Studying the efficiency and performance of the vehicle detection method based on feature fusion and attention enhancement

Ming Xue

Wuhan Fiberhome Technical Services Co., Ltd., 88 Youkeyuan Rd., Hongshan District, Wuhan, 430068, China

Abstract

Considering the cost deployment problem of traffic recognition algorithms, this paper considers YOLOv4 as the base architecture. The lightweight DenseNet is used as the backbone feature extraction network, and effective channel attention (ECA) and Adaptive Spatial Feature Fusion (ASFF) are used to enhance the PANet structure with attention-guided fusion. The weight ratio of the loss function is optimized and the mosaic method is used for training enhancement. The results show that the proposed algorithm improves both the detection accuracy and detection speed as well as reduces the number of parameters by 64. The research results provide some reference value for the traffic construction of smart cities.

Keywords

target detection, vehicle detection, YOLOv4, feature fusion, attention mechanism, lightweighting

1. Introduction

As the pioneer of one-stage target detection, the YOLO series algorithm innovates on the detection principle of the Faster R-CNN series by abandoning the RPN approach and using regression to obtain the coordinate information of the bbox. YOLOv1, an algorithm that uses an end-to-end identification approach, is known as the one-stage target detection algorithm. This algorithm was quickly deployed in many real-world projects due to the dramatic increase in detection speed, and was even used in military devices. A large number of one-stage target detection algorithms have also emerged since then, and these algorithms have evolved through iterations in pursuit of faster and more accurate recognition results [1, 2, 3, 4].

Bochkovskiy et al. [5] proposed YOLOv4, the algorithm developed for one-stage target detection. It is based on the architecture of the classical YOLO target detection family, published in 2020 and endorsed by the authors of YOLOv3. Such algorithms concentrate both target classification and localization in the same network architecture, enabling end-to-end detection.

The YOLOv4 algorithm consists of the CSPDarknet53 backbone network, SPPNet, PANet feature fusion network and the YOLO-Head detection head module that is used in YOLOv3. Its network structure is shown in figure 1.

In this paper, we further improve the training and inference speed of the one-stage detection algorithm by modifying the backbone network of the model, based on the YOLOv4 algorithm, and improve the model structure using the attention mechanism and feature fusion module to enhance the detection performance of the algorithm.

doors-2025: 5th Edge Computing Workshop, April 4, 2025, Zhytomyr, Ukraine
mingxue202@gmail.com (M. Xue)
0009-0002-6581-9888 (M. Xue)

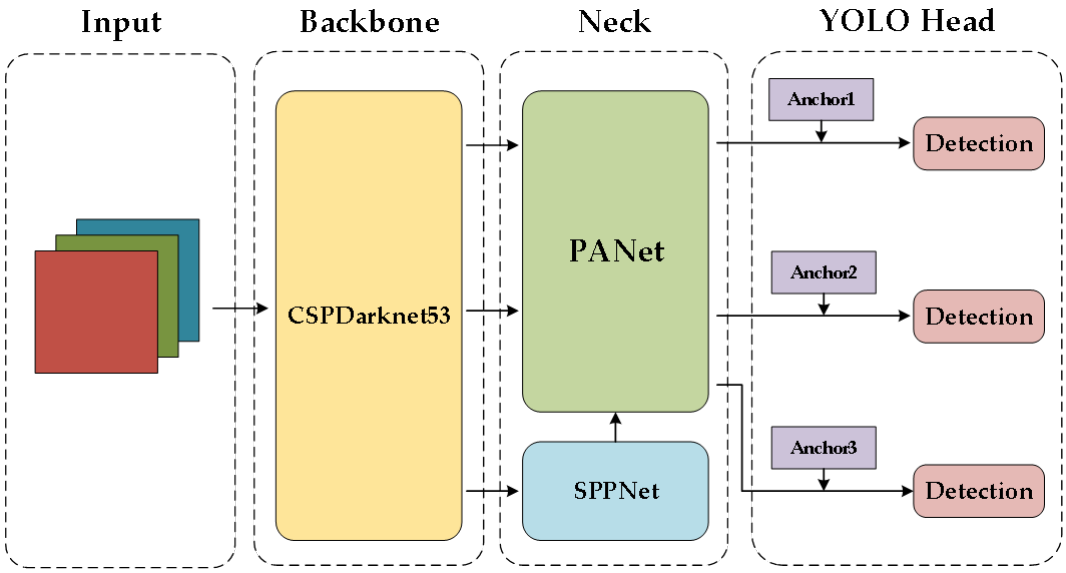

Figure 1: YOLOv4 network structure.

2. Improved YOLOv4 method

2.1. Feature Parymid Network (FPN)

Feature Pyramid Representation (FPN) is a common approach to address the challenge of scale variation in target detection. Its structural layer design allows the model to more fully utilize the feature information extracted from the backbone network.

Various FPNs are designed to maximize the utilization of the multi-scale feature maps from backbone, and its optimization leads to significant performance improvement of object detection. Therefore, the algorithms in this paper work in concert with the fusion of PANet and ASFF to enhance the reuse and extraction of feature maps and avoid the loss of effective information [6, 7, 8, 9].

2.2. Attentional mechanisms

The attention mechanism focuses on local information while suppressing distracting information. Attention mechanisms have made important breakthroughs in recent years in areas such as image and natural language processing, and have widely demonstrated their effectiveness in improving model performance.

From a mathematical point of view, the attention mechanisms provides a weight-based model to perform operations. The attention mechanism uses the network layer to calculate the weight values corresponding to the relevant feature maps, and then applies these weights to the feature maps, so that the feature maps with a large role in extracting information become somewhat more influential on the overall. With respect to the content of interest, attention mechanisms can be split into three types: channel attention mechanism, spatial attention mechanism, and mixed spatial and channel attention mechanism.

2.2.1. The spatial attention mechanism

The Spatial Transformer Network (STN) [10] proposed by Google DeepMind is a spatial-based attention by learning the shape change of the input so as to accomplish preprocessing operations suitable for a specific task. The ST module consists of localisation net, grid generator and sample. The localisation net determines the parameter θ of the input required transformation. The grid generator finds the mapping $T(\theta)$ of the output to the input features by θ and the defined transformation. The sample combines the

location mapping and transformation parameters to select the input features and combine them with bilinear interpolation for the output.

2.2.2. Channel attention mechanism

Sequeeze and Excitation Net (SENet) [11] is a channel type Attention model, which automatically enhances or suppresses channels after model learning by modeling the importance of each feature channel. It divides a bypass branch after the normal convolution operation, and this branch is compressed and fully connected to obtain a set of weight values.he importance of the different channels can be learned by applying this set of weights to each of the original feature channels.

