The Journal of Supercomputing (2023) 79:2108–2136
https://doi.org/10.1007/s11227-022-04726-7

MAN and CAT: mix attention to nn and concatenate attention to YOLO

Runwei Guan[1,2,3] · Ka Lok Man[2] · Haocheng Zhao[1,2,3] · Ruixiao Zhang[4] · Shanliang Yao[2] · Jeremy Smith[1] · Eng Gee Lim[2] · Yutao Yue[3]

Accepted: 13 July 2022 / Published online: 6 August 2022

Abstract
CNNs have achieved remarkable image classification and object detection results over the past few years. Due to the locality of the convolution operation, although CNNs can extract rich features of the object itself, they can hardly obtain global context in images. It means the CNN-based network is not a good candidate for detecting objects by utilizing the information of the nearby objects, especially when the partially obscured object is hard to detect. ViTs can get a rich context and dramatically improve the prediction in complex scenes with multi-head self-attention. However, it suffers from long inference time and huge parameters, which leads ViT-based detection network that is hardly be deployed in the real-time detection system. In this paper, firstly, we design a novel plug-and-play attention module called mix attention (MA). MA combines channel, spatial and global contextual attention together. It enhances the feature representation of individuals and the correlation between multiple individuals. Secondly, we propose a backbone network based on mix attention called MANet. MANet-Base achieves the state-of-the-art performances on *ImageNet* and *CIFAR*. Last but not least, we propose a lightweight object detection network called *CAT-YOLO*, where we make a trade-off between precision and speed. It achieves the *AP* of 25.7% on *COCO 2017 test-dev* with only 9.17 million parameters, making it possible to deploy models containing ViT on hardware and ensure real-time detection. CAT-YOLO could better detect obscured objects than other state-of-the-art lightweight models.

Keywords Attention mechanism · Object detection · Object recognition · Plug-and-play NN · Lightweight NN

Runwei Guan and Ka Lok Man authors have contributed equally to this work.

✉ Yutao Yue
yueyutao@idpt.org

Extended author information available on the last page of the article

1 Introduction

Image classification and object detection have been developed rapidly in the past few years, mainly including two types of neural networks: a CNN based and a vision transformer based(ViT based).

For CNN, it is characterized by small size and fast computing speed, and the network has rich inductive biases, which can quickly converge on the training set and extract rich semantic features. However, due to its locality, CNN is not a good candidate for modeling the global feature. In image classification, CNN-based neural networks include ResNet [1], VGG [2], EfficientNet [3], etc. In object detection, CNN-based neural networks include Faster R-CNN [4], YOLOv3 [5], YOLOv4 [6], YOLOv5 [7], YOLOX [8], etc.

Vision transformer (ViT) abandons inductive bias and treats each image as a sequence. Compared with CNN, ViT has a more extensive model capacity, extracting contour information well and recognizing objects in complex scenes more robust [9]. It also has better generalization. However, ViT is stuck in its substantial parameters and long inference time. ViT-based classification models include ViT [10], Swin [11], Twins [12], DeiT [13], CeiT [14] and TNT [15], etc. In object detection, ViT-based models include DETR [16], YOLOS [17], Deformable DETR [18], etc.

To sum up, we conclude that CNN and ViT have good complementarity. CNN is fast but lacks generalization and robustness compared with ViT. ViT has strong generalization and robustness due to its multi-head attention mechanism, but prolonged inference time.

Because of these shortcomings of CNN, many plug-and-play attention modules are designed to improve the accuracy of CNN-based neural networks, such as SENet [19], SKNet [20] and CBAM [21]. SENet and SKNet incorporate spatial information into channel features and compute corresponding attention maps using multi-layer perceptron (MLP) layers [22]. CBAM provides a solution that sequentially embeds channel and spatial attention modules. However, we have found that almost all the plug-and-play modules proposed so far are still based on pure CNN, which means that they still retain some features of CNN. They are still unstable when recognizing some complicated objects.

Although large models can provide high detection accuracy, many areas in the industry are temporarily not suitable or unable to use large models due to the limitations of hardware memory and hashrate, such as product detection on factory production lines [23], traffic perception [24], medical image [25] and assisted driving systems [26]. Many scenes in these fields have high requirements for real-time detection quality [27]. Engineers usually like to use a lightweight CNN-based network, which is more convenient for deploying and achieving real-time object detection at high speed. However, its accuracy needs to be improved, especially when detecting obscured targets in some complex scenes. It is still a difficult task for traditional tiny CNN-based object detection networks. CNNs are not robust enough to prevent these noises. ViTs can model the global dependency and have very high robustness due to their multi-head attention mechanism, which can well exclude noise. However, the pure ViT-based model cannot exactly achieve real-time detection on high-performance GPU. For instance, the FPS of YOLOS [17] is only 5.7 on one 1080Ti GPU, which is unable for real-time detection, let alone on the hardware with low-memory and low-hashrate.

In this paper, we focus on improving neural networks by mixing three attention mechanisms. We design a mix attention module, where we integrate a lightweight ViT-based network into CNN elegantly. We add as-few-as-possible parameters to improve the precision of lightweight NNs for detecting obscured objects in complex scenes. At the same time, we guarantee the real-time detection. The main contributions of this paper can be summarized as follows:

- We design a plug-and-play layer and module called mix attention layer (MAL) and mix attention module (MAM). MAL and MAM both combine the feature with spatial, channel and global contextual attention. We can plug them into any CNN-based network to enhance the feature representation in the channel, space and context with low cost.
- We propose a hybrid-structure image classification network(backbone) called MANet based on MAL, which has the state-of-the-art performance on image classification.
- We propose a lightweight real-time object detection network called CAT-YOLO, which elegantly concatenates the MAL and MAM to YOLOv4-Tiny after the trade-off between precision and speed. CAT-YOLO gets the AP of 25.7% in *COCO 2017 test-dev*, 3.9% higher than YOLOv4-Tiny. In our own obscured-object dataset, CAT-YOLO gets the AR_{50} of 48.7%, which is 5.6% higher than YOLOv4-Tiny.

2 Related work

2.1 Development process of object detection network

Object detection based on deep learning has developed rapidly over the past few years, during which period many novel works have emerged while the speed and accuracy have greatly improved. We summarize these works as follows:

- Evolution of two-stage networks to one-stage networks.
- Importance of each component in the object detection network.
- Exploration of ViT-based neural network in object detection.

For the stage paradigm of object detection network based on CNN, one-stage networks that Fig. 1 shows have become the mainstream object detection paradigms and gradually replaced two-stage networks in both academia and industry.

For the two-stage object detection network, R-CNN [28] is one of the originators. It combines a CNN with an SVM classifier. The CNN is used to extract image features while a large number of proposals are obtained by selective search. After that, proposals are sent to SVM for classification and bounding box regression. Fast

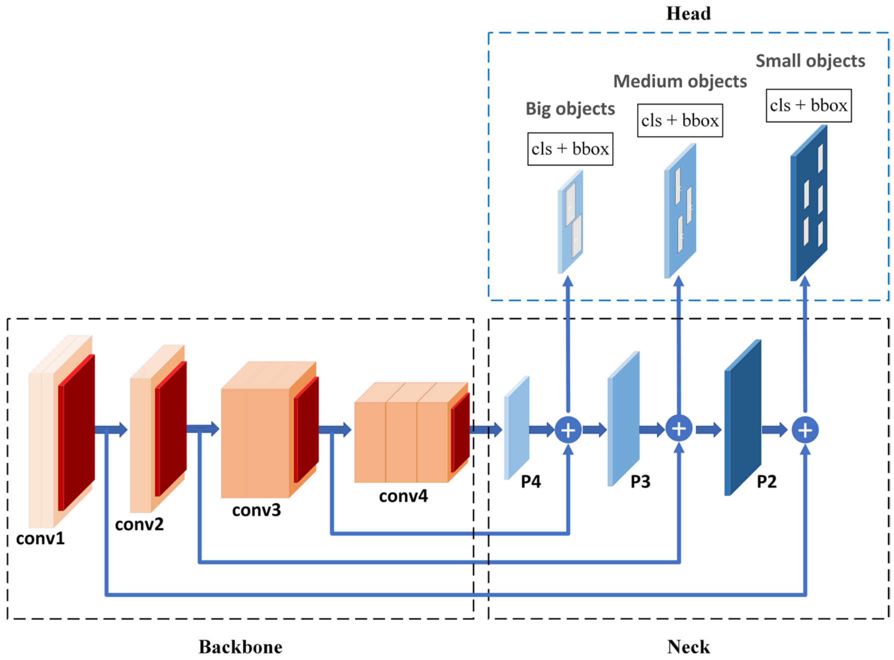

Fig. 1 Typical one-stage detection network

R-CNN replaces SVM classifier in R-CNN with ROI pooling and fully connected layers. Faster R-CNN [29] replaces selective search with a region proposal network to generate proposals for object detection. However, two-stage networks are very slow and can hardly do real-time detection in some scenarios, which is a key reason why one-stage object detection networks spring up.

