BA-Net: Bridge Attention for Deep Convolutional Neural Networks

Yue Zhao[1,2], Junzhou Chen[1,2]*, Zirui Zhang[1,2], and Ronghui Zhang[1,2]

[1] School of Intelligent Systems Engineering, Shenzhen Campus of Sun Yat-sen University, No. 66, Gongchang Road, Guangming District, Shenzhen, Guangdong 518107, P.R. China

[2] Guangdong Provincial Key Laboratory of Fire Science and Intelligent Emergency Technology, Sun Yat-sen University, Guangzhou, 510006, P.R. China

{zhaoy376, zhangzr23}@mail2.sysu.edu.cn

{chenjunzhou, zhangrh25}@mail.sysu.edu.cn

Abstract. In attention mechanism research, most existing methods are hard to utilize well the information of the neural network with high computing efficiency due to heavy feature compression in the attention layer. This paper proposes a simple and general approach named Bridge Attention to address this issue. As a new idea, BA-Net straightforwardly integrates features from previous layers and effectively promotes information interchange. Only simple strategies are employed for the model implementation, similar to the SENet. Moreover, after extensively investigating the effectiveness of different previous features, we discovered a simple and exciting insight that bridging all the convolution outputs inside each block with BN can obtain better attention to enhance the performance of neural networks. BA-Net is effective, stable, and easy to use. A comprehensive evaluation of computer vision tasks demonstrates that the proposed approach achieves better performance than the existing channel attention methods regarding accuracy and computing efficiency. The source code is available at `https://github.com/zhaoy376/Bridge-Attention`.

Keywords: Channel attention mechanism, Deep neural networks architecture, Networks optimization

1 Introduction

Deep convolutional neural networks (CNNs) are widely used in the computer vision community [31], showing excellent performance on various tasks, e.g., image classification, object detection, instance segmentation, and semantic segmentation. Since the appearance of AlexNet [14], numerous researches have dedicated to boosting the performance of CNNs [7, 12, 24, 27].

In recent years, the attention mechanism has attracted much attention as a novel technique to enhance performance. It learns attention weights from the adjacent convolution layer, thus concentrating on more important features. The

∗ Corresponding author.

channel attention mechanism is one of the attention mechanisms with the most representative method, such as squeeze-and-excitation networks (SENet) [11], which learns the channel attention from an average pooled output on each map, bringing considerable performance gain in various CNNs. Fig.1 shows the block architecture of most attention methods, which consists of stacked convolution layers and one attention layer.

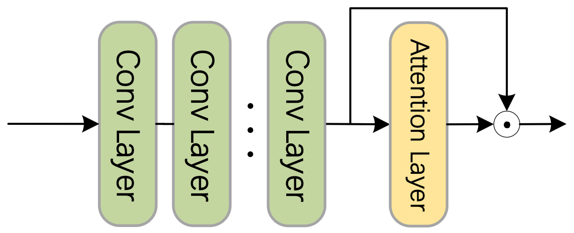

Fig. 1. The block structure of most existing attention methods. The output, coming from stacked convolution layers, passes through the attention layer,thus the attention weights is obtained.

In order to obtain better attention, many methods like [1, 4] attempt to use sophisticated strategies on the adjacent convolution output while bringing high model complexity and computation cost. Essentially, attention generation is a process of heavy feature compression, in which the convolution output converts to a vector or a map for channel or spatial attention, respectively. So if we want to get better attention only from a single convolution output, the computational efficiency has to be sacrificed.

To address this issue, we provide new thinking that utilizes previous features. From the view of information transfer, the attention also has an implicit correlation to the previous convolution layers since convolution layers are stacked. Thus bridging previous features can straightforwardly supplement valuable information for the attention. Moreover, the pieces of information from different layers are well interchanged in the attention layer. In this way, we propose the Bridge Attention Net(BA-Net), in which previous features are integrated into the attention layer via simple strategies similar to the SENet. As a result, its model complexity regarding the number of parameters, computation cost, and inference speed is comparable to the SENet. Experimental evaluations on image classification, object detection, and instance segmentation show that BA-Net can perform better than existing channel attention methods.

The major contributions of this article can be summarized as follows:

- We analyze the limitation of traditional channel attention mechanisms and empirically demonstrate that bridging previous features can straightforwardly supplement efficient information from the view of information transfer.
- We propose the Bridge Attention Net (BA-Net) and implement the basic module using only simple strategies. Moreover, we extensively investigate

the effectiveness of different previous features and give out the best BA module structure.

– Comprehensive experiments on various computer vision tasks are carried out and show that the proposed BA-Net can achieve higher performance with low model complexity and fast speed compared with existing channel attention methods.

2 Related Work

We mainly revisit attention mechanisms and cross-layer interaction applied in Convolutional Neural Networks(CNNs) in the existing literature.

