electronics

Article

Attention-Enhanced Lightweight One-Stage Detection Algorithm for Small Objects

Nan Jia [1], Zongkang Wei [1] and Bangyu Li [2,*]

[1] BeiJing Institute of Aerospace Control Device, Beijing 100854, China
[2] Institute of Automation Chinese Aacademy of Sciences, Beijing 100098, China
* Correspondence: bangyu.li@ia.ac.cn

Abstract: The majority of object detection algorithms based on convolutional neural network are focused on larger objects. In order to improve the accuracy and efficiency of small object detection, a novel lightweight object detection algorithm with attention enhancement is proposed in this paper. The network part of the proposed algorithm is based on a single-stage framework and takes MobileNetV3-Large as a backbone. The representation of shallower scale features in the scale fusion module is enhanced by introducing an additional injection path from the backbone and a detection head specially responsible for detecting small objects is added. Instead of pooling operators, dilated convolution with hierarchical aggregation is used to reduce the effect of background pixels on the accuracy of small object locations. To improve the efficacy of merging, the spatial and channel weights of scale features are modified adaptively. Last but not least, to improve the representation of small objects in the training datasets, the Consistent Mixed Cropping method is also proposed. The small labels of standard datasets are expanded with the self-collected samples for the training of the algorithm network. According to the test results and visualization on the 64-Bit Extended (X86-64) platform and embedded Advanced RISC Machine (ARM) platform, we find that the average accuracy (mAP) of the proposed algorithm is 4.6% higher than YOLOv4 algorithm, which achieves better small object detection performance than YOLOv4 algorithm, and the computational complexity is only 12% of YOLOv4 algorithm.

Keywords: small object detection; attention mechanism; lightweight; embed inference test

Citation: Jia, N.; Wei, Z.; Li, B. Attention-Enhanced Lightweight One-Stage Detection Algorithm for Small Objects. *Electronics* **2023**, *12*, 1607. https://doi.org/10.3390/electronics12071607

Academic Editor: Jungong Han

Received: 3 March 2023
Revised: 24 March 2023
Accepted: 28 March 2023
Published: 29 March 2023

1. Introduction

The detection of small objects has important implications for research and practical applications. There will be modest obstacles on the airport runway's road pavement, such as cobblestones, screws, and nails. If these minute foreign objects on the runway can be precisely detected, major accidents can be avoided. In the field of automatic driving, it is vital that the vehicle's vision sensor accurately detects small objects that could cause traffic accidents. In the field of industrial automation, tiny defects (such as burrs and fractures) visible on the surface of the material are detected using small object detection technology, thereby preventing larger financial losses. Finding microscopic lesions in medical image analysis is crucial for identifying early diseases and enhancing the effectiveness of therapy. In conclusion, small target detection technology has a multitude of application possibilities, application value, and significant implications for research across a wide spectrum of disciplines [1–4].

Due to the small proportion of small objects within image pixels and the absence of evident pixel characteristics, the detection of small objects remains a challenge in the field of object detection. First, the outline of small objects in the image is not distinct, making it challenging to train a neural network to differentiate between objects and backgrounds. Second, in the actual world, small objects, such as pedestrians, vehicles, birds, etc., are prone to congregate in the form of aggregations. Small objects of the same type are typically

ignored by neural networks or confused with other objects, resulting in missed objects or erroneous alarms. In addition, large objects in the datasets are typically designated as positive samples, whereas the proportion of negative samples (small objects) is not particularly high. The sample imbalance can readily hinder the ability of one single threshold to differentiate between positive and negative samples, resulting in the loss of small objects [4].

Since object detection is the starting point for more difficult tasks like semantic perception and object tracking, it is a major research area in the field of computer vision. In the field of computer vision, there are two types of definitions for minuscule objects based on the pixel size of the object in the image: relative scale and absolute scale definition. The relative scale specification for small objects is the ratio of pixels covered by the object to the total number of pixels in the image in the range of 0.08%∼0.58%. Objects with resolutions smaller than 32 × 32 from the Microsoft Common Objects in Context [5] (MS-COCO) datasets are the most prevalent example of absolute scale. In this paper, the definition of absolute scale serves as the dividing line between minor objects.

Object detection algorithms based on deep learning typically consist of one or two stages. The primary difference between the two was whether or not the location of objects was inferred by a convolutional neural network branch operating alone. Region Convolutional Neural Network [6](R-CNN) and its improved algorithms [7,8] are exemplary of two-stage object detection techniques. This method generates a provisional prediction of the object location by first constructing a suggestion box for the object using a region proposal network. After modifying the boundary location of the region features derived by clipping the suggestion box, the object category is regressed. Common one-stage algorithms include You Only Look Once [9–11] (YOLO) and its derivative series, as well as Single Shot MultiBox Detector [12,13] (SSD). The SSD employs the frame mechanism of Faster Region Convolutional Neural Network (FasterRCNN) as a reference among them in order to anticipate quickly and determine the object location with reasonable precision. After SSD incorporates a multi-scale feature layer for object detection, the subsequent YOLO derivative algorithm employs additional high-level properties of the backbone network to improve the detection capability of small objects.

We propose a one-stage algorithm for the detection of small objects based on YOLOv4. Through network structure and dataset augmentation, the proposed algorithm improves the precision and efficiency of small object detection. First, in terms of the algorithm network, we employ the MobileNetV3-Large [14] as the backbone and the depthwise separable convolution and channel attention mechanisms to enhance the efficacy of feature extraction. Due to the shallow features' smaller receptive fields, small objects can be detected more effectively. We have therefore increased the injection of shallow features in the scale fusion module and the depth of the feature layer. At the end part of the scale fusion module, a head for detecting small objects is also added. The spatial attention mechanism is used to increase the shallow network's tendency to fixate on objects and to improve the efficiency of scale fusion. We use dilated convolution with hierarchical aggregation to improve the spatial structure information of the feature layer and eliminate the interference of background pixels on small objects.

In terms of datasets augmentation, in addition to introducing the Mixup [15] and Mosaic [11] methods, we also proposed the Consistent Mixed Cropping (CMC) method to improve the network's ability to identify small objects and increase the diversity of small labels. In addition, we have counted the distribution of sample labels in PASCAL VOC [16] (PASCAL VOC represents Pattern Analysis, Statistical Modelling, and Computational Learning Visual Object Classes. Subsequently referred to as VOC) datasets and discover that the labels of a large object made up the majority in VOC datasets (small objects make up 3.5%). Large objectives are straightforward to identify as positive samples [17]. In order to reduce the disparity between positive and negative samples, we added self-collected supplementary data to the training procedure to increase the number of small samples.

