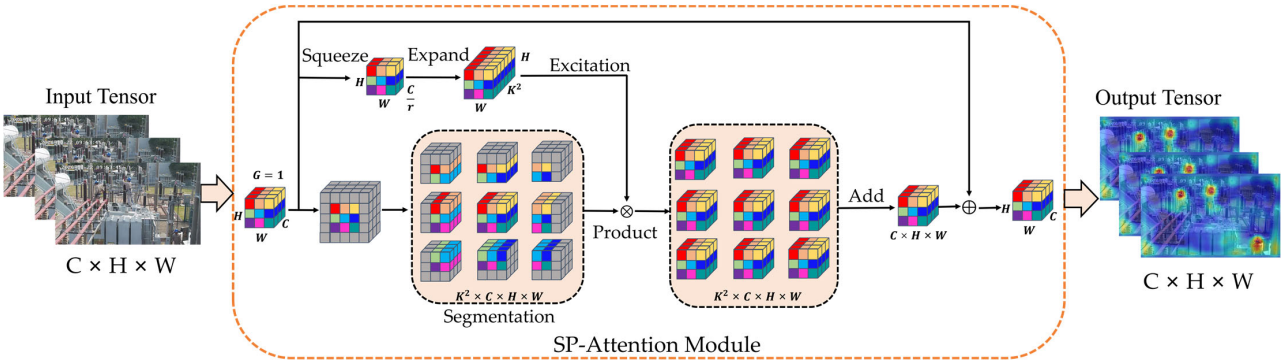


Figure 3. The structure of SP Attention module. The different colored blocks on the cubes represent parameters at different positions in a certain feature channel.

The SP Attention Mechanism first segments the input feature map into several feature maps with different spatial locations and then assigns different weights to each feature map to filter out important target features in different spatial positions. Taking Figure 3 as an example, the SP attention module takes an input image of size $C \times H \times W$. The hyperparameters in the figure are set as follows: $W = 3$, $H = 3$, $C = 3$, $K = 3$, $r = 2$, and $G = 1$. Here, K represents the size of the sliding window used to segment the feature map, r is the compression factor of the input feature map channel, and G is the number of groups to which the input feature map is sliced in the channel dimension. The mechanism performs the same attention operation on each group. The specific implementation steps of the attention mechanism are as follows:

(1) Segmentation: The input tensor is passed through the bottom branch. In the bottom branch, the input tensor is segmented into K^2 feature maps using a sliding window of size $K \times K$. Each feature map has a size of $C \times H \times W$.

(2) Squeeze, Expand, and Excitation: The input tensor is passed through the top branch. In the top branch, the channel number of the input tensor first is compressed to $1/r$ through a 1×1 convolution and then expanded to K^2. Finally, the attention filter is generated through the excitation operation.

(3) Product: The output of the top branch (the attention filter) and the bottom branch (the segmented feature maps) is multiplied element-wise using a product operation. As shown in Figure 4, the product operation between the attention filter and the segmented feature maps can be equivalently expressed as follows: Firstly, the attention filter is sequentially unfolded along the channel dimension and the segmented feature maps are divided into C groups according to the channel order. Then, the attention filter is multiplied with each group of the segmented feature maps element-wise, that is, the elements at the same position are multiplied. Finally, the weighted feature maps are output. The product operation realizes the filtering of the attention filter on the feature maps representing different regions.

(4) Add: The resulting feature maps of size $K^2 \times C \times H \times W$ generated by the product operation are added for normalization, and finally added to the input feature map to obtain the output feature map with the same shape as the input.

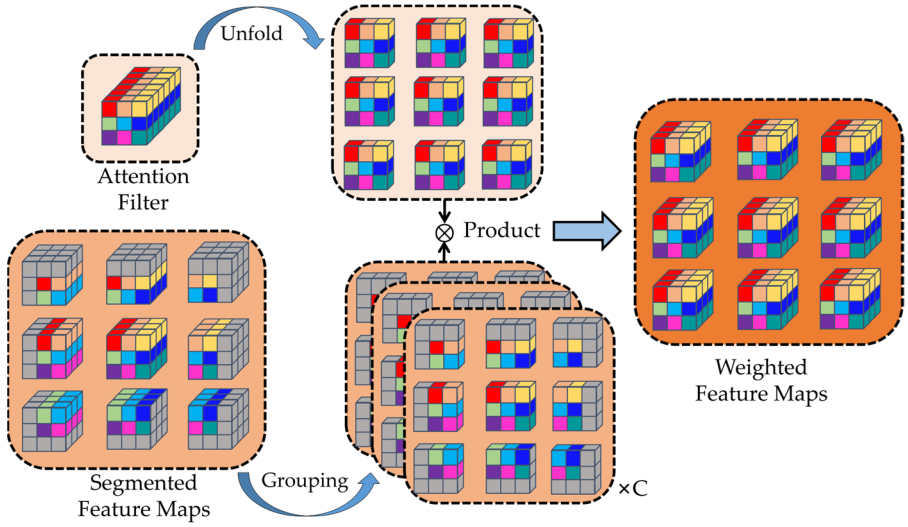


Figure 4. Diagram of the product operation.

The computation and parameter volume of this attention mechanism are constrained by hyperparameters, and the size of the model can be adjusted according to actual needs. The Parameter and FLOPs (Floating Point of Operations) of the mechanism are shown in Equations (1) and (2), respectively, and the scale of both primarily depends on hyperparameters r, G, and K. In network design, we can effectively change the size of the attention module by adjusting r, G, K, and the insertion position of the attention module, thus better balancing the contradiction between accuracy improvement and computational burden. This attention mechanism can achieve comprehensive and effective extraction of local image features while balancing accuracy, parameter volume, and computation. Therefore, compared with most other attention mechanisms, it is more suitable for lightweight object detection models.

$$\text{Parameter}_{\text{SP\_Attention}} = (1 \times 1 \times C \times \frac{C}{r}) + (1 \times 1 \times \frac{C}{r} \times \text{GK}^2) = \frac{C^2 + \text{CGK}^2}{r} \tag{1}$$

where C represents the number of channels in the input tensor, K represents the size of the sliding window used in the segmentation operation, r represents the compression ratio factor of the input feature map channels, and G represents the number of groups.

$$\begin{aligned} &\text{FLOPs}_{\text{SP\_Attention}} \\ &= \left(1 \times 1 \times C \times H \times W \times \tfrac{C}{r}\right) + \left(1 \times 1 \times \tfrac{C}{r} \times H \times W \times GK^2\right) + \left(G \times K^2 \times H \times W \times C\right) + \beta \\ &= \left(\tfrac{C^2 + CGK^2 + rCGK^2}{r}\right) HW + \beta \end{aligned} \tag{2}$$

where H and W represent the height and width of the input tensor, respectively, while β represents the computational cost of addition and other operations.