Attention mechanisms. The attention mechanism is capable of enhancing the assignment of the most informative feature representations while suppressing the less useful ones, thus allowing the model to focus on the important regions in the context adaptively. The pioneering SENet [11] is the cornerstone of the attention mechanism research field. The method extracted channel-wise features by simple global average pooling and full connection layers, significantly improving the performance of many CNNs with few parameters and computing costs added. The SKNet [16] enhances the expressiveness of the model by passing the feature map through two convolution layers of different kernel sizes, followed by the extraction of channel attention. While extracting channel attention, BAM [19] and CBAM [29] utilize the spatial information and generate spatial attention using convolution. DA-Net Attention [3] concentrates on the relevance of local and global features and combines the two features by summing the attention modules of two branches. ResNeSt [32] adopts a similar split-attention block, which enables the fusion of attention between different groups of the input feature maps. GSoPNet proposes the global second-order pooling to introduce higher-order representation to improve the non-linear capability of CNNs. GSoPNet [4] obtains attention by fully using the second-order statistics of the holistic image. Fca-Net [20] revisits channel attention using frequency analysis and generalizes the pre-processing of channel attention mechanism in the frequency domain. Some methods explore lightweight strategies to reduce the parameters and computing cost of the model with attention. ECA-Net Attention [26] proposes local cross-channel interaction and generates attention by 1D convolution. SA-Net is also a lightweight attention structure inspired by channel shuffling. Half of the features are used to generate spatial attention in SA-Net, and the other half is used to generate channel attention. At the end of the block, features are shuffled along the channel. The above mechanisms provide many novel ways to generate channel or spatial attention. However, one thing in common among them is that they only focus on the features of the layer adjacent to the attention layer. The features in previous layers are ignored. In this paper, we will explore the effect of features in previous layers on the attention mechanism.

Cross-layer integration. It is a common strategy to improve network representation by skip connection, which can solve the problem of gradient dispersion and the disappearance of deep networks to some extent. This strategy can train deep networks more adequately, making deeper network structures feasible, and has been widely used in the design of neural network models. The ResNet [7]network first proposed a residual module, which facilitates the fusion of information between different layers. DenseNet [12] also uses a similar structure but differs from ResNet [7]in the form of concatenating for feature stitching. In U-Net [22], which is commonly used in the field of medical segmentation, the decoder-encoder module is connected through a skip connection to make feature extraction achieve higher accuracy.

Actually, cross-layer integration has been used to improve the performance of attention mechanisms. Duo L. et al. [15] propose the DREAL method to optimize parameters of arbitrary attention modules, in which LSTM [9] is used to integrate previous attention weights, and deep reinforce learning is used to update parameters of LSTM and attention layers. DIANet [13] also utilizes LSTM module to integrate previous attention weights and directly outputs attention weights in current block by LSTM. DIANet visualizes the effect of previous features acting on the current attention layer and shows the effect on stabilizing Training. Yu. W. et al. [28] propose the evolving attention to improve the performance of transformers, named EA-AANet. Attention maps in a preceding block are integrated with ones in the current layer by residual connection and 2D convolution. Compared to these works, the proposed BA-Net in this paper has a similar motivation, but this approach integrates the features in previous layers of the current block. The higher performance of our models is displayed in Table.2 demonstrates that feature integration of our method is more effective.

3 Approach

In this section, we first revisit traditional channel attention mechanisms(i.e., SENet [11]) and give out the common form of the mechanisms. Then we demonstrate the limitation of the mechanisms through empirical analysis. It inspires us to come up with the Bridge Attention mechanism, and we will concretely introduce the implementation of the proposed module.

3.1 Traditional channel attention mechanisms

Revisit SENet. Let the output of the SE block be $X \in \mathbb{R}^{C \times H \times W}$, where C, H and W are channel, height and width dimension of the output. Accordingly, the generated attention weights can be computed as:

$$\omega = \sigma(\mathcal{F}_{\mathcal{C}}(gap(X))) \tag{1}$$

where $gap(X) = \frac{1}{HW}\sum_{i=1,j=1}^{H,W} X_{i,j}$ is channel-wise global average pooling, $\sigma(\cdot)$ represents Sigmoid function. $\mathcal{F}_{\mathcal{C}}(\cdot)$ represents two stacked Full Connection(FC) layers, which can be expressed as followed:

$$\mathcal{F}_{\mathcal{C}}(y) = (\boldsymbol{W_2})ReLU(\boldsymbol{W_1}y) \tag{2}$$

In Eqn.2, ReLU [18] represents an activation function. $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$ are matrix used to form the attention weights. The two matrix respectively have size of $C \times (\frac{C}{r})$ and $(\frac{C}{r}) \times C$, in which the reduction factor r is used to avoid heavy computation and high complexity of the attention layer.