The proposed algorithm is trained on 64-Bit Extended (x86-64) platform and evaluated on embedded Advanced RISC Machine (ARM) platform. According to the test results, the Mean Average Precision (mAP) of the proposed algorithm is 4.6% greater than that of the original algorithm, YOLOv4. Among these, the Average Precision (AP) of the top six categories with small objects has been improved. Then, we administer inference tests using randomly selected test sets images. The visualization results reveal that the proposed algorithm is still capable of accurate recognition despite the difficulty of judging dense small object networks (the confidence level is close to 0.5). The algorithm has 76.4% fewer multiplier accumulators and 32.5% fewer network parameters than YOLOv4, respectively. It demonstrates that the proposed algorithm is less complex than the original algorithm. The convolutional neural network of proposed algorithms, YOLOv4 and YOLOv4-tiny, are then evaluated on the x86-64 and embedded ARM platform, respectively. The proposed algorithm's average frame rate is comparable to YOLOv4-tiny.

The contributions in this paper are listed as follows:

1. By introducing an attention mechanism, enriching shallow scale features, and boosting context interaction, a novel lightweight small object detection algorithm based on a one-stage framework is proposed, which improves the accuracy and efficiency of small object detection;
2. We propose a new method for expanding the diversity of small samples in training data in order to increase the network's ability to learn from small samples. Simultaneously, we utilize self-collected sample data to increase the number of small identifiers in the training data;
3. On the embedded ARM platform, the effectiveness of the proposed algorithm is evaluated on Neural Compute Stick (NCS2), and it can serve as a reference for future work.

2. Related Work

2.1. Multi-Scale Semantic Fusion

Object detection was made easier with the use of multi-scale fusion [4,9–11,15]. Images of various sizes were created by scaling the source image. Following that, feature maps of various sizes were created based on images of those sizes. Scale-by-scale processing and feature splicing were the last steps in processing the feature map. Features fusion forces feature maps to consider both high-level semantics and shallow details. At the same time, there was no tradeoff made in terms of expression capacity, speed, or resource usage. Feature Pyramid Networks (FPN) [18] was a common multi-scale fusion model, as seen in Figure 1. By stacking adjacent layer features, FPN improved the ability of feature maps to express multi-scale semantic information. Inner Outside Net (ION) [19] separated ROI (Region of Interest) from the feature map of various sizes and detected objects after fusing features from multiple scales. Any size image could be processed using Spatial Pyramid Pooling (SPP) [20], which used spatial pyramid pooling to fuse multi-scale information and enhance object detection performance. In order to improve the ability to extract features, the Atrus Spatial Pyramid Pooling (ASPP) [21] used in the DeepLabV3 algorithm swapped out the three spatial pooling layers in the SPP for three dilated convolutions with different expansion rates. By overcoming the conflict between picture resolution and receptive field, ASPP enhanced object detection performance.

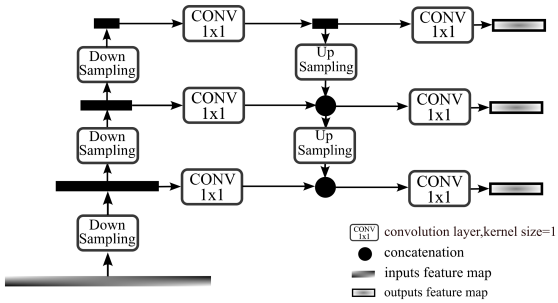

Figure 1. The overview of FPN. The FPN architecture integrates both high-level semantics and shallow details in the form of a feature graph pyramid.

2.2. Attention Mechanism

The network could achieve adaptive attention using the attention mechanism [22–24]. The emphasis of the network could change in accordance with the input image samples, which enabled the neural network to identify useful patterns in challenging situations. By dynamically altering the 'relevant information' of the input image, the attention mechanism improves the deep neural network's learning effectiveness. Convolutional Block Attention Module (CBAM [22], as shown in Figure 2), which combined the Spatial Attention Module (SAM) and Channel Attention Module (CAM), was a common attention mechanism module. To increase the effectiveness of object recognition, the Efficient Channel Attention (ECA) [23] module of the ECAnet employed the non-dimensional local attention interaction technique and adaptive 1×1 convolution. The Split-Attention (SA) module in Split-AttentionNetworks (ResNeSt) [25] divided the feature map into groups based on the channel dimension and weights it based on the overall context. The aforementioned algorithms transformed the deep learning model into a particular pattern by adding an attention strategy to the network, increasing the precision and effectiveness of object recognition.

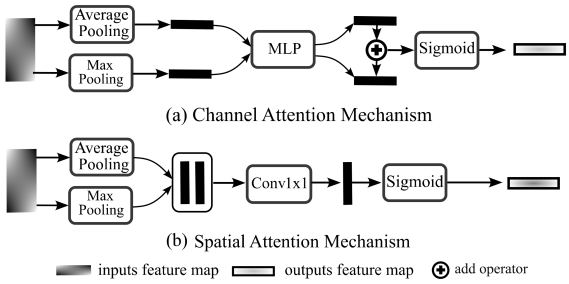

Figure 2. The overview of CAM and SAM. Whereas SAM focuses on spatial direction, CAM emphasizes the significance of channel direction.

2.3. Depthwise Separable Convolution and MobileNet

The main principle of deepwise separable convolution was to divide the conventional convolution operation into depthwise and pointwise convolutions. Deep separable convolution had fewer parameters and lower operating costs than traditional convolution. It also significantly reduced the number of network structure blocks and improves the effect of feature extraction. The Inverted Residuals (IR, as depicted in Figure 3, a module of MobilenetV2 [26] and the attention mechanism of Squeeze-and-Excitation Networks (SENet) [24] channel were combined in MobilenetV3 [14]. To enhance the quality of the representation produced by the network, SENet explicitly models the correlation between network convolution feature channels. According to the relevance of each feature channel, which was automatically determined through learning, relevant characteristics were then promoted, while features that are not pertinent to the work at hand are suppressed [24].

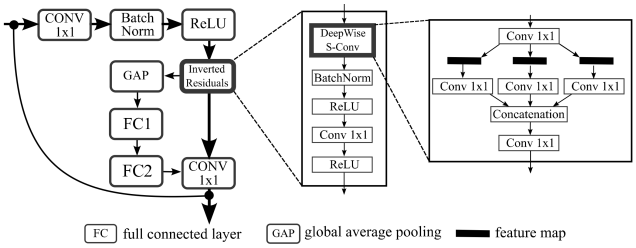

Figure 3. The overview of IR block. IR block implemented deepwise separable convolution and drew on the idea of the SENetalgorithm. DeepWise S-Conv represents a deepwise separable convolution block.