YOLO-based networks are undoubtedly the representations of one-stage network, which extract ROI, object classification and bounding box regression at the same time. YOLO official series includes YOLOv1 [30], YOLOv2 [31] and YOLOv3 [5]. YOLOv4 [6] firstly introduces the path aggregate network(PAN) [32] as the neck to fuse features of different scales and enrich semantic information. Secondly, it uses many data augmentation methods, such as Mixup [33], Cutmix [34] and Mosaic [6]. Thirdly, CIoU [35] loss replaces the traditional IoU loss to optimize the bounding box regression. In addition, label smoothing [36] is used to process the prediction results to alleviate the over-confidence of the neural network. After that, YOLOX [8] re-introduces anchor-free and decouples the prediction head.

Backbone occupies more parameters and computation in traditional object detection networks. However, GiraffeDet [37] redesigns the paradigm for object detection, which proposes a lightweight backbone called S2D-chain while attaching importance to the neck.

With the rise of ViT [10], we have noticed that the multi-head self-attention mechanism could improve the accuracy of image classification. For object detection, an object detection network based on the encoder–decoder paradigm is also proposed. DETR [16] uses a transformer for bounding box regression and object classification. YOLOS [17] uses a whole-transformer architecture from feature extraction to target detection. Deformable DETR [18] draws on the idea of the deformable convolutional network(DCN) [38] and proposes a deformable attention mechanism. Although self-attention could help the network improve the precision of object detection in some complex scenes, ViT-based networks still suffer from huge parameters and slow inference speed.

2.2 ViT-CNN hybrid backbone

ViTs have many advantages on the recognition of complex samples. Paul et al. [39] argue ViT is very robust to adversarial attacks and noises. Nasser et al. [40] find ViT is good at the recognition of obscured objects. CNN needs much less data than ViT to train for its inductive bias. ViT-CNN hybrid network is proposed as a new-paradigm backbone, which could take advantage of both. BoTNet [41] adds the multi-head self-attention to the end of each stage. Visformer [42] inserts multi-head self-attention in each basic convolution block. Most ViT-CNN hybrid backbone actually only take multi-head self-attention into CNN instead of the whole ViT structure.

2.3 Attention of CNNs

Almost all plug-and-play modules are based on CNN. CNN has its own attention mechanism. SENet [19] proposes a channel-based attention mechanism, which enables the network to combine the semantic information learned by each channel. By introducing the inception module, SKNet [20] fuses the results of convolutions of different sizes to compute spatial attention features. ECA-Net [43] uses 1*1 convolution kernels to replace the fully connected layers in SENet, which significantly reduces the number of parameters. It is a lightweight channel attention network.

2.4 Attention of ViTs

Attention mechanism in ViT-based neural network is called self-attention. Self-attention mechanism was first proposed in Transformer [44], which is used to compute word vector representations based on full-text data.

In the conventional vision transformer, the attention mechanism is used to perform regional attention on the patch sequence of the image, so as to obtain the attention, location and category features of all patches in each image. Eq.1 shows the attention formula.

$$Att(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

where Q, K and V are the patch sequence of the image multiplied, respectively, by the query, key and value matrix, which are learnable parameters. $\sqrt{d_k}$ is a scale factor for reducing the variance of the result of QK^T. The *softmax* function is for normalization.

ViT-based neural network tends to stack multiple self-attention heads to get richer feature. We call these heads as multi-head self-attention(MHSA). Eq.4 shows the formula of multi-head self-attention.

$$MHSA = Att(QW_1^Q, KW_1^K, VW_1^V) \oplus \cdots \oplus Att(QW_i^Q, KW_i^K, VW_i^V) \tag{2}$$

where $W_i^Q \in R^{d_{model} \times d_k}$, $W_i^K \in R^{d_{model} \times d_k}$, $W_i^V \in R^{d_{model} \times d_v}$ are learnable parameters for generalization. i denotes the number of MHSA. $\oplus$ denotes the operation of stack.

3 Mix attention

As a hybrid mode, mix attention pioneers the combination of the three attention mechanisms in neural networks: channel attention, spatial attention and global contextual attention. Mixing three attention mechanisms could comprehensively improve the feature representation in the network and effectively model the feature map's channel, space and global context, which improves the feature representation of complex, blurred and obscured objects.

In order to meet the needs of different degrees, we design a mix attention layer (MAL) and a mix attention module (MAM), respectively, both of which are plug-and-play. MAL can replace any bottleneck in the neural network while MAM can be inserted into any position in the neural network.

As follows, we mix three attention mechanisms and adopt the parallel architecture:

1. For CNNs, the channel corresponds to the semantic information of the feature, and the spatial position corresponds to the absolute position encoding of the feature. These constitute the absolute feature of the object. CNNs with rich inductive bias are very good at implementing the above. Nevertheless, in object detection, we need some relative features to construct the feature map of the object, especially when the detection of the object is difficult. In addition to the feature extraction of the object itself, we need to use some extra information around the object to describe the object and model the feature map. It is equivalent to being able to instruct the machine that there might be an object involved. That is the meaning of the relative feature. ViT will perform global modeling when processing features, and the multi-head attention mechanism will calculate the correlation between different features to achieve relative features of objects. The relative feature could effectively improve the detection of objects in complex scenes. Therefore, channel, spatial and global contextual attention enable global feature modeling of the object itself, location and importance of the object in the entire space. That is why we mix the three types of attention mechanisms together. Theoretically, assuming there is a one-channel absolute-feature map $X \in R^{1 \times W \times H}$, where W denotes the width and H is the height. The inner product result of X with itself in the vector space is $S \in R^{1 \times W \times W}$:

$$S = XX^T, S \in R^{1\times W\times W}. \tag{3}$$

In a vector space, the geometric meaning of S is the degree of correlation or similarity between X and X^T. Since X and X^T are the same thing, we could also call S as self-correlation. Then, we do dot product between S and X and get $A \in R^{1\times W\times H}$:

$$A = S^T X, A \in R^{1\times W\times H}. \tag{4}$$

A is a feature map with self-correlation representation. Actually, it is the skeleton of self-attention mechanism, which intuitively means in addition to the object itself, the model needs to pay attention to other things that are highly relevant to the object.

2. CNNs have rich inductive bias, such as locality, translation invariance and affine invariance. ViTs could learn the global contextual feature. They have different processing mechanisms for the same feature, and the series-mode may cause instability during training and even lead to over-fitting. The parallel structure allows them to stably extract and process feature representations in their own way.
3. Park et al. [45] argued that the convolution is a high-pass filter while the vision transformer is a low-pass filter, which means CNNs are good at extracting texture features while ViTs focus more on shape features.

3.1 Mix attention layer

Fig.2 shows the structure of MAL. There are three kinds of basic units in MAL, which are Basic Convolution Block(BC), Compound Residual Block(CR) and Attention Aggregation.

3.1.1 Basic convolution block

A BC consists of one convolution layer with 3*3 kernels, one batch-norm and one LeakyReLU for activation. BC is used to adjust the channel number and extract feature at the same time. The batch-norm (BN) normalizes the value of feature map to avoid the exploding and vanishing gradient, which would avoid over-fitting in some extent. The *LeakyReLU* can enhance the generalization of the network. Given an image $x_i \in R^{C\times W\times H}$, C is the channel number. The feature map x_{i+1} processed by a basic convolution block is:

$$x_{i+1} = BC(x_i) = LeakyReLU(BN(Conv(x_i))) \tag{5}$$

where the size of x_{i+1} is $R^{n\times W\times H}, n \in N^+$. The width and height remain unchanged for the convolution layer includes a padding operation.

3.1.2 Compound residual block

Before loading the feature map into spatial and channel attention branches, there is a CR, which could extract features with different scales. In addition, the residual block could also significantly alleviate the phenomena of explosion and vanishing gradient.

