A Novel Block-wise Attention Method: Focus-and-Association Networks

Deyang Liu, Guoan Yang*, Zhengzhi Lu, Yong Yang, Chuanbo Zhou
School of Automation Science and Engineering
Xi'an Jiaotong University
Xi'an, Shaanxi 710049, China
yohn08@qq.com, gayang@mail.xjtu.edu.cn, lu947867114@stu.xjtu.edu.cn, 294575885@qq.com, 332290570@qq.com
* Guoan Yang: gayang@mail.xjtu.edu.cn

***Abstract*—Deep residual networks have the problem of diminishing feature reuse, meaning that as the network depth increases, the accuracy does not increase linearly but gradually becomes smooth. Recent work has demonstrated that in addition to increasing the depth, the performance of networks can also be improved by explicitly embedding learning mechanisms. A broad range of prior research has investigated channel and spatial attention mechanisms to strengthen the representational power of convolutional neural networks (CNNs). In this paper, we focus instead on the block relationship. Inspired by the human visual attention mechanism, we propose a novel block-wise attention module, which we term the "focus-and-association" (FA) module, that performs dynamic block-wise feature recalibration by explicitly modelling interdependencies between blocks. We show that FA module can be applied to various existing state-of-the-art CNNs with few additional parameters and slight computational burden, and it effectively generalizes across different datasets. Extensive experiments show the effectiveness of the focus-and-association network (FANet). In addition, we further demonstrate the compatibility of the FA module by combining it with the existing state-of-the-art embedded algorithm unit, and achieve further improvements in accuracy. Finally, we visualize part of the details from the network to further explore the mechanism of action of the FA module.**

Key words: focus-and-association; block-wise feature recalibration; attention mechanism; convolutional neural networks

I. INTRODUCTION

Deep learning algorithms based on convolutional neural networks (CNNs) have made great progress in image classification tasks[1][2][3]. A CNN is essentially an input-to-output projection that learns the mapping relationship from a large amount of data without precise mathematical expressions[4][5], meaning that CNN extracts feature representations from the image in an implicit way. With the increasing of the network depth, the extracted features continuously become more abstract, and finally they are transformed into semantic information. It has been proved in a large number of experiments that a deeper network is the key to success. In[6], He et al. proposed a "residual learning"-based structure, which greatly inhibits the "degradation" of the network, making it feasible to stack thousands of layers in the networks. Although the residual structure can extend the network to an extremely deep form with consistent improvements in accuracy, each fraction of a percent of improved accuracy costs nearly double the number of layers. Veit et al. [7] pointed out that the residual network introduced a series of shorter paths into the network and these paths had certain redundancy, indicating that the reuse rate of features in the deep residual network is gradually diminished, that is, a large number of features cannot be fully utilized.

Recent studies have demonstrated that in addition to increasing the depth of networks, the performance can also be improved by explicitly embedding learning mechanisms. One such approach are the Inception architectures [2][8], which introduce multi-scale mechanisms into the networks, improving the representational power of the network. More recent works [9] [10] focused on channel and spatial dependencies, establishing attention learning mechanisms at channel and spatial aspects and improving the efficiency of feature utilization for the network. Zhang et al. [11] pointed out that not all the layers in CNNs play the same role; different layers make different contributions to the process of feature extraction. Therefore, we propose an explicit block-wise attention module: the focus-and-association (FA) module. The block here can theoretically contain any convolutional layer, any embedded algorithm unit or any combination of them. From the perspective of blocks in the network, we explicitly model the interdependencies among them and provide a feature recalibration mechanism. The proposed FA module can selectively emphasize the informative blocks and suppress the less useful blocks based on the global information of each block and the associated information of multiple blocks to reduce the redundancy among network blocks.

The structure of the FA module is illustrated in Fig. 1. A given block $\mathbf{B}_i$ ($i \in [0, \mathrm{n}-1]$) of the network, it can be represented as a transformation $\mathbf{F}_{tri} : \mathbf{X}_i \to \mathbf{U}_i$, $\mathbf{X}_i \in \mathbb{R}^{H_i' \times W_i' \times C_i'}$, $\mathbf{U}_i \in \mathbb{R}^{H_i \times W_i \times C_i}$. To model the interdependencies between $\mathbf{B}_i$ and $\mathbf{B}_0$, $\mathbf{B}_1$, …, $\mathbf{B}_{i-1}$, when $i > 1$, we create FA mechanism for $\mathbf{B}_i$, including two steps of "focus"($\mathbf{F}_{fo}$ in Fig. 1) and "association"($\mathbf{F}_{as}$ in Fig. 1). First, in the "focus" step, the input feature $\mathbf{X}_i$ (i.e., the final recalibrated output feature of block $\mathbf{B}_{i-1}$) of block $\mathbf{B}_i$ is passed through a global pooling operation across spatial dimensions $H_i' \times W_i'$ and transformed into a channel descriptor $\boldsymbol{l} \in \mathbb{R}^{C_i'}$, which is also a descriptor

978-1-6654-9567-7/22/$31.00 ©2022 IEEE
DOI 10.1109/MLKE55170.2022.00011

for the output of block $\mathbf{B}_{i-1}$. Then it is followed by a tiny inference network composed of 2 fully connected (FC) layers, further

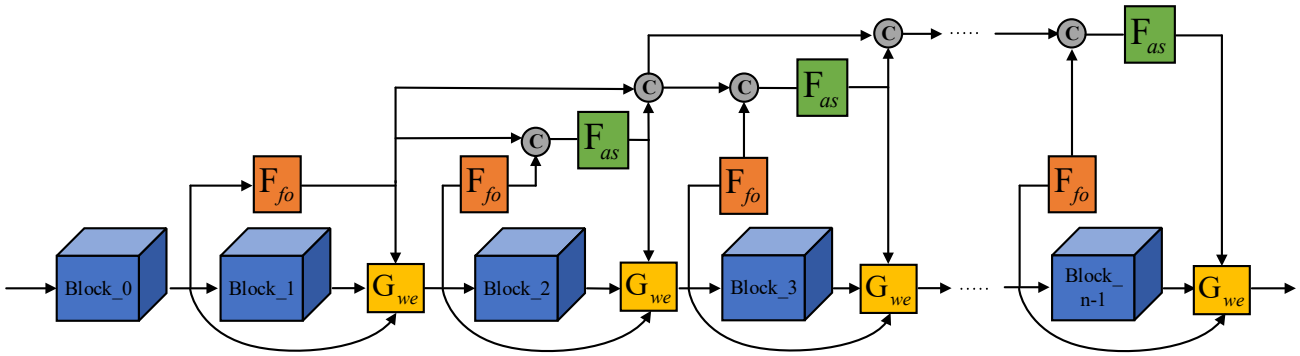

Fig. 1. Schematic diagram of a network with the FA module. Ⓒ represents the concatenation operation of tensors.

abstracting the channel descriptor $\boldsymbol{I}$ into a global descriptor $k_i' \in \mathbb{R}^1$ of $\mathbf{X}_i$. In this paper, k_i' is also called "focused attention". Until here, the "focus" step of $\mathbf{B}_i$ realizes an overall perception of the output tensor of $\mathbf{B}_{i-1}$. Second, the "association" step is used to generate the "associative attention" k_i. Before block $\mathbf{B}_i$, a set of "associative attention" [k_1, k_2, …, k_{i-1}] are generated from [$\mathbf{B}_1$, $\mathbf{B}_2$,…, $\mathbf{B}_{i-1}$] by the corresponding FA mechanisms. We concatenate [k_1, k_2, …, k_{i-1}] with the focused attention k_i' of $\mathbf{B}_i$, constructing an attention descriptor $\mathbf{s}_i' \in \mathbb{R}^i$. Then $\mathbf{s}_i'$ is passed through another tiny inference network that is composed of 2 FC layers and generates the output k_i. Finally, $\mathbf{U}_i$ is recalibrated by k_i through the weighting process ($\mathbf{G}_{we}$ in Fig. 1).

The FA module is a lightweight method, that involves only few additional parameters and slight computational burden, and it has good extensibility. By appropriately partitioning a network into blocks (i.e., how most networks were originally designed), the FA module can be used directly in various state-of-the-art architectures and effectively improve the performance. In addition, FA module has good compatibility, it can be used in conjunction with the existing state-of-the-art algorithm to achieve further performance improvement. To support these claims, we conduct multigroup experiments on various networks(ResNet [6], VGGNet [12], Inception-v4 [13] and ResNeXt [14]) with various datasets(CIFAR-10[15], CIFAR-100[15], Tiny ImageNet and ImageNet[16]), which show that FA Module has consistent performance improvements and is applicable to different networks and datasets.

Section II of this paper introduces related works. Section III provides the detailed structure of the proposed FA module. Section IV conducts experiments to verify and analyze the performance of FA Module. Section V concludes this paper and provides future research direction.

II. Related Work

A. Deep networks

Since large-scale neural network has been successfully applied in visual tasks [1], deeper networks have become an important concern of researchers. VGGNet [12] explored the relationship between the depth and the performance of networks, successfully constructing deep CNNs with the deepest layer up to 19, and indicats that increasing the depth of the network can improve the performance to a certain degree. Inception architectures[2][8] used multiple convolution kernels to extract and fuse the information from different scales of the feature map, generating a better feature representation. By introducing skip connections, ResNet [6] greatly inhibits the "degradation" of networks and made it feasible to construct extremely deep CNNs. Combined with the basic residual structure of ResNet, various new networks have been proposed, such as WideResNet [17], Inception-ResNet[13] and ResNeXt[14]. WideResNet is a residual network with increased width (i.e., more convolutional filters) and reduced depth, demonstrating the effectiveness of the wider network depth. Inception-ResNet combins the Inception units with skip connections, increasing the depth and width of the network, and further optimized the Inception units on the basis of [2], making it more concise and efficient. ResNeXt, based on ResNet, uses the group convolution method to improve the performance of networks without increasing the complexity of the parameters, and this structure has strong extensibility to be applied to various existing residual-based architectures.

The above methods mainly focus on factors such as depth, width and the complexity of the network. In this paper, we focus on the "attention" aspect of the network, and use the attention mechanism to make the network pay more attention to important information, thereby improving the network's abilities of feature extraction and representation.

B. *Attention mechanism*

Attention mechanism is an important part of human cognitive behavior[18][19][20], and humans tend to consciously focus on the information with more significance.

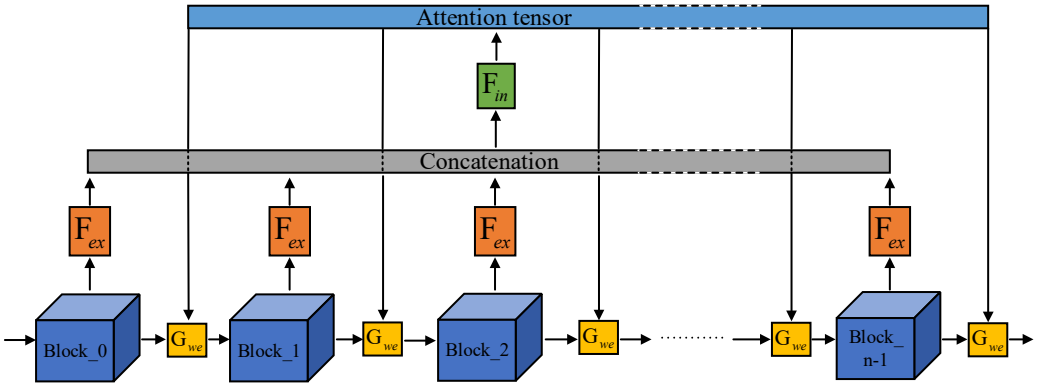

Fig. 2 Schematic diagram of the ideal block-wise attention module.

The importance of the attention mechanism has been extensively studied in previous work[21][22][23][24]. Long short-term memory (LSTM)[25] uses a gating mechanism, which is a kind of bottom-up attention mechanism based on saliency, to "memorize" important information and "forget" secondary information, and shows a huge advantage in processing long sequences. Highway networks[26] introduced a method similar to the gating mechanism in LSTM, through a "transform gate" and a “carry gate” to control “how much of the output is produced by transforming the input and carrying it”, which allows the features to be flexibly selected and learned by the network. Wang et al. [27] proposed an attention-based residual learning method, using an encoder-decoder-style attention module to refine the feature maps, which not only improved the precision but also the robustness to noise. Different from the above methods, which directly calculate the attention map of the three-dimensional feature, “squeeze-and-excitation” (SE) network SENet [9] and convolutional block attention module (CBAM) [10] separately calculate the channel attention and spatial attention by explicitly modelling the interdependencies among channels and among spatial pixels. SENet focuses on channel-wise attention information, extracting and expressing channel features through the SE block, which gives networks the capability to learn channel-wise interdependences from the training data. In addition to channel attention, CBAM uses spatial attention, which plays an important role in finding where to focus [28]. CBAM connects the channel attention block and the spatial attention block in series, and performed feature recalibration along the channel and spatial dimensions. In addition, recent works have shown that self-attention is also a feasible method to build image processing models [29][30]. In a variety of tasks, it replaces recurrent and convolutional models and achieves powerful performances [31][32][33][34].

In this paper, we focus on the interdependencies of the blocks which can theoretically be any convolutional layer, any embedded algorithm unit or any combination of them. By viewing the block as a whole unit, we establish a block-wise attention mechanism to perform feature recalibration to the blocks, making the network pay more attention to important blocks.

III. Block-wise Attention Module

A given block $\mathbf{B}_i$ ($i \in [0, n-1]$) of the network can be represented as a transformation $\mathbf{F}_{tri}: \mathbf{X}_i \to \mathbf{U}_i$, $\mathbf{X}_i \in \mathbb{R}^{H_i' \times W_i' \times C_i'}$, $\mathbf{U}_i \in \mathbb{R}^{H_i \times W_i \times C_i}$. Ideally, $\mathbf{F}_{tri}$ can be any convolutional layer, any embedded algorithm unit or any combination of them. The block-wise attention module is used to dynamically control the effectiveness of block $\mathbf{B}_i$, which can be described as:

$$\tilde{\mathbf{U}}_i = \mathbf{F}_{bw}(\mathbf{X}_i, \mathbf{U}_i) = \mathbf{X}_i + (\mathbf{U}_i - \mathbf{X}_i) \times k_i \qquad (1)$$

For block $\mathbf{B}_i$, whose input feature is $\mathbf{X}_i$ and output feature is $\mathbf{U}_i$, its effectiveness can be described as $(\mathbf{U}_i - \mathbf{X}_i)$. The transformation $\mathbf{F}_{bw}$ in equation (1) uses a weighting factor $k_i \in (0,1)$ to perform feature recalibration, recalibrating the response of $\mathbf{B}_i$. Ideally, the block-wise attention module should have the capability to fully capture the interdependences among blocks, making the blocks influence and compete with each other, and finally generate attention for each block.

A. *Ideal Structure*

According to our vision, ideally, the structure of the block-wise attention module could have the form like Fig. 2. $\mathbf{F}_{ex}$ represents the feature extraction process, which extracts global information from the output feature map of each block, generating feature descriptors. Then all the descriptors are passed into $\mathbf{F}_{in}$, which is an inference process, generating attention for each block. Finally, the effectiveness of each block is adjusted by recalibrating the output feature map of each block through each corresponding $\mathbf{G}_{we}$, which is a

weighting process, recalibrating the feature based on attention information.

However, this module exists only in ideal situations, and cannot be practically realized. The network is constructed serially while this module requires parallel feature processing. Therefore, we propose an FA module, which is a feasible form of the Block-wise Attention Module.