We consider that attention mechanism can be divide into two parts, **Integration** and **Generation**. In SENet, the output X is first squeezed by average pooling $gap(\cdot)$ and fully integrated among channels by matrix $\boldsymbol{W_1}$, considered as **Integration** $\mathcal{I}(\cdot)$. And then the features are sequentially fed into $RELU$, matrix $\boldsymbol{W_2}$, $\sigma(\cdot)$, to get the final attention weights, considered as **Generation** $\mathcal{G}(\cdot)$. Thus the common form of attention mechanism can be expressed as :

$$\mathcal{I}(\cdot) = \boldsymbol{W_1}(gap(\cdot)),\ \mathcal{G}(\cdot) = \sigma(\boldsymbol{W_2}(ReLU(\cdot))) \tag{3}$$

$$\omega = \mathcal{G}(\mathcal{I}(X)) \tag{4}$$

In our method, richer features of previous layers are bridged and integrated into $\mathcal{I}(\cdot)$ beside features of the adjacent layer. Thus better attention is generated.

Limitation. Fig.1 shows the block architecture of most existing attention methods, which includes the convolution part and an attention layer. Let $\mathcal{F}(\cdot)$ and $att(\cdot)$ represent the convolution part and the attention layer respectively, thus the whole process can be expressed as:

$$\mathcal{F}_{att}(\cdot) = \mathcal{F}(\cdot) \odot att(\mathcal{F}(\cdot)) \tag{5}$$

$\odot$ represents the element-wise multiplication.

Generally, the convolution part consists of several stacked convolution layers. We assume that the total number of convolution layers is n, thus:

$$\mathcal{F}(\cdot) = F_n(F_{n-1}(\cdots F_2(F_1))) \tag{6}$$

$F_i(\cdot)$ represents the certain convolution layer, where $1 \leq i \leq n$.

Considering the distance, we assume that the outputs of $\mathcal{F}(\cdot)$ are more implicitly correlated with the previous q layers, thus Eqn.6 can be approximately equal to:

$$\mathcal{F}(\cdot) \approx F_n(F_{n-1}(\cdots F_{n-(q-1)}(F_{n-q}))) \Rightarrow \mathcal{F}(F_n, F_{n-1}, \ldots, F_{n-(q-1)}, F_{n-q}) \tag{7}$$

In most existing attention methods, the attention layer only extracts features from the adjacent layer $F_n(\cdot)$. Some methods even have complicated calculations in $att(\cdot)$ for richer information, which weakens the correlation with the previous q convolution layers:

$$att(\cdot) = att(F_n(F_{n-1}(\cdots F_{n-(q-1)}(F_{n-q})))) \approx att(F_n) \tag{8}$$

According to Eqn.8, the generated attention weights lack correlation with previous layers, resulting in insufficient adaptive to outputs of $\mathcal{F}(\cdot)$.

In fact, [13] [28] have noticed the above issue, but only the attention weights of previous blocks are fed into the current attention layer. In our method, features of the previous layers inside the block are bridged to the current attention layer.

3.2 Implementation of the BA module

In this part, we concretely introduce how previous features are integrated into our method and give out the implementation of the Bridge Attention module. Let the output of $F_i(\cdot)$ inside the block be $X_i \in \mathbb{R}^{C_i \times H \times W}$. The outputs are first global average pooled (gap) to the size of $C_i \times 1 \times 1$, and then fed into respective matrices of size $C_i \times (\frac{C_n}{r})$ to get the squeezed features S_i. The size $\frac{C_n}{r} \times 1 \times 1$ is the same as the squeezed feature from the output of $F_n(\cdot)$, which is followed by $att(\cdot)$. Thus the squeezed features from different layers are directly added, and the final integrated feature is obtained.

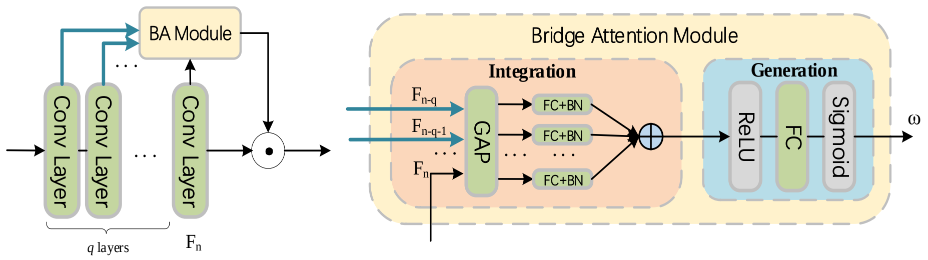

Fig. 2. The overview of Bridge Attention module. Blue arrows indicate that features from previous q layers are bridged to the attention layer. **FC** indicates the matrix to squeeze features and **BN** is BatchNorm Layer.