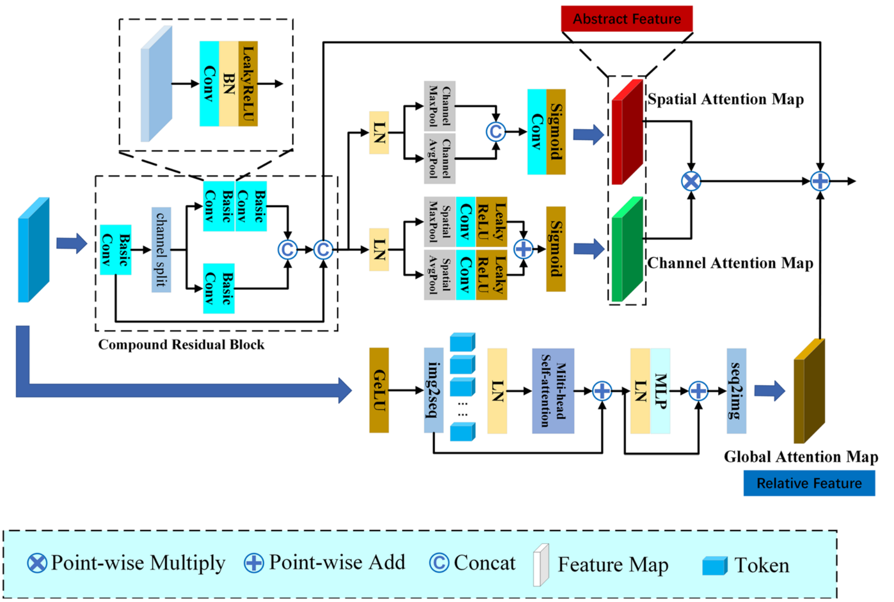

Fig. 2 In mix attention layer(MAL), there are three branches following the compound residual block, which are spatial attention, channel attention and global attention branch. Although the module looks a little bit complex, the parameters are mainly located in convolution blocks and multi-head self-attention block

In CR, there is a small residual unit in a big residual block. Firstly, the feature map $x_i \in R^{C\times W\times H}$ is fed into a BC, then the channels of the feature map are split equally. After that, half of the feature map goes through two BCs while the other half is processed by one BC, which are concatenated to constitute the small residual unit.

$$\begin{aligned} x_{i0} &= BC(BC(\phi(BC(x_i))_0)), x_{i0} \in R^{\frac{C}{2}\times W\times H} \\ x_{i1} &= BC(\phi(BC(x_i))_1), x_{i1} \in R^{\frac{C}{2}\times W\times H} \end{aligned} \tag{6}$$

where $\phi(\cdot)$ refers to the operation of spitting channels, the result of which is a two-element array, containing the first half feature map x_{i0} and the second half feature map x_{i1}.

The result of the small residual unit is concatenated to the feature map before the operation of the channel split. The whole process can be expressed as:

$$x_{i+1} = CR(x_i) = Cat(BC(x_i), Cat(x_{i0}, x_{i1})), x_{i+1} \in R^{2C\times W\times H} \tag{7}$$

where we use $Cat(\cdot)$ as the concatenation operation.

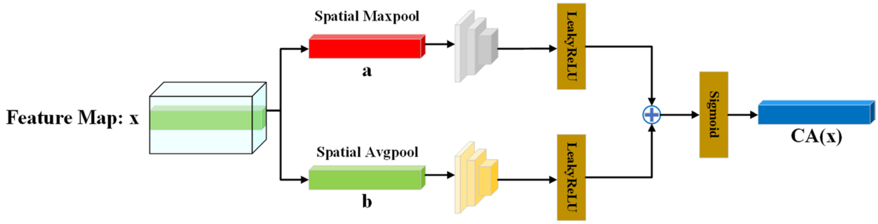

Fig. 3 Channel attention

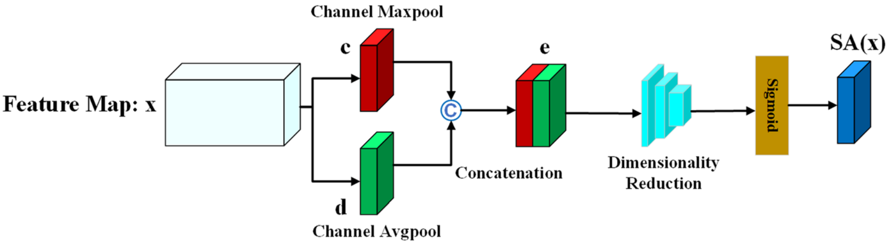

Fig. 4 Spatial attention

3.1.3 Attention aggregation

Three attention branches include the channel attention(CA), spatial attention(SA) and global attention(GA). Each branch pays attention to different features with respective attention mechanisms and fuses together at last.

As Fig. 3 shows, CA pays attention to the inter-channel relationship and channels that matter, which corresponds the texture feature of the image. We empirically argue that max-pooling can obtain the unique information of each channel, and average-pooling can obtain the overall information of each channel. Given a feature map $x \in R^{2C\times W\times H}$, we use max-pooling and average-pooling to jointly extract the features along the spatial axis, then we get the feature maps $a \in R^{2C\times 1\times 1}$ and $b \in R^{2C\times 1\times 1}$. They are, respectively, fed into a 1×1 convolution layer. After that, we perform a *LeakyReLU* activation instead of *ReLU* and do element-wise addition between them. Finally, we use a *sigmoid* to get the channel attention map, which can be called channel attention map. The process is shown in Eq. 8:

$$\begin{aligned} a &= LeakyReLU(Conv(MaxPool(x))), a \in R^{2C\times 1\times 1} \\ b &= LeakyReLU(Conv(AveragePool(x))), b \in R^{2C\times 1\times 1} \\ CA(x) &= \sigma(a \oplus b), CA(x) \in R^{2C\times 1\times 1} \end{aligned} \tag{8}$$

where $\oplus$ indicates the element-wise addition and σ indicates the *sigmoid*.

As Fig.4 shows, SA pays attention to the informative location in the feature map. Given a feature map $x \in R^{2C\times W\times H}$, we first implement the max-pooling and average-pooling to extract the features along the channel axis, then we get feature maps

$c \in R^{1\times W\times H}$ and $d \in R^{1\times W\times H}$. After that, we concatenate c and d in channel axis and get feature map $e \in R^{2\times W\times H}$, then a convolution layer is used to reduce the channel number. Finally, we use a *sigmoid* on the feature map. Eq.9 shows the whole process of generating the spatial attention map, which can be called spatial attention map.

$$
\begin{aligned}
c &= MaxPool(x), c \in R^{1\times W\times H} \\
d &= AveragePool(x), d \in R^{1\times W\times H} \\
e &= Cat(c, d), e \in R^{2\times W\times H} \\
SA(x) &= \sigma(Conv(e)), SA(x) \in R^{1\times W\times H}.
\end{aligned} \tag{9}
$$

GA is constructed by a ViT-based neural network. The feature map $x \in R^{C\times W\times H}$ is firstly flattened into a patch sequence $x_p \in R^{C\times P^2}$, where (P, P) indicates the resolution of each patch. Besides, due to the patch sequence has no positional information, we add the learnable position tokens $pos \in R^{C\times P^2}$ to the patch sequence to get the attention sequence $s_{att} \in R^{C\times P^2}$ with layer-norm(LN). The multi-head attention mechanism is applied to the patch sequence to calculate the feature representation with the similarity between patches $s_{mhsa} \in R^{2C\times P^2}$. Eq.10 shows the whole process:

$$
\begin{aligned}
x_p &= Flatten(x), x_p \in R^{C\times P^2} \\
s_{att} &= LN(x_p \oplus pos), s_{att} \in R^{C\times P^2} \\
s_{mhsa} &= MHSA(s_{att}) \oplus s_{att}, s_{mhsa} \in R^{2C\times P^2}.
\end{aligned} \tag{10}
$$

Then, we add a layer-norm and an MLP layer to fuse the features extracted from different self-attention heads. Besides, a residual side is also considered to alleviate the phenomenon of explosion and vanishing gradient. Finally, the sequence is transformed into the format of the feature map $GA \in R^{2C\times W\times H}$. Eq.11 shows the whole process:

$$
\begin{aligned}
s'_{mhsa} &= MLP(LN(s_{mhsa})) \oplus s_{mhsa}, s'_{mhsa} \in R^{2C\times P^2} \\
GA &= Seq2Img(s'_{mhsa}), GA \in R^{2C\times W\times H}.
\end{aligned} \tag{11}
$$

After we got the channel attention map $CA \in R^{2C\times 1\times 1}$ and the spatial attention map $SA \in R^{1\times W\times H}$, two feature maps are multiplied, where CA broaden the channels of SA. It intuitively indicates that CA tells SA the importance of different channels. After we get the channel–spatial attention map $CSA \in R^{2C\times W\times H}$, a long residual side is used to add the CR and CSA and get $CSA' \in R^{2C\times W\times H}$, which could also dramatically prevent gradient vanishing. Finally, we get the mix attention map $MA \in R^{2C\times W\times H}$ by adding the global attention map GA to CSA'. As Eq.12 shows:

$$
\begin{aligned}
CSA &= SA \otimes CA, CSA \in R^{2C\times W\times H} \\
CSA' &= CSA \oplus CR, CSA' \in R^{2C\times W\times H} \\
MA &= CSA' \oplus GA, MA \in R^{2C\times W\times H}
\end{aligned} \tag{12}
$$

where $\otimes$ denotes the multiply operation with the broadcasting mechanism in Python.

3.2 Mix attention module

As Fig.5 shows, mix attention module(MAM) is a module where MAL drops CR. MAM is an attention-only module that could be plugged in any position of CNNs. Assuming there is an input map $x_i \in R^{C \times W \times H}$, the channel number of the output map x_{i+1} can be any number greater than the number of self-attention heads(SAH) in the module, which is shown in Eq.13.

$$x_{i+1} = MAM(x_i), x_{i+1} \in R^{n \times W \times H}, n \geq len(SAH) \tag{13}$$

where $len(\cdot)$ denotes the calculation of a map's length.