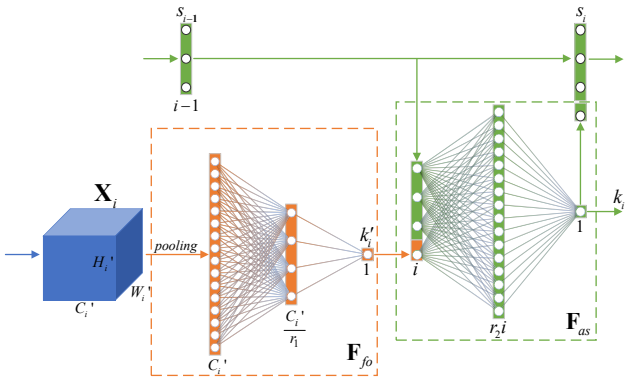

Fig. 3. The schematic diagram of the FA module. For block $\mathbf{B}_i$ of the network, $\mathbf{X}_i$ is its input feature. In this figure, the orange part represents the "focus" step $\mathbf{F}_{fo}$ and the green part represents "association" step $\mathbf{F}_{as}$. These colors correspond to those in Fig. 1.

B. Focus-and-Association Module

For human visual attention mechanism, in addition to screening information of interest based on focuses of attention, we also tend to generate associations among them, thereby obtaining a more reliable judgment. According to Treisman and Gelade's research[35] on feature integration theory, the human visual attention mechanism can be divided into two steps: (1) saliency-based feature perception; (2) fusion of salient features. Inspired by this mechanism, we proposed a focus-and-association (FA) module to generate block-wise attention by associating features from different blocks. This process $\mathbf{F}_{bw}$ can be described as:

$$\begin{aligned}\tilde{\mathbf{U}}_i &= \mathbf{F}_{bw}(k_1,k_2,...,k_{i-1},\mathbf{X}_i,\mathbf{U}_i)\\ &= \mathbf{G}_{we}(\mathbf{F}_{as}(k_1,k_2,...,k_{i-1},\mathbf{F}_{fo}(\mathbf{X}_i)),\mathbf{X}_i,\mathbf{U}_i)\\ &= \mathbf{G}_{we}(\mathbf{F}_{as}(k_1,k_2,...,k_{i-1},k_i'),\mathbf{X}_i,\mathbf{U}_i)\\ &= \mathbf{G}_{we}(k_i,\mathbf{X}_i,\mathbf{U}_i)\end{aligned} \tag{2}$$

Here $\mathbf{F}_{fo}$ represents the "Focus" step. For the input feature $\mathbf{X}_i$ of $\mathbf{B}_i$, $\mathbf{F}_{fo}$ is a mapping from $\mathbf{X}_i \in \mathbb{R}^{H_i' \times W_i' \times C_i'}$ to the focused attention $k_i' \in (0,1)$, which includes a preliminary feature extraction and inference process, similar to the first response of human visual attention mechanism. $\mathbf{F}_{as}$ represents the "association" step, which is a mapping from $[k_1,k_2,...,k_{i-1},k_i'] \in \mathbb{R}^i$ to the associative attention $k_i \in (0,1)$, which is an associated attentional inference process, similar to the association of human visual attention mechanism. $\mathbf{G}_{we}$ represents a weighting process that performs feature recalibration to the output of $\mathbf{B}_i$ based on k_i to dynamically control the effectiveness of $\mathbf{B}_i$. The overall diagram is shown in Fig. 1. When $i=0$, we do not recalibrate $\mathbf{B}_i$ with the FA module, but regard it as the benchmark block. When $i=1$, equation (2) equals:

$$\begin{aligned}\tilde{\mathbf{U}}_i &= \mathbf{F}_{bw}(\mathbf{X}_i,\mathbf{U}_i)\\ &= \mathbf{G}_{we}(\mathbf{F}_{as}(\mathbf{F}_{fo}(\mathbf{X}_i)),\mathbf{X}_i,\mathbf{U}_i)\end{aligned} \tag{3}$$

In this case, the input of $\mathbf{F}_{as}$ only contains $\mathbf{F}_{fo}(\mathbf{X}_i)$ subject to $\mathbb{R}^1$, and because the output of $\mathbf{F}_{as}$ also subject to $\mathbb{R}^1$, $\mathbf{F}_{as}$ is only a multiple mapping of the input, which is meaningless for the network. Therefore, we skip $\mathbf{F}_{as}$, letting $k_i = k_i'$, and equation (3) will be simplified to:

$$\begin{aligned}\tilde{\mathbf{U}}_i &= \mathbf{F}_{bw}(\mathbf{X}_i,\mathbf{U}_i)\\ &= \mathbf{G}_{we}(\mathbf{F}_{fo}(\mathbf{X}_i),\mathbf{X}_i,\mathbf{U}_i)\end{aligned} \tag{4}$$

In addition, when $i>1 \bigcap i \in \mathbb{Z}$, FA module can be strictly designed according to equation (2).

1)Focus: Saliency-based Feature Perception

Given a neural network with n blocks, we view the output $\mathbf{U}_i$ of each block $\mathbf{B}_i$ as a feature, then the network can be regarded as a feature set $\mathbf{M}$ containing n features. [35] pointed out that "without focused attention, features cannot be related to each other." For a feature $\mathbf{X}_i$ (i.e., the final recalibrated output of block $\mathbf{B}_{i-1}$) in $\mathbf{M}$, the proposed "focus" step is corresponding to the first step of the human visual attention mechanism: saliency-based feature perception. It generates a focused attention k_i' based on the saliency of the feature. Specifically, it contains two sub-processes: (1) saliency feature extraction and (2) saliency map inference. The detailed composition of the "focus" step is illustrated in Fig. 3 (the orange part) and can be expressed as:

$$k_i' = \mathbf{F}_{fo}(\mathbf{X}_i) = \mathbf{F}_{in}(\mathbf{F}_{ex}(\mathbf{X}_i)) \tag{5}$$

$\mathbf{F}_{ex}$ realizes the first subprocess: Saliency feature extraction. For CNNs, there is richer spatial information in the shallow layer of the network, and richer semantic information in the deep layer. The continuous convolution operation is essentially a process that compressing the spatial information of the spatial dimension into the channel dimension to generate semantic information. Here we concern more about semantic information, therefore, for $\mathbf{F}_{ex}$ we suggest compressing the spatial information into the channels. The input feature $\mathbf{X}_i$ can be viewed as a set of channel descriptors, whose statistics are prevalent in feature engineering works [36][37][38]. Here, we use global average pooling, which is a common method for the abstraction of spatial information. Zhou et al. [39] pointed out that global average pooling can be used to explore the extent of the target object, and Hu et al. [9] and Woo et al. [10] both used it to calculate spatial statistics. We perform global average pooling to $\mathbf{X}_i$ across spatial

dimensions $H_i' \times W_i'$ to get a channel descriptor $\boldsymbol{l}$, and the j-th element l_j of $\boldsymbol{l}$ can be calculated by:

$$l_j = \mathbf{F}_{ex}(\mathbf{X}_{i,j}) = \frac{1}{H_i' \times W_i'} \sum_{p=1}^{H_i'} \sum_{q=1}^{W_i'} \mathbf{X}_{i,j}(p,q) \tag{6}$$

$\mathbf{F}_{in}$ realizes the second subprocess: saliency map inference, where the saliency map is an attention value subject to $\mathbb{R}^1$. [9] proved that a network composed of two FC layers can learn the nonlinear interaction relationship from channels, and we adopt a similar structure for $\mathbf{F}_{in}$:

$$\mathbf{F}_{in}(\boldsymbol{l}) = \sigma(\mathbf{W}_2 \delta(\mathbf{W}_1 \boldsymbol{l})) \tag{7}$$

Where δ denotes the rectified linear unit (ReLU) activation function [40], σ denotes the sigmoid activation function, $\mathbf{W}_1 \in \mathbb{R}^{\frac{C_i'}{r_1} \times C_i'}$, and $\mathbf{W}_2 \in \mathbb{R}^{1 \times \frac{C_i'}{r_1}}$. $\mathbf{F}_{in}$ includes 2 FC layers and reduces $\boldsymbol{l} \in \mathbb{R}^{C_i'}$ to the focused attention $k_i' \in \mathbb{R}^1$. To limit the complexity and aid generalization, we create a dimensionality-reduction layer (the first FC layer) with reduction ratio r_1, followed by a ReLU and another dimensionality-reduction layer (the second FC layer) with the output subjected to $\mathbb{R}^1$. Finally, it's activated by sigmoid function, outputting the focused attention k_i'.

Discussion. The proposed "focus" step is a process of saliency-based feature perception, including saliency feature extraction and saliency map inference. In essence, this process achieves a high degree of abstraction from the input feature, which contains spatial and semantic information, to the focused attention that contains only semantic information.

2)Association: Fusion of Salient Features

The "association" step is corresponding to the second step of the human visual attention mechanism: fusion of salient features. It generates an associative attention k_i by associating features from different blocks.

Before block $\mathbf{B}_i$, a set of "associative attention" $\mathbf{s}_{i-1}$ (composed of [k_1, k_2, …, k_{i-1}]) is generated from [$\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{i-1}$] by the corresponding FA mechanisms. We concatenate $\mathbf{s}_{i-1}$ with the focused attention k_i' of $\mathbf{B}_i$, constructing an attention descriptor $\mathbf{s}_i' \in \mathbb{R}^i$, which contains the attention information of [$\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_i$]. Then $\mathbf{s}_i'$ is passed through the "association" step $\mathbf{F}_{as}$, which is illustrated in Fig. 3 (the green part), and it can be described as:

$$k_i = \mathbf{F}_{as}(\mathbf{s}_i') = \sigma(\mathbf{W}_4 \delta(\mathbf{W}_3 \mathbf{s}_i')) \tag{8}$$

Where $\mathbf{W}_3 \in \mathbb{R}^{r_2 i \times i}$, $\mathbf{W}_4 \in \mathbb{R}^{1 \times r_2 i}$. Similar to $\mathbf{F}_{in}$, $\mathbf{F}_{as}$ is also composed of 2 FC layers. The difference is that here, the input $\mathbf{s}_i'$ is a feature with a very small size while contains a large amount of information (attention for the blocks). To fully extract interdependences among blocks, we create a dimensionality-increasing layer (the first FC layer) with increasing ratio r_2, which improves the ability to utilize features. The rest part of $\mathbf{F}_{as}$ is the same as $\mathbf{F}_{in}$: ReLU activation, a dimensionality-reduction layer (the second FC layer) and sigmoid activation. Finally, the associative attention k_i is generated and is concatenated with $\mathbf{s}_{i-1}$ to construct $\mathbf{s}_i$ which is useful for later FA modules.

Discussion. The proposed "association" step is a process of fusion of salient features, which integrates interdependences among blocks, generating reliable associative attention for the block. On the other hand, from the perspective of backpropagation, it provides multiple paths for blocks with different feature levels, thereby making the learning process more coordinated.

3)Weighting: Feature Recalibration

The weighting process is used to control the effectiveness of block $\mathbf{B}_i$ based on the associative attention k_i, recalibrating the output feature $\mathbf{U}_i$ of $\mathbf{B}_i$ and generating the recalibration result $\tilde{\mathbf{U}}_i$, which is also the input feature $\mathbf{X}_{i+1}$ of block $\mathbf{B}_{i+1}$. This process is defined as:

$$\mathbf{X}_{i+1} = \tilde{\mathbf{U}}_i = \mathbf{G}_{we}(k_i, \mathbf{X}_i, \mathbf{U}_i) = \mathbf{X}_i + (\mathbf{U}_i - \mathbf{X}_i) \times k_i \tag{9}$$

A similar feature weighting method is also used in [26], and it has been proved to have good generality in feature selection, which is defined as:

$$\tilde{\mathbf{U}}_i = \mathbf{U}_i \cdot \mathbf{T}(\mathbf{X}_i) + \mathbf{X}_i \cdot \mathbf{C}(\mathbf{X}_i) \tag{10}$$

[26] explain equation (10) from the perspective of the gating mechanism, where $\mathbf{T}(\mathbf{X}_i)$ is called the "transform gate" and $\mathbf{C}(\mathbf{X}_i)$ is called the "carry gate". When $\mathbf{C} = 1 - \mathbf{T}$,

$$\tilde{\mathbf{U}}_i = \mathbf{U}_i \cdot \mathbf{T}(\mathbf{X}_i) + \mathbf{X}_i \cdot (1 - \mathbf{T}(\mathbf{X}_i)) \tag{11}$$

Apparently, after expanding equations (9) and (11), they will have a similar form. However, since our k_i is different from $\mathbf{T}(\mathbf{X}_i)$ both in size and calculation process, we explain the mechanism of equation (9) from another point of view: For the block $\mathbf{B}_i$, $(\mathbf{U}_i - \mathbf{X}_i)$ represents all its effectiveness. From the perspective of the effectiveness of the block, we rescale $(\mathbf{U}_i - \mathbf{X}_i)$ with the associative attention k_i to control how much $\mathbf{B}_i$ works. In particular, there are the following extreme cases:

$$\tilde{\mathbf{U}}_i = \begin{cases} \mathbf{X}_i, & if\ k_i = 0, \\ \mathbf{U}_i, & if\ k_i = 1. \end{cases} \tag{12}$$

When $k_i = 0$, the effectiveness of $\mathbf{B}_i$ is completely suppressed, which is equivalent to skipping $\mathbf{B}_i$. When $k_i = 1$, the effectiveness of $\mathbf{B}_i$ is completely released, which is equivalent to skipping $\mathbf{G}_{we}$. Therefore, depending on the value of k_i, the effectiveness of $\mathbf{B}_i$ can be smoothly controlled by the weighting process $\mathbf{G}_{we}$. In addition, $\mathbf{U}_i$ and $\mathbf{X}_i$ in

equation (9) must have the same shape. When mismatching, we use the method mentioned in [6], performing a linear projection with convolutions to match the features.

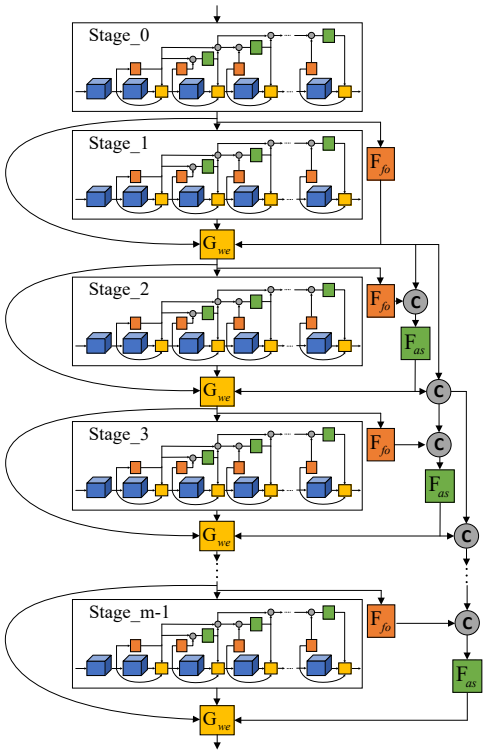

Fig. 4. A multi-stage deployment form for the FA module

C. Architecture Arrangement

The proposed FA module can be applied to most existing state-of-the-art CNNs. Because it's a block-wise attention mechanism, here different definitions of the block lead to different forms of deployment.

For VGGNet[12], which has 5 blocks, the FA module can be easily deployed by directly constructing the FA module for each block. The detailed structure is the same as the case of $n=4$ in Fig. 1. For networks with multiple stages, such as ResNet[6], Inception-v4[13] and ResNeXt[14], their architectures are not just a stack of multiple blocks; these networks are divided into multiple stages, (i.e., the image processing task is divided into multiple stages), and each stage contains multiple blocks. In this case, we make corresponding adjustments to the deployment of FA module. In section III we mentioned that $\mathbf{B}_i$ can contain any convolutional layer, any embedded algorithm unit or any combination of them. Therefore, the stage can also be viewed as a block, based on which, we proposed a multi-stage deployment form for the FA module in Fig. 4, which contains 2 steps: First, we separately build the FA module for the blocks in each stage, which is same as Fig. 1; Then we regard each stage as a block unit and repeatedly build the proposed FA module at the stage aspect. In this way, we establish connections for different blocks and stages with FA module. In addition, we take the Residual block [6] and SE-Residual block [9] for example, showing the corresponding combination with FA module in Fig. 5.

D. Implementation

In this paper, FA module is applied to 4 architectures: VGG [12], ResNet [6], Inception-v4 [13] and ResNeXt [14]. For each of them, we use same optimizing strategy. During the training process on CIFAR-10 and CIFAR-100 datasets [15], we use the same data augmentation method with [6]: performing 4 pixels zero padding operations on each side of the images and taking a random 32×32 crop after horizontal flipping. Mean channel subtraction is also applied to perform normalization. When training the Tiny-ImageNet, we make adaptive adjustments based on the image size: 8 pixels zero padding and 64×64 crop size. When testing we evaluate only the single view of the original images (unless specifically pointed). The Models are trained using stochastic gradient descent (SGD) with a momentum 0.9 and a mini-batch size of 128. We use a weight decay of 0.0005 and start with the learning rate of 0.1, dividing it by 10 every 60 epochs. All the models are trained until converging, with a maximum of 240 epochs.

We compare the performances between networks deployed with FA module (FANet) and SE block [9] (SENet). For a fair comparison with SENet, we set the reduction ratio r (reported in[9]) to be 4 and 8, and take the one with better performance as the final result. For the proposed FANet, to show the general effect of this structure, we set the ratio (mentioned in section III.B) $r_1=r_2=4$ to perform experiments.

IV. EXPERIMENTS

In this section, we perform experiments to verify the effectiveness of FA module in a range of datasets and architectures.

A. Experiments on CIFAR-10 and CIFAR-100

1)CIFAR-10:

We first perform experiments on the CIFAR-10 dataset [15], which contains 10 classes, 50K training images and 10K test images with an image size of 32×32. We separately experiment on VGG[12], ResNet[6], Inception-v4[13] and ResNeXt[14], deploying FA module into the networks and comparing it with SENet[9].

For VGG-19, we add batch normalization [8] before the activation function in each block to accelerate the training process. We delete two FC layers, leaving only one FC layer as the classifier. In addition, we use the pretrained weights on the ImageNet 2012 classification dataset[16] to initialize convolutional layers. The structure of FA-VGG-19 has the same form with the case of $n=4$ in Fig. 1.

For ResNet, we reproduce the network according to the structure [6], which contains 3 stages. Each stage contains n residual blocks, and each residual block contains 2 3×3 convolutions. In addition to the first convolutional layer and the last FC layer, there are totally $6n+2$ layers in the network. We deploy FANet according to the structure shown in Fig. 4.

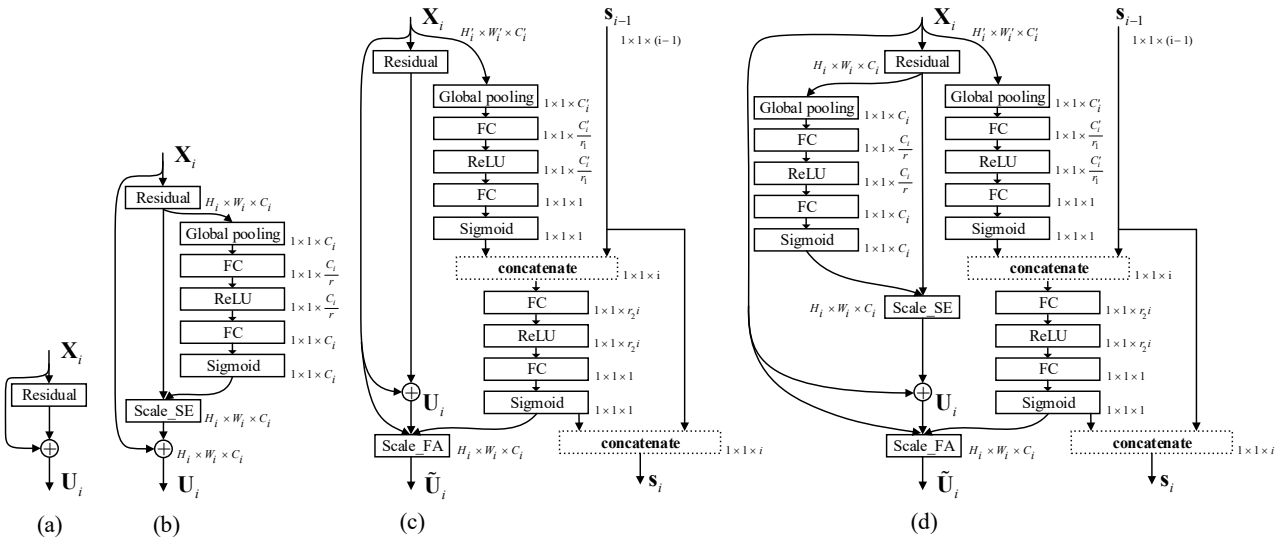

Fig. 5. Original blocks and combinations with FA module. (a): The original residual block [6]. (b): The SE-residual block, which is reported in [9]. (c): The proposed FA-residual block, which is suitable for the cases where $i > 1$. (d) The proposed FA-SE-residual block, which is suitable for the cases where $i > 1$. "Scale_SE" means the scaling process reported in [9] and "Scale_FA" means the weighting process $\mathbf{G}_{we}$ mentioned in section III.B.

Inception-v4 [13] contains a total of 3 stages, including 4 Inception-A blocks, 7 Inception-B blocks and 3 Inception-C blocks. Similar to the case of ResNet, we respectively deploy the FA module to stages and blocks. In addition, we use zero padding expanding the input image size to 96×96, and we fine-tune the reduction part (valid padding to same padding) and Inception-B block (kernel size of [1,7],[7,1] to [1,5],[5,1]) of Inception-v4 to fit the small size of input images. For ResNeXt, we follow the settings of [14], build an ResNeXt-29 which has 3 stages with 3 residual blocks in each stage. We set cardinality to 8 and width to 16 for ResNeXt in the experiments. ResNeXt has similar structure with ResNet, therefore it is deployed in the same way.

As is shown in Table I, apart from the comparison with the corresponding baseline, FANet is also compared with the state-of-the-art architecture SENet [9]. We first perform experiments on ResNets across different depths. Obviously the proposed FANet achieves consistent accuracy improvement in ResNet with various depths, and FA-ResNet-110 achieves a top-1 error of 5.57%, bringing only an extremely small increase in amount of parameters and computational complexity at the same time. Compared with SENet, FANet consumes similar additional parameters and computations. When the baseline network is shallow, SENet shows a larger improvement: SE-ResNet-20 achieves a 0.07% accuracy improvement over FA-ResNet-20. Under the baseline of ResNet-32, FANet achieves almost the same accuracy as SENet. When the baseline network is deep, FANet shows better performance than SENet. When the baselines are ResNet-44, ResNet-56 and ResNet-110, FANet achieves 0.19%, 0.24%, and 0.04% improvements in accuracy compared to SENet. Remarkably, FA-ResNet-56 achieves a higher accuracy than ResNet-110 (top-1 error of 5.86 versus top-1 error of 5.97) with only half of the total computational burden (6.278 MFLOPs versus 12.33 MFLOPs) and parameters (0.892M versus 1.751M).