We notice that distributions among the squeezed features can be massive differences due to the squeezing process, so we apply Batch Normalization for the features to make them in similar distributions, thus the integration can be more effective. In addition, Batch Normalization improves the nonlinear representation of the features, which also benefits network parameters updating. As a whole, the **Integration** part can be expressed as:

$$S_i = BN_i(\boldsymbol{W_i}(gap(F_i))) \tag{9}$$

$$\mathcal{I}_{BA}(\cdot) = \sum_{n-q}^{n} S_i \tag{10}$$

Then we input the integrated feature into **Generation** part and get the final attention weights. The overview of Bridge Attention module is shown in Fig.2.

3.3 The best structure of the BA module

Since there are various previous features that can be bridged to the attention layer, we investigate their effectiveness to explore the best structure of the BA module. We evaluate the image classification task based on the backbone of ResNet-50, and the result is shown in Table.1. The row$_3$ achieves higher accuracy than the row_2, although they are at the same position. Because the attention weights are used to rescale the feature maps, compared with the convolution outputs, they contribute less helpful information to the attention layer. Besides, bridging the proximate features can achieve better performance. Due to the heavy feature compression in the attention layer, more proximate features can straightforwardly supplement more relevant information.

Table 1. The comparison when bridging different features. *att*: the attention weights, *conv*: the convolution outputs, *prev*: the previous block, *curr*: the current block, $conv_i$: the i-th convolution layer, *end*: the end of the block.

Backbone	Type	Position	Param.	TOP-1(%)
SE ResNet-50	*None*	—	28.07M	78.14
BA ResNet-50	*att*	$prev, conv_3$	29.16M	78.41
	conv	$prev, conv_3$	29.16M	78.49
	conv	$prev, end$	29.16M	78.54
	conv	$curr, conv_1$	28.39M	78.78
	conv	$curr, conv_2$	28.39M	78.77
	conv	$curr, conv_{1\&2}$	28.71M	**78.85**

In conclusion, bridging the closer convolution outputs can achieve better performance. So we merely consider the features within the block, avoiding a significant increase in configuration complexity. Generally, the blocks of existing CNNs contain no more than three convolution layers, such as ResNet [7], ResNext [30], MoblileNetv3 [10], EffcientNet [25], so we bridge all convolution outputs before the attention layer.

For example, the block of the ResNet-50 contains three convolution layers, where the number of channels in each output is C, C, and $4C$. The BA layer follows the last layer. Each output is processed by GAP and converted to a vector of size of $\frac{4C}{r}$ by FC, where r is always set to 16. Then, the vectors will be batch normalized and summed. Another FC converts the sum to $4C$ vector. The total number of neurons in a BA layer is $(C + C + 4C) \times \frac{4C}{r} + 4C \times \frac{4C}{r}$, besides BN.

4 Experiments

In this section, we evaluate our method on three computer vision tasks, including image classification, object detection, and instance segmentation. We first demonstrate the implementation details of the experiments. Then, we give out the performance comparison of our method with other attention methods.

4.1 Implementation details

For image classification, we evaluate the performance on ImageNet-1K [23] dataset, where we apply our method on various backbone architectures, including ResNet

[7], MobileNet-v3, EfficientNet and ResNeXt. We take the same strategy of the data augmentation and hyperparameter settings in [7] and [8]. The training images are cropped randomly to 224×224 with random horizontal flipping, while the testing images are resized to 256×256 and cropped from the center to 224×224. We use an SGD optimizer with a momentum of 0.9 and a weight decay of 1e-4. In the training phase, the initial learning rate is set to 0.1 for a batch size of 256. All models are trained within 100 epochs with cosine learning rate decay following FcaNet [20].

For object detection and instance segmentation, we evaluate our method on the MS COCO2017 dataset [17]. Faster R-CNN [21] and Mask R-CNN [6] are used as detectors while BA-Net-50 & 101 pretrained on ImageNet-1K are used as backbone. We used MMDetection toolkit [2] to implement all detectors and follow the default settings. The shorter side of input images is resized to 800. All models are optimized using SGD with weight decay of 1e-4, the momentum of 0.9, and batch size set to 8. The total number of training epochs is 12, and the initial learning rate is 0.01, decreased by a factor of 10 at the 8th and 11th epoch, respectively. We construct all models based on the PyTorch framework and experiment on four Nvidia RTX 3090Ti GPUs.