4 MANet

MANet is an image classification network that consists of two parts, a backbone and a head. For the backbone, it includes basic convolution layers(BC) and mix attention layers(MAL). The head is specifically added for image classification. Figure 6 shows the structure of MANet-Small.

Park [45] has proved that multi-head self-attention in deep layers of NN greatly improves the prediction performance than in shallow layers. Therefore, we firstly adopt two basic convolution layers instead of MALs because the shallow layers' features consist of fragmented information without any semantic information, so it is not an excellent choice to directly apply the attention

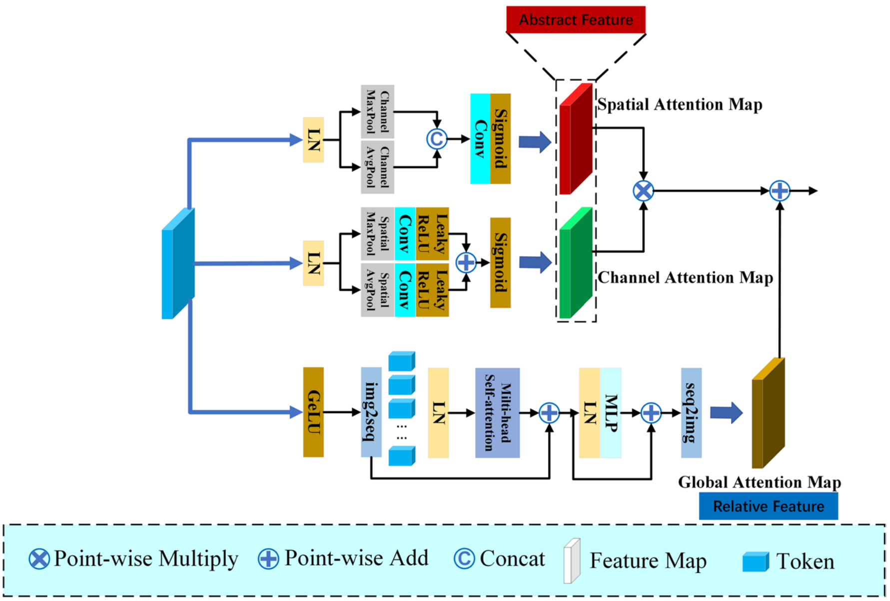

Fig. 5 Mix attention module(MAM)

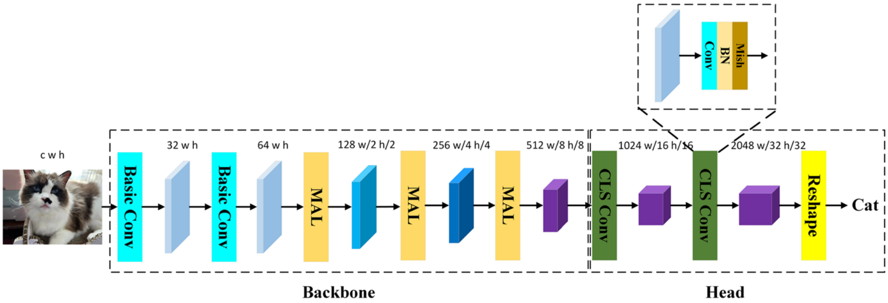

Fig. 6 The structure of MANet-Small. In the backbone, there are 2 basic convolution layers, 3 mix attention layers, 2 cls convolution layers. Assuming the dilation ratio of channel is 2, the image with size $3 \times W \times H$ processed by the backbone of MANet-Small would be resized as $2048 \times \frac{W}{32} \times \frac{H}{32}$

mechanism to them, which will cause longer training time. In addition, if not appropriately trained, directly involving the attention mechanism would cause over-fitting, which means the attention mechanism would mislead the network to pay attention to meaningless areas. Therefore, we first use CNN to deepen the network. As the network deepens, the receptive field of the network gradually becomes immense. At this time, the feature map of the network has learned rich semantic features.

We add three MALs to help the network learn the features self-adaptively, focusing on meaningful semantic feature. The attention mechanism extracts features to the maximum extent from space, channel and global context. In this process, the attention mechanism removes redundant features as much as possible and finely models valuable features.

CLS convolution layers(CC) follow the mix attention blocks, which further process the attention feature maps from MAL and prepare for the classification. We do not use the fully connected layers but two CCs, which considerably reduce unnecessary parameters.

A CC consists of a convolution, a batch-norm and a Mish [46]. Given a feature map $x_i \in R^{C\times W\times H}$, the feature map x_{i+1} processed by a CC is:

$$x_{i+1} = CC(x_i) = Mish(BN(Conv(x_i))), x_{i+1} \in R^{2C\times\frac{W}{2}\times\frac{H}{2}}. \tag{14}$$

The output of MANet is $x_{out} \in R^{1\times1\times N_{cls}}$ after a reshaping operation, where N_{cls} denotes the number of categories.

4.1 Model detail

For a better comparison with other models, we design three models of different sizes, whose main parameters are shown in Table 1. We divide them into 3 levels according to the number of parameters, which are Tiny(T), Small(S) and Base(B) version.

Table 1 The detail of MANet's variants

	BC	MAL			CC
Model	Block	Head	Embed	Block	Block
MANet-T(4.3M)	2	4	256	3	1
MANet-S(23.4M)	2	8	384	3	2
MANet-B(69.3M)	4	16	512	4	3

The kernel sizes of all convolution layers are 3×3 and 1×1. Heads indicates the number of the multi-heads in a vision transformer block. Embed indicates the number of patch embeddings. 4.3M, 23.4M and 69.3M denote the number of the model parameters. The channel in each block could be modified

5 CAT-YOLO

CAT-YOLO in Fig.7 is a lightweight CNN-ViT network for object detection. It consists of 3 parts like YOLOv4-Tiny, which are a backbone, a neck and prediction heads.

5.1 Backbone

We use MANet-Tiny as the backbone, which includes 2 BCs and 3 MALs. Assuming there is an image $img \in R^{3\times512\times512}$, after five-stage feature extraction, we get a feature map $f \in R^{512\times16\times16}$, which has the same downsampling ratio with CSPDarknet53-Tiny in YOLOv4-Tiny. MANet-Tiny could extract richer context feature by attention aggregation in MALs.

5.2 Neck

Neck can get the multi-scale features and fuse them, which is prepared for the object detection in prediction heads. In CAT-YOLO, firstly we use a tiny feature pyramid network(FPN) [47] instead of the combination of SPP [48] and PAN [32], which can reduce parameters. Secondly, we exclude the pyramid-3($P3$) and only include a pyramid-4($P4$) and a pyramid-5($P5$) in the FPN. $P4$ has the feature map with larger width and height than $P5$, so $P4$ is for small object detection while $P5$ is for big object detection with the larger receptive field. To fuse multi-scale features in $P5$ to $P4$, there are some blocks between $P4$ and $P5$, which includes an upsampling block, a mix attention module(MAM).

The upsampling block uses the nearest neighbor interpolation to upsample the feature map of $P5$. The MAM could further extract the feature with attention and enhance the semantic feature.

Assuming there is a $P5_{mix} \in R^{C\times W\times H}$ and a $P4 \in R^{\frac{C}{2}\times2W\times2H}$, we will get a feature map $P4_{Mix} \in R^{\frac{3C}{2}\times2W\times2H}$. The whole process is shown in Eq. 15.