Then we also perform experiments on VGG-19, Inception-v4 and ResNeXt-29, in which cases, the FANets achieve consistent accuracy improvements compared to the baselines and SENets. In particular, FA-ResNeXt-29 achieves a minimum top-1 error rate of 5.37%. From these results, the proposed FA module shows great potential to be deployed in various architectures and produce consistent performance improvements.

TABLE I. Top-1 error rates (%) on the CIFAR-10 dataset and complexity comparisons.

Baseline	original	re-implementation			SENet[9]			FANet (Proposed)		
	top-1 err.	*top-1 err.*	*MFLOPs*	*#param*	*top-1 err.*	*MFLOPs*	*#param*	*top-1 err.*	*MFLOPs*	*#param*
ResNet-20[6]	8.75	7.70	2.063	0.293M	**7.25**	2.122	0.301M	7.32	2.093	0.297M
ResNet-32[6]	7.51	7.05	3.431	0.487M	**6.50**	3.482	0.495M	6.51	3.484	0.495M
ResNet-44[6]	7.17	6.43	4.800	0.682M	6.33	4.870	0.692M	**6.14**	4.879	0.693M
ResNet-56[6]	6.97	6.31	6.168	0.876M	6.10	6.259	0.889M	**5.86**	6.278	0.892M
ResNet-110[6]	6.43	5.97	12.33	1.751M	5.61	12.51	1.778M	**5.57**	12.67	1.802M
VGG-19[12]	-	7.34	140.2	20.03M	6.57	142.4	20.34M	**6.45**	143.9	20.56M
Inception-v4[13]	-	17.9	258.1	36.84M	12.7	284.5	40.60M	**12.3**	284.2	40.56M
ResNeXt-29[14]	-	5.93	39.78	5.647M	5.70	47.04	6.685M	**5.37**	46.91	6.665M

The "original" column gives the results reported in original paper. To make a fair comparison, we retrain the baseline models and report the scores in the re-implementation column. We compare the performance of the proposed FANet with the state-of-the-art SENet [9]. The SENet column refers to the corresponding architectures where the SE block has been deployed and the FANet column refers to the corresponding architectures where the FA module has been deployed.

TABLE II. Top-1 error rates (%) on the CIFAR-100 dataset and complexity comparisons.

	MFLOPs	**#param**	**top-1 err.**
VGG-19[12]	140.6	20.08M	28.58
VGG-19[12]+SE[9]	142.7	20.39M	27.34
VGG-19[12]+FA	144.2	20.60M	**26.61**
ResNet 56[6]	6.209	0.882M	29.16
ResNet-56[6]+SE[9]	6.300	0.895M	**27.93**
ResNet-56[6]+FA	6.319	0.898M	28.01
Inception-v4[13]	259.1	36.97M	61.56
Inception-v4[13]+SE[9]	285.4	40.74M	44.38
Inception-v4[13]+FA	285.1	40.70M	**38.20**
ResNeXt-29[14]	40.43	5.739M	22.92
ResNeXt-29[14]+SE[9]	47.68	6.777M	**20.80**
ResNeXt-29[14]+FA	47.55	6.757M	22.47

2)CIFAR-100

To verify the performance of the FA module in a complex dataset with a larger number of classes, we also perform experiments on CIFAR-100 dataset, which contains 100 classes, 50K training images and 10K test images with an image size of 32×32. When deploying the FA module, the settings are the same as those on the CIFAR-10 dataset, except that the last FC layer is adaptively modified to 100 neurons to match the number of classes. As is shown in Table II, for each baseline network, the corresponding FANet achieved a consistent accuracy improvement, in particular, a 23.36% accuracy improvement was achieved in Inception-v4 network. Compared with SENet, FANet achieves better performances than SENet in the baseline of VGG-19 and Inception-v4 (0.73% and 6.18% respectively, over SENet) while in the baselines of ResNet and ResNeXt, SENet achieves a better performance because FANet improves the performance mainly by reducing the redundancy among blocks. For ResNet and ResNeXt, which have residual blocks, the original residual learning mechanism has already reduced this redundancy to a certain extent; therefore the effectiveness of the corresponding FANet is not as remarkable as SENet, which mainly reduces redundancy among channels. Especially in the ResNeXt baseline, which has far more channels than ResNet (1024 versus 64 for the last convolutional layer), the corresponding SENet greatly reduce the channel-wise redundancy and achieves an improvement in accuracy of 1.67% compared to FANet.

B. Experiments on Tiny ImageNet

TABLE III. Top-1 error rates (%) on the Tiny ImageNet dataset and complexity comparisons.

	MFLOPs	**#param**	**top-1 err.**
VGG-19[12]	140.9	20.13M	55.30
VGG-19[12]+SE[9]	142.0	20.28M	**39.99**
VGG-19[12]+FA	144.6	20.65M	41.81
ResNet-56[6]	6.254	0.889M	45.44
ResNet-56[6]+SE[9]	6.345	0.902M	**44.49**
ResNet-56[6]+FA	6.364	0.904M	44.64
Inception-v4[13]	260.2	37.13M	93.72
Inception-v4[13]+SE[9]	286.5	40.90M	54.73
Inception-v4[13]+FA	286.2	40.85M	**51.69**
ResNeXt-29[14]	41.14	5.841M	36.54
ResNeXt-29[14]+SE[9]	48.40	6.880M	**35.04**
ResNeXt-29[14]+FA	48.27	6.860M	35.60

We also perform experiments on a more complex dataset, Tiny ImageNet, which contains 200 classes, 100K training images, 10K validation images and 10K test images with an image size of 64×64. We train the model on the training set and test the top-1 error rate on the validation set. The experiments are also performed on VGG[12], ResNet[6], Inception-v4[13] and ResNeXt[14]. We follow the settings in section IV.A and make adaptive adjustments to fit the Tiny ImageNet dataset: First, for each network, the last FC layer is adaptively modified to 200 neurons to match the class number. Second, for VGG-19, we add a global max pooling before the last FC layer. Third, for ResNet and ResNeXt, we adjust the stride of the first layer of 3×3 convolution from 1 to 2. The other configurations remain unchanged.

As is shown in Table III, for each baseline network, the corresponding FANet achieved a consistent accuracy improvement, in particular, a 42.03% accuracy improvement was achieved in Inception-v4 network. It should be noted that because of the high complexity both of the network and the dataset, in this case it's difficult for the baseline network to converge under our default parameter settings, while the corresponding SENet and FANet can significantly improve that. Compared with SENet, in the baseline of Inception-v4, FANet achieves a better performance (3.04% over SENet). In other baselines, SENets achieve a better performance.

Discussion. Based on the results of section IV.A and IV.B, we argue that the block-wise redundancy χ, the channel-wise redundancy η, the depth of the network and the class number of the dataset have certain connections: For $\gamma=\frac{\chi}{\eta}$, with an increase in the network depth and an decrease in the number of classes, γ will continue to increase; otherwise, γ will continue to decrease. The proposed FA module mainly reduce the block-wise redundancy χ, the state-of-the-art SE block mainly reduce the channel-wise redundancy η. These two kinds of redundancies are not completely independent and are related to each other. By reducing one of them, the other one is also reduced to a certain extent. How to quantify χ and η,

and jointly analyze them with the network depth and dataset complexity, will be one of our future research directions.

C. Combine FA Module with State-of-the-art

TABLE IV. Top-1 error rates (%) and complexity comparisons of SENet[9] and FA-SENet on the Tiny ImageNet dataset.

Baseline	SENet[9]		FA-SENet	
	top-1 err.	*#param*	*top-1 err.*	*#param*
Inception-v4[13]	54.73	40.90M	**48.29**	44.62M
ResNet-56[6]	**44.49**	0.902M	44.75	0.930M
ResNet-110[6]	43.96	1.790M	**43.90**	1.840M

FA-SENet represents the network deployed with both SE blocks and FA module, the diagram of FA-SE-Residual module is shown in Fig. 5. (d).