3.3. Lightweight Neck Network

In object detection tasks, feature fusion networks are widely used to improve the performance of object detection models. It is usually used as the Neck part of the model to fuse feature maps from different layers in the Backbone to obtain rich semantic and localization information. The Path Aggregation Network (PANet) [24] is a widely used feature fusion network in general-purpose object detection algorithms, which utilizes both top-down and bottom-up aggregation paths to fully fuse shallow high-resolution features and deep low-resolution features for more accurate object detection [25]. However, existing PANet feature fusion networks do not consider the model lightweight, and the computational and parameter costs need to be reduced and optimized. In this paper, we propose a lightweight

Neck network based on the PANet framework. For the network, we incorporate SP Attention modules and make them more lightweight by replacing the complex CSP module and conventional convolutions with Depthwise Separable Convolution. We also optimize the channel configuration in each layer and employ lighter-weight operators in the aggregation path. In contrast to the feature fusion networks commonly used in general-purpose object detection algorithms, this Neck network strikes a better balance between accuracy, speed, and complexity.

As shown in Figure 5, when a detection network is input with an image of resolution 640 × 640, the image will gradually decrease in resolution and become more abstract and semantically rich as it passes through the deeper layers of the Backbone network, while the localization information in the feature maps will gradually become blurry. The proposed Neck network in this paper introduces feature maps from different layers for information fusion, enabling the acquisition of more accurate and rich image features and target information.

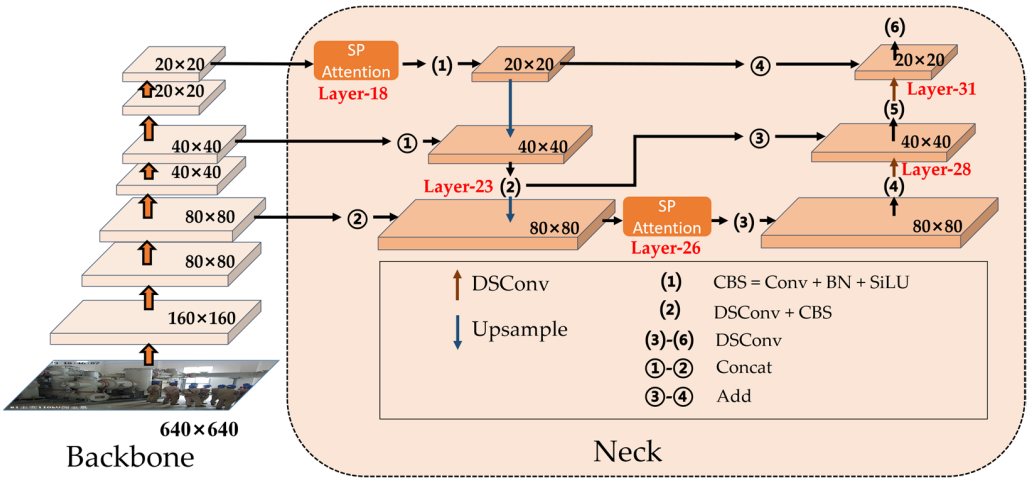


Figure 5. The structure of the lightweight Neck network. The red text indicates the layer number of the module within the entire model.

When constructing the Neck network, we first inserted SP attention modules at a connection between the Backbone and Neck (at the 18th layer), as well as within the Neck network (at the 26th layer), to enhance the model's ability to extract features from deep low-resolution features and shallow high-resolution features with relatively low computational cost. Next, we pruned the channel numbers in the Neck network's layers, while maintaining equal input and output channel numbers for the convolution modules at the 23rd, 28th, and 31st layers. This was performed to minimize the memory access cost of the network.

Finally, we lightened the Neck network by replacing the complex CSP module and the conventional convolution used as the upsampling operator with Depthwise Separable Convolution (DSConv). The specific structure of the DSConv used in this paper is shown in Figure 6, which consists of a Depthwise Convolution (DWConv) and a Pointwise Convolution (PWConv) [26]. Additionally, the Batch Normalization (BN) layer and Hard-Swish (H-Swish) activation function are connected after both convolutions to accelerate model training convergence and improve the model's expression ability [27]. In DWConv, each convolution filter only convolves with one channel of the input tensor, while in a conventional convolution, the filter needs to convolve with every channel of the input tensor. Therefore, compared with conventional convolution, DWConv significantly reduces computation and parameter costs. However, since DWConv independently convolves each channel of the input tensor, it cannot effectively extract the relationship information among different input channels. Therefore, after DWConv, PWConv is required to facilitate

inter-channel feature communication and compensate for the feature richness sacrificed by DWConv due to its lightweight design.

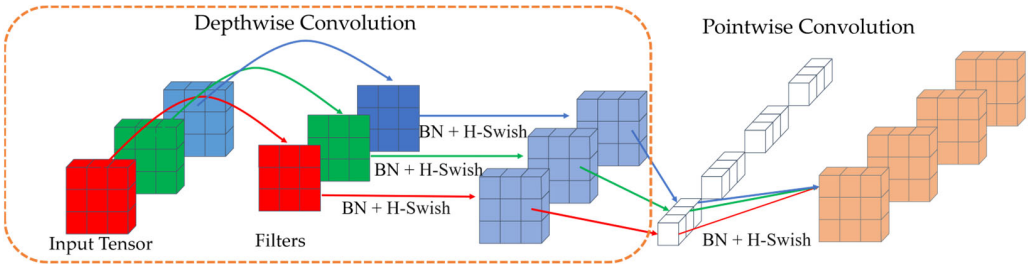


Figure 6. The structure of Depthwise Separable Convolution. The orange cuboids represent the output of this convolutional module.

