Pattern Recognition 148 (2024) 110148

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Improved channel attention methods via hierarchical pooling and reducing information loss

Meng Zhu [a], Weidong Min [a,c,d,*], Junwei Han [b,c,d], Qing Han [a,c,d], Shimiao Cui [a]

[a] School of Mathematics and Computer Science, Nanchang University, Nanchang 330031, China
[b] School of Software, Nanchang University, Nanchang 330047, China
[c] Institute of Metaverse, Nanchang University, Nanchang 330031, China
[d] Jiangxi Key Laboratory of Smart City, Nanchang 330031, China

ARTICLE INFO

Keywords:
Convolutional neural networks
Channel attention
Pooling
Reducing information loss
Information encoding

ABSTRACT

Channel attention has been demonstrated to improve performance of convolutional neural networks. Most existing channel attention methods lower channel dimension for reducing computational complexity. However, the dimension reduction causes information loss, thus resulting in performance loss. To alleviate the paradox of complexity and performance trade-off, we propose two novel channel attention methods named Grouping-Shuffle-Aggregation Channel Attention (GSACA) method and Mixed Encoding Channel Attention (MECA) method, respectively. Our GSACA method partitions channel variables into several groups and performs the independent matrix multiplication without the dimension reduction to each group. Our GSACA method enables interaction between all groups using a "channel shuffle" operator. After these, our GSACA method performs the independent matrix multiplication each group again and aggregates all channel correlations. Our MECA method encodes channel information through dual path architectures to benefit from both path topology where one uses the multilayer perception with dimension reduction to encode channel information and the other uses channel information encoding method without dimension reduction. Furthermore, a novel pooling operator named hierarchical pooling is presented and applied to our GSACA and MECA methods. The experimental results showed that our GSACA method almost consistently outperformed most existing channel attention methods and that our MECA method consistently outperformed the existing channel attention methods.

1. Introduction

With the development of computing hardware, deep convolutional neural networks (CNNs) have been extensively used and achieved notable success in a range of visual tasks [1–3]. Given a loss function, a CNN can adaptively fit any continuous function with any given accuracy by the gradient descent (or ascent) algorithm. This indicates that CNNs have the ability to finish many real tasks, because any real task can be regarded as a type of function calculation.

Recently, various channel attention methods have demonstrated to improve performance of CNNs [4–9]. These channel attention methods are aimed at adaptively recalibrating channel information, thus augmenting feature representations of the objects of interest. Most existing channel attention methods lower channel dimension for reducing computational complexity [4,5,9]. *E.g.*, the Squeeze-and-Excitation (SE) method reduced channel dimension via setting $r \geq 4$ (r denotes the reduction ratio).

However, the dimension reduction causes information loss, thus resulting in performance loss. To alleviate the paradox of complexity and performance trade-off, we propose two novel channel attention methods named Grouping-Shuffle-Aggregation Channel Attention (GSACA) method and Mixed Encoding Channel Attention (MECA) method, respectively. A novel multilayer perception named Grouping-Shuffle-Aggregation Multi-Layer Perception (GSAMLP) is presented and applied to our GSACA method. As described in Algo. 1, our GSAMLP partitions channel variables into several groups and performs the independent matrix multiplication without dimension reduction to each group. Our GSAMLP method enables interaction between all groups using a "channel shuffle" operator. After these, our GSAMLP method performs independent matrix multiplication without dimension reduction on each group once more and aggregates all correlations. Using this way, there is no reduction in channel dimension for each group, and interdependence between all channel variables is also built.

Consider that using the MLP to encode channel information can efficiently build channel correlations and that channel grouping without dimension reduction can efficiently reduce information loss, a novel

* Corresponding author at: School of Mathematics and Computer Science, Nanchang University, Nanchang 330031, China.
E-mail addresses: minweidong@ncu.edu.cn (W. Min), Junweihan2010@gmail.com (J. Han).

https://doi.org/10.1016/j.patcog.2023.110148
Received 25 March 2023; Received in revised form 17 September 2023; Accepted 20 November 2023
Available online 22 November 2023

mixed encoding unit is presented and applied to our MECA method. As described in Algo. 2, our mixed encoding unit encodes channel information through dual path architectures to benefit from both path topology where one uses the MLP with dimension reduction to encode channel information and the other uses channel grouping without dimension reduction to encode channel information.

Furthermore, a novel pooling operator named hierarchical pooling is presented and applied to our GSACA and MECA methods. As described in Section 3.2, our hierarchical pooling method consists of two steps. In the first step, our hierarchical pooling uses smooth max pooling to extract information. Smooth max pooling takes into account the differences between pixels, which increases the response to pixels with larger differences. Therefore, smooth max pooling can better preserve the fine details and texture features in the image. In the second step, our hierarchical pooling uses average pooling to extract information. In some tasks, average pooling may lead to better performance and generalization ability. By using a two-step strategy, our hierarchical pooling method can simultaneously utilize the advantages of both types of pooling.

The main contributions of this paper are summarized as follows.

(1) We propose the GSAMLP. Our GSAMLP uses jointly channel grouping, channel shuffle and channel aggregation to encode channel information. Using this way, our GSAMLP not only can avoid the channel dimension reduction to reduce information loss, but also can efficiently build channel correlations.
(2) We propose the mixed encoding unit to encode channel information. Our mixed encoding unit benefit from both using the MLP with dimension reduction to encode channel information and using channel grouping without dimension reduction to encode channel information. Using this way, our mixed encoding unit not only can efficiently build channel correlations, but also can efficiently reduce information loss.
(3) We propose hierarchical pooling. Our hierarchical pooling first uses smooth max pooling to extract information and then uses average pooling to extract information. In this way, our hierarchical pooling not only can better preserve the fine details and texture features in the image, but also can have better generalization ability.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces the proposed GSACA method in detail. Section 4 introduces the proposed MECA method in detail. Section 5 shows experimental results. Section 6 shows ablation studies. Section 7 makes the discussion. Section 8 draws the conclusion.

2. Related work

Different objects are typically represented by different channels in different feature maps of deep neural networks. Channel attention can adaptively adjust the weight of each channel, which can be seen as a process of selecting objects and determining what to focus on. The concept of channel attention was first proposed via Hu et al. [4], who presented SENet for this purpose. In this section, we summarize the representative channel attention works and discuss various channel attention methods along with their development process.