2.2.3. Fusion of spatial and channel attention mechanisms

Convolutional Block Attention Module (CBAM) [12] is a representative network that combines spatial and channel attention mechanisms. It uses a channel-then-space approach for collocation, so that the model models the important information of channel and spatial locations separately. Besides these, there are many other attention mechanisms related to research, such as residual attention mechanism, multi-scale attention mechanism, recursive attention mechanism, etc. [13, 14, 15, 16, 17].

2.3. Related methods

Figure 2 shows the overall architecture of the proposed algorithm in this paper. The algorithm in this paper takes the one-stage target detection algorithm YOLOv4 as the reference architecture and divides the algorithm framework into four parts: data pre-processing and input, backbone network, FPN structure and prediction network. In this paper, the pre-processed images are first passed into the backbone network, which adopts a lightweight DenseNet structure consisting of different numbers of dense blocks and transition layers. Depending on the number of sub-module overlays, the backbone network extracts the feature information at different scales and passes it into the FPN network. Before this information is passed into SPPNet and PANet, the feature information will be further filtered and refined by three ECA attention modules. Then the information output from the bidirectional fusion-type network PANet is fed into the complex fusion network ASFF, which makes the feature map information at different scales form the interaction. Finally, the information extracted from the ASFF network is fed into the YOLO detection head for detection, and the prediction results of the image are obtained after the information decoding and other operations. Next, the backbone network, FPN structure and loss function of the algorithm in this paper are described in more detail, respectively.

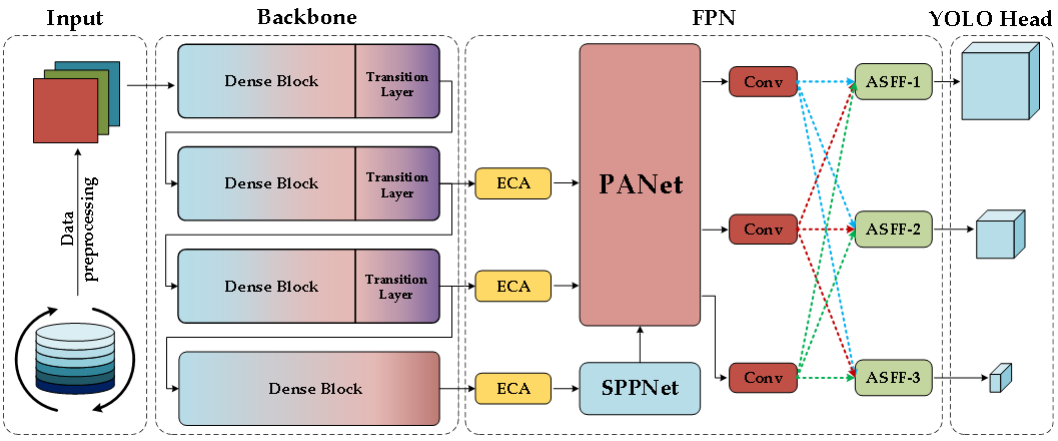

Figure 2: Overall algorithm structure.

2.3.1. Lightweighting of the backbone

The backbone network is replaced with the lightweight network DenseNet-121, and the rest of the architecture is optimized on the basis of YOLOv4.

The main components of the DenseNet network are both the dense blocks and transition layers.

The dense block is composed of several bolt necks. Each block uses the same number of output channels, and then uses a loop to connect the input and output of each block in the channel dimension. The structure of bolt neck is shown in the upper part of figure 3. Each bottle neck contains two convolutions, the first one is a 1*1 convolution, which has $4k$ output channels. Here, k is a feature map growth factor, which is the number of feature maps contributed by each bottle neck. The second 3*3 convolution has k output channels. Finally, the input of the module and the output of the 3*3 convolution are concat stacked to obtain the overall number of output channels of the module as $C' + k$.

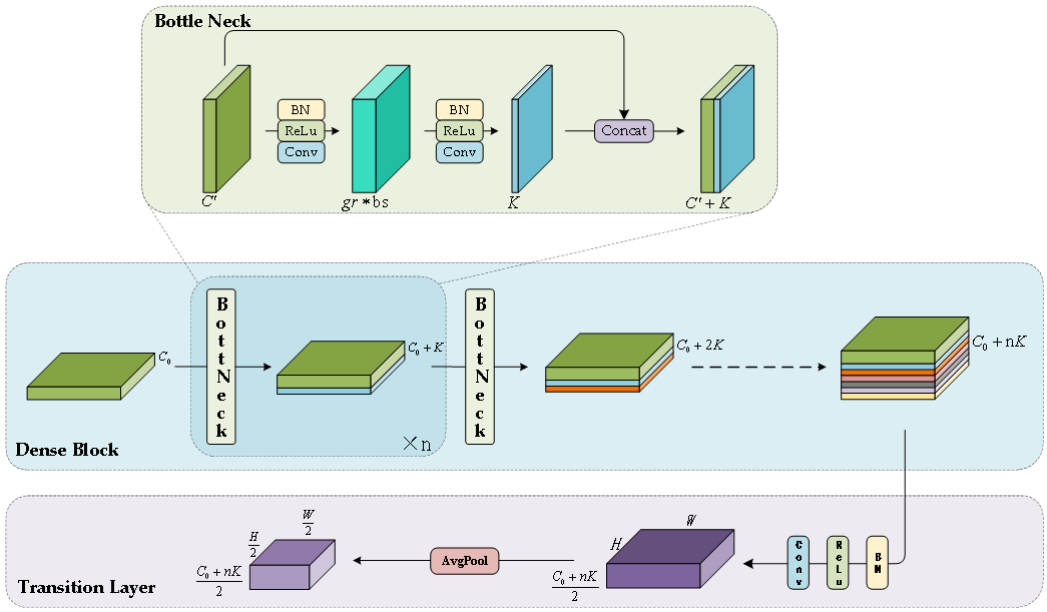

Figure 3: DenseNet main structure.

The dense block structure is shown in the middle part of figure 3, which consists of several bottle neck. If the number of input channels of the whole dense block is C_0. Since the output of Bottle Neck stacks the output and input of the final convolutional structure in its interior, the number of feature channels will be increased by k for each bottle neck that passes through it. Therefore, the number of final output feature maps of a dense block composed of n bottle neck is $C_0 + nk$.

By looking at the dense block structure, it can be seen that the input of each bottle neck is a stack of all the outputs of its preceding layers. This essentially densely connected network structure is the reason why DenseNet can achieve good results.

The transition layer is used to control the model complexity, and its structure is shown in the lower part of figure 4. Since the number of channels increases with each dense block connection, using too many would result in an overly complex model. Therefore, the transition layer first reduces the number of channels by a 1*1 convolution layer, and then to compress the height and width of the feature map, an average pooling layer with stride=2 is used for downsampling, which further reduces the model complexity.