In addition, we replaced the original Concat operation with the Add operation, which has smaller FLOPs and Parameter. The calculation processes for these two operations are shown in Equations (3) and (4), respectively. A comparison shows that, for the same size of input tensor, the Add operation does not increase the number of channels in the input tensor, and therefore, the number of convolutional kernels in the subsequent layers is only half that of the Concat operation. Consequently, the corresponding FLOPs and Parameter are also only half of that of the Concat operation.

$$\text{Calculation}_{\text{concat}} = \sum_{i=1}^{c} X_i * K_i + \sum_{i=1}^{c} Y_i * K_{i+c} \tag{3}$$

where X_i and Y_i are input feature maps from different branches, c is the number of channels in the input feature map, $*$ is the convolution operation, and K_i is the number of convolution kernels in the subsequent convolution layer.

$$\text{Calculation}_{\text{add}} = \sum_{i=1}^{c} (X_i + Y_i) * K_i = \sum_{i=1}^{c} X_i * K_i + \sum_{i=1}^{c} Y_i * K_i \tag{4}$$

4. Experimental Results

4.1. Experimental Setting

The experimental environment for this paper is Windows 10, 2.10 GHz Intel Xeon Silver 4110 CPU, NVIDIA GeForce RTX 2080 Ti GPU, Python 3.8.15, CUDA 11.3.1, cuDNN 8.2.1, and torch 1.13.0. We adopt transfer learning to train the model by loading pre-trained weights before training. The optimizer of the neural network model is SGD, with a learning rate of 0.01. The input size of the images is 640 × 640, and the batch size is 32. A total of 300 epochs of iterative training was conducted.

4.2. Experimental Data

4.2.1. VOC Dataset

The PASCAL VOC (Visual Object Classes) dataset is a classic benchmark dataset for object detection and semantic segmentation, which has been widely used for evaluating object detection algorithms. Therefore, we evaluated the general performance of our proposed algorithm on this dataset. The VOC dataset consists of seven versions from 2005 to 2012, and we used the commonly used 2007 and 2012 versions of the PASCAL VOC training and validation sets for model training. After model training, we performed a final evaluation on the test set of PASCAL VOC 2007. The VOC dataset used in this paper contains target information for 20 categories such as a car, dining table, and bus. The dataset includes a total of 21,503 images, consisting of 17,202 training images, 2150 validation images, and 2151 testing images. A sample of the dataset is shown in Figure 7.

Figure 7. PASCAL VOC dataset.

4.2.2. Security Monitor for Power Construction Dataset

As there is currently no publicly available dataset suitable for detecting violations of power grid workers, we require the construction of a dataset for training the model. In the process of constructing the dataset, we conducted a great deal of work, including image acquisition, image cleaning and annotation, and image normalization. We first collected 6550 pictures of power grid work sites from a certain power company, which were divided into two categories. The first category is high-resolution pictures taken by on-site inspection personnel with cameras, consisting of 5250 images with a resolution of 5184 × 3888 pixels, usually taken from horizontal or oblique angles relative to the target object. The second category is single-frame images captured from the power grid monitoring video, consisting of 1300 images with relatively small resolutions of 1280 × 720 pixels or 1920 × 1080 pixels, usually taken from an overhead angle relative to the target object. Afterward, we cleaned and annotated the collected images to obtain the Security Monitor for Power Construction dataset suitable for this study, which contains 1499 images and 7 categories, including Hat, Human, No-neckline, No-cuff, Safety belt, Seine, and Fence. A sample of the dataset is shown in Figure 8. The algorithm trained on the SMPC dataset is designed to detect violations in the attire and behaviors of workers. Specifically, it can detect whether a safety helmet is properly worn, whether the safety clothing is correctly worn (e.g., cuff and neckline), and determine if a worker has violated restricted areas by detecting their relative position to the Seine and Fence.

Figure 8. SMPC dataset.

In order to enhance the robustness and generalization of the trained algorithm and improve its performance in the presence of noise, environmental changes, and other abnormal circumstances, while avoiding overfitting during training, we employed data augmentation methods on the SMPC dataset. We applied various data augmentation operations such as Blurring, Salt-and-pepper noise addition, Brightening, Darkening, Equalization, Rotation, Flipping, Gaussian noise addition, and Translation, as illustrated in Figure 9. Through these data augmentation operations, we can simulate various abnormal conditions that may occur in real-world scenarios, including changes in brightness, viewpoint, and noise interference. Additionally, these approaches greatly increase the number of samples in the SMPC dataset, allowing the model to learn a more diverse range of real-world scenario information during training, and making the detection system deployed with this algorithm can operate continuously and efficiently. After data augmentation, the final dataset contained 10,693 images, including 7802 training images, 873 validation images, and 2018 testing images.

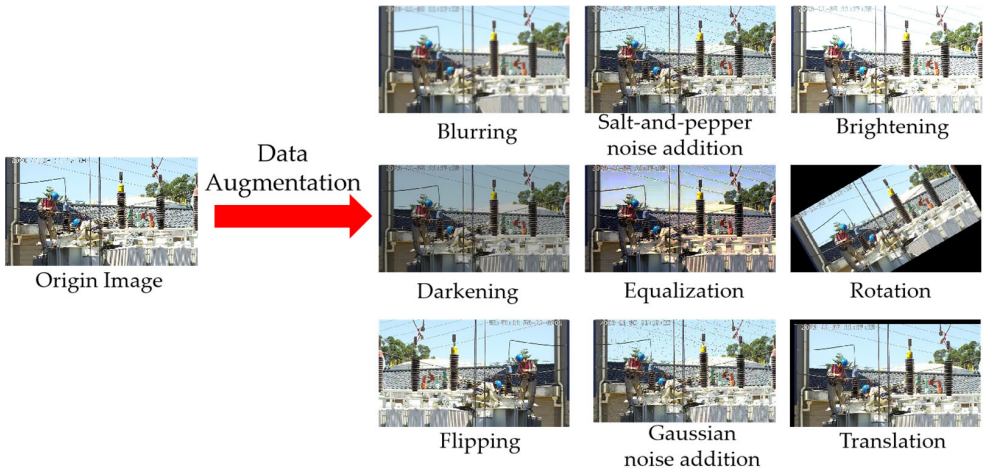


Figure 9. Data Augmentation methods for SMPC dataset.

4.3. Evaluation Indicators

4.3.1. Evaluation Metrics for Accuracy

The evaluation metrics for accuracy adopted in this paper include Precision, Recall, and mean Average Precision (mAP). Precision refers to the ratio of the number of correctly detected positive samples to the total number of detected positive samples by the model. Its calculation formula is shown in Equation (5).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5}$$

where TP denotes the number of true positives detected as positive and FP represents the number of true negatives detected as positive.