3. Proposed Algorithm

Figure 4 illustrates the division of our algorithm into the training and prediction operations. In order to increase the variety of samples in training datasets, we introduce a series of data augmentation methods (including Mixup [15], Mosaic [11], and Random Transformations (RT) of lighting and geometry) during the training phase. We also suggest a data augmentation method Consistent Mixed Cropping, which uses the prior knowledge of objects to strengthen the interaction between small sample labels and the background. In order to remedy the imbalance between positive and negative samples, which is detrimental to small object detection, we also increased the number of small labels in the sample image by self-collected samples.

The anchor mechanism and single-level object detection framework YOLOv4 are used as references by the algorithm network. Smaller receptive fields and higher resolution are correlated with shallow features in the deep learning network, which is advantageous for the detection of small objects. The shallow features also include a large number of parameter correlations, which makes the network model relatively complex and slows down inference. We make a trade-off between the proportion of shallow features and the speed of inference to resolve this issue. We take MobileNetV3-Large, a compact and effective backbone network. The decoding module's task is to decode the characteristics produced by the algorithm network's detection head and apply Non-Maximum Suppression (NMS) [9–11] to the prediction frame. We include a header for recognizing small objects in the decoding module using the YOLOv4 approach as a model.

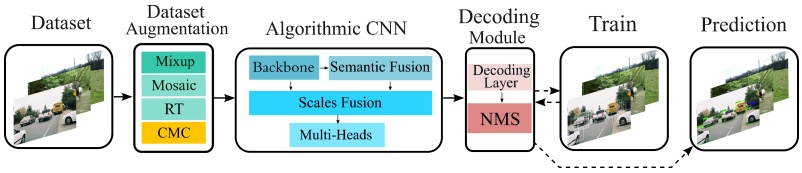

Figure 4. The pipeline of the proposed algorithm. The CMC represents Consistent Mixed Cropping. NMS represents non-maximum suppression. The RT represents Random Transformation.

3.1. Overview of Algorithmic Convolutional Neural Network

Based on the one-stage framework, the algorithmic convolutional neural network is shown in Figure 5. To enhance the capacity to detect small objects, we add a low-level feature injection in the scale fusion part to improve the expression of shallow features in the scale features. Additionally, a detection head made to look for small objects is also included. Context-Enhanced Semantic Fusion (CESF) module and Attention-Enhanced Scale Fusion (AESF) module are included in the neck portion. Blind coverage of small object features may come from pooling deep semantic features. We also believe that keeping a wider receiving field and more organized information about context features in the feature map is helpful for lowering the noise floor around weak objects. As a result, we use hierarchical aggregation instead of spatial pyramid pooling in the semantic context fusion section. In the scale fusion part, we apply the attention enhancement mechanism at the spatial and channel levels to the feature map of each scale. So that the network has the ability to

adaptively adjust the weight for the small object response area and improve the recognition accuracy of small objects.

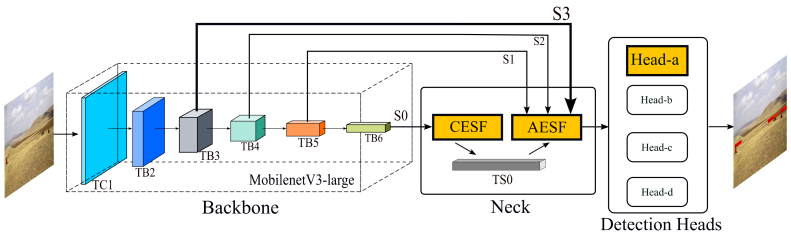

Figure 5. The archiecture of proposed Algorithm CNN (in Figure 4). The primary contribution in this paper is the section of the figure that is yellow and highlighted.

3.2. Efficient Context Enhanced Semantic Fusion

The SPP [20] module in the YOLOv4 algorithm used a pooling layer to distill feature output, which might lead to the loss of fine-grained information, making semantic features unable to fully express small objects. In order to solve this, we design the context semantic fusion module Context Fusion Module (CFM) to replace the SPP network in the original network, using the concept from reference [27]. In order to accomplish context semantic fusion, CFM employs dilated convolutions with various expansion rates. This method effectively mines spatial structure features by capturing more spatial location information about the input feature layer while expanding its receptive field. The improved network can better identify background pixels and improves the recognition ability of small objects.

The architecture of the CFM module is shown in Figure 6. For the input $TB6$ feature, the channel is adjusted by convolution blocks, and then the receptive field is expanded by three expansion convolutions with ratios of 1, 3, and 5. For the chessboard effect caused by dilated convolution, we use Hierarchical Feature Fusion (HFF) [28] structure to suppress it by hierarchical fusion. The expansion volume is then integrated into features $DF1$, $DF3$, and $DF5$. The 3×3 convolution blocks are used to smooth the above features. The feature maps of three levels are stacked by tensor splicing for fusion, and two convolution blocks are used for feature concentration and channel adjustment to finally obtain the semantic fusion feature $TS0$.

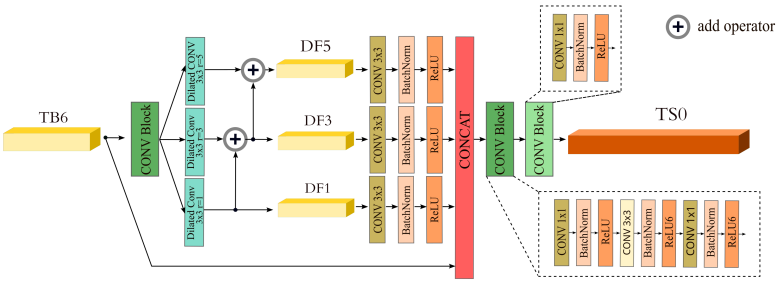

Figure 6. The archiecture of CFM.

3.3. Attention Enhanced Spatial Scales Fusion

The convolutional neural network contains less semantic information than the shallow feature map, but the object location was still very accurate. The minute object features progressively became unnoticeable despite the deep feature map's semantic information being relatively dense. Shallow features were commonly used for small object recognition, which makes it necessary to incorporate the semantic information of deep features.

In order to boost semantic feature fusion's effectiveness, we apply the spatial and channel attention method [22] to the scale feature layer, on the basis of a multi-path aggregation network, to enhance the recognition capability of small objects. We specifically deepen the hierarchical structure of scale features by injecting a shallower feature map

into the spatial scale fusion module. The effectiveness of the merging of shallow feature maps and high-level semantic maps is simultaneously improved by the employment of the attention mechanism. We proposed the Spatial Scale Attention Module (S2AM) to improve the emphasis on shallow features and the capacity to recognize small objects by adaptively adjusting the weight of space and channels in the process of scale fusion. Figure 7 displays the entire spatial scale fusion module organization.

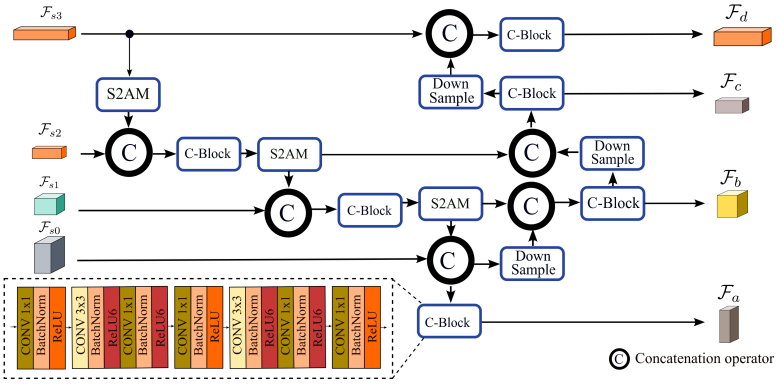

Figure 7. The layout of the AESF module.

The archiecture of Spatial Scale Attention Module is shown in Figure 8. The adaptive guiding features are aggregated in key directions by the scale attention module in the spatial scale fusion module, which cascades through the attention modules at the two levels of space and channel. The global maximum pooling and the global average pooling are executed, respectively, for the input feature $\mathcal{F}_i$. Two $\times$ 1 convolutions are used to tweak the resulting feature layer before sigmoid activation is used to create the channel attention tensor. By aggregating with the feature layer $\mathcal{F}_i$, the output feature layer $\mathcal{F}_{mid}$ is created. The two output features are then tensor layered after $\mathcal{F}_{mid}$ is pooled globally to large and globally to average along the spatial direction of each layer of channels. The spatial attention tensor is obtained after the stacking features have been modified and turned on by the channel, and the output feature $\mathcal{F}_o$ is obtained following the aggregation with $\mathcal{F}_{mid}$. In order to increase the effectiveness of fusion with underlying semantic features, SAM modules have enhanced the features of each scale with spatial attention and channel attention. After attention improvement, the feature picture is sampled using bilinear interpolation and then combined with the top feature.

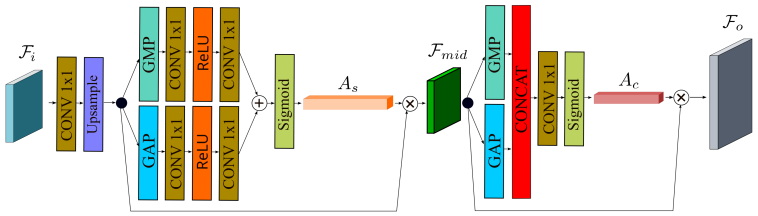

Figure 8. The archiecture of S2AM.

3.4. Data Augmentation

We introduce some effective data augmentation methods, such as Mixup [15], Mosaic [11], and random transformation, to increase the input datasets in order to increase the learning effectiveness of the network. In order to improve training performance, the network has to learn more features connected to samples. Figure 9 displays the sample case after data augmentation. The amplified samples with category labels created after stacking the first four samples using the Mixup method are shown in the figure's first column. The mixed image created by utilizing the Mosaic augmentation approach is displayed in the second column. The sample produced after the affine transformation, illumination

transformation, and scaling operation is represented by the third column. A popular data augmentation technique in deep learning networks is Mixup. By proportionally combining two pairs of sample data from the same category, it creates new samples. Another quick and efficient form of data augmentation is Mosaic. To splice and synthesize a new sample, it employs four genuine frames from different samples. Following normalization, it is the same as learning four samples simultaneously while enhancing the variety of the sample backdrop. To increase the diversity of the input samples, lighting, and geometric alteration are also applied.

We also suggest the Consistent Mixed Cropping (CMC) method (Figure 9c) to increase the effectiveness of learning minor goals. Consistent Mixed Cropping is inspired by Mixup. But instead of haphazardly chopping and splicing samples, we attach the object to its matching background in accordance with the object's meaning. To enhance the shape of small objects, we scale and geometrically transform the object once more. We believe that using the incorrect hybrid splicing technique could cause misunderstanding in online learning. For instance, finding small objects at sea has a much higher likelihood than detecting a ship or a bird. These prior details will lower the likelihood of network inference errors.

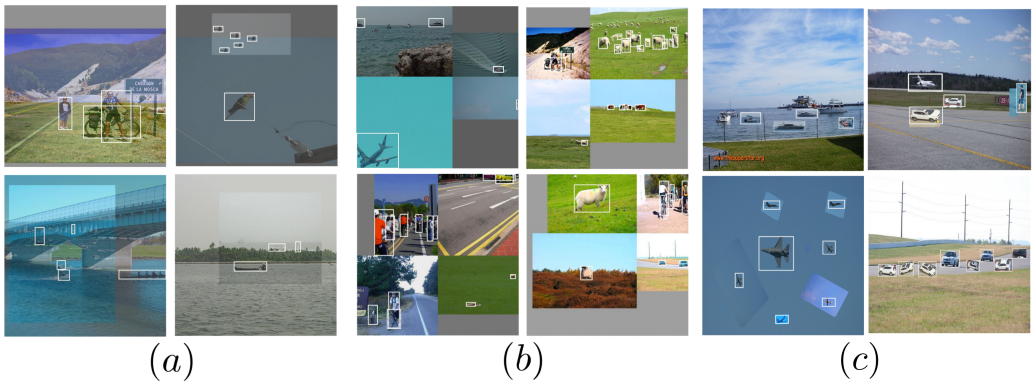

Figure 9. Visualization of multiple data augmentation methods. Mixup, Mosaic, and Consistent Mixed Cropping are listed in each column from left to right represented by (**a**–**c**).

4. Experiments of Training

In this section, the model of the proposed algorithm convolutional neural network is built and trained on a Linux server using PyTorch [29]. The training experiments are divided into two sections. One component entails evaluating the performance enhancement of the final model of the proposed algorithm for convolutional neural networks. The second component is training for the ablation experiment, which will be used to evaluate the contribution of the proposed enhanced modules CFM and S2AM.

4.1. Training Environment

Cuda11.8, Torch1.13.0, Cudnn8.6, and Python 3.10 are running on Intel i7-10700K at Nvidia Geforce Rtx3090 hardware in the training environment for the proposed algorithm neural networks. We use 10% of the datasets as the test sets during training and the remaining serving as both the training sets and the verification sets. We employ the pre-training weight of the backbone network to fine-tune the training process in an effort to increase training efficiency. The model parameters are iterated using the Adaptive moment estimation (Adam) [30] algorithm, and the learning rate is dynamically changed during training using the cosine annealing approach. The model of algorithm convolutional neural network has been trained for 200 epochs in total.