TABLE V. Top-1 error rates (%) and complexity comparisons of SENet[9] and FA-SENet on the CIFAR-100 dataset.

Baseline	SENet[9]		FA-SENet	
	top-1 err.	*#param*	*top-1 err.*	*#param*
VGG-19[12]	27.34	20.39M	**26.39**	20.91M
ResNet-110[6]	26.50	1.784M	**25.92**	1.834M

In section IV.B, we analyzed the mechanism of the FA module (proposed) and the State-of-the-art SE block [9], indicating that they can reduce the block-wise and channel-wise redundancies, respectively. In this section, we further deploy the FA module to SENet to simultaneously reduce network redundancies from both the block and channel aspects, and it is compared with the original SENet. Taking ResNet as an example, we illustrate the schema of an FA-SE-Residual block in Fig. 5 (d). We perform experiments on the Tiny ImageNet and CIFAR-100 datasets, and the settings are same as those in section IV.A and IV.B.

We first experiment on Tiny ImageNet dataset, and the results is shown in Table IV. In the baseline network of Inception-v4, FA-SENet achieves an improvement of 6.44% over SENet while in the baseline network of ResNet56, the accuracy of FA-SENet is even lower than that of SENet. Based on our reasoning in section IV.B, we argue that deeper networks have a higher degree of block-wise redundancy on which the proposed FA module works. Therefore, we further experiment on ResNet110 and this time FA-SENet achieves an improvement of 0.06% over SENet, which is in line with our expectations, and the variation tendency in accuracy shows the potential of FA the module to be applied to extremely deep networks. To further verify the effectiveness of FA-SENet, we also perform experiments on CIFAR-100 dataset. As shown in Table V, FA-SENet achieves a consistent accuracy improvement. When the baseline is VGG-19 and ResNet-110, FA-SENet achieves improvement of 0.95% and 0.58% respectively, over SENet.

Discussion. The mechanism of FA module reduces the block-wise redundancy of the network, and in section III we mentioned that this block can contain any convolutional layer, any embedded algorithm unit or any combination of them. In this section, we view SE block, which is a state-of-the-art embedded algorithm unit, as part of the original block and deploy the FA module. The results in this section show the great potential of the FA module in compatibility, deployed with state-of-the-art embedded algorithm units at the same time. In addition, the performance of the FA module in the deep network (ResNet110) shows that its application in the extremely deep network is worth anticipating.

D. Visualization and Analysis

In this section, we visualize part of the details in the network to further explore the mechanism of action of the FA module. We extract 5 classes from the training set of the ImageNet 2012 classification dataset [16] (in this paper, we call this subset as ImageNet-5 dataset), and take 80% of the dataset (for each class, i.e., 1040 images) as the training set, and 20% (for each class, i.e., 260 images) as the test set. During training, we resize the shorter side of the image to 256 pixels while maintain the aspect ratio of the image, and then perform random cropping to obtain images with a size of 224×224 pixels. When evaluating the model, we use the center-crop to obtain the single-crop top-1 error rate. We use ResNet-56 [6] as the baseline, which has the same architecture as is mentioned in section IV.B, except that the last FC layer is adaptively modified to 5 neurons to match the number of classes. In addition, the mini-batch size is set to 32. The other settings are the same as those in section III. D.

TABLE VI. Top-1 error rates (%) on the ImageNet-5 dataset and complexity comparisons.

	MFLOPs	#param	top-1 err.
ResNet-56[6]	6.174	0.876M	10.62
ResNet-56[6]+SE[9]	6.265	0.889M	9.23
ResNet-56[6]+FA	6.285	0.892M	10.31
ResNet-56[6]+SE[9]+FA	6.375	0.905M	**9.08**

We train and evaluate 4 networks: ResNet-56, SE-ResNet-56, FA-ResNet-56, and FA-SE-ResNet-56. As is shown in Table VI, FANet achieves consistent accuracy improvement. The corresponding FANets of ResNet-56 and SE-ResNet-56 (FA-ResNet-56 and FA-SE-ResNet-56) achieve accuracy improvements of 0.31% and 0.15%, respectively, with only a few additional parameters (0.016M) and slight computational burden (0.11 MFLOPs), and particularly, a minimum top-1 error of 9.08% is achieved in FA-SE-ResNet-56. To explore how the FA module works, we first draw training curves of top-1 error rate and loss both on the training set and test set. As is shown in Fig. 6, FANet produces consistent gains in performance that are sustained throughout the training process. The corresponding FANet has significantly faster convergence rate than ResNet-56 and SE-ResNet-56. In addition, we can see from (a) and (b) that in the early stage of training, FANet can significantly reduce the fluctuation amplitude of the curve, which indicates that the FA module makes the network converge in the direction with better generalizability.

We also applied gradient-weighted class activation mapping (Grad-CAM) [41] to the networks to visualize the images in the test set. Grad-CAM uses gradients to calculate the importance of spatial position in the convolutional layer. By observing the regions that network has considered important for predicting a class, we try to analyze how the FA module affects the ability of the network to extract features. We compare the visualization results of ResNet-56, SE-ResNet-56, FA-ResNet-56 and FA-SE- ResNet-56. To explore

how the FA module works, we select some representative results to show in Fig. 7, and the softmax scores for target classes are also given.

In Fig. 7, the FA module makes the network more adept at perceiving more detailed features from the whole space region and associating these features with each other. For example, for the visualization results in column 3, although ResNet-56 and SE-ResNet-56 can identify cats well (the corresponding score P is close to 1), the focuses are scattered and independent. However, FA-SE-ResNet-56 can continuously focus on the entire body of the cat, and although the corresponding P is not maximized, we argue that this classification result is the most robust. Remarkably, for the visualization results in the last 2 columns, where the target color is similar to the background color, ResNet-56 and SE-ResNet-56 cannot locate the target, while the corresponding FANets (FA-ResNet-56 and FA-SE-ResNet-56) can accurately focus on it. We believe that this feature has a necessary connection with the "association" step of the FA module. The "association" step integrates the interdependencies among blocks and establishes connections for different features of different levels. When performing backpropagation, the features of multiple levels and their connections are learned integrally. In response to the results of

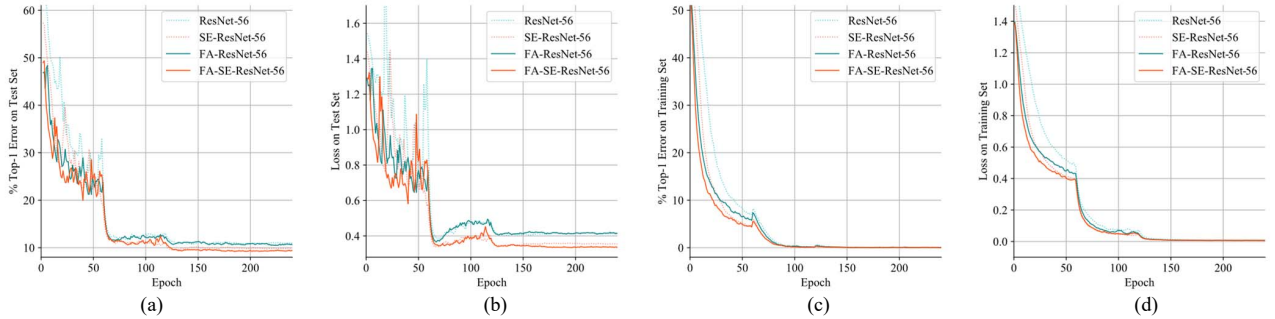

Fig. 6. Training curves of top-1 error rate and loss on ImageNet-5 dataset. (a) Top-1 error rate on test set. (b) Loss on test set. (c) Top-1 error rate on training set. (d) Loss on training set. The results of ResNet-56, SE-ResNet-56, FA-ResNet-56 and FA-SE-ResNet-56 are shown.