4.2 Image Classification on ImageNet-1K

Performance comparison with other methods. Firstly, we evaluate our method under the backbones of ResNet-50&101, which are the most common backbones used to apply attention mechanisms. Besides traditional channel mechanisms like SENet [11], ECA-Net [26], FcaNet [20], we also compare the performance with the methods using cross-layer integration, like DREAL [15], DIANet [13] and EA-AANet [28]. We give out the metrics from their origin papers. In addition, we noticed that different training settings are used in different mechanisms in their origin papers, so we retrained part of attention models that is reproducible, following the setting of FcaNet [20]. Observed in Table.2, BA-Net has higher performance than other attention mechanisms in $org.$ metrics, specifically BA-Net, significantly outperforms SENet by 2.14% and 1.41% in $org.$ TOP-1 under the backbones of ResNet-50 and ResNet-101, respectively. Under the same training setting, our method also performs better than SENet, ECA-Net, and FcaNet. Specifically, BA-Net outperforms SENet by 0.71% and 0.62% in $self.$ TOP-1 under the two backbones, respectively.

Computing cost. Observed in Table.2, parameters of BA-Net are slightly larger than parameters of SENet since the features of previous layers are bridged to the attention layer, while FLOPs of them are almost the same. To further illustrate the computation cost of BA-Net, we make additional comparisons on graphics memory usage and speed when training and testing. In Table.3, memory usage when training and testing are slightly increased while training speed and testing speed are slightly decreased. With comparable computing costs, BA-Net outperforms SENet by 0.71% and 0.62% under the two backbones.

Table 2. Performance comparisons of different attention methods on ImageNet-1K in terms of network parameters(Param.), floating point operations per second (FLOPs), and Top-1/Top-5 accuracy. The term *self.* means that the metrics came from the experiments retrained by ourselves while *org.* means that the metrics came from the original paper. Our method and the best records are marked in **bold**.

Attention Method	Backbone	Param.	FLOPs	TOP-1(%)		TOP-5(%)	
				self.	*org.*	*self.*	*org.*
SENet	ResNet-50	28.07M	4.13G	78.14	76.71	94.05	93.38
ECA-Net		25.56M	4.13G	77.98	77.43	93.94	93.65
FcaNet		28.07M	4.13G	78.57	78.52	94.16	94.14
SENet+DREAL		28.12M	4.13G	—	77.85	—	94.05
DIANet		28.36M	4.13G	78.31	77.24	94.08	—
EA-AANet		25.80M	4.35G	—	78.22	—	94.21
BA-Net(ours)		28.71M	4.13G	**78.85**		**94.28**	
SENet	ResNet-101	49.29M	7.86G	79.41	77.62	94.62	93.93
ECA-Net		44.55M	7.87G	79.23	78.65	94.45	94.34
FcaNet		49.29M	7.86G	79.63	79.64	94.66	94.63
SENet+DREAL		49.36M	7.87G	—	79.27	—	94.59
DIANet		47.35M	7.86G	79.47	—	94.66	—
EA-AANet		45.40M	8.60G	—	79.29	—	94.81
BA-Net(ours)		50.49M	7.87G	**80.03**		**94.83**	

Table 3. Computing cost comparisons of BA-Net and SENet in temrs of memory usage and speed(frame per second, FPS) when train and test. M. represents memory usage and S. represents speed.

Method	Backbone	Train M.	Train S.	Test M.	Test S.	TOP-1(%)
SENet	ResNet-50	34.74G	656 FPS	7.49G	1315 FPS	78.14
BA-Net(Ours)		34.85G	612 FPS	7.67G	1280 FPS	**78.85**
SENet	ResNet-101	46.86G	397 FPS	8.96G	845 FPS	79.41
BA-Net(Ours)		47.54G	362 FPS	9.26G	792 FPS	**80.03**

Table 4. Performance comparisons of BA-Net application on different backbone architectures. The backbones with Bridge Attention and the best records are marked in **bold**.

BackBone	Type	Param.	TOP-1(%)	TOP-5(%)
ResNeXt-50	Origin	25.05M	78.77	94.18
	+BA	28.85M	**79.58**	**94.69**
EfficientNet-b0	Origin	5.29M	70.11	89.45
	+BA	8.42M	**71.70**	**90.21**
MobileNetv3-small	Origin	2.54M	**65.87**	86.26
	+BA	2.79M	65.85	**86.48**
MobileNetv3-large	Origin	5.48M	73.29	91.18
	+BA	6.03M	**73.50**	**91.22**

Application on other backbones. To further verify the capability of Bridge Attention on other backbone architectures, we apply our method on ResNext, EfficientNet, and MobileNetv3. We retrained all the original backbones while the types with Bridge Attention added, comparisons are shown in Table.4. For ResNeXt-50, BA improves the model by 1.19% on TOP-1 and 0.51% on TOP-5. EfficientNet-b0 with BA significantly outperforms the origin model by 1.59% and 0.76% on TOP-1 and TOP-5, respectively. However, Bridge Attention seems to have little effect on light-weight backbone like MoblileNetv3, with only a 0.21% improvement on TOP-1 of the large type. So we consider that BA can help improve performance more significantly on heavy backbone architectures.