The core of SENet was the SE block, which collected global information, captured channel-wise relationships, and improved representation ability. SE blocks were consisted of two parts: the squeeze module and the excitation module. The squeeze module gathered global spatial information using global average pooling (GAP). The excitation module built channel-wise relationships and produced an attention vector using fully-connected layers and nonlinear layers (ReLU [10] and Sigmoid). Each channel of the input feature was then scaled via multiplying its corresponding element in the attention vector. Due to its low computational resource requirements, a SE block could be added after each residual unit [11]. However, SE blocks have limitations. GAP in the squeeze module is too simple to gather complex global information, and fully-connected layers in the excitation module increase model complexity. Subsequent works have attempted to improve the outputs of the squeeze module (*e.g.*, GSoP-Net [5]), reduce model complexity by means of enhancing the excitation module (*e.g.*, ECANet [7]), or improve both the squeeze and excitation modules (*e.g.*, SRM [6]).

The ability of GAP is limited to gather global information. To overcome this limitation, Gao et al. [5] proposed a global second-order pooling (GSoP) block, which could model high-order statistics while gathering global information. However, this improvement comes at the cost of additional computation. Qin et al. [9] demonstrated that GAP is a discrete cosine transform (DCT) special case, which they employed to propose a new multi-spectral channel attention method.

To improve both the squeeze and excitation modules, Lee et al. [6] proposed the lightweight style-based recalibration module (SRM). Its main innovation was style pooling, which used the mean and standard deviation of the input features to enhance its ability to gather global information. Moreover, SRM replaced the original fully-connected layer with a lightweight channel-wise fully-connected layer to reduce computational demands. Yang et al. [8] proposed the gated channel transformation (GCT) to reduce computational cost and efficiently gather channel information. GCT first collected global information via computing the l2-norm of each channel and applied a learnable vector $\boldsymbol{\alpha}$ to scale the feature. Then, it built competition mechanism using channel normalization. GCT used a Tanh activation to normalize the output attention vector. Finally, it multiplied the input using the attention vector and added a residual connection.

To avoid high model complexity, SENet reduces the channel dimension. However, this strategy causes channel information loss, thus leading to a reduction in the quality of results. To address this issue, Wang et al. [7] proposed the efficient channel attention (ECA) block, which used a one-dimensional convolution to build the interaction between channels instead of relying on dimension reduction. The ECA block only considered the direct interaction between each channel and its k-nearest neighbors. Luo et al. [12] drew inspiration from ECANet and further proposed a channel attention guided module (CAG). Its main improvement was using GAP and global max pooling separately to aggregate two distinct spatial context information based on an ECA block.

3. Proposed grouping-shuffle-aggregation channel attention method

First, the overview of our GSACA method is given in Section 3.1. Then, our hierarchical pooling is introduced in Section 3.2. Finally, our GSAMLP is introduced in Section 3.3.

3.1. Overview of our proposed method

The overview of our GSACA method is shown in Fig. 1. There are three sub-units in our GSACA method. **Squeeze.** $\boldsymbol{z} = \mathrm{GHP}(\boldsymbol{X})$. **Excitation.** $\boldsymbol{a} = \sigma(\mathrm{GSAMLP}(\boldsymbol{z}))$. **Scale.** $\boldsymbol{Y} = \boldsymbol{X} \odot \mathrm{Reshape}(\boldsymbol{a})$.

3.2. Hierarchical pooling

In most existing channel attention methods, GAP has been widely-used to gather multi-dimensional position information into one variable of the channel vector due to its simple implementation and great performance [4,7]. Assume that the input is $\boldsymbol{X} \in \mathbb{R}^{C\times H\times W}$. Let $\boldsymbol{\mu} \in \mathbb{R}^{C}$ denote the average pooled feature tensor by means of reducing the H-axis and W-axis of the input, GAP is formulated as Eq. (1),

$$\mu_k = \mathrm{GAP}(\boldsymbol{X}) = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W} x_{k,i,j}. \tag{1}$$

Here $x_{k,i,j}$ is the element at (i, j) within the kth feature map. μ_k is the mean of all elements within the kth feature map.

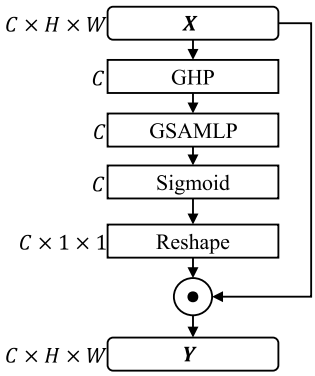

Fig. 1. Overview of our GSACA method. GHP denotes global hierarchical pooling. ⊙ denotes broadcast element-wise multiplication. H denotes image height, W denotes image width, and C denotes the number of image channels.

Yet, GAP takes all the low magnitudes into consideration and the contrast of the new feature map after pooling is reduced [13]. To alleviate the drawback of GAP, some existing attention methods additionally used global max pooling (GMP) to gather global position information and concatenated the average pooled and max pooled features [14–16], as formulated in Eq. (2). Let that $\boldsymbol{m} \in \mathbb{R}^C$ denotes the max pooled feature tensor by means of reducing the H-axis and W-axis of the input, and that $\boldsymbol{g} \in \mathbb{R}^{C\times 2}$ denotes concatenation feature by means of concatenating the average pooled and max pooled features,

$$\begin{aligned} \mu_k &= \mathrm{GAP}(\boldsymbol{X}) = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W} x_{k,i,j}, \\ m_k &= \mathrm{GMP}(\boldsymbol{X}) = \max_{i=1,\dots,H}\max_{j=1,\dots,W} x_{k,i,j}, \\ z_k &= [\mu_k, m_k]. \end{aligned} \tag{2}$$

Here m_k is the maximum value among all elements within the kth feature map.

However, the mixed pooling formulated as Eq. (2) has two drawbacks. One is that max(.) is non-derivable. The other is that the concatenation operation additionally spend computational cost. Inspired by mixed pooling [13], we propose a novel pooling method named hierarchical pooling method. Our hierarchical pooling benefits from both average pooling and smooth max pooling. And our hierarchical pooling is derivable.

Our hierarchical pooling first applies smooth max pooling to the input tensor $\boldsymbol{X}$ to obtain the output tensor $\boldsymbol{M} \in \mathbb{R}^{C\times\widehat{H}\times\widehat{W}}$, and then applies average pooling to tensor $\boldsymbol{M}$. Overall, our hierarchical pooling is formulated as Eq. (3),

$$\begin{aligned} m_{k,p,q} &= \tau \log\left(\sum_{(i,j)\in R_{(p,q)}} \exp\left(\frac{x_{k,i,j}}{\tau}\right)\right), \\ \mu_{k,u,v} &= \frac{1}{\left\|R_{(u,v)}\right\|}\sum_{(i,j)\in R_{(u,v)}} m_{k,i,j}. \end{aligned} \tag{3}$$

Here $x_{k,i,j}$ is the element at (i,j) within the pooling region $R_{(p,q)}$ which represents a local neighborhood around the position (p,q) in the tensor $\boldsymbol{X}$. $m_{k,i,j}$ is the element at (i,j) within the pooling region $R_{(u,v)}$ which represents a local neighborhood around the position (u,v) in the tensor $\boldsymbol{M}$. $\mu_{k,u,v}$ is the output of the hierarchical pooling operator related to the kth feature map. $\left\|R_{(u,v)}\right\|$ denotes the size of the pooling region $R_{(u,v)}$. τ denotes temperature coefficient and its default value is $1e{-}12$.