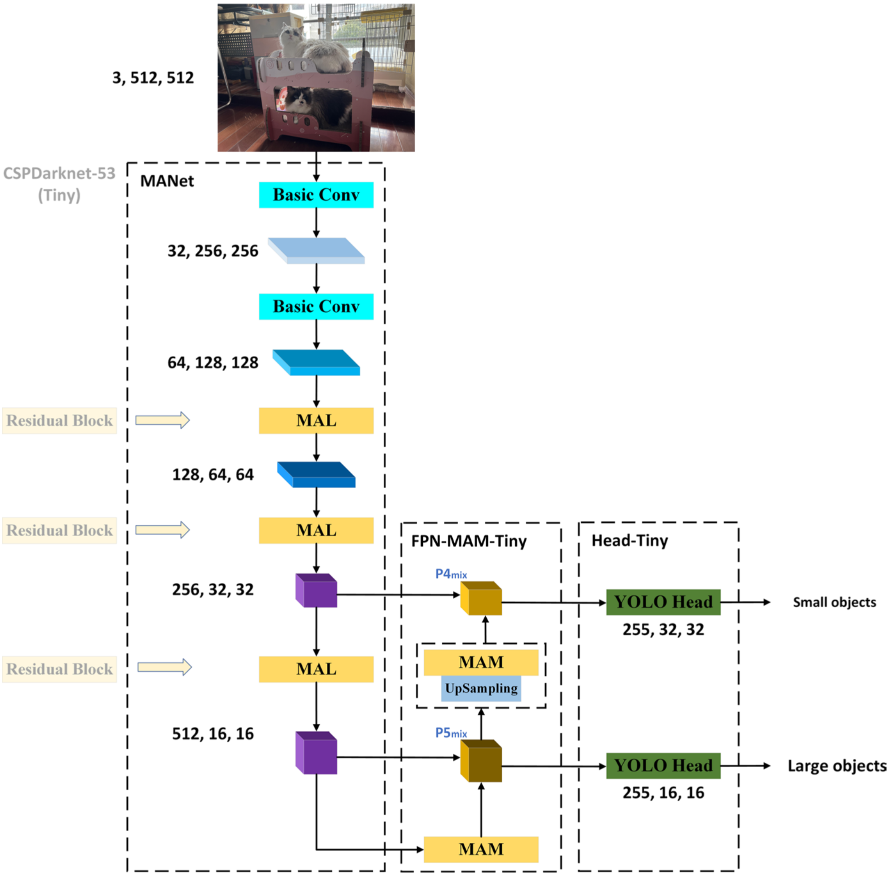

Fig. 7 The structure of CAT-YOLO. It includes a backbone(MANet-Tiny), a neck(FPN-MAM-Tiny) and two YOLO prediction heads. We replace the last three residual blocks with mix attention blocks. In addition, we fine-tune some parameters of convolution layers in the basic convolution blocks to reduce the size of feature map

$$\begin{aligned} P5'_{mix} &= UpSampling(P5_{mix}), P5'_{mix} \in R^{C\times 2W\times 2H} \\ P5'_{MAM} &= MAM(P5'_{mix}), P5'_{MAM} \in R^{C\times 2W\times 2H} \\ P4_{Mix} &= Cat(P4, P5'_{MAM}), P4_{Mix} \in R^{\frac{3C}{2}\times 2W\times 2H} \end{aligned} \tag{15}$$

5.3 Prediction head

We use the YOLO-style heads for object detection, including bounding box regression and object classification. Each object that the head predicts includes three kinds of elements, which are the object coordinates, a confidence score and a classification label. We take *COCO 2017* as an example, the channel number of heads is 255(85 × 3), where 85 includes 80 classes, offset of x, offset of y, height, width and confidence while 3 denotes that there are 3 prior anchor boxes in each channel. There are 2 prediction heads in CAT-YOLO, the head connected to $P4_{mix}$ is to detect small objects while the head connected to $P5_{mix}$ is to detect big objects.

5.4 Loss function

The conventional loss function of the object detection includes 2 parts, the loss of bounding box regression and classification loss.

We use CIoU [35] as the loss function of the bounding box regression. The regression loss of the bounding box is between the ground-truth bounding box and the prior bounding box. In CAT-YOLO, we will first divide the image into $N \times N$ grids, where the center of each prior bounding box is located in the corresponding grid. We design prior bounding boxes with 3 scales and aspect-ratios for better prediction, which is shown in Fig. 8. In order to accelerate the convergence and accuracy in the regression of bounding boxes with different aspect-ratios, $CIoU$ takes the aspect-ratio of bounding box into account, which is shown in Eq. 16.

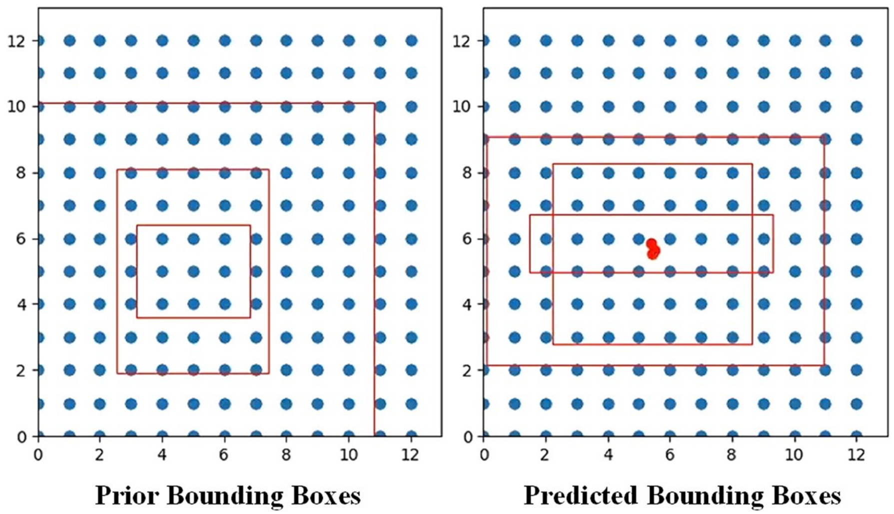

Fig. 8 Examples of prior bounding boxes and predicted bounding boxes in prediction heads. The center of the prior bounding box is the blue dot while the center of the predicted bounding box is the red dot, which is generated from the blue dot with an offset

$$
\begin{aligned}
IoU &= \frac{|b \cap b^{gt}|}{|b \cup b^{gt}|} \\
v &= \frac{4}{\pi^2}(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h})^2 \\
R_{CIoU} &= \frac{\rho^2(b_c, b_c^{gt})}{c^2} + \alpha v \\
L_{CIoU} &= 1 - IoU + R_{CIoU}
\end{aligned}
\tag{16}
$$

where b refers to the predicted bounding box while b^{gt} indicates the ground-truth bounding box. w and h denote the width and height of the bounding box, respectively. v is used to measure the similarity of aspect-ratios between b^{gt} and b, where α is a weight coefficient. For the regularizer R_{CIoU}, $\rho(\cdot)$ denotes the Euclidean distance between the center points of b and b^{gt} while c is the minimum diagonal length of the box enclosing two bounding boxes.

We use focal loss with the label smoothing as the classification loss. Focal loss could alleviate the negative effect of unevenly distributed data samples while label smoothing could significantly avoid over-confidence in the model as much as possible. The equation of the focal loss is shown in Eq.17.

$$
\begin{aligned}
p_i &= \frac{e^{\frac{x_i}{T}}}{\sum_{j=1}^{K} e^{\frac{x_j}{T}}}, \forall 1, 2, \dots, K \\
FL(p_i) &= -(1 - p_i)^\gamma log(p_i)
\end{aligned}
\tag{17}
$$

where T is a scalar temperature parameter above 1.0. It can smooth the output of the softmax function. In focal loss, γ is a hyper-parameter and we empirically set it to 2.

Therefore, the whole loss function is:

$$
L(b, b^{gt}) = L_{CIoU} + FL. \tag{18}
$$

5.5 Optimal bounding box match

The number of bounding boxes that the model predicts in the training process is much more than object number in the image. Therefore, we use non-maximum suppression (NMS) algorithm to filter redundant bounding boxes and leave the best matching box for each object. The Python-style NMS algorithm is shown in Algorithm 1.

Algorithm 1 Non-Maximum Suppression(NMS)

We have a predicted bounding box collection *oldboxes*, *Oldboxes*[0]=[x, y, w, h, iou, confidence]
We define a new bounding box collection *newboxes*
Require: $thres\_conf = 0.5, thres\_iou = 0.4$
sort(*oldboxes*, reverse=True, key= lambda x : x[5])
remove(*oldboxes*, Threshold=*thres_conf*)
while $len(oldboxes) > 0$ **do**
  current_box = *oldboxes*.pop(0)
  newboxes.append(current_box)
  for *box* in *oldboxes*:
   if *current_box*[4] == *box*[4] **then**:
    iou = IOU(*current_box*[:4], *box*[:4])
    if $iou > thres_{iou}$ **then**:
     oldboxes.remove(box)
    end if
   end if
end while

6 Experiments

We divide the experiments into 3 parts, which, respectively, test and verify the performance of MANet(MAL), MAM and CAT-YOLO.

6.1 MANet

We train and test on several classification benchmarks including *ImageNet-1K* [49], *CIFAR-10* [50], *CIFAR-100* [50] and *Tiny-ImageNet* [51].

6.1.1 Implementation details

For different datasets, we adopt different training strategies. Taking *ImageNet* [49] as an example, we firstly resize the image as 224 × 224(px). We use SGDM optimizer with a scheduler of cosine learning rate and use Mixup, etc. as the augmentation. We adopt 128 as the batch size and train for 300 epochs on 4 RTX

Table 2 Implementation details on *ImageNet*

Optimizer	Base LR	Scheduler	Weight decay	Momentum	Batch size	Epochs
SGDM	5e-3	Cosine	1e-3	0.937	128	300

Table 3 Data augmentation for image classification[1]

Method	Mixup	Flip	Padding	ColorJitter	Distortion	Rotation
IsUsed	✓	✓	✓	✓		✓

[1]The data augmentation methods are for all benchmarks, *ImageNet*, *CIFAR-10*, *CIFAR-100* and *Tiny-ImageNet*

Table 4 Top-1 accuracy of MANet on *ImageNet-1K*[1]

Model[2]	Params(M)	Top-1 Accuracy(%)
MANet-B	69.3	81.7
ViT-B [10]	86	79.6
MANet-S	23.4	78.3
DeiT-S [13]	22	76.8
MANet-T	4.3	73.1
CCT-6 [52]	3.17	72.5

[1]*ImageNet-1K* has 1.28 million images with 1000 classes

[2]Models are trained on four RTX 3090Ti GPUs

3090Ti GPUs. The details of the implementation and data augmentation are in Tables 2 and 3.