4.3 Object detection on COCO2017

In this subsection, we evaluate our BA-Net on object detection tasks using Fast R-CNN and Mask R-CNN. We mainly compare BA-Net with ResNet, SENet, ECA-Net, and FcaNet. We transferred our BA-Net models on the COCO2017 training set and gave out the metrics tested on the validation set. As shown in Table.5, most metrics of BA-Net achieve the highest performance. For Fast R-CNN, our BA-Net outperforms SENet by 1.8% and 2.1% in terms of mAP with the backbones ResNet-50 and ResNet-101, respectively. For Mask R-CNN, our BA-Net outperforms SENet by 2.1% in terms of mAP with ResNet-50.

Table 5. Performance comparisons of different attention methods on obeject detection task. Average Precision(AP) is the main comparison metric.

Backbone	Detector	Param.	FLOPs	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50	Faster-RCNN	41.53M	215.51G	36.4	58.2	39.2	21.8	40.0	46.2
SENet		44.02M	215.63G	37.7	60.1	40.9	22.9	41.9	48.2
ECA-Net		41.53M	215.63G	38.0	60.6	40.9	23.4	42.1	48.0
FcaNet		44.02M	215.63G	39.0	61.1	42.3	23.7	42.8	49.6
BA-Net		44.66M	215.68G	**39.5**	**61.3**	**43.0**	**24.5**	**43.2**	**50.6**
ResNet-101		60.52M	295.39G	38.7	60.6	41.9	22.7	43.2	50.4
SENet		65.24M	295.58G	39.6	62.0	43.1	23.7	44.0	51.4
ECA-Net		60.52M	295.58G	40.3	62.9	44.0	24.5	44.7	51.3
FcaNet		65.24M	295.58G	41.2	63.3	44.6	23.8	45.2	53.1
BA-Net		66.44M	295.70G	**41.7**	**63.4**	**45.1**	**24.9**	**45.8**	**54.0**
ResNet-50	Mask-RCNN	44.17M	261.81G	37.2	58.9	40.3	22.2	40.7	48.0
SENet		46.66M	261.93G	38.4	60.9	42.1	23.4	42.7	50.0
ECA-Net		44.17M	261.93G	39.0	61.3	42.1	24.2	42.8	49.9
FcaNet		46.66M	261.93G	40.3	**62.0**	44.1	**25.2**	43.9	52.0
BA-Net		47.30M	261.98G	**40.5**	61.7	**44.2**	24.5	**44.3**	**52.1**

4.4 Instance segmentation on COCO2017

For instance segmentation task, we take Mask R-CNN as the detector for evaluation and the result is shown in Table.6. mAP of BA-Net achieved 36.6% and

38.1% under the two backbones, which performs better than other attention methods. Compared with SENet, our BA-Net notably outperformed by 1.2% and 1.3% in terms of mAP, respectively. Besides image classification, BA-Net also performs well on object detection and instance segmentation tasks, which verifies that our BA-Net has good generalization ability for various tasks.

Table 6. Performance comparisons of different attention methods on instance segmentation task.

Backbone	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50	34.2	55.9	36.2	16.1	37.5	46.3
SENet	35.4	57.4	37.8	17.1	38.6	51.8
ECA-Net	35.6	58.1	37.7	17.6	39.0	51.8
FcaNet	36.2	58.6	38.1	—	—	—
BA-Net	**36.6**	**58.7**	**38.6**	**18.2**	**39.6**	**52.3**
ResNet-101	35.9	57.7	38.4	16.8	39.7	49.7
SENet	36.8	59.3	39.2	17.2	40.3	53.6
ECA-Net	37.4	59.9	39.8	18.1	41.1	54.1
BA-Net	**38.1**	**60.6**	**40.4**	**18.7**	**41.5**	**54.8**

5 Analysis

5.1 Effectiveness of Bridge Attention

To further analyze how Bridge Attention affects the feature map, we visualize the attention weights distribution of BA-Net and compare it with SENet. Concretely, we randomly sample four classes from ImageNet, which are American chameleon, castle, paintbrush, and a prayer mat, respectively. All images of each class are collected from the validation set of ImageNet, and some example images are shown in Figure 3.

Fig. 3. Example images of four classes from ImageNet. The images from left to right are American chameleon, castle, paintbrush and prayer mat, respectively.