Let l denote the loss value, the back propagation of our hierarchical pooling is formulated as Eq. (4),

$$\frac{\partial l}{\partial x_{k,m,n}} = \frac{\partial l}{\partial \mu_{k,u,v}}\frac{1}{\left\|R_{(u,v)}\right\|}\frac{\exp\left(\frac{x_{k,m,n}}{\tau}\right)}{\sum_{(i,j)\in R_{(p,q)}}\exp\left(\frac{x_{k,i,j}}{\tau}\right)}. \tag{4}$$

Here $x_{k,m,n}$ is the element at (m,n) within the pooling region $R_{(p,q)}$.

The global version of our hierarchical pooling is formulated as Eq. (5). Firstly, global hierarchical pooling (GHP) uses a 3×3 smooth max pooling operation with padding being $(1,1)$ and stride being $(3,3)$ to obtain tensor $\boldsymbol{M} \in \mathbb{R}^{C\times\widehat{H}\times\widehat{W}}$. Here, $\widehat{H} = \lfloor (H+2)/3 \rfloor$, $\widehat{W} = \lfloor (W+2)/3 \rfloor$. Then, GHP applies GAP to obtain channel-level global information. Using this approach, GHP leverages the advantages of both smooth max pooling for capturing information and average pooling for capturing information,

$$m_{k,p,q} = \tau \log\left(\sum_{(i,j)\in R_{(p,q)}} \exp\left(\frac{x_{k,i,j}}{\tau}\right)\right), \mu_k = \frac{1}{\widehat{H}\widehat{W}}\sum_{i=1}^{\widehat{H}}\sum_{j=1}^{\widehat{W}} m_{k,i,j}. \tag{5}$$

3.3. Grouping-shuffle-aggregation multilayer perception

Most existing channel attention methods used the MLP to encode channel information [4,9]. To reduce computational complexity, the channel dimensionality is reduced in the MLP. However, this causes information loss, thus resulting in performance loss. To overcome the paradox of complexity and performance trade-off, we propose the GSAMLP. The overview of our GSAMLP is shown in Fig. 2. The details of our GSAMLP is described in Algo. 1.

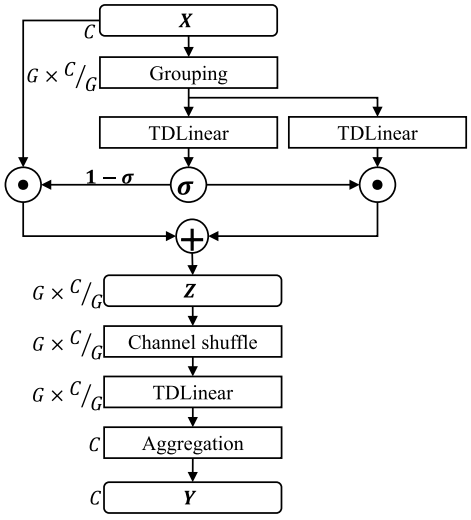

Fig. 2. Overview of our GSAMLP. TDLinear denotes Time Distributed Linear. G denotes the number of grouping.

Channel grouping. To avoid dimension reduction while not increasing parameter complexity, our GSAMLP groups channels, allowing each group of channels to independently perform matrix operations. Channel grouping techniques have been widely used in state-of-the-art CNN backbone networks [18–20]. In each layer of the residual module, the hyperparameter G increases as the number of channels increases, *e.g.*, G can be set as $G = C/w$, where w is a fixed constant and $w \in \{16, 32, 64, \dots\}$.

Residual gated linear unit. Inspired by Highway Networks [21] and the Gated Linear Unit (GLU) [22], our GSAMLP adopts the residual gated linear unit, defined as $\boldsymbol{z} = \boldsymbol{x} \odot (\boldsymbol{1} - \boldsymbol{g}) + \boldsymbol{u} \odot \boldsymbol{g}$, which not only reduces the vanishing gradient problem, but also enables information to be transmitted across multiple channels.

Channel shuffle. Due to channel grouping, the channel information within each group does not interact with each other. Therefore, our GSAMLP uses channel shuffle [17] to promote inter-group interaction, and then each group performs independent matrix operations again, allowing each output component to contain contributions from all input components.

4. Proposed mixed encoding channel attention method

First, the overview of our MECA method is given in Section 4.1. Then, our mixed encoding unit is introduced in Section 4.2.

Algo. 1: Details of our GSAMLP

Input: The input vector $\boldsymbol{x} \in \mathbb{R}^C$, the trainable weights $\boldsymbol{\Theta}, \boldsymbol{\Omega}, \boldsymbol{W} \in \mathbb{R}^{G\times(C/G)\times(C/G)}$, the hyperparameter G, $p = 0.2$.

Output: The output vector $\boldsymbol{y} \in \mathbb{R}^C$.

/* Channel grouping and building channel correlations for each group. */

$[\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_G] = \boldsymbol{x}$, $\boldsymbol{x}_i \in \mathbb{R}^{C/G}$;

for $i \leftarrow 1$ **to** G **do**

  $\boldsymbol{u}_i = \boldsymbol{\theta}_i \boldsymbol{x}_i$, $\boldsymbol{u}_i \in \mathbb{R}^{C/G}$;

  $\boldsymbol{v}_i = \boldsymbol{\omega}_i \boldsymbol{x}_i$, $\boldsymbol{v}_i \in \mathbb{R}^{C/G}$;

end

/* Residual gated linear unit. */

$\boldsymbol{g} = \sigma(\text{Dropout}([\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_G]), p)$;

$\boldsymbol{z} = \boldsymbol{x} \odot (\boldsymbol{1} - \boldsymbol{g}) + \boldsymbol{u} \odot \boldsymbol{g}$;

/* Enabling interaction between all groups using the channel shuffle [17] operator. */

$\text{ChannelShuffle}([\boldsymbol{z}_1, \boldsymbol{z}_2, \dots, \boldsymbol{z}_G])$;

/* Building channel correlations for each group again. */

for $i \leftarrow 1$ **to** G **do**

  $\boldsymbol{y}_i = \boldsymbol{w}_i \boldsymbol{z}_i$, $\boldsymbol{y}_i \in \mathbb{R}^{C/G}$;

end

/* Aggregation. */

$\boldsymbol{y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_G]$.

4.1. Overview of our proposed method

The overview of our MECA method is shown in Fig. 3. There are three sub-units in our MECA method. **Squeeze.** $\boldsymbol{z} = \text{GHP}(\boldsymbol{X})$. **Excitation.** $\boldsymbol{a} = \text{MixedEncoding}(\boldsymbol{z})$. **Scale.** $\boldsymbol{Y} = \boldsymbol{X} \odot \text{Reshape}(\boldsymbol{a})$.

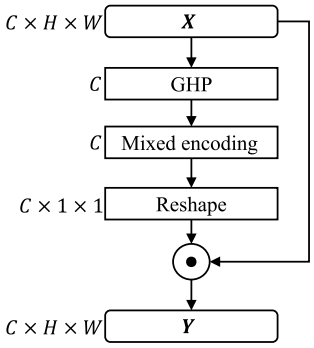

Fig. 3. Overview of our MECA method.

4.2. Mixed encoding unit using dual path architectures

The methods to encode channel information can be mainly divided into four categories. The first is using the MLP to encode channel information [4,9], the second is using the cross-correlation operator to encode channel information [7], the third is using the element-wise multiplication to encode channel information [6,8], and the fourth is using channel grouping to encode channel information, *e.g.*, our GSACA. Each method to encode channel information has its own advantage and disadvantage. Since that, the mixed encoding has the ability to combine their advantages. Therefore, we present mixed encoding unit to encode channel information.

The overview of our mixed encoding unit is shown in Fig. 4. The details of our mixed encoding unit is described in Algo. 2. There are two branches. One branch encodes channel information using a standard MLP. The other branch encodes channel information through channel grouping. The outputs of the two branches are then aggregated using random average fusion. Using this way, our mixed encoding unit not only can efficiently build channel correlations, but also can efficiently reduce information loss.

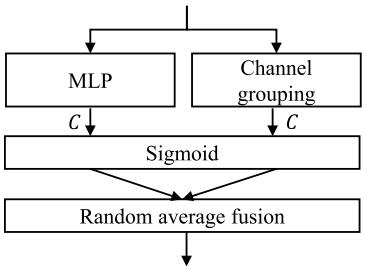

Fig. 4. Overview of our mixed encoding unit.

Algo. 2: Details of our mixed encoding unit

Input: The input vector $\boldsymbol{x} \in \mathbb{R}^C$, the trainable weights $\boldsymbol{W}_1, \boldsymbol{W}_2 \in \mathbb{R}^{C\times(C/r)}$, $\boldsymbol{W}_3 \in \mathbb{R}^{(C/r)\times C}$, $\boldsymbol{\Theta} \in \mathbb{R}^{G\times(C/G)\times(C/G)}$, the hyperparameters r, G.

Output: The output vector $\boldsymbol{y} \in \mathbb{R}^C$.

/* Encoding channel information using the MLP. */

$\boldsymbol{u} = \sigma(\boldsymbol{W}_3((\boldsymbol{W}_2\boldsymbol{x}) \odot \sigma(\boldsymbol{W}_1\boldsymbol{x})))$;

/* Encoding channel information using the channel grouping. */

$[\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_G] = \boldsymbol{x}$, $\boldsymbol{x}_i \in \mathbb{R}^{C/G}$;

for $i \leftarrow 1$ **to** G **do**

  $\boldsymbol{v}_i = \sigma(\boldsymbol{\theta}_i \boldsymbol{x}_i)$, $\boldsymbol{v}_i \in \mathbb{R}^{C/G}$;

end

$\boldsymbol{v} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_G]$;

/* Random average fusion. */

During training time: $\boldsymbol{y} = (1 - \lambda)\boldsymbol{u} + \lambda\boldsymbol{v}$, where λ is a random value being either 0 or 1. During test time: $\boldsymbol{y} = (\boldsymbol{u} + \boldsymbol{v})/2$.

Settings of hyperparameters. The setting of hyperparameter G is identical to that in GSAMLP. As the mixed encoding unit involves two branches, the hyperparameter r can be set to twice the value used in SE to align with SE's parameter complexity.

Gated liner unit. For aligning with SE's parameter complexity, the hyperparameter r in our mixed encoding unit is set to twice the value used in SE. To compensate for the performance loss caused by parameter reduction, our mixed encoding unit uses the GLU, defined as $(\boldsymbol{W}_2\boldsymbol{x}) \odot \sigma(\boldsymbol{W}_1\boldsymbol{x})$. $\boldsymbol{W}_2\boldsymbol{x}$ does not have any activation function applied, making it less prone to gradient vanishing.

Random average fusion. The inspiration for the random average fusion comes from the random Dropout and DropConnect [23] methods. During training with Dropout, a random subset of activation values in each layer are set to zero. Similarly, with DropConnect, a random subset of weights within the network are set to zero. Both of these techniques have been shown to effectively regularize neural networks. The value of λ, which is randomly set to either 0 or 1, determines whether using the MLP to encode channel information or using the channel grouping to encode channel information is used. In other words, the random average fusion stochastically changes the encoding regulation scheme, which can partially address the issues encountered with using the MLP to encode channel information and using the channel grouping to encode channel information.

Comparison between different fusion methods. To explore the effect of different fusion methods on the performance of our mixed encoding unit, we conducted comparison experiments on CIFAR-100 [24] using EfficientNet-B0 [19]. The experimental configurations were the

Table 1
Comparison experiments between different fusion methods. None means the attention mechanism used in the original backbone network is removed.

Model	Top-1 err. (%) ↓	Top-5 err. (%) ↓
EfficientNet-B0 [19] + none	38.43	13.92
+ Mul.	37.95	13.19
+ Avg.	38.02	13.97
+ Adaptive Avg.	38.46	13.88
+ Random Avg.	36.81	13.12

Table 2
Comparison experiments between each branch.

Model	Top-1 err. (%) ↓	Top-5 err. (%) ↓
EfficientNet-B0 [19] + none	38.43	13.92
+ MLP	38.84	13.97
+ channel grouping	38.32	13.92
+ mixed encoding	36.81	13.12
REGNETY-800MF [20] + none	42.06	16.33
+ MLP	41.77	15.47
+ channel grouping	40.82	15.49
+ mixed encoding	41.04	15.41

same as those described in Section 5.1. The results were listed in Table 1, which indicated that using random average fusion allowed our mixed encoding unit to achieve optimal performance.

Exploring the complementarity of two branches. To explore the complementarity of two branches in our mixed encoding unit, we conducted comparison experiments on CIFAR-100 [24] using EfficientNet-B0 [19] and REGNETY-800MF [20]. The experimental configurations were the same as those described in Section 5.1. The results were listed in Table 2. On EfficientNet-B0, significant performance boosted when both branches were used jointly. On REGNETY-800MF, the joint use of both branches outperformed the only use of MLP and almost matched the only use of channel grouping. Therefore, the fusion of these two methods always performed more robustly on different models.