Recall is the proportion of true positives that were correctly detected by the model. In other words, Recall measures the model's ability to correctly identify positive instances. Its formula is shown in Equation (6).

$$\text{Recall} = \frac{TP}{TP + FN} \tag{6}$$

where FN represents the number of true positives that are incorrectly classified as negatives.

mAP is the average of the Average Precision (AP) for all categories. AP is the average Precision calculated at different Recall. Therefore, mAP is the average performance of the detection algorithm on different categories, as shown in Equation (7). mAP@0.5 is the mAP calculated at an IoU of 0.5, while mAP@0.5:0.95 refers to dividing the IoU threshold into 11 thresholds with an interval of 0.05 from 0.5 to 0.95, respectively, calculating the mAP at each threshold and averaging these mAP values.

$$\text{mAP} = \frac{\sum_{i=1}^{K} AP_i}{K} \tag{7}$$

where K is the number of all categories and AP_i represents the Average Precision of the i-th category.

4.3.2. Evaluation Metrics for Lightweight

FLOPs (Floating Point of Operations) and Parameter are the two most commonly used evaluation metrics for model complexity. FLOPs refer to the computational power required for the network's forward propagation [28], while Parameter refers to the total number of parameters involved in the computation of the model. If the detection algorithm includes L

convolution layers, the FLOPs of the algorithm can be calculated according to Equation (8) [29].

$$\text{FLOPs} \approx \sum_{l=1}^{L} \text{FLOPs}^l = 2 \times \sum_{l=1}^{L} \frac{\left(K^l\right)^2 \cdot H^l \cdot W^l \cdot C_{\text{in}}^l \cdot C_{\text{out}}^l}{\left(S^l\right)^2} \tag{8}$$

where K^l represents the convolution kernel size of the l-th convolutional layer, while H^l and W^l represent the height and width of the input tensor, respectively. C_{in}^l and C_{out}^l represent the number of input and output channels of the l-th convolutional layer, respectively, while S^l represents the convolution stride.

To measure the total number of learnable parameters in a detection model, we use Parameter as the evaluation metric. If the detection model contains L convolutional layers, its total number of parameters can be calculated using Equation (9).

$$\text{Parameter} = \sum_{l=1}^{L} \text{Parameter}^{\,l} \approx \sum_{l=1}^{L} \left(K^l\right)^2 \cdot C_{\text{in}}^l \cdot C_{\text{out}}^l \tag{9}$$

In addition, we introduce FPS [30] as a metric to evaluate the actual performance of the model, which represents the number of images that the model can detect per second. Its formula is as Equation (10):

$$\text{FPS} = \frac{T_{\text{total}}}{N_{\text{iamges}}} \tag{10}$$

where T_{total} represents the total inference time of the model and N_{images} represents the number of images that the model processes during inference.

4.4. SP Attention Module Experimental Analysis

To evaluate the impact of the proposed SP attention module on the SP-YOLO-Lite detection model, we first conducted experiments under four different conditions: "without inserting the SP attention module", "inserting an SP attention module at the 18th layer", "inserting an SP attention module at the 26th layer", and "inserting two SP attention modules". "Inserting two SP attention modules" refers to simultaneously inserting an SP attention module into the 18th and 26th layers of the network. We conducted these comparative experiments to verify the superiority of the attention module insertion method used in the proposed SP-YOLO-Lite.

The experimental results are shown in Tables 2 and 3. The results demonstrate that "inserting two SP attention modules" can achieve better detection accuracy compared with other insertion methods for both datasets, albeit with a slight increase in computation and parameter costs as well as a small decrease in FPS.

Table 2. The results of experiments on the VOC dataset with different attention module insertion methods and various attention modules. "Without attention" indicates that no attention module is inserted into the model, "18" and "26" represent the insertion positions of the attention modules at the 18th and 26th layers of the network, respectively.

Model	mAP @0.5(%)	mAP @0.5:0.95(%)	FLOPs ($\times 10^9$)	Paraments (Byte $\times 10^6$)	FPS (t/n)
SP-YOLO-lite (without attention)	77.1	52.1	4.8	2.32	46.5
+SP Attention (18)	78.2	54.1	4.9	2.39	47.4
+SP Attention (26)	78.1	54.1	5.0	2.33	47.8
+SP Attention (18,26)	78.4	54.3	5.0	2.40	43.5
+SE Attention (18,26)	75.1	48.4	4.8	2.36	45.2
+ECA Attention (18,26)	75.1	48.6	4.8	2.32	41.0
+CBAM Attention (18,26)	75.0	48.4	4.9	2.36	37.2

Table 3. The results of experiments on the SMPC dataset with different attention module insertion methods and various attention modules.

Model	mAP @0.5(%)	mAP @0.5:0.95(%)	FLOPs ($\times 10^9$)	Paraments (Byte $\times$ 10^6)	FPS (t/n)
SP-YOLO-lite (without attention)	82.5	55.2	4.7	2.30	40.3
+SP Attention (18)	83.3	55.2	4.8	2.37	36.5
+SP Attention (26)	83.6	55.1	4.9	2.32	30.8
+SP Attention (18,26)	84.8	57.6	5.0	2.39	39.4
+SE Attention (18,26)	82.8	55.3	4.8	2.34	37.6
+ECA Attention (18,26)	83.3	56.3	4.7	2.30	37.3
+CBAM Attention (18,26)	82.8	54.6	4.8	2.34	39.8

For the VOC dataset, compared to "without inserting the SP attention module", the mAP@0.5 and mAP@0.5:0.95 of the model using "inserting two SP attention modules" increased by 1.3% and 2.2%, respectively. Compared to "Inserting an SP attention module at the 18th layer", the mAP@0.5 and mAP@0.5:0.95 of the model increased by 0.2% and 0.2%, respectively, while compared to "Inserting an SP attention module at the 26th layer", the mAP@0.5 and mAP@0.5:0.95 of the model increased by 0.3% and 0.2%. Similarly, for the SMPC dataset, using "inserting two modules" increased the mAP@0.5 and mAP@0.5:0.95 by 2.3% and 2.4%, respectively, compared to "without inserting the module". Compared to "Inserting one module at the 18th layer", it increased the mAP@0.5 and mAP@0.5:0.95 by 1.5% and 2.4%, respectively, while compared to "Inserting one module at the 26th layer", the increase was 1.2% and 2.5%.

Moreover, we compared the performance of the proposed SP attention module with three mainstream attention modules, namely SE [18], ECA [20], and CBAM [19]. These three modules were inserted into the model's 18th and 26th layers in the same way as the SP-YOLO-Lite. The experimental results are shown in Tables 2 and 3. The results showed that the SP attention module can effectively improve the detection performance of the model compared to other attention modules in both datasets. For the VOC dataset, the model with SP modules inserted achieved mAP@0.5 and mAP@0.5:0.95 that exceeded those of the SE, ECA, and CBAM modules by 3.3%, 3.3%, 3.4%, and 5.9%, 5.7%, 5.9%, respectively. For the SMPC dataset, its mAP@0.5 and mAP@0.5:0.95 exceeded those of the SE, ECA, and CBAM modules by 2.0%, 1.5%, and 2.0%, and 2.3%, 1.3%, and 3.0%, respectively.

In addition, we use Grad-CAM (Gradient-weighted Class Activation Map) [31] to visualize the training weights of detection models with four types of attention modules: SE, ECA, CBAM, and SP (ours), in order to demonstrate the superiority of the SP attention module by comparison. The visualization results are shown in Figure 10, where the reddish areas indicate the regions that the attention mechanism pays more attention to, while the bluish areas receive less attention. The experimental results show that the SP attention mechanism can focus more on the target area compared to other attention mechanisms, thus enabling the detection model to extract more effective feature information.

4.5. Ablation Experiments

To validate the superiority of the lightweight Backbone and Neck networks proposed in this paper, we replaced the Backbone and Neck networks of YOLOv5s with the proposed Lite-Backbone and Lite-Neck networks, respectively, and conducted ablation experiments on the VOC and SMPC datasets. The experimental results are shown in Tables 4 and 5. Compared with the original YOLOv5s, the YOLO model using Lite-Backbone significantly reduced FLOPs and Parameters by approximately 40% and 30%, respectively, on both datasets, while maintaining comparable accuracy. The model using Lite-Neck sacrificed some accuracy for significant reductions in FLOPs and Parameters by approximately 30% and 40%, respectively, on both datasets. Moreover, when applying both Lite-Backbone and

Lite-Neck, FLOPs and Parameters of the model were greatly reduced by approximately 70%, with a small decrease in accuracy of approximately 3% on both datasets.

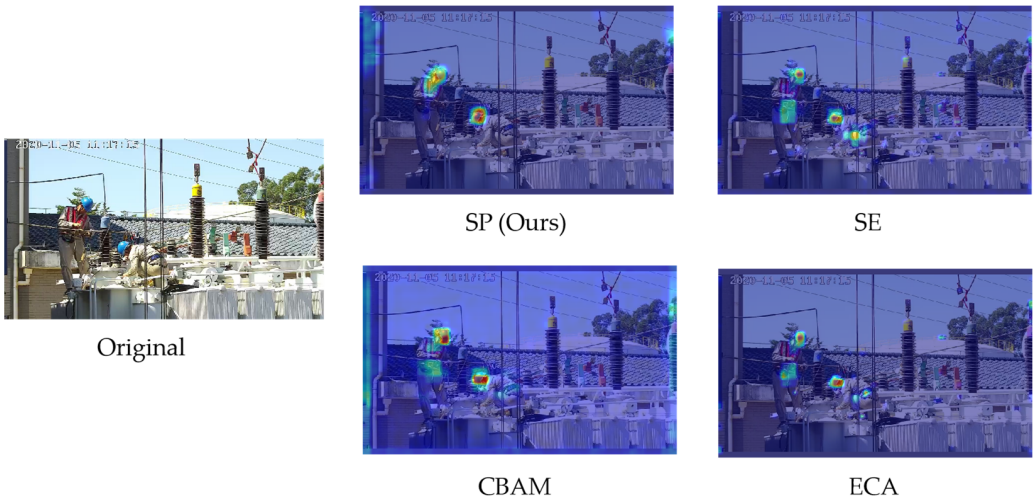


Figure 10. Grad-Cam Heat map comparison of various attention mechanisms.

Table 4. Ablation experiments of SP-YOLO-Lite on the VOC dataset.

Model	mAP @0.5(%)	mAP @0.5:0.95(%)	FLOPs ($\times 10^9$)	Paraments (Byte $\times 10^6$)	FPS (t/n)
YOLO V5s	81.5	59.6	15.9	7.06	49.3
+Lite-Backbone	79.3	57.0	9.5	5.03	37.5
+Lite-Neck	75.5	50.0	11.5	4.45	43.7
+Lite-Backbone + Lite-Neck	78.4	54.3	5.0	2.40	43.5

Table 5. Ablation experiments of SP-YOLO-Lite on the SMPC dataset.

Model	mAP @0.5(%)	mAP @0.5:0.95(%)	FLOPs ($\times 10^9$)	Paraments (Byte $\times 10^6$)	FPS (t/n)
YOLO V5s	87.6	65.9	15.8	7.03	44.1
+Lite-Backbone	86.3	62.4	9.5	4.99	50.0
+Lite-Neck	77.3	45.3	11.4	4.43	38.9
+Lite-Backbone + Lite-Neck	84.8	57.6	5.0	2.39	39.4

4.6. Comparison Experiments

To further validate the superiority of the proposed SP-YOLO-Lite model, we conducted performance comparison experiments on the VOC and SMPC datasets with other YOLOv5 series models, including YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, as well as YOLOv5-Lite$_e$, YOLOv5-Lite$_s$, YOLOv5-Lite$_c$, and YOLOv5-Lite$_g$ from YOLOv5-Lite [16] series. As shown in Tables 6 and 7, the experimental results demonstrate that the SP-YOLO-Lite model better balances the relationship between detection accuracy, model size, computational complexity, and detection speed (FPS) compared to other models. With smaller parameters and computational complexity, the SP-YOLO-Lite model achieves high detection accuracy and speed on both datasets.

Table 6. Comparison experiments of different target detection algorithms on the VOC dataset. "Size" refers to the storage space occupied by a model, which is the number of weight parameters in the model.

Model	mAP@0.5 (%)	mAP @0.5:0.95(%)	Size (MB)	FLOPs ($\times 10^9$)	Paraments (Byte $\times 10^6$)	FPS (t/n)
YOLOv5-Lite$_e$	69.4	42.9	1.8	2.7	0.73	41.2
YOLOv5-Lite$_s$	74.9	49.6	3.3	3.8	1.57	32.6
YOLOv5n	75.7	51.7	3.9	4.2	1.79	33.9
SP-YOLOv5-Lite (Ours)	78.4	54.3	5.1	5.0	2.40	43.5
YOLOv5-Lite$_c$	79.1	56.3	9.1	8.9	4.43	48.5
YOLOv5s	81.5	59.6	13.8	15.9	7.06	49.3
YOLOv5-Lite$_g$	81.7	60.9	11.3	15.9	5.51	45.5
YOLOv5m	84.4	65.7	40.3	48.1	20.93	47.6
YOLOv5l	86.6	68.8	88.7	108.0	46.21	26.2
YOLOv5x	87.5	70.5	165.3	204.2	86.30	21.6

Table 7. Comparison experiments of different target detection algorithms on the SMPC dataset.

Model	mAP@0.5 (%)	mAP @0.5:0.95(%)	Size (MB)	FLOPs ($\times 10^9$)	Paraments (Byte $\times 10^6$)	FPS (t/n)
YOLOv5-Lite$_e$	82.0	50.5	1.8	2.7	0.72	45.0
YOLOv5-Lite$_s$	82.1	56.1	3.5	3.7	1.55	42.6
YOLOv5n	83.1	56.5	3.9	4.2	1.77	47.2
SP-YOLOv5-Lite (Ours)	84.8	57.6	4.9	5.0	2.39	39.4
YOLOv5-Lite$_c$	87.3	65.2	9.2	8.7	4.39	36.6
YOLOv5s	87.6	65.9	14.4	15.8	7.03	44.1
YOLOv5-Lite$_g$	87.5	66.7	11.4	15.8	5.48	33.6
YOLOv5m	87.7	70.3	42.2	47.9	20.88	28.4
YOLOv5l	88.0	73.3	92.9	107.7	46.14	23.8
YOLOv5x	88.1	73.9	173.1	203.9	86.21	19.6

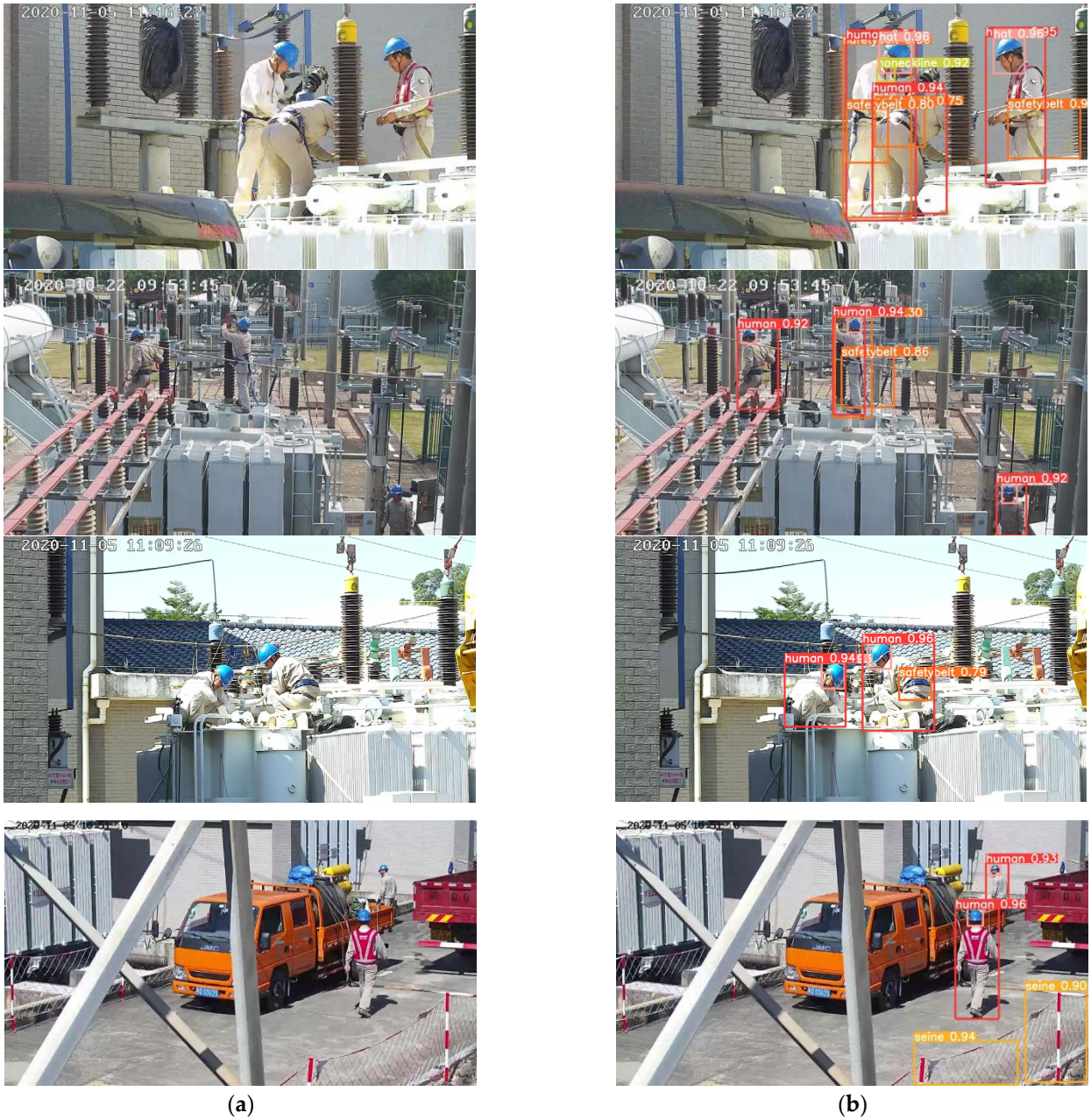
As shown in Table 6, compared to the YOLOv5-Lite$_e$, YOLOv5-Lite$_s$, and YOLOv5n with similar sizes, the SP-YOLO-Lite model achieved an improvement of 9%, 3.5%, 2.7% in mAP@0.5 and 11.4%, 4.7%, 2.6% in mAP@0.5:0.95, respectively, on the VOC dataset, and an improvement of 2.3, 10.9, and 9.6 in detection speed, respectively. Compared to larger models such as YOLOv5-Lite$_c$, YOLOv5s, YOLOv5-Lite$_g$, YOLOv5m, YOLOv5l, and YOLOv5x, the SP-YOLO-Lite model achieved high accuracy and detection speed while significantly reducing FLOPs and Parameters, by 42.5–97.6% and 45.6–97.2%, respectively.

Similarly, as shown in Table 7, compared to YOLOv5-Lite$_e$, YOLOv5-Lite$_s$, and YOLOv5n with similar sizes, the model achieved an improvement of 2.8%, 2.7%, and 1.7% in mAP@0.5 and 7.1%, 1.5%, and 1.1% in mAP@0.5:0.95, respectively, on the SMPC dataset. Compared to YOLOv5-Lite$_c$, YOLOv5s, YOLOv5-Lite$_g$, YOLOv5m, YOLOv5l, and YOLOv5x, the model significantly reduced FLOPs and Parameters, by 42.5–97.5% and 45.6–97.2%, respectively.