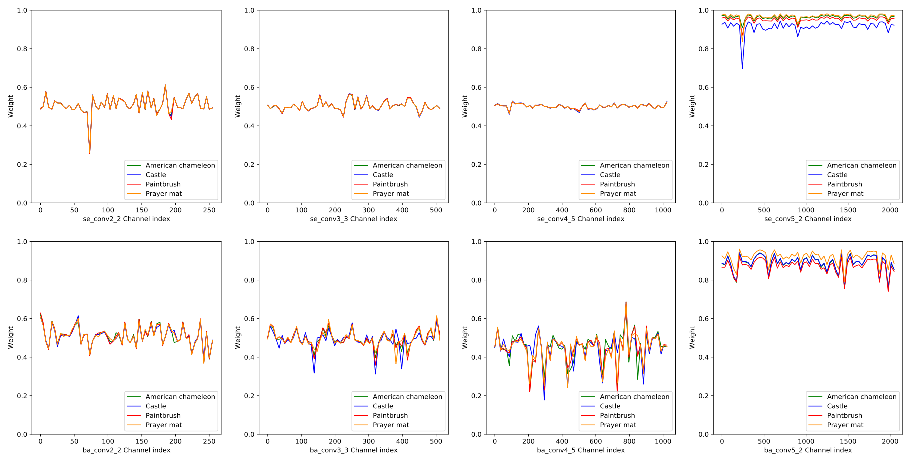

Fig. 4. Visualization of channel attention weights of Block-i_j, where i indicate the i-th stage and j is j-th block in i-th stage. The weights learned by SENet50 blocks and BA-Net50 are illustrated in top and bottom row, respectively.

We put the images of the same class into the pretrained BA-Net-50 and SENet-50, then compute the channel attention weights of convolution blocks on average. Figure 4 visualizes the attention weights of four blocks, and each is the last block of four stages. The attention weights of SE blocks are illustrated in the top row, while BA blocks' are illustrated in the bottom row.

In the first stage, the weights distributions of both models are similar, showing that enhancement from Bridge Attention is not significant for coarse feature extraction. Nevertheless, in the later stages for detailed feature extraction, the variance of the weight distribution in BA-Net increases significantly, indicating that the weights become more diverse, especially in the third stage. It demonstrates that BA-Net can effectively capture the more essential features while filtering out the less important ones, thus enhancing the representation of the feature maps. In addition, the SENet's weights curves of different classes in the first three stages almost overlap, while the BA-Net's curves of different classes are clearly distinguishable on some channels. This indicates that BA-Net can distinguish detailed features of different classes sharply. In general, the bridged features of previous convolution layers effectively enhance the representation ability of the output feature maps.

5.2 Importance of the integrated features

In our method, the features of different convolution layers in the block are integrated into the attention layer, so we want to explore the relationship between the integrated features and the attention weights and how the features contribute to the attention weights. We consider using the random forest model to reveal the relationship and take the Gini importance [5] from the model as the measurement of feature importance.

The calculating process is referred to as Algorithm 1. There are 16 blocks in the BA-Net-50, where each block contains three convolution layers, and the attention layer follows the third layer. We also use the validation set of ImageNet to inference in the model, and then we get the squeezed features S^i and attention weights ω^i in each block. We fit a random forest model for each block using its S^i and ω^i, and then visualize the feature importance as shown in Fig.5. We notice that not all S_3 contribute the most to the attention weights, which are adjacent to the attention layer. For example, the contribution of S_2 is comparable to S_3 in B12, or even the integrated features from previous convolution layers are more than important S_3, such as B11, B16. The results demonstrate that the features from previous convolution layers also effectively contribute to the attention weight and even play a dominant role among the integrated features in a particular block.

Algorithm 1: Calculate importance of the squeezed features

```
Data:
$S^i$: consisting of $S_1^i, S_2^i, S_3^i$, indicating all squeezed features in block $i$;
#The size of $S^i$ is $(b \times c^i \times 3)$, $b$: training samples, $c^i$: channels of $S^i$
$\omega^i$: Channel attention weights in block $i$;
$N$: the number of blocks in BA-Net50 backbone;
Result:
The hotmap $G$ about importance of squeezed features in block $i$;
initialization $G = \emptyset$;
for $i = 1$ to $N$ do
    $x \longleftarrow [S_1^i, S_2^i, S_3^i]$;
    $x \longleftarrow x$.reshape$(b, (c^i \times 3))$;
    $y \longleftarrow \omega^i$;
    Model $\longleftarrow$ RandomForestRegressor();
    Model.fit$(x, y)$
    Importances $\longleftarrow$ Model.feature_importances_ ;
    # The length of Importances is $(c^i \times 3)$ ;
    $res = \emptyset; s = 0; cnt = 0$ ;
    for $k = 0$ to $(c^i \times 3)$ do
        $s \longleftarrow s +$ Importances$(k)$;
        $cnt \longleftarrow cnt + 1$;
        if $cnt = c^i - 1$ then
            res.add$(s)$;
            $s \longleftarrow 0$;
            $cnt \longleftarrow 0$;
        end
    end
    $G$.add(res/max(res));
end
```