6.1.2 Comparison with the state-of-the-art

We compare three MANet variants with different CNNs and ViTs, which are all the state-of-the-art models in leaderboards of different benchmarks.

Table 4 shows the comparisons on *ImageNet-1K*. MANet-B with fewer parameters achieves 81.7% top-1 accuracy, 2.1% higher than ViT-B. MANet-S outperforms than DeiT-S about 1.5% top-1 accuracy. MANet-T has 73.1% top-1 accuracy, which is higher than 72.5% top-1 accuracy of CCT-6.

Table 5 shows the comparisons on *CIFAR-10*. MANet-B achieves 97.2% top-1 accuracy, 0.5% higher than ViT-B. MANet-T also outperforms than CCT-6 about 1.1% top-1 accuracy.

Table 6 shows the comparisons on *CIFAR-100*. MANet-B has 88.7% top-1 accuracy. MANet-S and MANet-T, respectively, get 86.5% and 81.6% top-1 accuracy.

Table 5 Top-1 Accuracy of MANet on *CIFAR-10*[1]

Model[2]	Params(M)	Top-1 Accuracy(%)
MANet-B	69.3	97.2
ViT-B [10]	86	96.7
MANet-S	23.4	95.1
DeiT-S [13]	22	94.7
MANet-T	4.3	93.4
CCT-6 [52]	3.17	92.3

[1]*CIFAR-10* has 60k images with 10 classes

[2]Models are trained on one RTX A4000 GPU

Table 6 Top-1 Accuracy of MANet on *CIFAR-100*[1]

Model[2]	Params(M)	Top-1 Accuracy(%)
MANet-B	69.3	88.7
ViT-B [10]	86	87.6
MANet-S	23.4	86.5
DeiT-S [13]	22	85.7
MANet-T	4.3	81.6
CCT-6 [52]	3.17	80.1

[1]*CIFAR-100* has 60k images with 100 classes.

[2]Models are trained on one RTX A4000 GPU

Table 7 Top-1 Accuracy of MANet on *Tiny-ImageNet*[1]

Model[2]	Params(M)	Top-1 Accuracy(%)
MANet-B	69.3	85.7
ViT-B [10]	86	84.3
MANet-S	23.4	82.1
DeiT-S [13]	22	81.6
MANet-T	4.3	76.3
CCT-6 [52]	3.17	76.1

[1]*Tiny-ImageNet* has 100k images with 200 classes

[2]Models are trained on one RTX TITAN GPU

Table 7 shows the comparisons on *Tiny-ImageNet*. MANet-B gets 85.7% top-1 accuracy in *Tiny-ImageNet*, 1.4% higher than ViT-B. MANet-S and MANet-T, respectively, get 82.1% and 76.3% top-1 accuracy.

Fig. 9 These attention maps are learned by the mix attention in MALs

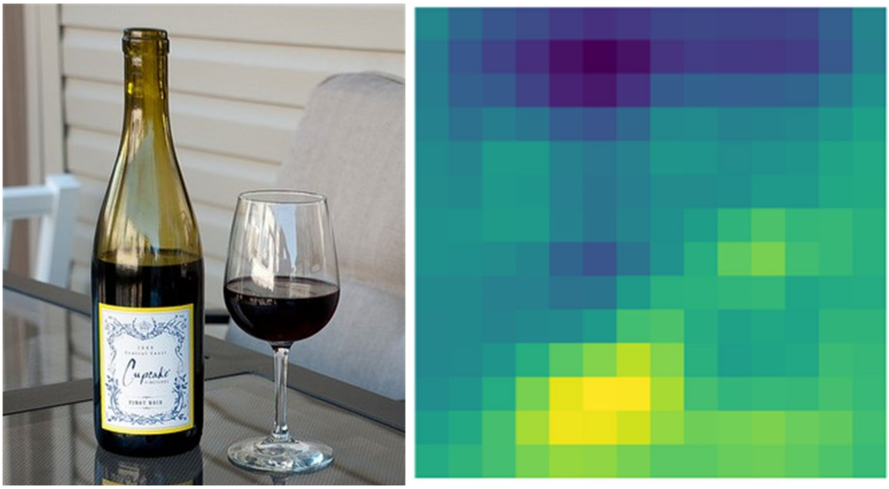

Fig. 10 CNN channel–spatial attention map of the last MAL in MANet-Tiny

6.1.3 Visualization of mix attention map

To further validate the effectiveness of our MANet, we firstly pretrain MANet-B on *Tiny-ImageNet* and a fraction of images in the training set of *ImageNet*, then we load test images to the model to calculate the mean of all attention matrices in all mix attention layers and map them to input images. As Fig. 9 shows, the mix attention including spatial, channel and global contextual attention could perceive features and edge gradients of objects well.

We visualize the feature map of channel–spatial attention in the last MAL. As Fig. 10 shows, the bottle mouth is a typical feature of the wine bottle, and the channel–spatial attention mechanism pays attention to the bottle mouth. At the same time, the channel–spatial attention also notices that the wine in the bottle and glass has the same color. In the pretraining dataset, wine bottles and wine glasses frequently co-occur, but convolution channel–spatial attention does not model their contextual connections well.

In addition, we also calculate and visualize the feature similarity matrices of image patches from multi self-attention heads in the global attention branch, which is learned by ViT-based neural network. Since MANet is a series structure, the input of the vision transformer branch comes from the feature map which is generated by CNN and vision transformer jointly in the last block. As Fig. 11 shows, the feature similarity maps generated from seven self-attention heads in the last MAL have clear context information and feature differentiation degrees, which means the following:

- The global contextual attention from ViT could nicely model the global dependency and correlation among objects. Intuitively, ViT can observe and analyze more about potential relationships between objects. The 3rd and 5th maps show that MANet regards the wine bottle and the wine glass as a whole. It indicates MANet has learned the correlation between objects from the data. With the help of spatial–channel attention, MANet keeps their individual features well.
- ViT-based network could nicely coexist with CNN-based network in MAL.
- The combination of 3 different attention mechanisms works to a certain extent.

6.2 Comparisons with other plug-and-play modules

As a plug-and-play module in neural network, MAM is compared with other plug-and-play modules including SENet and CBAM on *CIFAR-10* and *CIFAR-100*. We use ResNet-101 [1] and WRN-18 [53] as our basic models. We replace the last layer of each bottleneck in these two models with mix attention block.

Table 8 shows top-1 errors of different plug-and-play attention modules on *CIFAR-10*. The combination of ResNet-101 + MAM has the lowest top-1 error among all combinations, which is 0.15% lower than ResNet-101 + CBAM and 0.42% lower than ResNet-101 + SENet. The parameter number is just 0.74 million more than other two models. The combination of WRN-18 + MAM gets the lowest top-1 error(4.77%). The parameters of WRN-18 + MAM are only 1.03 million more than WRN-18 itself.

Table 9 shows top-1 errors of different plug-and-play attention modules on *CIFAR-100*. ResNet-101 + MAM achieves the lowest top-1 error of 23.19% in the ResNet-101 series. For the WRN-18 series, the combination of WRN-18 + MAM achieves the lowest top-1 error of 19.11% among all models on *CIFAR-100*. The results sufficiently prove the effectiveness of MAM with the comparison with other two plug-and-play modules.

6.3 CAT-YOLO

We train and test CAT-YOLO on *COCO 2017* [54]. CAT-YOLO achieves the state-of-the-art results among all lightweight object detection models.