4.2. Training Datasets

VOC07+12 datasets [16] and self-built small object supplementary sample images are used for training. The two datasets contain 21,925 images from 20 object categories. Among

them, 21,505 images are from VOC2007 and VOC2012 datasets, and 420 images are from self-built datasets. According to the label size of the object, we calculate the distribution of small objects in each category in the VOC datasets part of the training data, as shown in Figure 10a–d. The top six categories of small object labels in VOC datasets are person, car, sheep, bottle, pottedplant and bird.

According to Figure 10a, we find that the percentage of small tags in the datasets is generally low. Large objects are easy to be recognized as positive samples (samples with a confidence level greater than 0.5) in the recognition process. The low number of small labels would cause the ratio of positive samples to negative samples to be unbalanced, which would cause the positive samples with a large intersection ratio to submerge small objects in the process of non-maximum suppression [31]. In order to improve the imbalance between positive and negative labels, we then collected 420 additional images containing small objects. There are 8136 small object labels in total, including person (6316), car (1624), boat (146), and bird (50). Among the supplementary small object labels, 4258 labels are used in the sample image of the additional datasets by Consistent Mixed Cropping. In the datasets finally used for training (Figure 10e), the number of small object labels in the top six categories are car, person, sheep, boat, bird, bottle.

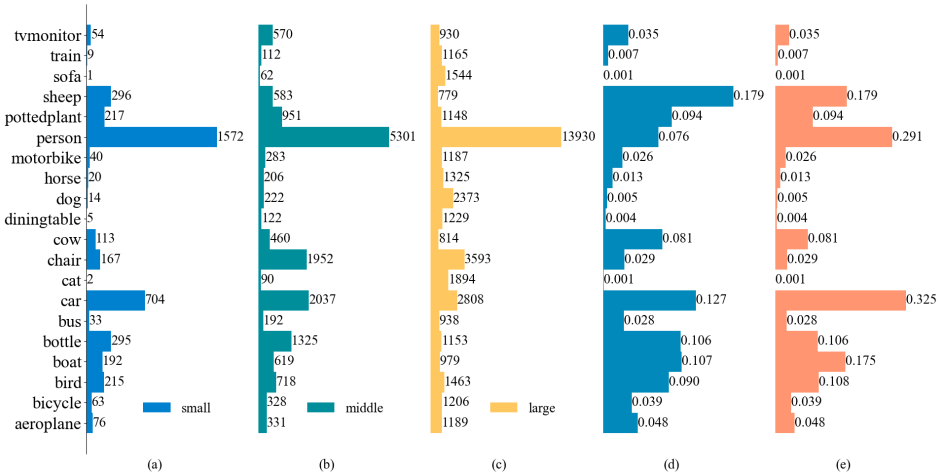

Figure 10. The distribution of the sample labels for VOC07+12. The number of small, medium, and large object labels in VOC datasets for each category are displayed in (**a**–**c**). The percentage of small object labels in each category is displayed in (**d**). The percentage of small object labels in training datasets is displayed in (**e**).

4.3. Performance Evaluation

On the test sets, we evaluate the accuracy of the proposed algorithm and other classical target detection algorithms. The comparison accuracy results are shown in Table 1. According to the results, the mAP of the proposed algorithm is 4.6% larger than the mAP of the initial YOLOv4 framework. Compared to other single-level algorithms, the accuracy of detection is enhanced. The detection accuracy is higher than SSD and FSSD algorithms by 13.0% and 8.5%, respectively. Compared to YOLOv3 [10], the proposed algorithm is 13.7% higher. In addition, the accuracy is 14.0% (VGG16 [32]) and 6.7% (ResNet101 [33]) higher than the two-stage detection method FasterR-CNN, and 19.0% higher than FastR-CNN.

Table 1. The comparison results of mAP in the test sets.

Algorithm	Backbone	mAP (%)
FasterR-CNN [8]	VGG16 [32]	76.5
FasterR-CNN [8]	ResNet101 [33]	83.8
FastR-CNN [7]	ResNet101 [33]	71.5
FSSD [13]	VGG16 [32]	82.0
YOLOv3 [10]	Darknet53 [10]	76.8
YOLOv4 [11]	CSPDarknet53 [34]	85.9
SSD [12]	VGG16 [32]	77.5
proposed	MobilenetV3-Large [14]	90.5

We compare the performance of the proposed algorithm model and the YOLOv4 model in the first six training datasets categories with a significant proportion of small objects (Table 2). The P_{50} and R_{50} in the table represent the predicted category accuracy and recall at the threshold of 0.5 confidence. In the aforementioned categories, the detection accuracy of the proposed algorithm has been enhanced. Each category's average accuracy increased by 4.1%, 4.8%, 3.9%, 5.1%, 1.2%, and 3.7%, respectively. In Figure 11, we visualize the P-R curve of the proposed algorithm on the test sets. The P-R curve depicts the precision and recall value at various confidence levels. The closer the P-R curve is to the upper right corner, the better the performance of the algorithm. The area below the P-R curve represents the average precision of the category (AP in Table 2). It can be seen from the figure that the proposed algorithm has better performance in categories with a higher proportion of small objects compared with YOLOv4.

Table 2. The top 6 categories in the test sets with the average accuracy of small objects.

Category	Percentage	Proposed Algorithm			YOLOv4		
		AP (%)	P_{50} (%)	R_{50} (%)	AP (%)	P_{50} (%)	R_{50} (%)
car	32.5	92.69	95.55	80.40	88.57	94.51	75.77
person	29.1	92.13	93.83	77.67	87.34	93.82	73.25
sheep	17.9	90.35	87.93	82.93	86.44	86.67	70.73
boat	17.4	83.10	89.69	63.54	78.01	91.95	58.39
bird	10.8	88.61	95.29	76.78	87.36	91.43	75.83
bottle	10.6	81.49	90.10	66.54	77.77	93.33	59.92

The AP represents the average value of category; P_{50} represents the value of accuracy with the confidence level at 0.5; R_{50} represents the value of recall with the confidence level at 0.5; Percentage represents the proportion of small labels in corresponding categories.

We randomly pick nine images from the test sets that have the smallest object tags for visualization in order to further validate the performance of the proposed algorithm (Figure 12). The chosen sample images are typical image samples with a large number of image samples or a high ratio of small objects to overall object labels. In order to visually view the detection effect, the size of the object anchor in each sample image is counted and marked on the right side of the image. We find that although the proposed algorithm makes difficult judgments on some small objects (the confidence level was close to 0.5), the overall judgment is accurate, and the proposed algorithm improves the detection performance of small objects.