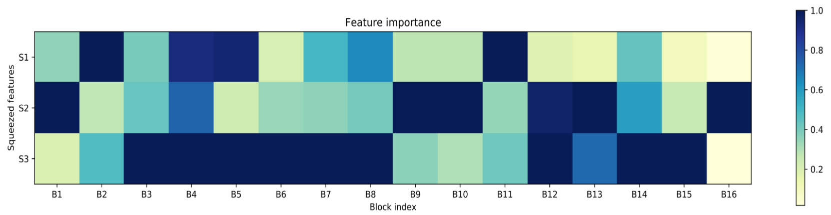

Fig. 5. Visualization of Gini importance of the integrated features. Bi indicates the i-th block in BA-Net50 model and Si is i-th squeezed feature in a certain block. The deeper the color of the square, the more important it is.

6 Conclusion

Traditional channel attention mechanisms heavily rely on the output of the adjacent convolution layer. Faced with this limitation, this paper proposes a novel idea named Bridge Attention to enrich the information for better channel weight estimation. We design the Bridge Attention Module with simple strategies by integrating the features from multiple layers. Experimental evaluation shows that the BA-Net achieves significant performance on various computer vision tasks. Moreover, we verify the features from multiple layers also effectively contribute to the attention weights. In future work, we will consider extending the Bridge Attention by exploring feature integration from the previous block, thus further improving the neural network's performance.

7 Acknowledgement

This work was partially supported by the Shenzhen Fundamental Research Program (No. JCYJ20200109142217397), Guangdong Natural Science Foundation (No. 2021A1515011794, and 2021B1515120032), Shenzhen Key Science and Technology Program (No. JSGG20210802153412036), and National Natural Science Foundation of China (No.52172350).

References

1. Bello, I., Zoph, B., Vaswani, A., Shlens, J., Le, Q.V.: Attention augmented convolutional networks. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 3286–3295 (2019)
2. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., et al.: Mmdetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019)
3. Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., Lu, H.: Dual attention network for scene segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3146–3154 (2019)

4. Gao, Z., Xie, J., Wang, Q., Li, P.: Global second-order pooling convolutional networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3024–3033 (2019)
5. Gregorutti, B., Michel, B., Saint-Pierre, P.: Correlation and variable importance in random forests. Statistics and Computing **27**(3), 659–678 (2017)
6. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
8. He, T., Zhang, Z., Zhang, H., Zhang, Z., Xie, J., Li, M.: Bag of tricks for image classification with convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 558–567 (2019)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
10. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al.: Searching for mobilenetv3. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1314–1324 (2019)
11. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7132–7141 (2018)
12. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)
13. Huang, Z., Liang, S., Liang, M., Yang, H.: Dianet: Dense-and-implicit attention network. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 4206–4214 (2020)
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25**, 1097–1105 (2012)
15. Li, D., Chen, Q.: Deep reinforced attention learning for quality-aware visual recognition. In: European Conference on Computer Vision. pp. 493–509. Springer (2020)
16. Li, X., Wang, W., Hu, X., Yang, J.: Selective kernel networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 510–519 (2019)
17. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
18. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Icml (2010)
19. Park, J., Woo, S., Lee, J.Y., Kweon, I.S.: Bam: Bottleneck attention module. arXiv preprint arXiv:1807.06514 (2018)
20. Qin, Z., Zhang, P., Wu, F., Li, X.: Fcanet: Frequency channel attention networks. arXiv preprint arXiv:2012.11879 (2020)
21. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. IEEE transactions on pattern analysis and machine intelligence **39**(6), 1137–1149 (2016)
22. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)

23. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
25. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
26. Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., Hu, Q.: Eca-net: efficient channel attention for deep convolutional neural networks, 2020 ieee. In: CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (2020)
27. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7794–7803 (2018)
28. Wang, Y., Yang, Y., Bai, J., Zhang, M., Bai, J., Yu, J., Zhang, C., Huang, G., Tong, Y.: Evolving attention with residual convolutions. arXiv preprint arXiv:2102.12895 (2021)
29. Woo, S., Park, J., Lee, J.Y., Kweon, I.S.: Cbam: Convolutional block attention module. In: Proceedings of the European conference on computer vision (ECCV). pp. 3–19 (2018)
30. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1492–1500 (2017)
31. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. In: International conference on machine learning. pp. 2048–2057. PMLR (2015)
32. Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J., Manmatha, R., et al.: Resnest: Split-attention networks. arXiv preprint arXiv:2004.08955 (2020)