4.7. Visualization Results

Figure 11 shows the recognition results of SP-YOLO-Lite on four randomly selected images in the SMPC dataset. Figure 11b not only includes detection boxes for different targets but also contains information on the category and confidence of each detection box, where confidence indicates the predicted probability that the target belongs to that category. By comparing Figure 11a with Figure 11b, it can be observed that SP-YOLO-Lite accurately identifies targets such as a hat, human, and safety belt without missing any detection or causing false alarms. Moreover, SP-YOLO-Lite can accurately recognize small targets such

as a safety belt with partial occlusion and has high confidence, which further demonstrates the superiority of the SP-YOLO-Lite algorithm.



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

Figure 11. Visualization Results. (**a**) Original picture; (**b**) Visualization Results of our proposed SP-YOLO-Lite algorithm.

4.8. Robustness Study

To evaluate the performance of the proposed algorithm under various conditions and potential challenges, we visualized the recognition results of SP-YOLO-Lite on a sample image from the SMPC dataset, both in its original form and after undergoing data augmentation methods such as Blurring and Salt-and-pepper noise addition. The visualization results are shown in Figure 12. By comparing the algorithm's recognition results on the original and augmented images in Figure 12, it is evident that SP-YOLO-Lite can accurately detect objects in images processed with different data augmentation operations, without any missed detections or false positives. This demonstrates the algorithm's robustness and generalization capabilities, as it maintains good detection performance even in the presence

of noise interference and other abnormal circumstances. Therefore, the deployment of this algorithm in a detection system ensures it can easily handle most detection scenarios, without requiring frequent maintenance, while operating continuously and efficiently.

Figure 12. The visualization results of images processed using various data augmentation methods.

4.9. Algorithm Deployment

To evaluate the practical application performance of the proposed SP-YOLO-Lite algorithm, it was deployed on the NVIDIA Jetson AGX Orin embedded platform in this paper. The NVIDIA Jetson AGX Orin is an embedded AI computing device known for its low power consumption and compact size. The hardware configuration information of the platform is presented in Table 8.

Table 8. The hardware configuration information of the NVIDIA Jetson AGX Orin embedded platform.

Name	Configuration Information
GPU	NVIDIA Ampere Architecture GPU
Maximum GPU Frequency	930 MHz
CPU	Octa-core Arm® Cortex®-A78AE 64-bit CPU
Maximum CPU Frequency	2.2 GHz
Memory	32 GB
Storage	64 GB
Power	15 W–40 W

Due to the limited computational power of embedded platforms, deploying object detection algorithms on such a platform requires lower power consumption, smaller models, and faster detection speed. Therefore, in this paper, the SP-YOLO-Lite model was further optimized using the TensorRT deep learning inference framework specifically for the deployed NVIDIA platform. In this process, we not only reduced the precision of the model weights through quantization to improve model inference efficiency but also exported the model in a specific format for efficient execution on the targeted platform.

TensorRT is a deep learning inference engine developed by NVIDIA for high-performance inference. Models exported with TensorRT can be applied to NVIDIA Jetson series embedded platforms such as Jetson Nano and Jetson Xavier NX. For other embedded platforms, such as ARM, TPU, and FPGA, we can choose the appropriate inference framework (such as OpenVINO) that is compatible with the platform to achieve efficient deployment of the algorithm on different embedded platforms.

To evaluate the comprehensive performance of the algorithm on the embedded platform, we standardized the input image resolution to 640 × 640 and evaluated the overall system performance using the SMPC dataset. The specific performance results are shown in Table 9.

Table 9. The performance comparison of optimized models with different weight quantization precisions.

Weight Quantization Precision	mAP @0.5(%)	mAP @0.5:0.95(%)	FPS (t/n)	Power (W)	Efficiency (FPS/W)
FP32	79.80	52.30	55.25	20.112	2.747
FP16	79.70	52.40	60.61	18.992	3.191
INT8	70.00	40.80	57.14	17.885	3.195

According to Table 9, it can be observed that the optimized models with different weight quantization precisions can achieve high efficiency on embedded platforms with power consumption ranging from 17.885 W to 20.112 W, running at a speed of at least 55 FPS, while maintaining a detection accuracy of over 70%. Therefore, the SP-YOLO-Lite algorithm exhibits lower model complexity and computational resource requirements compared to general-purpose object detection algorithms, enabling real-time violation detection on most resource-limited embedded platforms.

5. Conclusions

To achieve low-cost deployment and efficient operation of object detection algorithms on embedded platforms for safety monitoring systems at construction sites, this paper proposes a lightweight violation detection algorithm called SP-YOLO-Lite. The algorithm is built on the YOLOv5s framework and integrates a lightweight Backbone network based on the LC-Block, a lightweight Neck network based on the DSConv module, and a new attention module called the Segmentation-and-Product attention module. The lightweight Backbone network is directly stacked with LC-Block, a lightweight network module based on Depthwise Separable Convolution. It exhibits low network fragmentation and high inference efficiency. By replacing the backbone of the YOLOv5 algorithm with the Lite-Backbone, the algorithm achieves a 30% reduction in Parameters and a 40% reduction in FLOPs while maintaining similar accuracy. The Segmentation-and-Product attention mechanism innovatively incorporates image segmentation operations. It segments the input image and performs attention operations on each segmented region, thereby capturing sufficient local spatial feature information and effectively improving the detection accuracy of the model in violation detection tasks. The lightweight Neck network is based on PANet and constructed by introducing the DSConv module and SP attention module. We further optimize the channel configuration of each layer of the Neck network. By replacing the Neck of the YOLOv5 algorithm with the Lite-Neck, the algorithm achieves a 40% reduction in Parameters and a 30% reduction in FLOPs while maintaining similar accuracy. Comparative experimental results show that the proposed SP-YOLO-Lite achieves similar detection accuracy on the VOC and SMPC datasets compared to the YOLO V5s baseline network while significantly reducing FLOPs and parameters by approximately 70%, resulting in a significant decrease in deployment costs.

In the future, we will further optimize the SP-YOLO-Lite algorithm to enhance its performance and broaden its application scope. Firstly, we will collect relevant images to further expand the SMPC dataset and include other types of violation behaviors that may occur in real-world scenarios, thereby increasing the detectable violation types of the SP-YOLO-Lite algorithm. Secondly, we will further optimize the existing network structure and parameter configuration of SP-YOLO-Lite to improve its detection accuracy and speed. Finally, we will explore the use of other inference frameworks such as OpenVINO to optimize the SP-YOLO-Lite algorithm for compatibility with other types of embedded platforms.