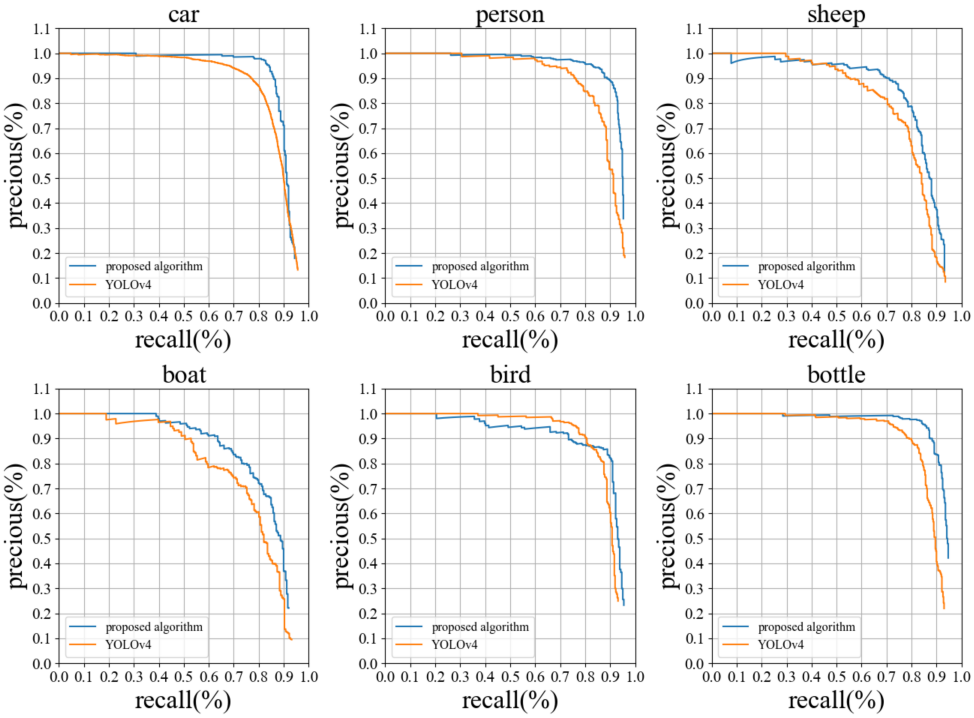

Figure 11. The visualization of P-R curves. The blue curve represents the proposed algorithm and the orange curve represents YOLOv4.

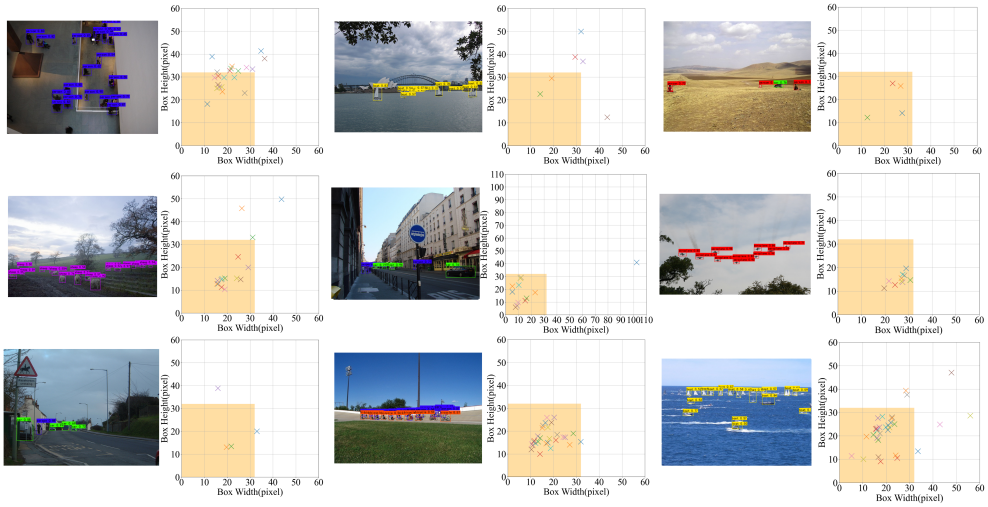

Figure 12. Visualization of small object detections. Each group's right sub-image shows the left image's object box's size dispersion. The threshold range of the small object box is represented by the yellow area.

4.4. Ablation Experiment

To validate the contribution of the proposed algorithm module, we trained six distinct models to compare their performance (Table 3). The results indicate that the CFM component can enhance the model's precision by 2.1% to 3.0%. The S2AM module can enhance model precision by 0.6% to 4.3%. The proposed CMC method can increase the model's accuracy by 1.7% to 4.3%.

Table 3. Performance evaluation of ablation experiments.

Method	mAP (%)
MobileNetV3-Large + CFM	88.0 (2.1↑)
MobileNetV3-Large + S2AM	86.5 (0.6↑)
MobileNetV3-Large + CMC	87.6 (1.7↑)
MobileNetV3-Large + CFM+S2AM	88.9 (3.0↑)
MobileNetV3-Large + CFM+CMC	88.5 (2.6↑)
MobileNetV3-Large + S2AM+CMC	90.2 (4.3↑)
proposed algorithm	90.5

As shown from Figure 13, we can find the characteristics of the CFM module when four different sets of input images are used to verify the effectiveness of CFM module fusion semantics. Each set of images in the figure has three columns. The original input image is in the first column, the feature map produced by the SPP module in the original network in the second column, and the feature TS0 of the sampling CFM module for context semantic fusion in the third column. The findings mentioned above show that the CFM module further improves the object area's response. It simultaneously increases the distance between the object region and the background and reduces background noise interference. The proposed CFM can increase the efficiency of small object detection when compared to SPP.

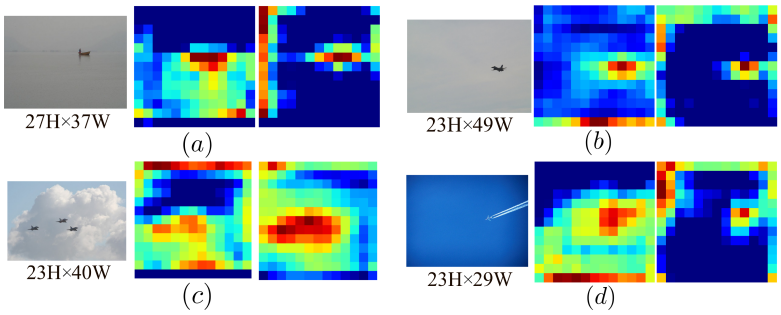

Figure 13. The comparison of SPP output and CFM output using heatmaps. The left image in each group represents the original sample, the middle image is the original frame's SPP output, and the right image is the CFM result. The object size in the photograph is indicated by the mark beneath it.

Figure 14 is the visualization of the network output following the addition of CFM and S2AM. Typical small object samples are represented by the first column in the picture, the enhancement algorithm results in the second column, and the original framework results in the third column. The object size in both sample images is 24 $\times$ 35 and 23 $\times$ 40. As compared to the second line, we find that the proposed module's identification of dense objects and detection of object stacking have both improved with the addition of CFM and SAM modules. The influence range of the object area in the algorithm (middle figure) in this work has been significantly increased, as can be observed from the third and first lines. The effectiveness of feature extraction has been increased, and small object identification has performed better because of the enhancement module.