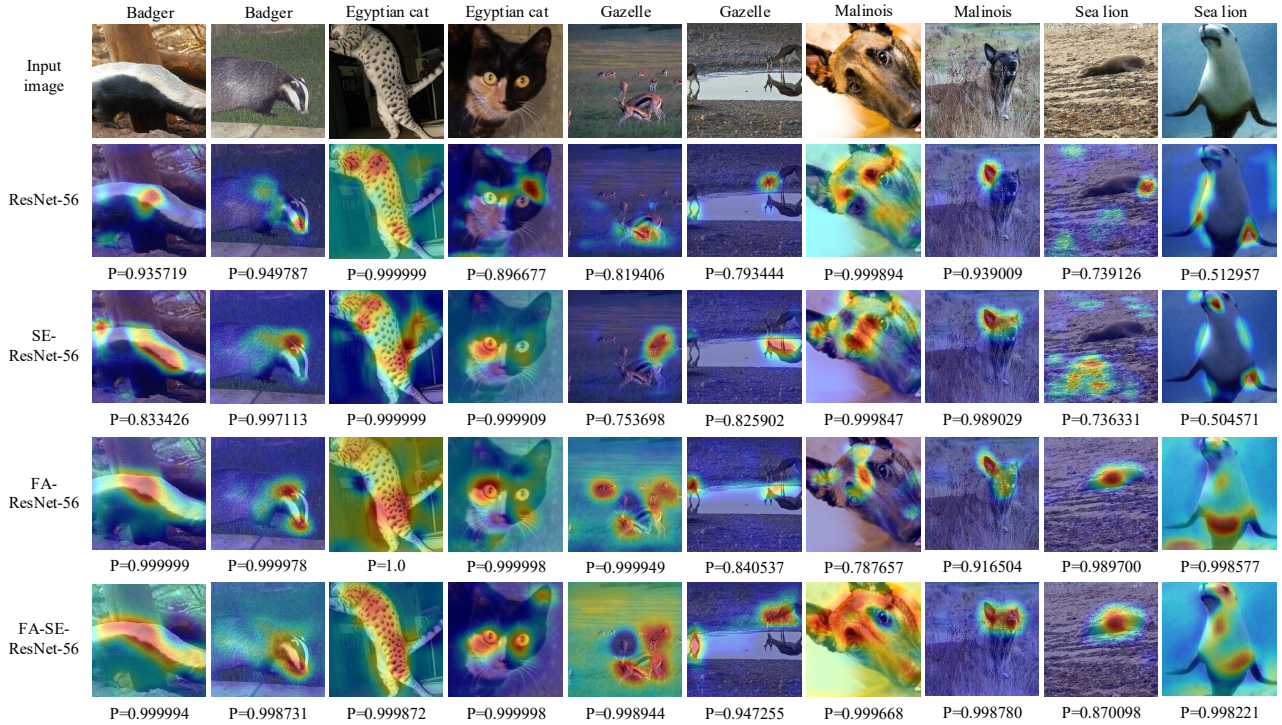

Fig. 7. Grad-CAM[41] visualization results. The visualization results of ResNet-56, SE-ResNet-56, FA-ResNet-56 and FA-SE-ResNet-56 are compared in the columns. The ground-truth label is above the input image, and P denotes the softmax score for the ground-truth class.

visualization, the network becomes more sensitive to the features, which are from different scales and spatial locations, and the associations among them in the image.

V.CONCLUSION

In this paper we propose the FA module, an explicit embedding learning mechanism, performing dynamic block-wise feature recalibration to reduce the block-wise redundancy and improve the representational power of a network. Experiments on multiple datasets and networks show the effectiveness of FANet, and it involves only few additional parameters and slight computational burden. In addition, the FA module has good compatibility. We combine it with the state-of-the-art SE block and achieve further improvements in accuracy. Finally, more experiments are performed to study how the FA module works, and we find that it makes the network more sensitive to perceive features and their connections, which may make a contribution to the field of image segmentation. In our future work, we will further explore the application of the FA module in extremely deep networks and the combination of FA module with various existing state-of-the-art methods.

ACKNOWLEDGMENT

This work was partially funded by the National Natural Science Foundation of China under grant numbers 61673314, 61573273 and the National Key Research and Development Program of China under grant 2018YFB1700104.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, 2017.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436–444, 2015.

[5] Y. Bengio, I. Goodfellow, and A. Courville, Deep learning. MIT press Massachusetts, USA:, 2016, ISBN 0262035618.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[7] A. Veit, M. Wilber, and S. Belongie, "Residual networks behave like ensembles of relatively shallow networks," Advances in neural information processing systems, vol. 29, pp. 550–558, 2016.

[8] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in International conference on machine learning, vol. 37. ICML, 2015, pp. 448–456.

[9] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks." IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 8, pp. 2011–2023, 2020.

[10] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 3–19.

[11] C. Zhang, S. Bengio, and Y. Singer, "Are all layers created equal?" arXiv preprint arXiv:1902.01996, 2019.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[13] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, 2017, pp. 4278–4284.

[14] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 5987–5995.

[15] A. Krizhevsky, G. Hinton et al., "Learning multiple layers of features from tiny images," in Technical Report, University of Toronto, 2009.

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein et al., "Imagenet large scale visual recognition challenge," International journal of computer vision, vol.115, no. 3, pp. 211–252, 2015.

[17] S. Zagoruyko and N. Komodakis, "Wide residual networks", in British Machine Vision Conference (BMVC), pp. 87.1–87.12, 2016.

[18] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," IEEE Transactions on pattern analysis and machine intelligence, vol. 20, no. 11, pp. 1254–1259, 1998.

[19] R. A. Rensink, "The dynamic representation of scenes," Visual cognition, vol. 7, no. 1-3, pp. 17–42, 2000.

[20] M. Corbetta and G. L. Shulman, "Control of goal-directed and stimulus-driven attention in the brain," Nature reviews neuroscience, vol. 3, no. 3, pp. 201–215, 2002.

[21] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in 3rd International Conference on Learning Representation, (ICLR), pp. 1–15, 2015.

[22] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention." in 3rd International Conference on Learning Representation, (ICLR), vol. 37, pp. 1–10, 2015.

[23] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," in International Conference on Machine Learning. ICML, 2015, pp. 1462–1471.

[24] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in Advances in Neural Information Processing Systems, NIPS2019, vol. 2, pp. 2017–2025, 2019.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," Advances in neural information processing systems, vol. 28, no. 2377-2385, NIPS2015, 2015.

[27] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 3156–3164.

[28] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 5659–5667.

[29] H. Hu, Z. Zhang, Z. Xie, and S. Lin, "Local relation networks for image recognition," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 3463–3472.

[30] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," , NIPS2019, pp. 68–80, 2019.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," Advances in neural information processing systems, NIPS2017, pp. 5998–6008, 2017.

[32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding,", in The Conference of the North American Chapter of the Association for Computational Linguistics(NAACL) pp. 4171–4186, 2019.

[33] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length

context," in 57th Annual Meeting of the Association-for-Computational-Linguistics (ACL2019), pp. 2978–2988, 2019.

[34] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in Advances in Information Processing Systems 32 (NIPS2019), pp. 1–11, 2019.

[35] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," Cognitive psychology, vol. 12, no. 1, pp. 97–136, 1980.

[36] J. Sanchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," International journal of computer vision, vol. 105, no. 3, pp. 222–245, 2013.

[37] L. Shen, G. Sun, Q. Huang, S. Wang, Z. Lin, and E. Wu, "Multi-level discriminative dictionary learning with application to large scale image classification," IEEE Transactions on Image Processing, vol. 24, no. 10, pp. 3109–3123, 2015

[38] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in 2009 IEEE Conference on computer vision and pattern recognition. IEEE, 2009, pp. 1794–1801.

[39] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2921–2929.

[40] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in Proceedings of the 27th International Conference on Machine Learning, 2010, pp. 1–8.

[41] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization." International Journal of Computer Vision, vol. 128, no. 2, pp. 336–359, 2020.