Author Contributions: Conceptualization, Z.H., J.W., and L.S.; methodology, Z.H. and J.W.; software, Z.H. and J.W.; validation, Z.H., J.W., and Y.X.; formal analysis, Z.H. and J.W.; investigation, Z.H. and J.W.; resources, Z.H. and J.W.; data curation, Z.H. and J.W.; writing—original draft preparation, Z.H. and J.W.; writing—review and editing, Z.H.; visualization, Z.H.; supervision, L.S., T.L., and X.H.; project administration, Z.H.; funding acquisition, L.S. and T.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant No. 61903315 and the Natural Science Foundation of the Department of Science and Technology of Fujian Province under Grant No. 2022J011255; and in part by the Foundation for Science and Technology Cooperation Program of Longyan under Grant No. 2020LYF16004.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

Acknowledgments: We thank all reviewers for their comments and Fujian Xiamen State Grid Corporation for their support.

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

References

1. Zhu, M.; Li, F.; Zhou, X. Research on the anti habitual violation of power enterprises. In Proceedings of the 2015 International Conference on Economics Social Science, Arts, Education and Management Engineering, Xi'an China, 12–13 December 2015; pp. 614–617.
2. Liu, X. Research on Assembly Line Dynamic Target Detection Algorithm Based on YOLOv3. Master's Thesis, Changchun University of Science and Technology, Changchun, China, June 2021.
3. Murthy, C.B.; Hashmi, M.F.; Bokde, N.D.; Geem, Z.W. Investigations of object detection in images/videos using various deep learning techniques and embedded platforms—A comprehensive review. *Appl. Sci.* **2020**, *10*, 3280. [CrossRef]
4. Hossain, S.; Lee, D.J. Deep learning-based real-time multiple-object detection and tracking from aerial imagery via a flying robot with GPU-based embedded devices. *Sensors* **2019**, *19*, 3371. [CrossRef] [PubMed]
5. Park, M.W.; Elsafty, N.; Zhu, Z. Hardhat-wearing detection for enhancing on-site safety of construction workers. *J. Constr. Eng. Manag.* **2015**, *141*, 04015024. [CrossRef]
6. Wang, L.; Duan, J.; Xin, L. YOLOv5 Helmet Wear Detection Method with Introduction of Attention Mechanism. *Comput. Eng. Appl.* **2022**, *58*, 303–312.
7. Zhang, B.; Sun, C.F.; Fang, S.Q.; Zhao, Y.H.; Su, S. Workshop safety helmet wearing detection model based on SCM-YOLO. *Sensors* **2022**, *22*, 6702. [CrossRef] [PubMed]
8. Wang, L.; Zhang, X.; Yang, H. Safety Helmet Wearing Detection Model Based on Improved YOLO-M. *IEEE Access* **2023**, *11*, 26247–26257. [CrossRef]
9. Shin, D.J.; Kim, J.J. A Deep Learning Framework Performance Evaluation to Use YOLO in Nvidia Jetson Platform. *Appl. Sci.* **2022**, *12*, 3734. [CrossRef]
10. 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, 26 June–1 July 2016; pp. 779–788.
11. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Hawaii, HI, USA, 21–26 July 2017; pp. 7263–7271.
12. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
13. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
14. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A single-stage object detection framework for industrial applications. *arXiv* **2022**, arXiv:2209.02976.
15. 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 Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
16. Chen, X.; Gong, Z. YOLOv5-Lite: Lighter, Faster and Easier to Deploy. 2021. Available online: https://pythonawesome.com/yolov5-lite-lighter-faster-and-easier-to-deploy/ (accessed on 25 May 2022).
17. Li, J.; Ye, J. Edge-YOLO: Lightweight Infrared Object Detection Method Deployed on Edge Devices. *Appl. Sci.* **2023**, *13*, 4402. [CrossRef]
18. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7132–7141.
19. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
20. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient channel attention for deep convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Online, 23–28 August 2020; pp. 11534–11542.

21. Cui, C.; Gao, T.; Wei, S.; Du, Y.; Guo, R.; Dong, S.; Lu, B.; Zhou, Y.; Lv, X.; Liu, Q.; et al. PP-LCNet: A lightweight CPU convolutional neural network. *arXiv* **2021**, arXiv:2109.15099.
22. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.
23. Huang, Z.; Su, L.; Wu, J.; Chen, Y. Rock Image Classification Based on EfficientNet and Triplet Attention Mechanism. *Appl. Sci.* **2023**, *13*, 3180. [CrossRef]
24. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8759–8768.
25. Shi, W.; Shi, Y.; Zhu, D.; Zhang, X.; Li, J. Traffic Sign Instances Segmentation Using Aliased Residual Structure and Adaptive Focus Localizer. In Proceedings of the 2022 26th International Conference on Pattern Recognition (ICPR), Quebec, QC, Canada, 21–25 August 2022; pp. 3676–3685.
26. Liu, C.; Yang, D.; Tang, L.; Zhou, X.; Deng, Y. A Lightweight Object Detector Based on Spatial-Coordinate Self-Attention for UAV Aerial Images. *Remote Sens.* **2022**, *15*, 83. [CrossRef]
27. Xie, Y.; Zhu, J.; Cao, Y.; Zhang, Y.; Feng, D.; Zhang, Y.; Chen, M. Efficient video fire detection exploiting motion-flicker-based dynamic features and deep static features. *IEEE Access* **2020**, *8*, 81904–81917. [CrossRef]
28. Nahmias, M.A.; De Lima, T.F.; Tait, A.N.; Peng, H.T.; Shastri, B.J.; Prucnal, P.R. Photonic multiply-accumulate operations for neural networks. *IEEE J. Sel. Top. Quantum Electron.* **2019**, *26*, 7701518. [CrossRef]
29. Li, Y.; Chen, Y.; Dai, X.; Chen, D.; Liu, M.; Yuan, L.; Liu, Z.; Zhang, L.; Vasconcelos, N. Micronet: Improving image recognition with extremely low flops. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 468–477.
30. Meng, C.; Wang, Z.; Shi, L.; Gao, Y.; Tao, Y.; Wei, L. SDRC-YOLO: A Novel Foreign Object Intrusion Detection Algorithm in Railway Scenarios. *Electronics* **2023**, *12*, 1256. [CrossRef]
31. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 618–626.