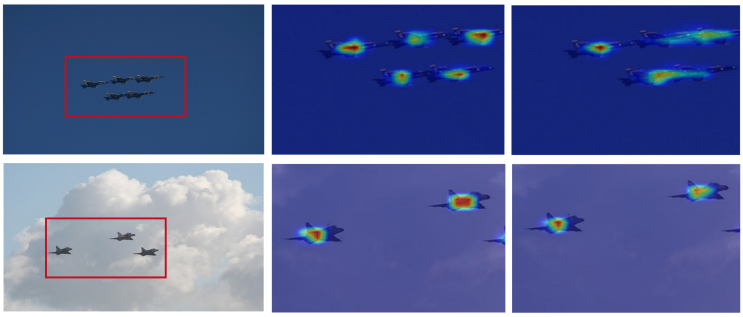

Figure 14. The comparison of the heat maps of CFM and S2AM with and without. The left image for each line represents the original sample, the middle image displays the results of a heat map using a few modules, and the right image does not.

5. Deployment Test and Discussion

In order to increase the precision and effectiveness of small object recognition, we propose an attention-enhanced lightweight small object detection algorithm based on the single-stage framework. In the training experiment section, the accuracy of the trained model's recognition of tiny objects is primarily evaluated. In light of the model's accuracy test results, the ablation experiment, and the visual small object prediction results, it is evident that the proposed algorithm can enhance the accuracy of small object recognition.

In this section, we deploy the model on the x86-64 platform to evaluate the inference efficiency of the proposed algorithm. As an extension test, we also deploy the model on embedded ARM platform to evaluate the inference efficiency of the proposed algorithm.

5.1. Deployment Environment

The x86-64 platform hardware environment used in the deployment test is the laptop with Intel I7-8750H Central Processing Unit (CPU) and Nvidia Rtx1050Ti Graphics Processing Unit (GPU). The embedded ARM platform(as shown in Figure 15) employs Raspberry4B which has a 4-cores Cortex-A72 processor and the second generation of Neural Network Stick (NCS2). The NCS2 is an artificial intelligence device that features a high-performance, low-power AI processor. It is specifically designed for deep learning inference and can be used to run neural networks on edge devices such as drones, smartphones, and other compute nodes. The device is capable of running advanced models quickly and efficiently, making it suitable for applications such as image classification, object detection, face recognition, and more. Additionally, it provides an API for easy integration with existing software solutions. We utilize OpenVINO [35] framework to create a deployment environment for embedded platform that serves as a model. Simultaneously, the proposed algorithm model is quantized and compressed in accordance with NCS2 usage requirements. To facilitate comparative testing, we select YOLOv4 and its lightweight model YOLOv4-tiny as references. All of the reference models are from the Intel Open Model Zoo depository [35].

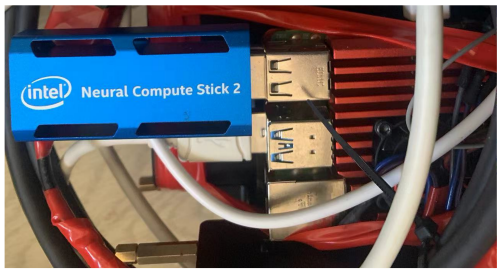

Figure 15. The equipments overview of embedded ARM platform. The Raspberry4B with Cortex-A72 is in the red section, and the blue part is NCS2.

5.2. Inference Efficiency

In the deployment tests, we gather a total of 50 sample images containing moderately sized objects from the test sets in order to determine the time required for network inference. According to the results in x86-64 platform in Table 4, we find that the average inference time of the proposed algorithm network is 51.8% less than the original framework YOLOv4 and 12.3% more than YOLOv4-tiny. Using the Torchstat utility of Pytorch, we then analyzed the computational complexity of algorithms and the quantity of the model parameters. We find that the number of Multiply–Accumulate Operations (MAdds) in the proposed algorithm's model is 87.9% less than that of YOLOv4 and similar to YOLOv4-tiny (5.8% higher than YOLOv4-tiny). The proposed algorithm model has 76% fewer parameters than YOLOv4 and 1.6 times as many as YOLOv4-tiny. The proposed algorithm model has an average single frame inference time of 29.2 ms, which is 51.8% less than YOLOv4 and 12.3% more than YOLOv4-tiny. The comparison between computational complexity and average frame rate demonstrates that the proposed algorithm has superior inference efficiency.

Table 4. The comparison of inference performance on x86-64 platform.

Algorithm	Params (M)	GMAdds	GFLOPs	Runtime (ms)	FPS
YOLOv4	64.5	59.8	29.9	60.6	16.5
YOLOv4-tiny	5.9	6.8	3.4	26.0	38.4
proposed	15.4	7.2	3.6	29.2	34.2

The Params in the table represents the parameter quantity of the network in millions ($1e^6$); The GMAdds represent giga($1e^9$) multiply-adds operations; The GFLOPs represents billion($1e^9$) floating-point operations; The Runtime represent the inference time which counted includes forward time and post-processing time; The FPS represents the number of image frames inferenced by the network per second. The inference time count includes forward time and post-processing time.

In order to evaluate the inference efficacy of the model on the embedded ARM platform, we quantified and compressed the model of the proposed algorithm into FP16 format according to the hardware requirements of NCS2. In addition, the inference efficacy of the model on the GPU and CPU hardware has also been tested using the OpenVINO framework as a point of comparison. The test results are shown in Table 5.

Comparing the FPS data in Tables 4 and 5, the quantized algorithm model inference using VPU on the embedded ARM platform is 80.9% lower than that of using CPU on the x86-64 platform. By contrasting the FPS results of the standard YOLOv4 and YOLOv4-tiny models, we find a similar result. The FPS of the YOLOv4 and YOLOv4-tiny models decreased by 96.9% and 85.4%, respectively, under the same comparison conditions. The inference performance of the proposed algorithm model using the OpenVINO framework is inferior to that of the Pytorch framework, and the model acceleration effect of NCS2 is insignificant. Subsequently, we meticulously analyzed the time consumption of each component of Latency. Based on the qualitative analysis of the Latency results in Table 5, we discover that the three models in the comparative test consumed the most time during forward propagation of the network under various hardware environments. In three hardware environments, the inference time of the proposed algorithm is comparable to the YOLOv4-tiny model provided by the Open Model Zoo of Intel. The inference time of the proposed algorithm model and lightweight model utilizing VPU for inference on the ARM platform is 91.7% and 91.2% less than that of YOLOv4, respectively.

Based on previous analysis and test data in Table 5, we discovered that, for the same model in the OpenVINO framework, the inference performance of NCS2 is typically slower than that of CPU and GPU. The inference acceleration impact of VPU on neural network models is not immediately apparent. We investigate the possibility that FP16-formatted model data of NCS2 is irreconcilable with the 64-bit instruction set in the embedded ARM environment. Next, we will attempt to replace the 32-bit instruction set and investigate the effect of multiple NCS2s accelerating simultaneously. Additionally, the lightweight model

provides more performance benefits on embedded ARM platform. The visualization of the embedded ARM deployment experiments are as shown in Figure 16.

Table 5. The results of inference test based on OpenVINO.

Algorithm	Inference Engine	Latency (ms)					FPS
		Decoding	Preprocessing	Inference	Postprocessing	Rending	
YOLOv4	VPU	8.8	5.2	1778.9	104.5	0.6	0.5
	CPU	4.8	1.3	554.5	12.5	0.1	1.8
	GPU	2.7	1.6	257.6	14.6	0.2	3.6
YOLOv4-tiny	VPU	5.6	5.6	155.6	17.1	<0.1	5.6
	CPU	2.8	1.1	44.9	1.9	<0.1	20.7
	GPU	2.7	1.3	17.4	1.5	<0.1	20.3
proposed *	VPU	/	/	146.1	/	/	6.5
	CPU	/	/	21.3	/	/	28.5
	GPU	/	/	35.8	/	/	18.5

VPU represents using NCS2; CPU represents using I7-8750H; GPU represents using Nvidia1050Ti. * quantization model of the proposed algorithm.

Figure 16. Visualization of inference results in ARM based on OpenVINO.

6. Conclusions

In this paper, a novel attention-enhanced lightweight small object detection method based on a one-stage framework is proposed. By introducing an attention mechanism, the proposed algorithm enhances the efficacy of scale feature extraction and fusion, expanding the receptive field of semantic features by expanding convolution, and introducing context to suppress the interference of background on small objects. The proposed algorithm also increases the proportion of small labels in the training datasets and enhances the detection ability of small objects. The results of the deployment on the x86-64 platform indicate that the mean average precision of the proposed algorithm is 4.6% greater than that of the YOLOv4 method prior to its enhancement, and that it performs better in the recognition of small targets. The deployment test results on the embedded ARM platform indicate that the acceleration effect of the quantized model employing a single NCS2 is inferior to those of the GPU and CPU, and that the proposed algorithm is nearly as quick as YOLOv4-tiny at inference.

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

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: The Institute of Automation Chinese Academy of Sciences was acknowledged by the authors for providing extra datasets for this article.

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

Abbreviations

The following abbreviations are used in this manuscript:

ARM	Advanced RISC Machine
CBAM	Convolutional Block Attention Module
CFM	Context Fusion Module
CMC	Consistent Mixed Cropping
CESF	Context Enhanced Semantic Fusion
NCS2	the second generation of Neural Network Stick
NMS	Non-Maximum Suppression
S2AM	Spatial Scale Attention Module
VPU	Video Processing Unit
CPU	Central Processing Unit
GPU	Graphics Processing Unit

References

1. Jawaharlalnehru, A.; Sambandham, T.; Sekar, V.; Ravikumar, D.; Loganathan, V.; Kannadasan, R.; Khan, A.A.; Wechtaisong, C.; Haq, M.A.; Alhussen, A.; et al. Target Object Detection from Unmanned Aerial Vehicle (UAV) Images Based on Improved YOLO Algorithm. *Electronics* **2022**, *11*, 2343. [CrossRef]
2. Haq, M.A. Planetscope Nanosatellites Image Classification Using Machine Learning. *Comput. Syst. Sci. Eng.* **2022**, *42*, 1031–1046. [CrossRef]
3. Haq, M.A. CNN Based Automated Weed Detection System Using UAV Imagery. *Comput. Syst. Sci. Eng.* **2021**, *42*, 837–849.
4. Liu, L.; Ouyang, W.; Wang, X.; Fieguth, P.; Chen, J.; Liu, X.; Pietikäinen, M. Deep Learning for Generic Object Detection: A Survey. *Int. J. Comput. Vis.* **2020**, *10*, 261–318. [CrossRef]
5. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 740–755. [CrossRef]
6. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587. [CrossRef]
7. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448. [CrossRef]
8. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788. [CrossRef]
9. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271. [CrossRef]
10. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018. . [CrossRef]
11. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
12. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.E.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the 2016 European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37. [CrossRef]
13. Li, Z.; Zhou, F. FSSD: Feature Fusion Single Shot Multibox Detector. *arXiv* **2017**, arXiv:1712.00960.
14. 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 2019 IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324. [CrossRef]
15. Zhang, H.; Cisse, M.; Dauphin, Y.N.; Lopez-Paz, D. Mixup: Beyond Empirical Risk Minimization. *arXiv* **2017**, arXiv:1710.09412.
16. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The PASCAL Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2009**, *88*, 303–338. [CrossRef]
17. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988. [CrossRef]

18. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125. [CrossRef]
19. Bell, S.; Zitnick, C.L.; Bala, K.; Girshick, R. Inside–Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2874–2883.
20. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *1*, 1904–1916. . [CrossRef]
21. Chen, L.C.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking Atrous Convolution for Semantic Image Segmentation. *arXiv* **2017**, arXiv:1706.05587. https://doi.org/10.48550/arXiv.1706.05587.
22. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 3–19. [CrossRef]
23. Wang, X.; Zhang, X.; Gong, Y.; Liu, W.; Shi, H. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 11534–11542.
24. 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. [CrossRef]
25. Zhang, H.; Wu, C.; Zhang, Z.; Zhu, Y.; Zhang, Z.; Lin, H.; Sun, Y.; He, T.; Muller, J.M.; Manmatha, R. ResNeSt: Split-Attention Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2022; pp. 2736–2746. [CrossRef]
26. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520. . [CrossRef]
27. Yu, F.; Koltun, V. Multi-Scale Context Aggregation by Dilated Convolutions. In Proceedings of the International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016. [CrossRef]
28. Mehta, S.; Rastegari, M.; Caspi, A.; Shapiro, L.; Hajishirzi, H. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
29. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019.
30. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
31. Neubeck, A.; Van, G. Efficient non-maximum suppression. In Proceedings of the 18th International Conference on Pattern Recognition, Hong Kong, China, 20–24 August 2006; Volume 3, pp. 850–855. [CrossRef]
32. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the 3rd International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015. [CrossRef]
33. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016. [CrossRef]
34. Wang, C. Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020. [CrossRef]
35. Official Documentation for OpenVINO by Intel. Available online: https://docs.openvino.ai/latest/home.html (accessed on 1 January 2022).