3. Citation of attentional mechanisms

To ensure the detection accuracy of the model while performing lightweight optimization of the model, this paper will intersperse the attention mechanism module in the network structure.

To keep the balance between model complexity and performance, this paper refers to an effective channel attention module (ECA) that contains only a small number of parameters while delivering significant performance gains.

SE-Net is the basis of ECA-Net optimization and its structure is shown in figure 4(a). Global average pooling is first performed separately for each input channel, followed by two fully connected layers using different activation functions. This computational process causes the channel features to be mapped from high to low and then to high dimensions. This dimensionality reduction operation reduces the complexity of the model, but it also hinders the correspondence generated between weights and weights, which may result in the loss of critical information.

Empirical data show ECA-Net, by observing SE-Net and improving it, that it is important to avoid dimensionality reduction when learning channel attention and a proper cross-channel interaction can increase the complexity of the model only slightly while maintaining performance. Its structural design is given in figure 4(b).

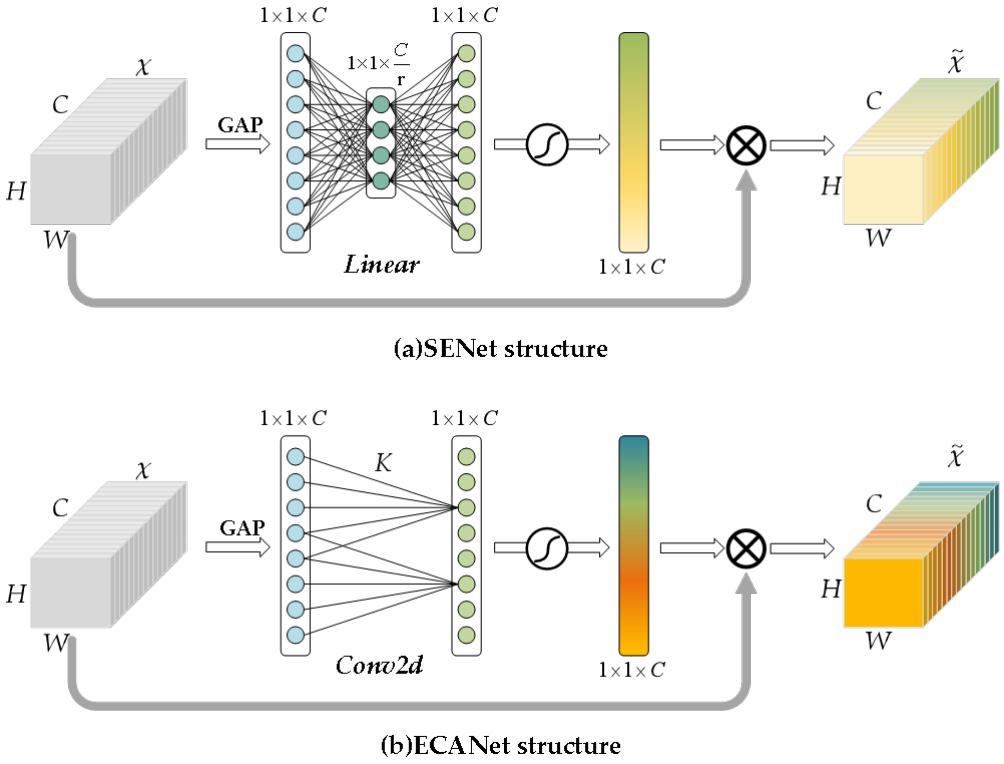

Figure 4: SENet and ECANet structures.

On the left is the feature of the original input image, which is first subjected to global average pooling (GAP) [18] to obtain a 1*1*C feature map, on which ECA obtains the local cross-channel interaction by fast one-dimensional convolution of size K. The parameter K can be developed by an adaptive function based on the dimension of the input channel C. This channel represents the local coverage of the cross-channel interaction. Then, the sigmoid activation function generates the weight share for every single channel. After that, the features with channel attention are obtained by merging the original input features with the channel weights. The network based on this module extracts discriminative features of images on the basis of channel dimensionality more easily.

In order to avoid the consumption of large arithmetic resources due to manual adjustment, the size of the parameter k can be generated adaptively by a function with the convolution kernel k:

$$k = \psi(C) = \left| \frac{\log_2(C)}{\gamma} + \frac{b}{\gamma} \right|_{\text{odd}} \tag{1}$$

where $|t|_{odd}$ denotes the odd number of t-nearest neighbors, γ is set to 2, and b is 1. From the equation, it is

clear that the communication range of the high-dimensional channel is longer, while the communication range of the low-dimensional channel is relatively contracted.

In this paper, three ECA layers are inserted at the connection between backbone and neck of the model to avoid dimensionality reduction while better bridging the two components, making the feature transfer of the model more efficient and preventing the disappearance of feature information. At the same time, the ECA layer allows the model to focus on more critical features and suppress unnecessary features, thus ignoring the interference brought by the image background, which enhances the accuracy of detection for the model even further.

4. Spatially adaptive fusion of feature layers

ASFF can further enhance the extraction capability of PANet and can fuse the information of multiple feature layers simultaneously. Its core idea is to adaptively adjust the spatial weights of each scale features in fusion by learning. Its underlying structure is shown in figure 5.

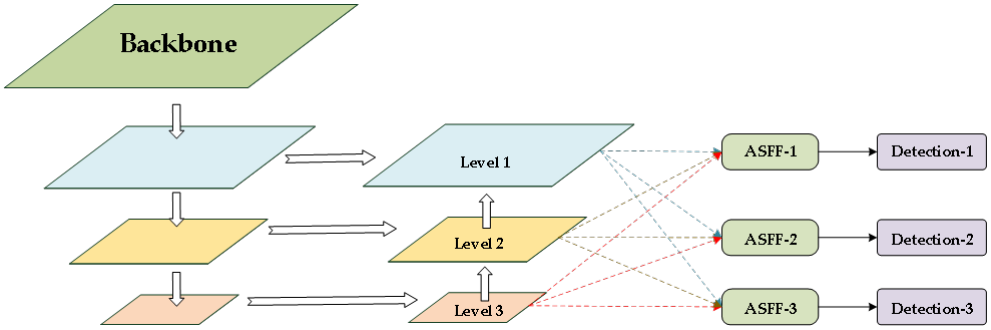

Figure 5: ASFF schematic.

The ASFF-3 is an example of a convolution with a convolution kernel of 3*3, a step size of 2, and a padding of 1. The $X2$ is scaled down to the same value as $X3$ with equal number of channels, and is denoted as level_1_resized. The number of channels and dimensionality of level_1_resized, level_2_resized, and $X3$ are the same. Finally, level_1_resized, level_2_resized, and $X3$ are multiplied by α, β and γ, respectively, and the values are summed, and the number of channels is adjusted by a final convolutional layer to obtain a new feature layer with multi-layer perceptual field fusion. The expression is as follows.

$$y_{ij}^{l} = \alpha_{ij}^{l} \cdot X_{ij}^{1 \to l} + \beta_{ij}^{l} \cdot X_{ij}^{2 \to l} + \gamma_{ij}^{l} \cdot X_{ij}^{3 \to l} \tag{2}$$

where y_{ij}^{l} represents the new feature map of a layer obtained by ASFF, α_{ij}^{l}, β_{ij}^{l}, and γ_{ij}^{l} represent the weight parameters learned through the three feature layers, and $\alpha_{ij}^{l} + \gamma_{ij}^{l} + \beta_{ij}^{l} = 1$ is guaranteed by the Softmax function.

5. Designing the loss function

The loss function based on the YOLOv4 algorithm contains three components: confidence error L_{conf}, classification error L_{cls}, and regression frame prediction error L_{loc}. Among them, the confidence error and classification error continue the design idea of YOLOv3 [19]. However, CIoU loss was used in the design of the regression frame prediction error. The CIoU is based on IoU, GIoU, and DIoU, and the CIoU takes into account three geometric factors, which are overlap area, centroid distance, and aspect ratio. They are calculated by the following equations [20, 21, 22].

$$
\begin{aligned}
L_{\text{conf}} = &- \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} \left[\overline{C_i^j} \log(C_i^j) + \left(1 - \overline{C_i^j}\right) \log(1 - C_i^j) \right] \\
&- \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{noobj}} \left[\overline{C_i^j} \log(C_i^j) + \left(1 - \overline{C_i^j}\right) \log(1 - C_i^j) \right]
\end{aligned} \tag{3}
$$

$$
L_{\text{cls}} = - \sum_{i=0}^{S^2} I_{ij}^{\text{obj}} \sum_{c \in \text{classes}} \left\{ \overline{P_i^j}(c) \log \left[P_i^j(c) \right] + \left[1 - \overline{P_i^j}(c) \right] \log \left[1 - P_i^j(c) \right] \right\} \tag{4}
$$

$$
\text{CIoU}(X, Y) = \text{IoU}(X, Y) - \frac{\rho^2(X_{\text{ctr}}, Y_{\text{ctr}})}{m^2} - \alpha v \tag{5}
$$

where: S^2 is the number of grids, B is the number of prediction frames in each grid, I_{ij}^{obj} and I_{ij}^{noobj} are the indicator values of the prediction frames containing and not containing the target, $\overline{C_i^j}$ is the confidence true value, C_i^j is the prediction confidence, λ_{noobj} is the penalty weight factor, $\overline{P_i^j}(c)$ is the actual probability that the target in the cell belongs to category c, $P_i^j(c)$ is the probability that the prediction is of category c, $\text{IoU}(X, Y)$ is the intersection ratio of the predicted frame X to the real frame Y, $\rho^2(X_{\text{ctr}}, Y_{\text{ctr}})$ is the Euclidean distance between the center point of the predicted frame and the real frame, m is the diagonal distance of the minimum closed region containing both the predicted and real frames, α is the balance adjustment parameter, v is the parameter measuring the consistency of the aspect ratio.

The regression frame prediction error L_{loc} is typically defined using the CIoU loss:

$$
L_{\text{loc}} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} (1 - \text{CIoU}(b_i^j, \hat{b}_i^j)) \tag{6}
$$

where b_i^j represents the predicted bounding box and $\hat{b}_i^j$ represents the ground truth bounding box.

In order to balance the loss sensitivity of different detection scales, in this paper, the three prediction heads in the network structure are multiplied with different weights when calculating the total loss. The weights assigned to Yolo Head1, Yolo Head2 and Yolo Head3 are 0.4, 1.0 and 4.0, respectively [23].

6. Experiment and analysis

6.1. Experimental setup

6.1.1. Dataset

In this paper, a variety of datasets will be used for performance evaluation.

1. The RSOV dataset is published by Brno University of Technology and consists of three sub-datasets with different viewpoints, namely the rear view shot dataset, the eye level view shot dataset, and the unconstrained shot dataset, each of which contains 5000 images of vehicles with annotations. Thus, the dataset has a total of 15,000 images containing information of 41 different brands and categories of vehicles. In this paper, we divide the dataset according to the ratio of 8:1:1, and finally get 12,000 training sets, 1,500 validation sets and 1,500 test sets.
2. BIT-Vehicle dataset is captured by two cameras at different times and locations, and these images vary in terms of lighting conditions, vehicle color, and camera viewpoint. All vehicles in the dataset are classified into six categories: Bus, Microbus, Minivan, Sedan, SUV, and Truck. The dataset has a total of 9850 images, and the dataset is divided according to the ratio of 8:1:1, resulting in 7880 training sets, 985 validation sets and 985 test sets.

3. To verify the detection generality of the proposed algorithm, the classical multi-category dataset PASCAL VOC is used. The PASCAL VOC Challenge is a world-class competition in computer vision covering several sub-tasks such as classification of images, detection and segmentation of targets, etc. VOC2007 and VOC2012 are two classical benchmark datasets publicly provided by the competition, including a total of 20 categories including people, airplanes, and cars, and each version of the dataset is produced in a uniform manner. In the paper, we use in total 16,551 images of trainval data from VOC2007 and VOC2012 as the overall dataset, which is randomly partitioned according to the ratio of 0.81:0.09:0.1, resulting in 13,405 training sets, 1,490 validation sets and 1,656 test sets.

6.1.2. Dataset pre-processing

Since there are large differences in the number of samples and uneven distribution of some images in some datasets, mosaic data enhancement is performed on the dataset before model training. The operation process is shown in figure 6.

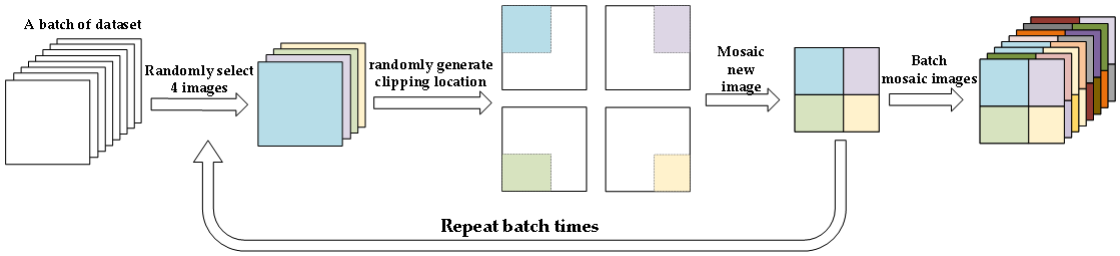

Figure 6: Principle of mosaic operation.

The mosaic data enhancement method first takes out a batch of data to be processed from the dataset, then randomly uses four images to be scaled or shifted in different proportions, placed in the direction of the four corners of the rectangle, crops the excess part of images that exceed the specified input size, and finally gets a new image as training data. The mosaic method for data preprocessing not only enhances the diversity of data and enriches the image dataset, but also improves the batchsize in disguise and enhances the efficiency of the model.

6.1.3. Evaluation index

1. **Evaluation metrics for model inference**
   For the target detection task, the precision P, recall R, and mean average precision mAP (mean Average Precision) are commonly used as evaluation metrics for model identification. The calculation of the relevant metrics is given below.

   a) Precision P and recall R.
   The precision is the ratio of correct predictions to the total number of predictions. It is one of the simplest metrics. Recall calculates the ratio of the number of predicted positive cases to the total number of positive case labels. They are calculated as follows:

$$P = \frac{TP}{TP + FP}, \tag{7}$$

$$R = \frac{TP}{TP + FN}, \tag{8}$$

   where TP denotes the number of positive samples correctly identified as positive, FP denotes the number of negative samples incorrectly identified as positive, and FN denotes the number of positive samples incorrectly identified as negative.

b) AP and mAP.
Average Precision (AP) is the area limited by the precision-recall curve. The better a classifier is, the higher the AP value. It is used to evaluate the detection accuracy of a class, and is calculated as follows:

$$AP = \int_0^1 P(R)\, dR \tag{9}$$

The mean Average Precision (mAP), which is the average value of multiple categories of AP. The mAP has a size in the interval $[0, 1]$ and the larger its value, the better, and this metric is the most important one in the target detection algorithm and is calculated as follows.

$$mAP = \frac{1}{n} \sum_{i=1}^{n} AP(i) \tag{10}$$

2. **Evaluation Index of Model Parameters**
In a deep learning model, the size of the number of parameters determines to some extent the depth of the model, the speed of inference and even the detection accuracy. A large deep learning architecture is often accompanied by high accuracy because it is closer to the neuronal composition of the human brain. However, a large number of parameters also means a sacrifice of inference time and response speed. Therefore, a small number of parameters is important for deploying deep learning models to embedded devices or platforms, and for handling large numbers of concurrent requests.
The metrics of floating point operations (FLOPs) is referred to as the number of float-point operations. It is understood as the number of computations, and the base unit is B. It can be used to measure the complexity of an algorithm/model.
Params is referred to as a total number of parameters to be trained, and the base unit is M.

$$\text{Params} = \sum_{l=1}^{L} P_l \tag{11}$$

where L is the total number of layers in the model and P_l is the number of parameters in layer l. For a typical convolutional layer, the number of parameters can be calculated as:

$$P_{\text{conv}} = (k_h \times k_w \times c_{\text{in}} + 1) \times c_{\text{out}} \tag{12}$$

where k_h and k_w are the kernel height and width, c_{in} is the number of input channels, c_{out} is the number of output channels, and the $+1$ term accounts for the bias parameter for each output channel.
For $FLOPs$ in a convolutional layer:

$$\text{FLOPs}_{\text{conv}} = h_{\text{out}} \times w_{\text{out}} \times (2 \times k_h \times k_w \times c_{\text{in}} - 1) \times c_{\text{out}} \tag{13}$$

where h_{out} and w_{out} are the height and width of the output feature map. The term $(2 \times k_h \times k_w \times c_{\text{in}} - 1)$ accounts for multiplication-addition operations for each output element.
For a fully connected layer:

$$P_{\text{fc}} = (n_{\text{in}} + 1) \times n_{\text{out}} \tag{14}$$

$$\text{FLOPs}_{\text{fc}} = (2 \times n_{\text{in}} - 1) \times n_{\text{out}} \tag{15}$$

where n_{in} is the number of input neurons and n_{out} is the number of output neurons.
The total computational complexity of a neural network model can then be expressed as:

$$\text{Total FLOPs} = \sum_{l=1}^{L} \text{FLOPs}_l \tag{16}$$

where FLOPs_l is the number of floating point operations in layer l.

Model efficiency can be characterized by the ratio:

$$\text{Efficiency} = \frac{\text{Accuracy}}{\text{FLOPs} \times \text{Params}} \tag{17}$$

This metric helps quantify the trade-off between model performance and computational resources required.

6.1.4. Training strategies

In the paper, we use migration learning to speed up the training of models. Migration learning is used to help train a new model by migrating the weight parameters of an already trained model to a new one. The basis of this approach is that most of the data or tasks are correlated, and by sharing some of the parameters of the pre-trained model to the new model to be trained, the training process of the new model can be significantly accelerated and optimized for the purpose of saving computational resources [24].

In deep neural networks, the previous convolutional layers generally learn shallow features with generality, while the later convolutional layers learn more targeted, higher-level abstract features for the current training target. Freezing some of the network layers first can speed up the training and also prevent the weights from being corrupted in the early stage of training. Migration learning can also effectively avoid the problem of poor generalization of the model due to the existence of local minima in the objective function. The migration learning strategy in this paper is as follows.

1. Selecting publicly available DenseNet weight files that have been trained through Imagenet or other large datasets as the parameter source for pre-trained weights for migration learning.
2. Load the weight files into the backbone network of this paper's model, and then freeze the backbone network without participating in back propagation. The other unfrozen network layers are trained with a certain number of epochs to perform gradient updates.
3. After a certain number of epoch iterations, the frozen layers are unfrozen and all network layers are involved in the backpropagation update to finally obtain the appropriate parameter matrix and bias vector.

6.1.5. Experimental conditions and parameter settings

The experiments in this paper are conducted under Linux with Intel(R) Xeon(R) CPU E5-2678 v3 @ 2.50GHz processor, 100GB RAM, NVIDIA GTX1070Ti graphics card, and Pytorch 1.8.0 framework for model training and testing. The training parameters were set as shown in table 1.

Table 1
Hyperparameter settings for model training.

Training strategies	Epoch	BatchSize	Optimizer	Weight decay	num_workers	Learning rate
Backbone-freeze	1–50	32	Adam	0	4	$5e^{-4}$
Backbone-unfreeze	51–100	8	Adam	0	4	$9.7e^{-6}$

6.2. Comparative experiments and discussion

6.2.1. Comparison with YOLOv4

Since the framework of this paper is inspired by the structure of YOLOv4, this paper mainly uses YOLOv4 as the comparison object to test the improvement results.

1. Training process comparison.

To show the convergence of the model, the YOLOv4 algorithm is compared with the proposed algorithm in terms of the loss values at training time. Since the initial training loss values are large, the curve generation will disturb the overall display, so the values of the first 5 epochs are removed from the loss curve graph. The final loss curves of the two models can be seen in figure 7.

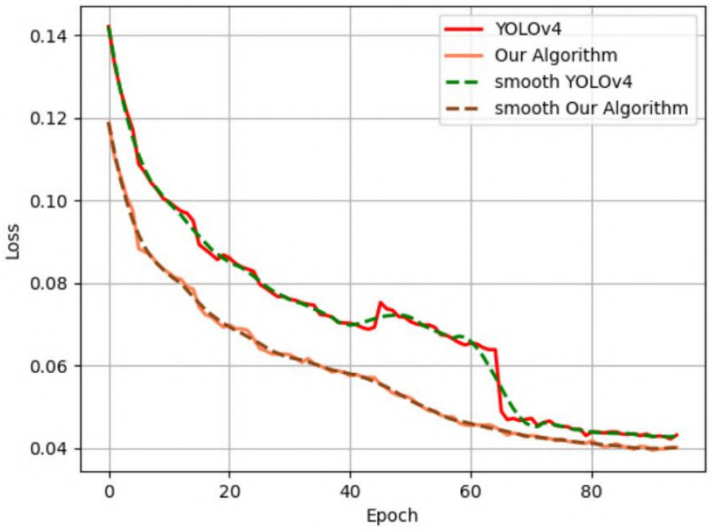

Figure 7: Loss trend comparison.

On the figure, we can see that the trend of the loss value tends to be smooth in the last 10 epochs, and there is almost no change in the magnitude, and the final convergence for the proposed algorithm is smaller than that of the original YOLOv4 model under the same loss function calculation for both models.

2. Model complexity comparison

The FLOPs and Params introduced in the previous section are used here to measure the complexity of the models.

According to the different model components and usage methods, the experimental groups are divided into four groups for comparison in this paper. The group marked as A1 is the original YOLOv4 algorithm; the group (A2) is based on YOLOv4 with the addition of an ECA layer in addition to replacing the backbone network with DenseNet; while the third group (A3) differs from A2 in that the ECA layer is turned into an ASFF structure; and finally, the fourth group (A4) has the proposed algorithm, with the replacement of the DenseNet backbone network and adding both ECA and ASFF layers. The results of each experimental group are given in table 2.

Table 2
Comparison of parameters of different models.

Experimental group	Method	FLOPs/G	Params/M
A1	YOLOv4	59.896	64.040
A2	DenseNet+ECA	25.880	16.116
A3	DenseNet+ASFF	30.796	23.293
A4	DenseNet+ECA+ASFF	30.799	23.293

As we can see, the original YOLOv4 algorithm has the highest complexity in the table, thus it takes longer time in training and inference, and the final generated weight file takes up more space. The comparison between A3 and A4 shows that the ECA attention structure not only works well, but also increases the computational pressure minimally; by comparing A2 and A4, it can be seen that the ASFF structure has a certain fraction on both FLOPs and Params when the computation of the ECA layer is

known to be tiny. By comparing A1 and A4, we can see that the computation of the proposed algorithm in this paper is only 51% of that for YOLOv4 and the number of parameters is only 36% of that of the original one, so the proposed algorithm can significantly reduce the usage cost of the model and is beneficial to the deployment of vehicle detection algorithms in practice.

3. Comparison of detection effect

In terms of detection effectiveness, YOLOv4 is used as baseline to compare with the proposed algorithm on a variety of datasets. The results are shown in table 3. As it can be seen from the table, the proposed algorithm has an improvement effect on different datasets compared to Baseline. In the tasks and datasets related to vehicle detection, the RSOV dataset improves by 3.53% and the BIT-Vehicle dataset improves by 2.46%. This confirms that the proposed algorithm has good applicability in the vehicle detection task. Meanwhile, the mAP of the PASCAL VOC dataset improves by 2.86%. This confirms that the proposed algorithm has good generalization ability and still performs well.

Table 3
mAP for the proposed algorithm and YOLOv4 with multiple datasets (%).

Algorithm	RSOV	BIT-Vehicle	PASCAL VOC
YOLOv4	95.17	93.55	84.83
Our	98.70	96.01	87.69

In figure 8, the recognition effects and detection heat maps of the two algorithms under the generic task are shown. The comparison between D1 and D2 shows that both algorithms have excellent performance in detecting people, however, D1 has the situation of missing detection for the obscured transportation, and it can also be seen from the heat map E1 that the YOLOv4 algorithm does not pay much attention to the target location of the transportation in the missing detection area. The recognition ability of D2 has been greatly improved. On E2 we see that the attention of the proposed algorithm to the missed detection region of D1 has been well improved.

6.2.2. Performance comparison with other algorithms

To further validate the downstream detection task and the proposed algorithm, the proposed algorithm is compared with other mainstream algorithms in with respect to detection effect, complexity and various other indicators. The experimental conditions and parameter settings are the same as in section 6.1.5 of in this paper, and the data sets are selected as RSOV for vehicle detection tasks and VOC data sets representing generic detection tasks, and the comparison results of various algorithms are illustrated in table 4. As can be seen, the proposed algorithm outperforms other algorithms on a variety of data sets. The YOLOv4-MobilenetV3 network, which combines the MobilenetV3 backbone network with YOLOv4, has the simplest model with a low number of parameters, but its mAP has a significant gap compared with other algorithms, making it difficult to meet the requirements for accurate vehicle recognition in traffic scenarios. EfficientDet-d3 and YOLOv4-MobilenetV3 have the same level of counts and excellent detection, but the algorithm takes longer time to train, on average 2-3 times longer than other one-stage algorithms, so it is not suitable for scenarios with strict speed requirements. YOLOv3 and YOLOv4 are both classic algorithms of the YOLO family, and although their detection capabilities perform well, the model parameters of both algorithms are relatively large, which can bring hardware expenses at the practical deployment level for traffic vehicle detection tasks. The more effective retinanet is reduced in the number of parameters compared to the former, but its computational effort is significantly aggravated, and the detection effect is also a certain distance from the proposed algorithm. Therefore, the above comparison results show that the proposed algorithm has a balance of computation, number of parameters and detection effect, not only has excellent detection effect, but also has advantages in training difficulty and model deployment cost compared with other algorithms.

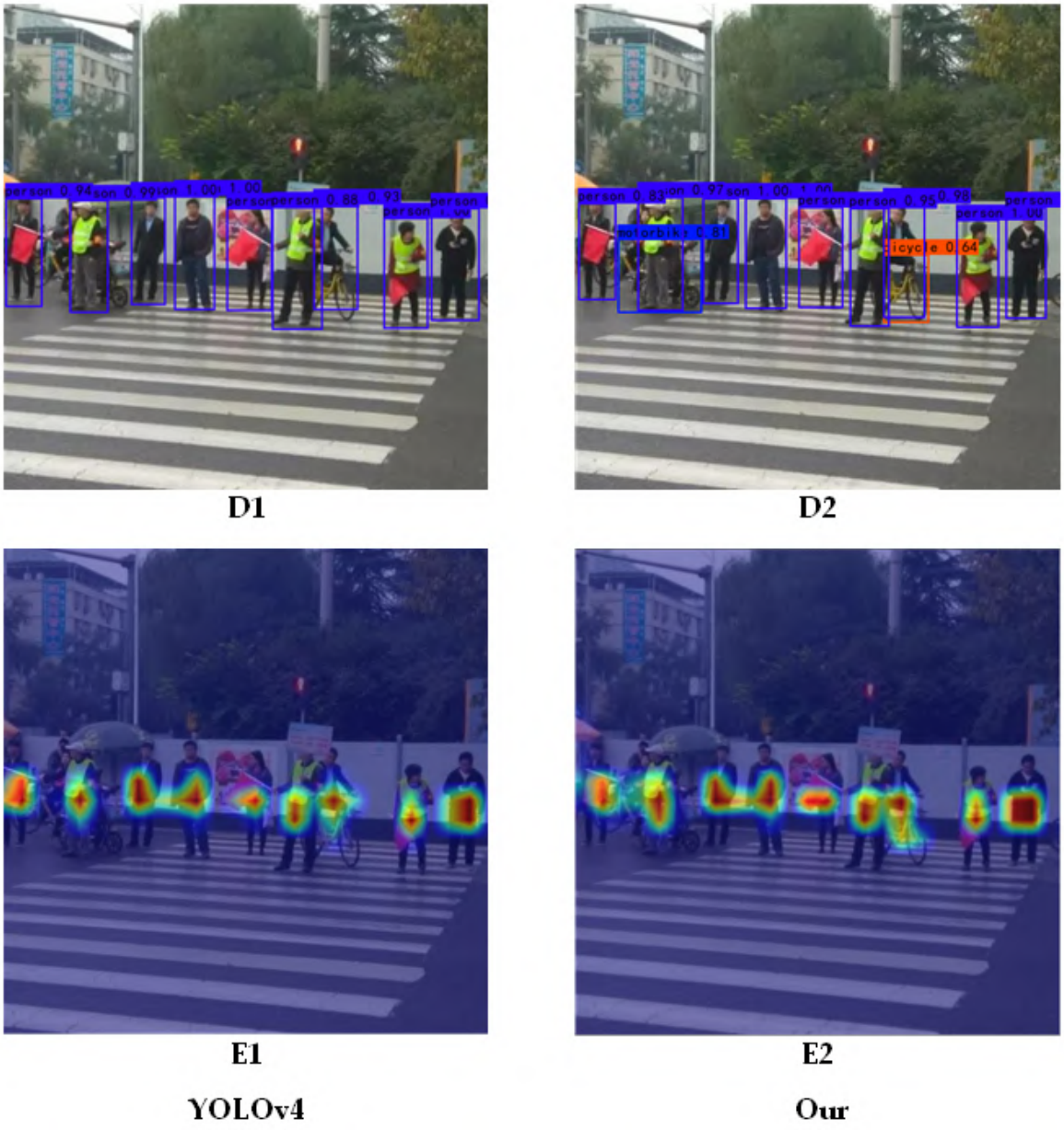

Figure 8: Visualization of general task detection effect.

6.3. Ablation experiments and analysis

Ablation experiments are mainly used to analyze the degree of influence of different components on the whole model. To highlight the effectiveness and synergy of the improvement points of the proposed algorithm, ablation experiments are conducted by using some of the improvement points in this paper. The dataset is selected as RSOV for vehicle detection task and VOC dataset representing generic detection task, where the experimental configuration and parameter settings for the ablation experiments are the same as in section 6.1.5 of this paper.

In this ablation experiment, the component modules are classified as follows: DenseNet backbone network module, ECA attention module, ASFF feature fusion module, and mosaic data enhancement module. The mAP performance of each of their groups is shown in table 5. In table 5, "✓" indicates that the method is used and "–" indicates that the method is not used.

Firstly, we can see there is a considerable improvement in the detection efficiency of the target when

Table 4
Comparison of the effects of different algorithms on multiple datasets.

Algorithm	mAP/%		FLOPs/G	Params/M
	RSOV	VOC07+12		
YOLOv3	93.99	83.15	65.658	61.626
YOLOv4	95.17	84.83	59.896	64.040
YOLOv4-MobilenetV3	86.63	73.67	7.162	11.406
Retinanet	96.14	85.16	151.013	36.724
EfficientDet-d3	96.76	83.11	46.871	11.931
Ours	98.70	87.69	30.799	23.293

Table 5
Results of ablation experiments.

Experience group	Component				mAP/%	
	DenseNet	ECA	ASFF	mosaic	RSOV	VOC07+12
G1	–	–	–	–	95.17	84.83
G2	✓	–	–	–	95.55	85.84
G3	✓	–	✓	✓	98.31	86.80
G4	✓	✓	✓	–	98.17	86.43
G5	✓	✓	–	✓	97.24	86.67
G6	✓	✓	✓	✓	98.70	87.69

all the improved modules are working in concert. From the experimental data of G1 and G2, it can be seen that the detection network using DenseNet lightweight backbone has good feature extraction ability for a variety of datasets based on migration learning. Comparing G4, G5 and G6, it can be seen that the obvious enhancements to mAP are the ASFF module and the mosaic data enhancement method. These two methods enhance the utilization of feature information and the generalization ability of the model, especially the synergistic effect of the two makes the enhancement more obvious. As can be seen in G3, the enhancement effect of the ECA attention module is small in percentage, however, its ability to refine the effective information in the channels allows it to enhance the mAP to a new superior value when used in combination with the ASFF module and the mosaic method. Thus, this ablation experiment is a good demonstration of the effectiveness of each component module of the proposed algorithm.

7. Conclusion

This paper focuses on the one-stage target detection method which has higher requirements for detection speed and deployment cost. Therefore, the proposed method uses YOLOv4 as the base architecture, and significantly reduces the number of parameters in the model by replacing the DenseNet, which has excellent performance, as the backbone feature extraction network; reconstructs the existing FPN network module, uses the ECA attention structure for the transition and transfer of feature information between backbone and neck, and adds the information cross-fusion function before the final detection layer of the network of the ASFF structure; while optimizing in terms of loss function and image preprocessing. The studies of the efficiency of the proposed method are carried out on RSOV, BIT-Vehicle and VOC datasets.

The training process converges faster with a lower value of the loss function for the proposed method. Comparison of complexity between the proposed method and the basic YOLOv4 shows the almost twofold decrease in complexity and the number of parameters is reduced by 64%. The detection accuracy in various datasets with different degrees, for example, the mAP reaches 98.70% on the test set of RSOV dataset. The research results provide some reference value for the traffic construction of smart cities.

For the data augmentation algorithm and traffic information recognition algorithm proposed in this article, the expansion and optimization of the dataset can be considered. By expanding and optimizing

the dataset, the accuracy and generalization of traffic information recognition can be improved. For example, images of different vehicle types can be added to enrich the dataset of traffic scenes, improve data diversity, and more data augmentation techniques such as scaling and random cropping can be used to increase data volume. We also considered the deployment possibility of models in embedded devices in vehicles, and therefore strictly controlled the complexity of algorithms.

Recently new versions of YOLO appeared. In our future studies, we will focus on implementing the developed algorithm in modern versions of YOLO.

Declaration on Generative AI: The author have not employed any generative AI tools.

References

[1] X. Dai, Y. Chen, B. Xiao, D. Chen, M. Liu, L. Yuan, L. Zhang, Dynamic Head: Unifying Object Detection Heads with Attentions, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 7369–7378. doi:10.1109/CVPR46437.2021.00729.

[2] M. Tan, Q. V. Le, EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, in: K. Chaudhuri, R. Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, volume 97 of *Proceedings of Machine Learning Research*, PMLR, 2019, pp. 6105–6114. URL: http://proceedings.mlr.press/v97/tan19a.html.

[3] X. Zhang, X. Zhou, M. Lin, J. Sun, ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 6848–6856. doi:10.1109/CVPR.2018.00716.

[4] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, CoRR abs/1704.04861 (2017). URL: http://arxiv.org/abs/1704.04861. arXiv:1704.04861.

[5] A. Bochkovskiy, C. Wang, H. M. Liao, YOLOv4: Optimal Speed and Accuracy of Object Detection, CoRR abs/2004.10934 (2020). URL: https://arxiv.org/abs/2004.10934. arXiv:2004.10934.

[6] Y. Gong, X. Yu, Y. Ding, X. Peng, J. Zhao, Z. Han, Effective Fusion Factor in FPN for Tiny Object Detection, in: 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), 2021, pp. 1159–1167. doi:10.1109/WACV48630.2021.00120.

[7] N. Guo, Z. Bai, Multi-scale Pulmonary Nodule Detection by Fusion of Cascade R-CNN and FPN, in: 2021 International Conference on Computer Communication and Artificial Intelligence (CCAI), 2021, pp. 15–19. doi:10.1109/CCAI50917.2021.9447531.

[8] T. Ahmad, X. Chen, A. S. Saqlain, Y. Ma, FPN-GAN: Multi-class Small Object Detection in Remote Sensing Images, in: 2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), 2021, pp. 478–482. doi:10.1109/ICCCBDA51879.2021.9442506.

[9] Y. Wang, A. Zell, Yolo+FPN: 2D and 3D Fused Object Detection With an RGB-D Camera, in: 2020 25th International Conference on Pattern Recognition (ICPR), 2021, pp. 4657–4664. doi:10.1109/ICPR48806.2021.9413066.

[10] M. Jaderberg, K. Simonyan, A. Zisserman, K. Kavukcuoglu, Spatial Transformer Networks, in: C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, R. Garnett (Eds.), Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, 2015, pp. 2017–2025. URL: https://proceedings.neurips.cc/paper/2015/hash/33ceb07bf4eeb3da587e268d663aba1a-Abstract.html.

[11] J. Hu, L. Shen, G. Sun, Squeeze-and-Excitation Networks, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132–7141. doi:10.1109/CVPR.2018.00745.

[12] S. Woo, J. Park, J. Lee, I. S. Kweon, CBAM: Convolutional Block Attention Module, in: V. Ferrari, M. Hebert, C. Sminchisescu, Y. Weiss (Eds.), Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII, volume 11211 of *Lecture Notes in Computer Science*, Springer, 2018, pp. 3–19. doi:10.1007/978-3-030-01234-2_1.

[13] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, X. Tang, Residual Attention Network for Image Classification, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6450–6458. doi:10.1109/CVPR.2017.683.

[14] H. Basak, R. Kundu, A. Agarwal, S. Giri, Single Image Super-Resolution using Residual Channel Attention Network, in: 2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS), 2020, pp. 219–224. doi:10.1109/ICIIS51140.2020.9342688.

[15] V. Mnih, N. Heess, A. Graves, K. Kavukcuoglu, Recurrent Models of Visual Attention, in: Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada, 2014, pp. 2204–2212. URL: https://proceedings.neurips.cc/paper/2014/hash/09c6c3783b4a70054da74f2538ed47c6-Abstract.html.

[16] W. Li, X. Zhang, Y. Peng, M. Dong, Spatiotemporal Fusion of Remote Sensing Images using a Convolutional Neural Network with Attention and Multiscale Mechanisms, International Journal of Remote Sensing 42 (2021) 1973–1993. doi:10.1080/01431161.2020.1809742.

[17] J. Deng, L. Cheng, Z. Wang, Attention-based BiLSTM fused CNN with gating mechanism model for Chinese long text classification, Comput. Speech Lang. 68 (2021) 101182. doi:10.1016/J.CSL.2020.101182.

[18] M. Lin, Q. Chen, S. Yan, Network In Network, in: Y. Bengio, Y. LeCun (Eds.), 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014. URL: http://arxiv.org/abs/1312.4400.

[19] N. K. Kim, H. K. Kim, Polyphonic Sound Event Detection Based on Residual Convolutional Recurrent Neural Network With Semi-Supervised Loss Function, IEEE Access 9 (2021) 7564–7575. doi:10.1109/ACCESS.2020.3048675.

[20] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, D. Ren, Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression, in: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, AAAI Press, 2020, pp. 12993–13000. doi:10.1609/AAAI.V34I07.6999.

[21] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, S. Savarese, Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 658–666. doi:10.1109/CVPR.2019.00075.

[22] Z. Zheng, P. Wang, D. Ren, W. Liu, R. Ye, Q. Hu, W. Zuo, Enhancing Geometric Factors in Model Learning and Inference for Object Detection and Instance Segmentation, IEEE Transactions on Cybernetics 52 (2022) 8574–8586. doi:10.1109/TCYB.2021.3095305.

[23] Z.-j. Jiang, Y.-f. Fan, Singularity Intensity Function Analysis of Autoregressive Spectrum and Its Application in Weak Target Detection Under Sea Clutter Background, Radio Science 55 (2020) e2020RS007108. doi:doi.org/10.1029/2020RS007108.

[24] X. Wei, S. Liu, Y. Xiang, Z. Duan, C. Zhao, Y. Lu, Incremental learning based multi-domain adaptation for object detection, Knowl. Based Syst. 210 (2020) 106420. doi:10.1016/J.KNOSYS.2020.106420.