Journal on Internet of Things
DOI: 10.32604/jiot.2022.025327
Article

Tech Science Press

HDAM: Heuristic Difference Attention Module for Convolutional Neural Networks

Yu Xue* **and Ziming Yuan**

School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing, 210044, China
*Corresponding Author: Yu Xue. Email: xueyu@nuist.edu.cn
Received: 15 November 2021; Accepted: 15 February 2022

Abstract: The attention mechanism is one of the most important priori knowledge to enhance convolutional neural networks. Most attention mechanisms are bound to the convolutional layer and use local or global contextual information to recalibrate the input. This is a popular attention strategy design method. Global contextual information helps the network to consider the overall distribution, while local contextual information is more general. The contextual information makes the network pay attention to the mean or maximum value of a particular receptive field. Different from the most attention mechanism, this article proposes a novel attention mechanism with the heuristic difference attention module (HDAM). HDAM's input recalibration is based on the difference between the local and global contextual information instead of the mean and maximum values. At the same time, to make different layers have a more suitable local receptive field sizes and increase the flexibility of the local receptive field design, we use genetic algorithm to heuristically produce local receptive fields. First, HDAM extracts the mean value of the global and local receptive fields as the corresponding contextual information. Then the difference between the global and local contextual information is calculated. Finally, HDAM uses this difference to recalibrate the input. In addition, we use the heuristic ability of genetic algorithm to search for the local receptive field size of each layer. Our experiments on CIFAR-10 and CIFAR-100 show that HDAM can use fewer parameters than other attention mechanisms to achieve higher accuracy. We implement HDAM with the Python library, Pytorch, and the code and models will be publicly available.

Keywords: Attention mechanism; convolutional neural network; genetic algorithm

1 Introduction

Convolutional Neural Networks (CNNs) [1] have achieved amazing development in the past 10 years. Due to the efficient representation, CNNs have achieved remarkable results in multiple downstream tasks, such as classification [2], detection [3] and segmentation [4]. Therefore, efforts to improve representation capabilities have never stopped. For example, in the early days of CNNs,

researchers found that the depth [5] of the network has a great impact on the performance of CNNs, because the deeper the network, the richer the high-dimensional information of the network. However, if the network reaches a certain depth, not only the computational cost of the network increases, and the inference is prolonged, but the performance of the network is severely degraded due to the vanishing or explosion of the gradient [6]. In addition to depth that affects the performance of the network, the width of the network is also an important factor that affects the performance of the network. Similar to increasing the depth, increasing the width [7,8] can also improve the representation ability of the network. But it also increases computing consumption and extends the inference time. Cardinality [9] diversifies the style of convolution in the same layer. This method can significantly improve the network without increasing parameters or adding a small number of parameters. All the above are to improve network performance by stacking, and skip connection [10] is to change the way of information transmission whose advantage is that no additional parameters are required, the gradient explosion and vanishing are solved at the same time, and the convergence speed of the network is improved. However, additional storage space is needed to store the skip-connected part during inference. Although the above methods can improve the performance of CNNs to a certain extent, they consume a lot of computing resources such as memory and floating-point calculations. The attention mechanism can improve CNNs with a small number of parameters and zero additional storage requirements. More importantly, the attention mechanism simulates the human visual system, that is, pay more attention to meaningful information rather than meaningless information.

Among them, SENet [11] is one of the most representative attention networks. SENet [11] proposes a channel attention mechanism and calculates the global average of each channel, which is used as contextual information (CI) to recalibrate the input. This can magnify the global feature of each channel. On this basis, CBAM [12] and BAM [13] additionally consider the spatial attention, and add the global maximum value to enrich the global CI, so that the network can find what and where to focus on more accurately. To make this attention more general, GENet [14] extracts local CI instead of global CI. Based on GENet [14], SPANet [15] extracts local CI from different spatial scales.

These attention strategies use few parameters to enhance the performance of CNNs. They recalibrate input pixels by multiplying the input pixels with global or local CI embedding, and integrate the CI into the network's information flow. Therefore, most attention mechanisms use either global CI or local CI. We review the meaning of global and local CI and conclude that global CI represents the average value of the entire image, reflecting the trend of the overall pixel value; the local CI describes the average value of the local receptive field, and represents the average value of the pixel values in a small area of the sample. The two are different, and the animal's visual system pays attention to this difference. The difference in color distribution between objects is a prerequisite for the observer to distinguish and pay special attention. And today's various attention strategies do not take this into consideration. Therefore, this paper proposes a novel attention strategy based on the difference between global and local CI, and this attention module is termed Heuristic Difference Attention Module (HADM). At the same time, to design a more reasonable local receptive field, we adopt a heuristic strategy, that is, to introduce genetic algorithm (GA) [16] to perform a heuristic search for the size of the local receptive field.

Specifically, we first extract global and local CI, obtain their embedding through the shared multi-layer perceptron (MLP) [17] of two layers, calculate the difference between global and local CI according to embedding, and use this difference recalibrate the input. At the same time, we encode the combination of local receptive field sizes of all layers in the network, and search for the best network local receptive field size combination through GA [16]. We validate HDAM on CIFAR-10 [18] and CIFAR-100 [18], and used accuracy, number of parameters as the measurement standards. The

results show that HDAM surpasses various current state-of-the-art network models. These networks include classic networks, attention networks, and networks based on neural network architecture search (NAS) [19,20].

2 Related Work

In this part, we will introduce the work related to HDAM from two aspects: Convolutional neural network and Attention mechanism.

2.1 Convolutional Neural Network

In the first decade of the 21st century, limited by hardware equipment, the development of CNNs has been at a low ebb. With the gradual increase in computing power, and due to the success of AlexNet [21] in 2011, the development of CNNs enteredthe spring. Since then, CNNs have been the main backbone of computer vision and made remarkable achievements. After AlexNet, researchers continued to improve the performance of CNNs. GoogleNet [8] and VGG [7] increased the depth of CNNs, and found that depth is an important factor affecting the performance of CNNs. However, the training of the model needs to be carefully designed, such as the initialization and learning rate settings, otherwise it is dificult to achieve the desired performance. Batch normalization (BN) [22] believes that this is because the convolutional layer in the model fits the input whose distribution is changing in each inference, that is, the input produces an internal covariate shift, so it proposes to normalize the data of each batch. This makes the training of the model easier and the performance is compelling. Although BN can make training easier, the explosion and vanishing of gradient caused by the increase in depth still affect the potential of CNNs. Therefore, the skip connection proposed by ResNet [10] solves this problem by a big margin, because it alleviates the gradient accumulation consequence caused by the chain rule. ResNet [10] provides an efficient network topology template for later CNNs design. In addition to depth, WideResNet [5] based on ResNet [10] believes that expanding the width is also an effective means to improve CNNs. Depth and width are important hyperparameters that affect the performance of CNNs. Besides, the convolution operation affects the performance of CNNs from another perspective. The depthwise separable convolution [23] uses fewer parameters to achieve similar accuracy to the general convolution. This type of convolution is mainly used on mobile devices. ResNeXt [9] uses multiple convolutions of different sizes in the same convolutional layer. Also, without using additional parameters or using few additional parameters, the performance of CNNs has been greatly improved. Different from the above methods, the current design of CNNs network is mainly focused on the performance improvement strategy of CNNs based on the attention mechanism. This paper also proposes a new type of attention network.

2.2 Attention Mechanism

The attention mechanism simulates how the animal visual system works, that is, paying attention to the more effective part. The performance of the model can be improved without increasing or increasing a few parameters. The attention mechanism mainly extracts the CI of the feature maps, and then multiplies CI back to the network to increase the network's sensitivity to this information. SENet [11] is a typical attention network. It extracts the result of global average pooling as CI. SPANet [15] and GENet [14] extract the local mean as the local CI, which makes the extraction method based on global CI more general. In addition to using the mean value as the CI, CBAM [12] and BAM [13] also use the maximum value as the component of the CI. Different from all existing attention mechanisms, we extract global and local CI at the same time, seek the difference between the two, and pass this difference back to the network. At the same time, to find the most suitable local receptive field size, we

use GA [16] based on heuristic search for the first time in the field of attention mechanism to generate the most suitable local receptive field size combination.

3 Proposed Algorithm

In this part, we discuss HDAM in detail. HDAM mainly includes four parts, namely global and local CI extraction, embedding and difference calculation, input recalibration, and best local receptive field search. To explain HDAM more accurately, we provide detailed formula derivation.

3.1 Contextual Information Extraction

CI extraction is an important operation of the attention mechanism. CI represents the concentration of a specific receptive field information and is the basis for embedding calculation.

We use the mean to represent the CI of the receptive field. First, we divide the input into non-overlapping patches, and each patch is a receptive field. We calculate the average value of the receptive field on each channel based on the channel, and use this as the CI in the receptive field on each channel. Given Input into $\in \mathbb{R}^{P\times C\times \widehat{H}\times \widehat{W}}$, where P means the number of local receptive fields (patches) and $\widehat{H}$ and $\widehat{W}$ denote the height and width of the patch. P equals $HW/\left(\widehat{H}\widehat{W}\right)$. With the addition of global receptive field, the final receptive field metric (RF) is $\in \mathbb{R}^{(P+1)\times C\times \widehat{H}\times \widehat{W}}$, then the CI is as follows:

$$CI = Mean\,(RF) \tag{1}$$

where *Mean*() calculates the mean of RF and CI is $\in \mathbb{R}^{P\times C}$. If RF is the global receptive filed, it means global CI, otherwise it means local CI.

3.2 Embedding and Difference Calculation

Embedding calculation maps the extracted *CI*. To control the number of parameters, we use two-layer shared MLP to map the extracted global and local *CI*. The ReLu [24] activation function is used after the first layer, and the softmax activation function is used after the second layer. Finally, we use cross entropy to calculate the difference between global embedding and local embedding as shown below (for clarity, bias is ignored):

$$Embedding = Softmax\,(W_2\,(ReLu\,(W_1\,(CI)))) \tag{2}$$

where *Embedding* is $\in \mathbb{R}^{P\times C}$ and W_1 and W_2 denote the two-layer MLP. *Embedding* is $\begin{pmatrix} Local\ Embedding \\ Global\ Embedding \end{pmatrix}$, where *Local Embedding* (*LE*) is $\in \mathbb{R}^{HW/\left(\widehat{H}\widehat{W}\right)\times C}$ and *Global Embedding* (*GE*) is $\in \mathbb{R}^{1\times C}$. The *difference coefficient* (*DC*) is calculated as follows:

$$DC = Crossentropy\,(GE, LE) \tag{3}$$

where *DC* is $\in \mathbb{R}^{HW/\left(\widehat{H}\widehat{W}\right)\times 1}$.

3.3 Recalibration

The recalibration is to multiply the difference coefficient with the input. This process makes the difference between the global and local CI flow into the network in the inference to enrich the subsequent feature processing, so that the gradient carries the difference information to optimize the network parameters.

We broadcast each DC obtained into a matrix with the same dimension and shape as its corresponding *local receptive filed*, and then the obtained matrix is multiplied by the *Input*. Finally, we

reshape the shape of *Output* into $\mathbb{R}^{C\times H\times W}$:

$$DC = Broadcast\,(DC)$$
$$Output = Input \times DC \tag{4}$$
$$Output = Reshape\,(Output)\,.$$

3.4 Local Receptive Filed Search

In this part, we will elaborate on the working principle of GA's heuristic search in local receptive field design. To facilitate our explanation, we use ResNet-50 [10] as the basic model for our explanation. As we all know, ResNet-50 [10] consists of four units, and each unit consists of several residual blocks. The number of residual blocks in each unit is three, four, six, and three, respectively. We only design HDAM on the input of each residual block. The input special sizes of all residual blocks in each of these four units are 16, 16, 8, and 4. Taking the first unit as an example, because the input special size of each residual block is 16, the range of the local receptive field size of each block can be [1/16, 1/8, 1/4, 1/2, 0], where the number represents the proportion of the input special size, 0 means that HDAM is not used, and the range of the local receptive field in the remaining units are [1/16, 1/8, 1/4, 1/2, 0], [1/8, 1/4, 1/2, 0] and [1/4, 1/2, 0].

We use an array with a length of 16, that is, the sum of the number of blocks in all units, to represent the local receptive field size combination of all patches in ResNet-50 [10]. Fig. 1 is an example:

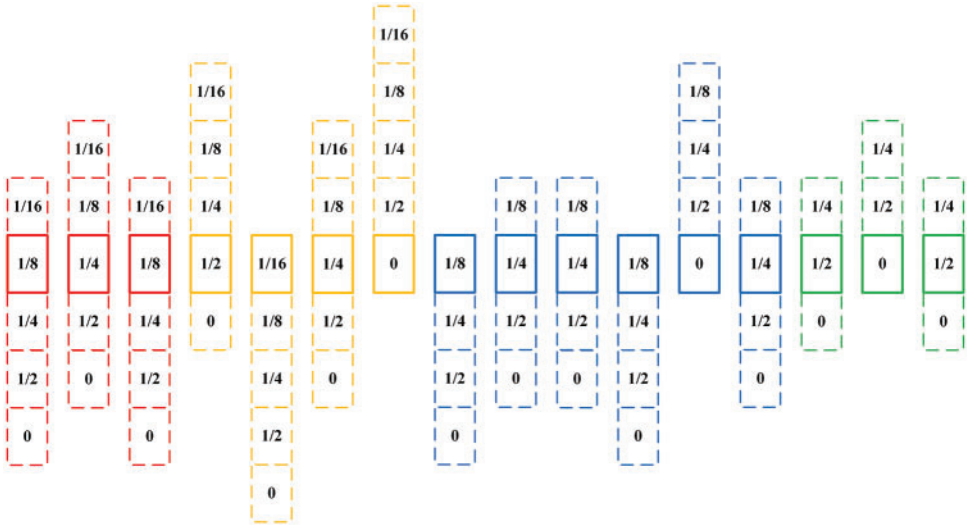


Figure 1: An encoded individual. The numbers in the dashed lines indicate the other size options

We represent a combination of local receptive fields as an individual and use GA [16] to search for the best combination. Algorithm 1 shows the entire process of GA [16].

Algorithm 1: Local receptive field search

Require: The population size N, the maximal generation number T, the crossover probability μ, the mutation of probability ν.

1:$P_0 \leftarrow$ Initialize N arrays as a population using encoding strategy;
2: Decode each individual and generate the corresponding CNN (ResNet-50 [10]). Train and validate each CNN, then take the highest accuracy as the fitness of each individual in P_0;
3: $t = 0$;
4: **while** $t < T$ **do**
5: $Q_t \leftarrow \emptyset$;
6: **while** $|Q_t| < N$ **do**
7: $p_1, p_2 \leftarrow$ Select two arrays from P_t with binary tournament selection;
8: $q_1, q_2 \leftarrow$ Generate two arrays by q_1 and q_2 by crossover operation with the probability μ, and mutation operation with the probability ν;
9: $Q_t \leftarrow Q_t \cup q_1 \cup q_2$;
10: **end while**
11: Train and evaluate CNNs' performance in Q_t;
12: $P_{t+1} \leftarrow$ Select N arrays from $P_t \cup Q_t$ by environmental selection;
13: $t \leftarrow t + 1$;
14: **end while**

Ensure: The architecture of a ResNet-50 [10] with the best combination of local receptive fields.

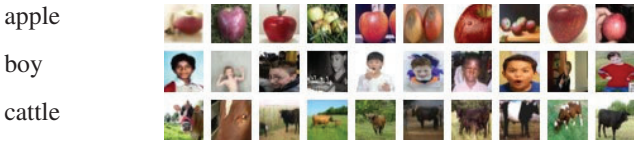
Before GA process, N denotes the population size, T denotes the maximal generation number, and the crossover and mutation probability are μ and ν, respectively. First, a population needs to be initialized. We encode the combination of a local receptive field size of a CNN as an array. We repeat this procedure N times to generate an initial population P_0. Second, we decode the individuals in the initial population to N CNNs. Then those CNNs are trained and evaluated and use the highest accuracy of each CNN on the validation dataset as its corresponding individual fitness value. Third, initialize a generation counter t to 0 and an empty set Q_t is initialized. Two individuals are selected as the parents from the population by binary tournament. Generate a number between 0 and 1 and if the number is less than μ, conduct the crossover operation on the two parent individuals. Determine in the same way whether to perform a mutation. After crossover and mutation, two offspring individuals are generated and merge them with Q_t. Repeat this process before the size of Q_t reaches N. Fourth, decode, train, and evaluate the individuals to obtain their fitness. Merge Q_t and P_t, then select N individuals from them by environmental selection to generate P_{t+1} and t increases by 1. Repeat these procedures until T generations.

4 Experiment Design

4.1 Dataset

We conduct our experiments on the two most popular datasets, CIFAR-10 [18] and CIFAR-100 [18]. The CIFAR [18] dataset is collected by Krizhevsky et al. and is divided in two subsets including CIFAR-10 [18] and CIFAR-100 [18] according to the number of categories. Each subset contains 60,000 images with the of 32 32, including 50,000 training images and 10,000 test images. The difference is that CIFAR-10 [18] contains 10 categories of images, each with 6,000 images, of which 5,000 are used for training and 1,000 are used for testing; CIFAR-100 [18] contains 100 categories, each with 600 images, of which 500 are used for training, 100 are used for testing. Fig. 2 shows the example of the CIFAR-10 and CIFAR-100.

(a) CIFAR-10 [18]



(b) CIFAR-100 [18]

Figure 2: Example of CIFAR-10 [18] and CIFAR-100 [18]

4.2 Peer Competitor

To illustrate the superior performance of HDAM, we select a variety of different CNNs models for comparison, including the classic CNNs, the CNNs searched by NAS, and the CNNs with mainstream attention mechanism. The CNN architectures searched by NAS includes the ones searched by semi-automatic NAS and fully automatic NAS. The classic CNN structure includes DenseNet [25], Maxout [26], VGG [7], Network in Network [27], Highway Network [28], All-CNN [29] and FractalNet [30]. The structures searched by semi-automatic NAS include Genetic CNN [31], EAS [32] and Block-QNN-S [33]. The structures found by the automatic search include Large-scale Evolution [34], CGP-CNN [35], NAS [36], MetaQNN [37] and AE-CNN [38]. CNNs based on the attention mechanism include SE-Net [11] and CBAM [12]. Except for the structure of CNNs based on the attention mechanism, we directly use their experimental results in the original paper, because these results are often the best. We retrain CNNs based on the attention mechanism.

4.3 Parameter Settings

We use ResNet-50 [10] as our basic model for embedding HDAM. According to the computing resource, two NVIDIA 2080TI graphic processing units (GPUs), we set the population size to 20 and the maximal generation to 20. The crossover and mutation probability are set to 0.9 and 0.2, respectively. We use the SGD with momentum as the optimizer. The momentum and weight decay are set to 0.9 and $5e-4$, respectively. A total of 250 epochs is set to train the individuals. The batch size is 128 and the learning rate is shown in Tab. 1. The training accuracy is recorded every 30 iterations in each epoch, and the test accuracy is recorded once in each epoch. Data augmentation includes random cropping, random horizontal flipping, and Cutout [39]. Random cropping fills four zeros on all borders of the image, and then randomly crops the image with a size of 32×32.

Table 1: Learning rate

Epoch	0∼50	50∼80	80∼120	120∼250
Learning rate	0.1	0.01	0.001	0.0008

5 Experiment Results

To void accidental factors, our experiments are conducted for 5 times, and the average value of these 5 times was taken as the final result. In addition to using accuracy as our evaluation index, the number of parameters is also used as one of the evaluation index. Tab. 2 shows the experimental results of HDAM on two datasets.

Table 2: Comparison between the proposed HDAM and the state-of-the-art peer competitors in terms of the classification accuracy, the number of parameters on the dataset CIFAR-10 [18] and CIFAR-100 [18]

Model	CIFAR-10 [18]	CIFAR-100 [18]	Parameter (M)	Style
DenseNet (k = 12) [25]	94.76	75.58	1.0	Hand-crafted
Maxout [26]	93.57	61.40	-	Hand-crafted
VGG [7]	93.34	71.95	20.04	Hand-crafted
Network in network [27]	91.19	64.32	-	Hand-crafted
Highway network [28]	92.28	67.61	-	Hand-crafted
All-CNN [29]	92.75	66.29	-	Hand-crafted
FractalNet [30]	94.78	77.70	38.6	Hand-crafted
Genetic CNN [31]	92.90	70.95	-	Semi-automatic
EAS [32]	95.77	-	23.4	Semi-automatic
Block-QNN-S [33]	95.62	79.35	6.1	Semi-automatic
Large-scale evolution [34]	94.60	-	5.4	Full-automatic
Large-scale evolution [34]	-	77.00	40.4	Full-automatic
CGP-CNN [35]	94.02	-	2.64	Full-automatic
NAS [36]	93.99	-	2.5	Full-automatic
MetaQNN [37]	93.08	72.86	-	Full-automatic
AE-CNN [38]	95.70	-	2.0	Full-automatic
AE-CNN [38]	-	79.15	5.4	Full-automatic
SE-ResNet-101 [11]	95.34	-	47.29	Attention
SE-ResNet-101 [11]	-	79.22	47.48	Attention
CBAM-ResNet-101 [12]	95.75	-	47.29	Attention
CBAM-ResNet-101 [12]	-	79.26	47.48	Attention
HDAM (Ours)	**96.10**	-	**23.65**	Attention
HDAM (Ours)		**79.79**	**23.83**	Attention

The first column of the table is the name of models, the second and third columns are the accuracy of CIFAR-10 [18] and CIFAR-100 [18] on each model, the fourth column is the number of parameters of each model, and the last column is the model category including hand-crafted, semi-automatic, and full-automatic. '-' means that the corresponding model has no public record.

The experimental results show that HDAM obtains the best accuracy of 96.10 on CIFAR-10 [18] and the highest accuracy of 79.79 on CIFAR-100 [18]. The accuracy of HDAM on CIFAR-10 [18] is 1.32 higher than the highest accuracy among hand-designed classic CNNs, 0.33 higher than the highest accuracy of CNNs generated by semi-automatic NAS, 0.4 higher than the highest accuracy of CNNs generated by full-automatic NAS and 0.35 higher than the highest accuracy of CNNs based on the attention mechanism. HDAM also obtains the highest accuracy of 79.79 on CIFAR-100 [18], which is 2.09 higher than the highest accuracy of hand-designed CNNs, 0.44 higher than the highest accuracy of CNNs generated by semi-automatic NAS, 0.64 higher than the highest accuracy generated by full-automatic NAS, and 0.53 higher than the highest accuracy of CNNs based on the attention mechanism. In terms of the number of parameters, HDAM has half the parameters of the attention network SE-ResNet-101 [11] and CBAM-ResNet-101 [12], which means that HDAM saves nearly half of the parameters and achieves higher performance.

6 Conclusion and Future Work

We propose a new attention mechanism module HDAM based on heuristics search and differences between the local and global CI. This module calculates global and local CI at the same time, but unlike any previous attention mechanism, HDAM does not use local or global CI to recalibrate the input, but calculates the difference between the two and recalibrates the input with the difference. In addition, to design a more reasonable local receptive field size, we first introduce heuristic search into the attention mechanism design. We encode the local receptive field of each convolutional layer into individuals, and use GA to search for the most suitable combination of local receptive fields. We use ResNet-50 as the base model to embed HDAM, and test HDAM on CIFAR-10 and CIFAR-100, respectively, and compare with four types of CNNs, including classic and state-of-the-art. The results show that HDAM surpasses all the above models on CIFAR-10 and CIFAR-100. Compared with the most popular attention mechanism-based models, HDAM can use nearly half of the parameters to obtain higher accuracy. For the future work, we will use weight inheritance to reduce the time spent searching for local receptive fields.

Acknowledgement: We have no contributors who do not meet the criteria for authorship.

Funding Statement: This work was partially supported by the National Natural Science Foundation of China (61876089, 61403206, 61876185, 61902281), the Opening Project of Jiangsu Key Laboratory of Data Science and Smart Software (No. 2019DS302), the Natural Science Foundation of Jiangsu Province (BK20141005), the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (14KJB520025), the Science and technology program of Ministry of Housing and Urban-Rural Development (2019-K-141), the Entrepreneurial team of sponge City (2017R02002), and the Priority Academic Program Development of Jiangsu Higher Education Institutions. (Corresponding Author: Yu Xue).

Conflicts of Interest: We give this statement to certify that all authors have seen and approved the manuscript being submitted and have no conflict of interest or personal relationships that could influence this paper.

References

[1] X. Wang, A. Bao, Y. Cheng and Q. Yu, "Multipath ensemble convolutional neural network," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 5, no. 2, pp. 298–306, 2021.

[2] H. Guo, S. Li, K. Qi, Y. Guo and Z. Xu, "Learning automata based competition scheme to train deep neural networks," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 2, pp. 151–158, 2020.

[3] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen *et al.,* "Deep learning for generic object detection: A survey," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 261–318, 2020.

[4] F. Sultana, A. Sufian and P. Dutta, "Evolution of image segmentation using deep convolutional neural network: A survey," *Knowledge-Based Systems*, vol. 201, pp. 106062, 2020.

[5] S. Zagoruyko and N. Komodakis, "Wide residual networks," arXiv preprint arXiv:1605.07146, 2016.

[6] T. Chen, Z. Zhang, X. Ouyang, Z. Liu, Z. Shen *et al.,* ""BNN-BN = ?": Training binary neural networks without batch normalization," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops*, Beijing, pp. 4619–4629, 2021.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Int. Conf. on Learning Representations*, San Diego, 2015.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed *et al.,* "Going deeper with convolutions," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Boston, pp. 1–9, 2015.

[9] S. Xie, R. Girshick, P. Dollar, Z. Tu and K. He, "Aggregated residual transformations for deep neural net-works," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Honolulu, Hawaii, pp. 1492–1500, 2017.

[10] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Las Vegas, pp. 770–778, 2016.

[11] J. Hu, L. Shen and G. Sun, "Squeeze-and-excitation networks," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Salt Lake City, pp. 7132–7141, 2018.

[12] S. Woo, J. Park, J. -Y. Lee and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. of the European Conf. on Computer Vision*, Munich, Germany, pp. 3–19, 2018.

[13] J. Park, S. Woo, J. -Y. Lee and I. S. Kweon, "BAM: Bottleneck attention module," in *European Conf. on Computer Vision*, Munich, Germany, pp. 3–19, 2018.

[14] J. Hu, L. Shen, S. Albanie, G. Sun and A. Vedaldi, "Gather-excite: Exploiting feature context in convolutional neural networks," arXiv preprint arXiv:1810.12348, 2018.

[15] J. Guo, X. Ma, A. Sansom, M. McGuire, A. Kalaani *et al.,* "Spanet: Spatial pyramid attention network for enhanced image recognition," in *IEEE Int. Conf. on Multimedia and Expo*, London, UK, pp. 1–6, 2020.

[16] S. Mirjalili, "Evolutionary algorithms and neural networks," *Evolutionary Algorithms and Neural Networks*, vol. 780, pp. 43–55, 2019.

[17] D. W. Ruck, S. K. Rogers and M. Kabrisky, "Feature selection using a multilayer perceptron," *Journal of Neural Network Computing*, vol. 2, no. 2, pp. 40–48, 1990.

[18] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[19] Y. Liu, Y. Sun, B. Xue, M. Zhang, G. G. Yen *et al.,* "A survey on evolutionary neural architecture search," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[20] Y. Xue, P. Jiang, F. Neri and J. Liang, "A multiobjective evolutionary approach based on graph-in-graph for neural architecture search of convolutional neural networks," *International Journal of Neural Systems*, vol. 31, no. 9, pp. 210035, 2021.

[21] A. Krizhevsky, I. Sutskever and G. E. Yuan Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, Lake Tahoe, Nevada, United States, vol. 25, pp. 1097–1105, 2012.

[22] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Int. Conf. on Machine Learning*, Lile, France, pp. 448–456, 2015.

[23] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1251–1258, 2017.

[24] X. Glorot, A. Bordes and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. of the Fourteenth Int. Conf. on Artificial Intelligence and Statistics*, Ft. Lauderdale, FL, USA, pp. 315–323, 2011.
[25] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Honolulu, Hawaii, pp. 4700–4708, 2017.
[26] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville and Y. Bengio, "Maxout networks," in *Int. Conf. on Machine Learning*, Atlanta, pp. 1319–1327, 2013.
[27] M. Lin, Q. Chen and S. Yan, "Network in network," in *Int. Conf. on Machine Learning*, Beijing, pp. 1–10, 2014.
[28] R. K. Srivastava, K. Greff and J. Schmidhuber, "Highway networks," in *Proc. of Int. Conf. on Learning Representations*, San Diego, 2015.
[29] J. T. Springenberg, A. Dosovitskiy, T. Brox and M. Riedmiller, "Striving for simplicity: The all convolutional net," in *Proc. of Int. Conf. on Learning Representations*, San Diego, 2015.
[30] G. Larsson, M. Maire and G. Shakhnarovich, "Fractalnet: Ultra-deep neural networks without residuals," in *Proc. of Int. Conf. on Learning Representations*, San Juan, Puerto Rico, pp. 1–11, 2016.
[31] L. Xie and A. Yuille, "Genetic cnn," in *Proc. of the IEEE Int. Conf. on Computer Vision*, Venice, Italy, pp. 1379–1388, 2017.
[32] H. Cai, T. Chen, W. Zhang, Y. Yu and J. Wang, "Efficient architecture search by network transformation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.
[33] Z. Zhong, J. Yan, W. Wu, J. Shao and C. L. Liu, "Practical block-wise neural network architecture generation," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Salt Lake City, pp. 2423–2432, 2018.
[34] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu *et al.,* "Large-scale evolution of image classifiers," in *Int. Conf. on Machine Learning*, Sydney, Australia, pp. 2902–2911, 2017.
[35] M. Suganuma, S. Shirakawa, and T. Nagao, "A genetic programming approach to designing convolutional neural network architectures," in *Proc. of the Genetic and Evolutionary Computation Conf.*, Berlin, Germany, pp. 497–504, 2017.
[36] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *Proc. of Int. Conf. on Learning Representations*, Toulon, France, pp. 1–16, 2017.
[37] B. Baker, O. Gupta, N. Naik and R. Raskar, "Designing neural network architectures using reinforcement learning," in *Proc. of Int. Conf. on Learning Representations*, Toulon, France, pp. 1–18, 2017.
[38] Y. Sun, B. Xue, M. Zhang and G. Yen, "Completely automated cnn architecture design based on blocks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 4, pp. 1242–1254, 2019.
[39] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," arXiv preprint arXiv:1708.04552, 2017.