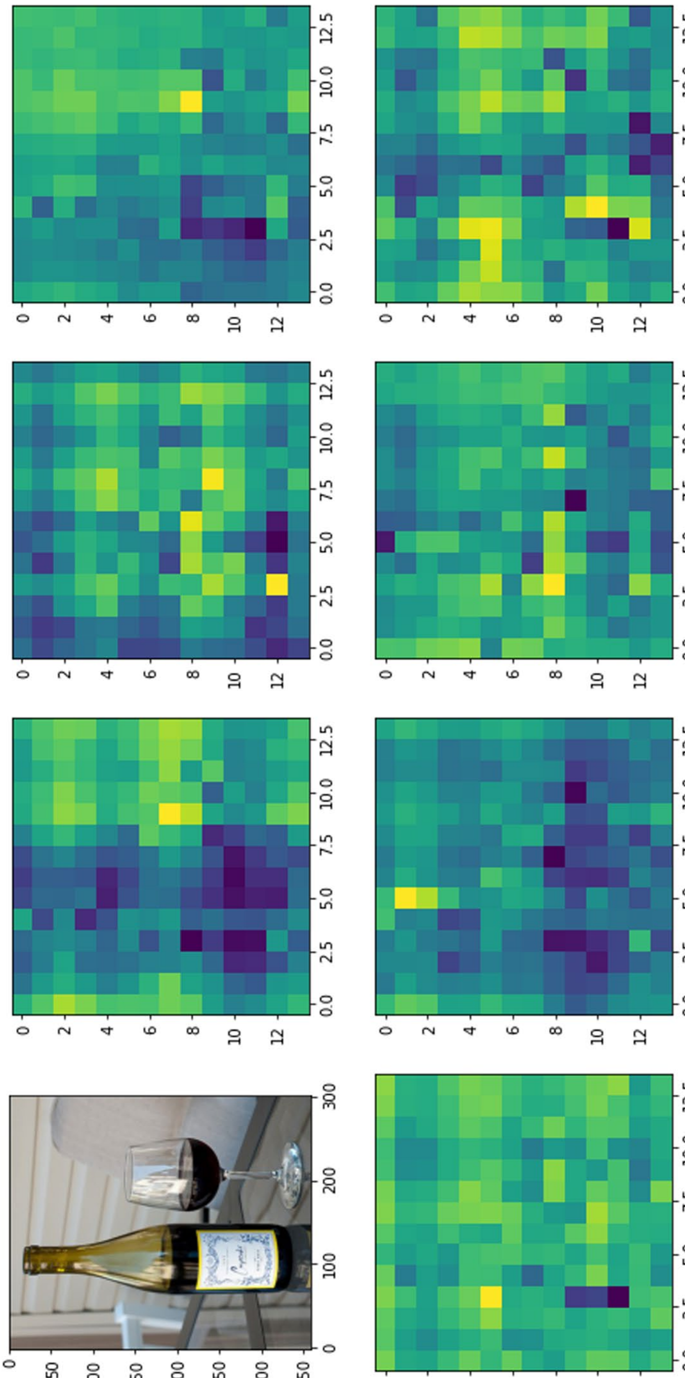

Fig. 11 We select seven self-attention heads of the vision transformer branch in the last MAL. We calculate the matrix QK^T of each head and visualize them as the heatmaps. These heatmaps show the feature similarity matrices learned by multi-head self-attention

Table 8 Top-1 error comparison with plug-and-play modules on *CIFAR-10*

Model[1]	Params(M)	Top-1 Error(%)
ResNet-101 [1]	44.55	7.24
ResNet-101 + SENet [19]	49.33	6.59
ResNet-101 + CBAM [21]	49.33	6.32
ResNet-101 + MAM	50.07	6.17
WRN-18 [53]	25.88	5.54
WRN-18 + SENet [19]	26.07	5.13
WRN-18 + CBAM [21]	26.08	5.06
WRN-18 + MAM	27.11	4.77

1 Models are trained on one RTX TITAN GPU

Table 9 Top-1 error comparison with plug-and-play modules on *CIFAR-100*

Model[1]	Params(M)	Top-1 Error(%)
ResNet-101 [1]	44.55	27.38
ResNet-101 + SENet [19]	49.33	24.62
ResNet-101 + CBAM [21]	49.33	24.56
ResNet-101 + MAM	50.07	23.19
WRN-18 [53]	25.88	22.81
WRN-18 + SENet [19]	26.07	20.66
WRN-18 + CBAM [21]	26.08	19.98
WRN-18 + MAM	27.11	19.11

1 Models are trained on one RTX TITAN GPU

Table 10 Implementation Details on *COCO 2017*

Optimizer	Initial LR	Min LR	Weight decay	Momentum	Batch size	Epochs
SGDM	1e–2	1e–4	5e–4	0.955	32	100

Table 11 Data Augmentation for Object Detection[1]

Method	Mosaic	Flip	Padding	ColorJitter	Distortion	Rotation
IsUsed	✓	✓	✓	✓	✓	✓

[1]The data augmentation methods are for *COCO 2017*

6.3.1 Implementation details

As Table 10 shows, we firstly resize the image as 512×512(px) and set the batch size to 32. We train for 100 epochs with SGDM as the optimizer, whose initial

Table 12 Comparison of *AP* on COCO 2017 test-dev

Model[1]	Backbone	AP(%)	AP_{50}(%)
CAT-YOLO	MANet-T	25.7	45.7
YOLOX-T [8]	Darknet-53-T	25.5	45.3
YOLOv4-T [6]	CSPDarknet-53-T	21.8	41.3
NanoDet-M [55]	ShuffleNetV2	21.7	40.2

[1]Models are trained on one RTX TITAN GPU

Table 13 Comparison of parameters and latency

Model	Params(M)	Latency(ms)[1]
NanoDet-M [55]	0.95	2.9
YOLOX-T [8]	5.06	7.7
YOLOv4-T [6]	6.06	7.9
CAT-YOLO	9.17	12.7

[1]We test and calculate the latency on one RTX TITAN without using tensorRT

learning rate is 0.01 and momentum value is 0.955. We set the weight decay as 5e-4 and use the cosine learning rate scheduler. The detail of data augmentation is shown in Table 11.

6.3.2 Comparison with the state-of-the-art

As a lightweight detection model, CAT-YOLO is compared with the tiny version of other models on *COCO 2017*. As Table 12 shows, CAT-YOLO got 25.7% AP, which is 0.2% higher than YOLOX-T and 3.9% higher than YOLOv4-T. CAT-YOLO has 45.7% AP_{50}, which is 0.4% higher than YOLOX-T and 4.4% higher than YOLOv4-T.

We also compare the parameter number and speed of these tiny models. The parameter number of CAT-YOLO is 3.11 million larger than YOLOv4-T and 4.11 million larger than YOLOX-T. It takes about 12.7 ms for CAT-YOLO to detect 416 × 416 images, which is 4.8ms longer than YOLOv4-T. CAT-YOLO is completely capable to detect objects in real time at the cost of slight more parameters and inference time than other tiny models (Table (13)).

6.3.3 Comparison in our obscured-object dataset

We collect 400 images consisting of obscured objects as a small dataset for evaluating obscured-object detection. Here, we use AR_{50} to evaluate the no-miss rate with a correct label and an IoU above 0.5. Eq.19 shows the recall as follows:

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

Table 14 Comparison of detecting obscured objects in our dataset of obscured objects

Model	$AR_{50}(\%)$[1]
NanoDet-M [55]	39.8
YOLOv4-T [6]	43.1
YOLOX-T [8]	46.6
CAT-YOLO	48.7

[1]The whole name of AR_{50} is the average recall with IoU > 0.5

where *TP* denotes the object detected with the correct classification label. *FN* denotes the missed object.

As Table 14 shows, CAT-YOLO outperforms among three models for obscured object detection, which gets the AR_{50} of 48.7%, 5.6% higher than YOLOv4-T and 2.1% higher than YOLOX-T.

6.3.4 Visualization of detection results

In addition to the test set of *COCO 2017*, we also test our model on our own test samples. As Figs. 12 and 13 show, CAT-YOLO's performance of detecting plain

Fig. 12 OOD test of obscured objects

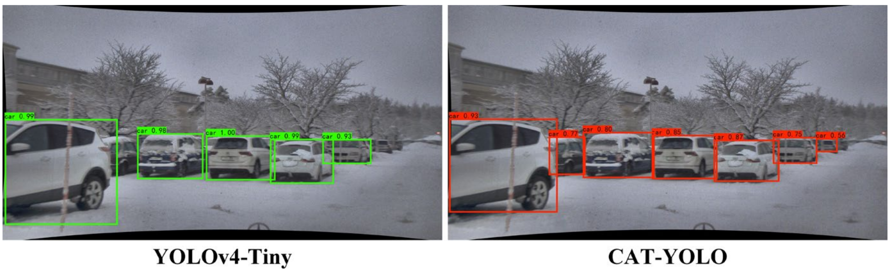

Fig. 13 OOD test of obscured, blurred and small objects

objects is as well as YOLOv4-Tiny. More importantly, CAT-YOLO performs better than YOLOv4-Tiny on detecting obscured, blurred and small objects. This shows that the mix attention layer (MAL) can enhance the feature representation of complex objects, and the mix attention module (MAM) can help neck to better aggregate features of different scales and efficiently detect objects to a certain extent.