5. Experiments

In this section, we evaluate the proposed methods on large-scale image classification, object detection using CIFAR-100 [24] and PASCAL VOC [25], respectively. Specifically, we first compare with state-of-the-art counterparts on CIFAR-100 using EfficientNet [19] and REGNET [20]. Then, we verify the effectiveness of our proposed methods on PASCAL VOC using SSDlite [26] and RetinaNet [27].

The hyperparameter G in our GSACA was set to $C/16$ for EfficientNet, g for REGNET, $C/32$ for SSDlite and $C/32$ for RetinaNet, respectively. The hyperparameters G and r in our MECA were set to $G = C/16, r = 8$ for EfficientNet, $G = g, r = 8$ for REGNET, $G = C/32, r = 8$ for SSDlite and $G = C/32, r = 18$ for RetinaNet, respectively.

5.1. Implementation details

To evaluate our proposed methods on CIFAR-100 [24] classification, we employed two widely used CNNs as backbone models, including EfficientNet [19] and REGNET [20]. The parameters of networks were optimized using Lion [28] with layer adaption [29] and mini-batch size of 128. All models were trained from scratch for 80 epochs. The initial learning rate was set to $2e{-}3$ for EfficientNet-B0 and EfficientNet-B4, $2e{-}3$ for REGNETY-800MF and REGNETY-1.6GF, respectively. If $\text{step} < 390$, the learning rate scaling factor linearly increased from 0 to 1. If $390 \leq \text{step} < 21\,450$, the learning rate scaling factor decayed inversely with step from 1 to 0.1. If $\text{step} \geq 21\,450$, the learning rate scaling factor remained constant at 0.1.

We further evaluated our methods on PASCAL VOC [25] using SSDlite [26] and RetinaNet [27]. For training SSDlite, the network was trained from scratch for 80 epochs using Adam [30] with mini-batch size of 128. The initial learning rate was set to $2e{-}3$. If $\text{step} < 133$, the learning rate scaling factor linearly increased from 0 to 1. If $133 \leq \text{step} < 7315$, the learning rate scaling factor decayed inversely with step from 1 to 0.1. If $\text{step} \geq 7315$, the learning rate scaling factor remained constant at 0.1. For training RetinaNet, the network was trained from scratch for 40 epochs using Adam with mini-batch size of 64. The initial learning rate was set to $2e{-}3$. If $\text{step} < 267$, the learning rate scaling factor linearly increased from 0 to 1. If $267 \leq \text{step} < 7209$, the learning rate scaling factor decayed inversely with step from 1 to 0.1. If $\text{step} \geq 7209$, the learning rate scaling factor remained constant at 0.1.

In each convolution layer and fully connected layer, the weight was initialized with truncated Xavier normal distribution [31,32], the bias if having was initialized with zeros. In each batch normalization [33] layer, the $\boldsymbol{\gamma}$ was initialized with ones and the $\boldsymbol{\beta}$ if having was initialized with zeros. In each GCT [8] method, $\boldsymbol{\alpha}$ was initialized with ones, $\boldsymbol{\gamma}, \boldsymbol{\beta}$ were initialized with zeros. In our GSACA method, $\boldsymbol{W}$ and $\boldsymbol{\Theta}$ were initialized with truncated Xavier normal distribution. In our MECA method, $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{W}_3$ and $\boldsymbol{\Theta}$ were initialized with truncated Xavier normal distribution.

All of experimental tasks were accomplished on one computer. Its configurations were as follows: CPU Intel(R) Core(TM) i7-11700K, GPU Nvidia RTX A5000, RAM 32 GB, Python 3.11.2, PyTorch 2.0.1, Cudatoolkit 11.8 and Cudnn 8.9.

5.2. Image classification

To evaluate the influence of our methods, we first performed experiments on the CIFAR-100 dataset. This dataset was composed of colored natural images with 32×32 pixels and 100 categories with 600 images each category. For training on CIFAR-100, we used standard data augmentation of random color space transformation and random horizontal flip. We reported the top-1 error and top-5 error as the evaluation metrics.

Integration with modern architectures. SE has become a basic component of EfficientNet and REGNETY. To study the integration of different attention methods with modern network architectures, we chose EfficientNet and REGNETY as benchmark networks, and replaced the original SE in the network with different attention methods. The results were reported in Table 3. When using different attention methods to replace the original SE in EfficientNet-B0, our MECA achieved the best top-1 error and reduced the top-1 error by 1.43% compared to SE, and our GSACA achieved the third-best top-1 error and reduced the top-1 error by 0.48% compared to SE. Our MECA achieved the second-best top-5 error and reduced the top-5 error by 0.52% compared to SE, and our GSACA increased the top-5 error by 0.21% compared to SE. When using different attention methods to replace the original SE in RegNETY-800MF, our GSACA achieved the best top-1 error and reduced the top-1 error by 1% compared to SE, and our MECA achieved the second-best top-1 error and reduced the top-1 error by 0.28% compared to SE. Our GSACA achieved the best top-5 error and reduced the top-5 error by 1.06% compared to SE, and our MECA achieved the third-best top-5 error and reduced the top-5 error by 0.34% compared to SE. It is worth noting that CAG did not perform dimensionality reduction, which theoretically can achieve optimal results, but it did not, possibly due to underfitting during training. Moreover, integrating CAG into modern network architectures added too much model complexity, which is not appropriate. Additionally, when our GSACA and MECA methods integrated into REGNETY could achieve better results than when integrated into EfficientNet. Overall, our methods could outperform SE while adding less model complexity.

5.3. Object detection

We next conducted experiments on the PASCAL VOC 2007/2012 [25]. This dataset was composed of 20 classes. The PASCAL VOC

Table 3
Comparison results between different channel attention methods on CIFAR-100 using different modern architectures. The **bold** denotes the best result under the same backbone network. The underline denotes the second best result under the same backbone network.

Model	Top-1 err. (%) ↓	Top-5 err. (%) ↓	Param. (M) ↓	FLOPs (G) ↓
EfficientNet-B0 [19] + none	38.43	13.92	3.499	0.119
+ SE [4]	38.24	13.63	4.126	0.120
+ GSoP [5]	39.82	14.14	5.132	0.132
+ SRM [6]	42.39	15.14	3.535	0.119
+ ECA [7]	38.62	14.09	3.499	0.120
+ GCT [8]	37.95	13.71	3.526	0.119
+ FCA [9]	39.22	14.49	4.126	0.120
+ CAG [12]	37.26	**12.75**	18.542	0.135
+ SFNEA [34]	39.15	14.30	3.499	0.119
+ GSACA (ours)	37.76	13.84	3.929	0.120
+ MECA (ours)	**36.81**	13.12	4.113	0.120
REGNETY-800MF [20] + none	42.06	16.33	4.880	0.266
+ SE [4]	41.32	15.75	5.720	0.267
+ GSoP [5]	44.01	17.09	7.370	0.292
+ SRM [6]	41.85	16.57	4.898	0.266
+ ECA [7]	42.18	16.43	4.880	0.266
+ GCT [8]	41.90	16.20	4.894	0.266
+ FCA [9]	43.07	16.91	5.720	0.267
+ CAG [12]	41.70	15.15	9.109	0.270
+ SFNEA [34]	46.08	18.58	4.880	0.266
+ GSACA (ours)	**40.32**	**14.69**	5.102	0.266
+ MECA (ours)	41.04	15.41	5.584	0.267

2012trainval dataset has 11,530 images containing 27,450 ROI annotated objects. We trained all detectors on VOC 2012trainval and evaluated the results on VOC 2007test for comparison. For training on PASCAL VOC, we used standard data augmentation of random color space transformation and random horizontal flip. We reported the metrics of COCO API as the evaluation metrics. The results were reported in Table 4. When using different attention methods to replace the original SE in the backbone network MobileNetV3-Large for SSDlite, integration of either SE, GCT, our GSACA or our MECA could improve performance of object detection by a margin. Meanwhile, our GSACA outperformed SE by 0.7% in terms of AP, 1.4% in terms of AP@50 and 0.5% in terms of AP@75 while using less parameter complexity. Our MECA outperformed SE by 0.4% in terms of AP, 1.1% in terms of AP@50 and 0.1% in terms of AP@75 while using lower parameter complexity. In particular, our methods achieved more gains for small objects, which were usually more difficult to be detected. When using different attention methods to integrate into the backbone network ResNet50 for RetinaNet50, integration of either SE, GSoP, ECA, CAG, our GSACA or our MECA could improve performance of object detection by a margin. Our GSACA achieved the best detection results. Our MECA achieved similar detection results with GSoP while using less model complexity. Our GSACA outperformed SE by 0.7% in terms of AP, 1.4% in terms of AP@50 and 0.4% in terms of AP@75 while using less parameter complexity. Our MECA outperformed SE by 0.2% in terms of AP and 0.8% in terms of AP@50 while using less parameter complexity.

6. Ablation studies

This section conducted ablation experiments to gain a better understanding of the relative importance of components in the our GSACA or MECA method.

6.1. Different pooling methods

To evaluate the influence of different pooling methods, we performed ablation experiments on CIFAR-100 [24] using EfficientNet-B0 [19] and REGNETY-800MF [20]. Optimization was performed using stochastic gradient descent (SGD) [35–37] with momentum 0.9 and a mini-batch size of 128. All models were trained for 80 epochs from scratch. The initial learning rate was set to 0.2. If $\text{step} < 390$, the learning rate scaling factor linearly increased from 0 to 1. If $390 \leq \text{step} < 21\,450$, the learning rate scaling factor decayed inversely with step from 1 to 0.01. If $\text{step} \geq 21\,450$, the learning rate scaling factor remained constant at 0.01. The results were reported in Table 5. While all GAP, GMP and GHP were effective, our GHP consistently achieved better performance on different models.

6.2. Standard MLP vs. our GSMLP

For fair comparison between our GSAMLP and the standard MLP, we conducted ablation experiments on CIFAR-100 [24] using ResNet50 [11]. Optimization was performed using Adam [30] a mini-batch size of 128. All models were trained for 80 epochs from scratch. The initial learning rate was set to $2e{-}3$. If $\text{step} < 390$, the learning rate scaling factor linearly increased from 0 to 1. If $390 \leq \text{step} < 21\,450$, the learning rate scaling factor decayed inversely with step from 1 to 0.1. If $\text{step} \geq 21\,450$, the learning rate scaling factor remained constant at 0.1. The results were reported in Table 6. Despite that the top-1 error of GSAMLP* was a little higher than that of the standard MLP when the hyperparameter in GSAMLP* was set to $G = C/32$ and the hyperparameter in SE was set to $r = 8$, the results of the other three error comparisons showed that GSAMLP* performed better than the standard MLP while using fewer parameters. For the two different hyperparameter settings, the top-1 error fluctuation of GSAMLP* was 0.08%, and the top-5 error fluctuation was 0.03%. The top-1 error fluctuation of the standard MLP was 0.77%, and the top-5 error fluctuation was 0.05%. Therefore, the performance of GSAMLP* is more robust for hyperparameter settings.

6.3. Standard MLP vs. our mixed encoding unit

For fair comparison between our mixed encoding unit and the standard MLP, we also conducted ablation experiments on CIFAR-100 [24] using ResNet50 [11]. The experimental configurations were the same as those described in Section 6.2. The results were reported in Table 7. Despite that the top-1 error of MECA* was a little higher than that of SE when the hyperparameters in MECA* were set to $r = 16, G = 32$ and the hyperparameter in SE was set to $r = 8$, the results of the other three error comparisons showed that MECA* performed better than SE while using fewer parameters. For the two different hyperparameter settings, the top-1 error fluctuation of MECA* was 0.21%, and the top-5 error fluctuation was 0.16%. The top-1 error fluctuation of SE was 0.77%, and the top-5 error fluctuation was 0.05%. Therefore, the performance of MECA* is more robust overall for hyperparameter settings.

Table 4
Comparison results between different channel attention methods on PASCAL VOC 2007test.

Model	AP (%) ↑	AP@50 (%) ↑	AP@75 (%) ↑	Param. (M) ↓	FLOPs (G) ↓
SSDlite320 [26] + none	14.2	28.4	12.3	1.650	0.463
+ SE [4]	14.6	28.9	13.3	2.466	0.465
+ GSoP [5]	13.3	25.8	12.0	4.109	0.598
+ SRM [6]	13.3	26.6	11.6	1.662	0.463
+ ECA [7]	14.0	28.2	12.1	1.650	0.464
+ GCT [8]	14.2	28.7	12.4	1.659	0.463
+ CAG [12]	13.9	27.5	12.6	4.907	0.467
+ SFNEA [34]	13.8	27.9	11.7	1.650	0.463
+ GSACA (ours)	**15.3**	**30.3**	**13.8**	1.954	0.463
+ MECA (ours)	15.0	30.0	13.4	2.398	0.463
RetinaNet50 [27]	12.8	25.5	11.5	14.542	25.035
+ SE [4]	14.0	27.6	12.6	15.799	25.046
+ GSoP [5]	14.0	28.5	12.5	18.016	25.439
+ SRM [6]	11.6	23.9	9.8	14.572	25.035
+ ECA [7]	13.9	27.9	12.3	14.542	25.045
+ GCT [8]	12.1	24.2	10.8	14.564	25.035
+ CAG [12]	13.6	26.9	12.5	24.602	25.055
+ SFNEA [34]	11.6	23.9	10.2	14.542	25.046
+ GSACA (ours)	**14.7**	**29.0**	**13.0**	15.267	25.037
+ MECA (ours)	14.2	28.4	12.5	15.727	25.038

Table 5
Ablation comparison between different pooling methods in the SE unit.

Model	Top-1 err.	Top-5 err.
EfficientNet-B0 [19] + GAP	37.72	12.64
EfficientNet-B0 [19] + GMP	38.06	13.20
EfficientNet-B0 [19] + GHP	**36.61**	**12.23**
REGNETY-800MF [20] + GAP	44.71	17.59
REGNETY-800MF [20] + GMP	43.32	**16.63**
REGNETY-800MF [20] + GHP	**42.89**	16.83

Table 6
Ablation comparison between the standard MLP and our GSAMLP. GSACA* denotes just replacing GHP with GAP in the GSACA.

Model	Hyper.	Top-1 err.	Top-5 err.	Params	FLOPs
SE [4]	$r = 8$	**36.19**	12.59	7.253	16.327
GSACA* (ours)	$G = C/32$	36.36	**12.46**	6.720	16.326
SE [4]	$r = 16$	36.96	12.64	6.624	16.326
GSACA* (ours)	$G = C/16$	**36.15**	**12.43**	6.358	16.326

Table 7
Ablation comparison between SE and our MECA. MECA* denotes just replacing GHP with GAP in the MECA.

Model	r/G	Top-1 err.	Top-5 err.	Params	FLOPs
SE [4]	$r = 8$	**36.19**	12.59	7.253	16.327
MECA* (ours)	$r = 16$, $G = 32$	36.42	**12.25**	7.180	16.237
SE [4]	$r = 16$	36.96	12.64	6.624	16.326
MECA* (ours)	$r = 32$, $G = 16$	**36.34**	**12.41**	6.588	16.326

7. Discussion

From the second paragraph to the fifth paragraph in Section 4.2, we have analyzed the advantages and disadvantages of different channel information encoding methods. Based on these analyses, we aim to improve the use of channel grouping to encode channel information along two different approaches. One approach involves channel shuffling and sequential stacking channel grouping layers (similar to increasing the receptive field by means of stacking convolution layers), while the other approach involves encoding channel information through two separate branches. Although the solutions are different, their goals are the same, *i.e.*, to reduce information loss while ensuring that each output component contained the participation of all input components to better establish channel correlations.

The limitation of our current methods is that hyperparameters G and r are manually set based on the existing network architectures. In the future, can the optimal settings in different network architectures be determined using network automatic search or meta-learning?

8. Conclusion

In this paper, we present a novel multilayer perception named GSAMLP, a novel unit to encode channel information named mixed encoding unit, and a novel pooling named hierarchical pooling. We apply them to channel attention and further propose two novel channel attention methods named GSACA method and MECA method, respectively. Compared with the existing channel attention methods, our proposed methods not only can reduce information loss but also can better establish channel correlations. The experimental results show that our GSACA method almost consistently outperformed most existing channel attention methods and that our MECA method consistently outperformed the existing channel attention methods. Our proposed GSAMLP and hierarchical pooling can not only be applied in channel attention but also in network architecture design, and we expect to see more applications in the future. While SE has become a basic component in some state-of-the-art convolutional neural networks, our proposed GSACA and MECA can serve as effective replacements for SE units in these networks.

Although channel attention has been shown to be successful, it focuses on what to pay attention to, while spatial attention considers where to pay attention. Based on this observation, we encourage considering whether there can be a universal attention framework that takes advantage of all kinds of attention mechanisms. For example, a soft selection mechanism (branch attention) can choose among channel attention and spatial attention according to the specific task undertaken. Attention mechanisms are inspired by the human visual system and are a step towards building an interpretable computer vision system. Typically, attention-based models are understood by rendering attention maps. However, this can only give an intuitive feel for what is happening, rather than precise understanding. However, in applications where security or safety are important, such as medical diagnostics and automated driving systems, often have stricter requirements. Better characterizing how methods work, including modes of failure, is needed in such areas. Developing characterizable and interpretable attention models can make them more widely applicable.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the link to my data in the manuscript.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62076117 and 62166026) and the Jiangxi Key Laboratory of Smart City (Grant No. 20192BCD40002).

Appendix. Proof of logsumexp(.) being the smooth version of max(.)

Assume $\boldsymbol{x} = [x_1, x_2, \dots, x_N]$, logsumexp(.) is formulated as Eq. (A.1),

$$\text{logsumexp}(\boldsymbol{x}) = \log(\sum_{i=1}^{N} \exp(x_i)). \tag{A.1}$$

Assume $x_{\max} = \max(x_1, x_2, \dots, x_N)$, then Eq. (A.2) obviously holds,

$$\exp(x_{\max}) \le \sum_{i=1}^{N} \exp(x_i) \le N \exp(x_{\max}). \tag{A.2}$$

Take log(.) to Eq. (A.2), then Eq. (A.3) holds,

$$x_{\max} \le \text{logsumexp}(\boldsymbol{x}) \le x_{\max} + \log(N). \tag{A.3}$$

Assume $\tau > 0$, then Eq. (A.4) holds,

$$\begin{aligned} \frac{x_{\max}}{\tau} &\le \text{logsumexp}(\frac{\boldsymbol{x}}{\tau}) \le \frac{x_{\max}}{\tau} + \log(N), \\ x_{\max} &\le \tau \text{logsumexp}(\frac{\boldsymbol{x}}{\tau}) \le x_{\max} + \tau \log(N). \end{aligned} \tag{A.4}$$

Eq. (A.4) shows that logsumexp(.) can be infinitely close to max(.) by means of reducing the value of τ.

References

[1] S.Q. Abbas, L. Chi, Y.P.P. Chen, Transformed domain convolutional neural network for alzheimer's disease diagnosis using structural mri, Pattern Recognit. 133, http://dx.doi.org/10.1016/j.patcog.2022.109031.
[2] Q. Wang, Y. Zhong, W. Min, et al., Dual similarity pre-training and domain difference encouragement learning for vehicle re-identification in the wild, Pattern Recognit. 139 (2023) 1–11, http://dx.doi.org/10.1016/j.patcog.2023.109513.
[3] D. Gai, R. Feng, W. Min, et al., Spatiotemporal learning transformer for video-based human pose estimation, IEEE Trans. Circuits Syst. Video Technol. http://dx.doi.org/10.1109/TCSVT.2023.3269666.
[4] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, IEEE Trans. Pattern Anal. Mach. Intell. 42 (8) (2020) 2011–2023, http://dx.doi.org/10.1109/TPAMI.2019.2913372.
[5] Z. Gao, J. Xie, Q. Wang, et al., Global second-order pooling convolutional networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3024–3033.
[6] H.J. Lee, H.E. Kim, H. Nam, Srm: A style-based recalibration module for convolutional neural networks, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1854–1862.
[7] Q. Wang, B. Wu, P. Zhu, et al., Eca-net: Efficient channel attention for deep convolutional neural networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11534–11542.
[8] Z. Yang, L. Zhu, Y. Wu, et al., Gated channel transformation for visual recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11794–11803.
[9] Z. Qin, P. Zhang, F. Wu, et al., Fcanet: Frequency channel attention networks, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 783–792.
[10] V. Nair, G.E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Proceedings of the International Conference on Machine Learning, 2010, pp. 807–814.
[11] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[12] Y. Luo, X. Cao, J. Zhang, et al., Ce-fpn: Enhancing channel information for object detection, Multimedia Tools Appl. 81 (21) (2022) 30685–30704, http://dx.doi.org/10.1007/s11042-022-11940-1.
[13] D. Yu, H. Wang, P. Chen, Z. Wei, Mixed pooling for convolutional neural networks, in: Proceedings of the Rough Sets and Knowledge Technology, 2014, pp. 364–375.
[14] S. Woo, J. Park, J.Y. Lee, et al., Cbam: Convolutional block attention module, in: Proceedings of the European Conference on Computer Vision, 2018, pp. 3–19.
[15] J. Park, S. Woo, J.Y. Lee, et al., Bam: Bottleneck attention module, in: Proceedings of the British Machine Vision Conference, 2018.
[16] D. Misra, T. Nalamada, A.U. Arasanipalai, Q. Hou, Rotate to attend: Convolutional triplet attention module, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 3139–3148.
[17] N.N. Ma, X.Y. Zhang, H.T. Zheng, J. Sun, Shufflenet v2: Practical guidelines for efficient cnn architecture, in: Proceedings of the European Conference on Computer Vision, 2018, pp. 122–138.
[18] S. Xie, R. Girshick, P. Dollar, et al., Aggregated residual transformations for deep neural networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2017, pp. 1492–1500.
[19] M. Tan, Q.V. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: Proceedings of the International Conference on Machine Learning, 2019, pp. 6105–6114.
[20] I. Radosavovic, R.P. Kosaraju, R. Girshick, et al., Designing network design spaces, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10428–10436.
[21] R.K. Srivastava, K. Greff, J. Schmidhuber, Training very deep networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2015.
[22] J. Gehring, M. Auli, D. Grangier, et al., Convolutional sequence to sequence learning, in: Proceedings of the International Conference on Machine Learning, 2017, pp. 1243–1252.
[23] L. Wan, M. Zeiler, S. Zhang, et al., Regularization of neural networks using dropconnect, in: Proceedings of the International Conference on Machine Learning, 2013, pp. 1058–1066.
[24] A. Krizhevsky, G.E. Hinton, Learning multiple layers of features from tiny images, 2009, URL https://citeseerx.ist.psu.edu.
[25] M. Everingham, J. Winn, The pascal visual object classes challenge 2012 (voc2012) development kit, Pattern Anal., Stat. Model. Comput. Learn. 8 (2011).
[26] A. Howard, M. Sandler, G. Chu, et al., Searching for mobilenetv3, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1314–1324.
[27] T.Y. Lin, P. Goyal, R. Girshick, et al., Focal loss for dense object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 017, pp. 2980–2988.
[28] X. Chen, C. Liang, D. Huang, et al.,
[29] R. Tian, A.P. Parikh, Amos: An adam-style optimizer with adaptive weight decay towards model-oriented scale, 2022, URL https://arxiv.org/abs/2210.11693.
[30] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of the International Conference on Machine Learning, 2015.
[31] D.R. Barr, E.T. Sherrill, Mean and variance of truncated normal distributions, Amer. Statist. 53 (4) (1999) 357–361, http://dx.doi.org/10.1080/00031305.1999.10474490.
[32] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.
[33] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: Proceedings of the International Conference on Machine Learning, 2015, pp. 448–456.
[34] H. Sun, B. Li, Z. Dan, et al., Multi-level feature interaction and efficient non-local information enhanced channel attention for image dehazing, Neural Netw. 163 (2023) 10–27, http://dx.doi.org/10.1016/j.neunet.2023.03.017.
[35] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat. 23 (3) (1951) 400–407.
[36] J. Kiefer, J. Wolfowitz, Stochastic estimation of the maximum of a regression function, Ann. Math. Stat. 23 (3) (1952) 462–466.
[37] I. Sutskever, J. Martens, G. Dahl, et al., On the importance of initialization and momentum in deep learning, in: Proceedings of the International Conference on Machine Learning, 2013, pp. 1139–1147.

Meng Zhu received the B.E. and M.E. degrees in computer science and technology from Nanchang University, China in 2018 and 2021, respectively. He is currently pursuing the Ph.D. degree at Nanchang University, China. His current research interests include computer vision, and natural language processing.

Weidong Min received the B.E., M.E. and Ph.D. degrees in computer application from Tsinghua University, China in 1989, 1991 and 1995, respectively. He is currently a Professor and the Dean, Institute of Metaverse, Nanchang University, China. He is an Executive Director of China Society of Image and Graphics. His current research interests include image and video processing, artificial intelligence, big data, distributed system, and smart city information technology.

Junwei Han received the B.S. and Ph.D. degrees from Northwestern Polytechnical University, Xi'an, China, in 1999 and 2003, respectively. He is currently a Full Professor with the School of Automation, Northwestern Polytechnical University, Xi'an, China. His current research interests include computer vision, multimedia processing, and brain imaging analysis. He is an Associate Editor of *IEEE Trans. Pattern Anal. Mach. Intell.*, *IEEE Trans. Neural Netw. Learn. Syst.*, *IEEE Trans. Multimedia*, and *etc.*.

Qing Han received the B.E. and M.E. degrees in computer application from Tianjin Polytechnic University, China in 1997 and 2006, respectively. She is currently an associate professor at School of Mathematics and Computer Science, Nanchang University, China. Her current research interests include image and video processing, and network management.

Shimiao Cui received the B.E. degree in computer science and technology from Nanchang University, China in 2021. He is currently pursuing the M.E. degree at Nanchang University, China. His current research interests include computer vision, and deep learning.