7 Conclusions and future work

In this paper, we have proposed a plug-and-play layer MAL and a module MAM, which mix 3 different attention mechanisms(mix attention): spatial, channel and global contextual attention. In MAL and MAM, spatial–channel attention could focus on the absolute feature while global contextual attention could focus on relative feature. In addition, we propose a CNN-ViT mixture backbone based on mix attention called MANet. MANet enhances the feature representation and achieves state-of-the-art performances on several benchmarks. Finally, we research concatenating mix attention on YOLOv4-Tiny-based network. CAT-YOLO performs much better than YOLOv4-Tiny on *COCO 2017* and significantly improves the detection of obscured objects by means of adding several mix attention blocks. These also show that CNN can be highly compatible with ViT. Under the mode in mix attention, ViT can enhance the feature representation and accuracy of pure CNN, while CNN can greatly reduce the training and inference speed of ViT. These two complement each other. CAT-YOLO will provide engineers and developers with a new mobile-deployable object detection model. In the future, we will design some lower-cost plug-and-play modules from the perspective of explainable AI to improve the performance of CNN-based semantic segmentation and object tracking neural networks.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11227-022-04726-7.

Acknowledgements The authors acknowledge XJTLU-JITRI Academy of Industrial Technology for giving the valuable support of the joint project. The authors also acknowledge Mr. Wenhan Tao for the support of GPU resources.

Data availability The datasets used and/or analyzed during the current study are available from their official websites and corresponding author on a reasonable request.

References

1. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 770–778
2. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. preprint arXiv:1409.1556

3. Tan M, Le Q (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, PMLR, pp 6105–6114
4. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. Advances in neural information processing systems. 28
5. Redmon J, Farhadi A (2018) Yolov3: an incremental improvement. Preprint arXiv:1804.02767
6. Bochkovskiy A, Wang C-Y, Liao H-YM (2020) Yolov4: optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934
7. Jocher G (2021) YOLOv5. https://github.com/ultralytics/yolov5
8. Ge Z, Liu S, Wang F, Li Z, Sun J (2021) Yolox: exceeding yolo series in 2021. Preprint arXiv:2107.08430
9. Shao R, Shi Z, Yi J, Chen P-Y, Hsieh C-J (2021) On the adversarial robustness of visual transformers. arXiv e-prints, 2103
10. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, et al (2020) An image is worth 16x16 words: transformers for image recognition at scale. Preprint arXiv:2010.11929
11. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B (2021) Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 10012–10022
12. Chu X, Tian Z, Wang Y, Zhang B, Ren H, Wei X, Xia H, Shen,C (2021) Twins: revisiting the design of spatial attention in vision transformers. Advances in Neural Information Processing Systems, 34
13. Touvron H, Cord M, Douze M, Massa F, Sablayrolles, A, Jégou H (2021) Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning, PMLR, pp 10347–10357
14. Yuan K, Guo S, Liu Z, Zhou A, Yu F, Wu W (2021) Incorporating convolution designs into visual transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 579–588
15. Han K, Xiao A, Wu E, Guo J, Xu C, Wang Y (2021) Transformer in transformer. Advances in Neural Information Processing Systems, 34
16. Carion N, Massa F, Synnaeve G, Usunier N, Kirillov A, Zagoruyko S (2020) End-to-end object detection with transformers. In: European Conference on Computer Vision, Springer, pp 213–229
17. Fang Y, Liao B, Wang X, Fang J, Qi J, Wu R, Niu J, Liu W (2021) You only look at one sequence: rethinking transformer in vision through object detection. Advances in Neural Information Processing Systems, 34
18. Zhu X, Su W, Lu L, Li B, Wang X, Dai J (2020) Deformable detr: deformable transformers for end-to-end object detection. Preprint arXiv:2010.04159
19. Hu J, Shen L, Sun G (2018) Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 7132–7141
20. Li X, Wang W, Hu X, Yang J (2019) Selective kernel networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 510–519
21. Woo S, Park J, Lee J-Y, Kweon IS (2018) Cbam: Convolutional block attention module. In: Proceedings of the European Conference on Computer Vision (ECCV), pp 3–19
22. Ramchoun H, Ghanou Y, Ettaouil M, Janati Idrissi MA (2016) Multilayer perceptron: architecture optimization and training
23. Saeed F, Paul A, Rho S (2020) Faster r-cnn based fault detection in industrial images. In: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Springer, pp 280–287
24. Rathore MM, Paul A, Rho S, Khan M, Vimal S, Shah SA (2021) Smart traffic control: identifying driving-violations using fog devices with vehicular cameras in smart cities. Sustain Cities Soc 71:102986
25. Nawaz H, Maqsood M, Afzal S, Aadil F, Mehmood I, Rho S (2021) A deep feature-based real-time system for alzheimer disease stage detection. Multimed Tools Appl 80(28):35789–35807
26. Robinson YH, Vimal S, Julie EG, Lakshmi Narayanan K, Rho S (2021) 3-dimensional manifold and machine learning based localization algorithm for wireless sensor networks. Wireless Personal Communications, 1–19
27. Fan S, Wang R, Wu Z, Rho S, Liu S, Xiong J, Fu S, Jiang F (2021) High-speed tracking based on multi-cf filters and attention mechanism. SIViP 15(4):663–671

28. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 580–587
29. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1440–1448
30. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 779–788
31. Redmon J, Farhadi A (2017) Yolo9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 7263–7271
32. Liu S, Qi L, Qin H, Shi J, Jia J (2018) Path aggregation network for instance segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 8759–8768
33. Zhang H, Cisse M, Dauphin YN, Lopez-Paz D (2017) mixup: Beyond empirical risk minimization. Preprint arXiv:1710.09412
34. Yun S, Han D, Oh SJ, Chun S, Choe J, Yoo Y (2019) Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 6023–6032
35. Zheng Z, Wang P, Liu W, Li J, Ye R, Ren D (2020) Distance-iou loss: Faster and better learning for bounding box regression. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 34, pp 12993–13000
36. Müller R, Kornblith S, Hinton GE (2019) When does label smoothing help? Advances in neural information processing systems, 32
37. Tan Z, Wang J, Sun X, Lin M, Li H, *et al* (2021) Giraffedet: A heavy-neck paradigm for object detection. In: International Conference on Learning Representations
38. Dai J, Qi H, Xiong Y, Li Y, Zhang G, Hu H, Wei Y (2017) Deformable convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp 764–773
39. Paul S, Chen P-Y (2021) Vision transformers are robust learners. Preprint<error l="301" c="bad csname" />arXiv:2105.07581**2(3**)
40. Naseer MM, Ranasinghe K, Khan SH, Hayat M, Shahbaz Khan F, Yang M-H (2021) Intriguing properties of vision transformers. Advances in Neural Information Processing Systems, 34
41. Srinivas A, Lin T-Y, Parmar N, Shlens J, Abbeel P, Vaswani A (2021) Bottleneck transformers for visual recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 16519–16529
42. Chen Z, Xie L, Niu J, Liu X, Wei L, Tian Q (2021) Visformer: The vision-friendly transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 589–598
43. Wang Q, Wu B, Zhu P, Li P, Zuo W, Hu Q Supplementary material for ‘eca-net: Efficient channel attention for deep convolutional neural networks. Technical report
44. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser, Ł, Polosukhin I (2017) Attention is all you need. Advances in neural information processing systems, 30
45. Park N, Kim S (2022) How do vision transformers work? Preprint arXiv:2202.06709
46. Misra D (2019) Mish: A self regularized non-monotonic activation function. Preprint arXiv:1908.08681
47. Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 2117–2125
48. He K, Zhang X, Ren S, Sun J (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans Pattern Anal Mach Intell 37(9):1904–1916
49. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, pp 248–255
50. Krizhevsky A, Hinton G, et al (2009) Learning multiple layers of features from tiny images
51. Le Y, Yang X (2015) Tiny imagenet visual recognition challenge. CS 231N 7(7):3
52. Hassani A, Walton S, Shah N, Abuduweili A, Li J, Shi H (2021) Escaping the big data paradigm with compact transformers. Preprint arXiv:2104.05704
53. Zagoruyko S, Komodakis N (2016) Wide residual networks. Preprint arXiv:1605.07146
54. Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: Common objects in context. In: European Conference on Computer Vision, Springer, pp 740–755

55. RangiLyu (2021) NanoDet-Plus: Super fast and high accuracy lightweight anchor-free object detection model. https://github.com/RangiLyu/nanodet

Authors and Affiliations

Runwei Guan[1,2,3] · Ka Lok Man[2] · Haocheng Zhao[1,2,3] · Ruixiao Zhang[4] · Shanliang Yao[2] · Jeremy Smith[1] · Eng Gee Lim[2] · Yutao Yue[3]

Runwei Guan
Runwei.Guan@liverpool.ac.uk

Ka Lok Man
Ka.Man@xjtlu.edu.cn

Haocheng Zhao
Haocheng.Zhao@liverpool.ac.uk

Ruixiao Zhang
rz6u20@soton.ac.uk

Shanliang Yao
shanliang.yao19@student.xjtlu.edu.cn

Jeremy Smith
J.S.Smith@liverpool.ac.uk

Eng Gee Lim
enggee.lim@xjtlu.edu.cn

1 Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3BX, UK

2 School of Advanced Technology, Xi'an Jiaotong Liverpool University, Suzhou 215123, China

3 Institute of Deep Perception Technology, JITRI, Wuxi 214000, China

4 School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK