Mathematical Biosciences and Engineering

MBE, 20(2): 2691–2715.
DOI: 10.3934/mbe.2023126
Received: 17 October 2022
Revised: 10 November 2022
Accepted: 14 November 2022
Published: 28 November 2022

http://www.aimspress.com/journal/MBE

Theory article

Neural architecture search based on dual attention mechanism for image classification

Cong Jin[1], Jinjie Huang[1,2,*], Tianshu Wei[1] and Yuanjian Chen[1]

[1] School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150006, China
[2] School of Automation, Harbin University of Science and Technology, Harbin 150006, China

* **Correspondence:** Email: huangjinjie163@163.com.

Abstract: Deep learning neural networks based on the manual design for image classification tasks usually require a large amount of a priori knowledge and experience from experts; thus, research on designing neural network architectures automatically has been widely performed. The neural architecture search (NAS) method based on the differentiable architecture search (DARTS) ignores the interrelationships within the searched network architecture cells. The optional operations in the architecture search space lack diversity, and the large parametric and non-parametric operations in the search space make the search process inefficient. We propose a NAS method based on a dual attention mechanism (DAM-DARTS). An improved attention mechanism module is introduced to the cell of the network architecture to deepen the interrelationships between the important layers within the architecture by enhancing the attention between them, which improves the accuracy of the architecture and reduces the architecture search time. We also propose a more efficient architecture search space by adding attention operations to increase the complex diversity of the searched network architectures and reduce the computational cost consumed in the search process by reducing non-parametric operations. Based on this, we further analyze the impact of changing some operations in the architecture search space on the accuracy of the architectures. Through extensive experiments on several open datasets, we demonstrate the effectiveness of the proposed search strategy, which is highly competitive with other existing neural network architecture search methods.

Keywords: neural architecture search; deep learning; image classification; DARTS; attention

1. Introduction

To date, deep learning has been widely used in many computer vision tasks, such as image classification [1–6], image recognition [7–9] and image segmentation [10–12]. However, deep neural networks based on artificial design, such as ResNet [13], DenseNet [14], Deeplab [15–17], GoogleNet [18–21], SENet [22], etc., require people to continuously do experiments using a large amount of prior knowledge and combine experiences to explore high-performance neural networks, which also consumes a lot of labor and time [23,24]. In addition, the designed neural networks are usually special because people have relatively limited knowledge and it is difficult to think outside of the inherent stereotypes. Thus, in recent years, many people have shifted their focus to how to design neural network architectures automatically, while also achieving network architectures with comparable performance to manually designed architectures on different tasks in various domains [25–27].

The research on automatic neural network architecture search (NAS) addresses three main aspects, namely, search strategy, search space and performance evaluation strategy. The search space that defines the search operations (e.g., convolution) required for a network architecture search and a rich and diverse search space can search for candidate architectures with higher complexity and better performance, and a more efficient search space also helps to simplify the search strategy and save search time. Redundant search operations can become a burden in the search process, which not only wastes a lot of search time, but it also affects the accuracy of the network architecture. The search space of a differentiable architecture search (DARTS) does not take into account the operations related to the attention mechanism and the impact of pooling operations on the architectural accuracy. We propose a new architectural search space that includes attention operations by adding two attention operations, i.e., the bottleneck attention module (BAM) [28] module and the convolutional block attention module (CBAM) [29]. We also eliminate the none operation from the DARTS search space, weigh the two operations of average pooling and maximum pooling and analyze them together with the more complex pooling operation. The new search space reduces the non-parametric operations in the network, improves the search accuracy and reduces the search time at the same time. The added search operations in the search space further improve the complexity of the searched network architecture and the overall performance of the network architecture.

Search strategies aim to automatically search for high-performance network architectures, and they are based on three main search strategies: evolutionary algorithms (EAs) [30–33], reinforcement learning (RL) [34–36] and DARTS [37–39]. The initial search strategies based on RL and EAs have shown significant results on classification tasks. But, they are computationally intensive and require thousands of GPUs, making them difficult to implement for most research workers. The search strategy based on differentiable weight sharing greatly reduces the computational cost. Therefore, in this paper, we adopt the differentiable weight sharing search strategy and use DARTS as the infrastructure for network architecture search. The DARTS method reduces the search time exponentially without reducing the network architecture accuracy by mapping the operation process into a directed acyclic graph (DAG) and converting the discrete search space into a continuous differentiable search space. However, the DARTS method ignores the interrelationship between different layers (nodes in the DAG) within the architecture. To address this problem, we add an attention module to generate a new cell of the network architecture, strengthen the interrelationship between layers and select the connected nodes and operations by different importance to improve the accuracy of the searched network architecture and further reduce the time spent on searching the network architecture. In addition, we

analyze the effect of changing the number of intermediate nodes, i.e., deepening the depth of the neural network, on the accuracy of the network architecture.

Performance estimation strategies are used to evaluate the performance of candidate network architectures and guide the search strategy to search for a neural network architecture with better performance. Many scholars have studied how to quickly converge the neural architecture to judge its performance, such as once-for-all (OFA) [40] network used sharing the weights of the small convolutional kernel with the large convolutional kernel, trained the large network architecture to convergence first and then shared the weights of the large network with the small network. The large convolutional kernel and the large network architecture provided good initialization for the small convolutional kernel and the small network architecture, accelerated the convergence of the network architecture and thus sped up the performance evaluation process. Many other works have attempted to speed up the estimation process. In this paper, we do not delve into performance evaluation strategies. We simply use a simple training of candidate architectures on a training dataset and then evaluate the architecture performance on a validation set. For evaluating the architecture performance metrics, we use deep learning performance metrics, i.e., training accuracy and testing accuracy metrics, to optimize the search strategy. In summary, we propose a dual attention mechanism-based neural architecture search method (DAM-DARTS). Our main contributions are as follows:

1) We propose a more efficient architecture search space that contains attention operations, reducing the redundant operations in the search space. The diversity of the space is increased, thus increasing the sophistication of the searched-out network architecture.

2) We introduce an improved attention module to the cell of the architecture, which enhances the interrelationship between the layers within the architecture by selecting the connected nodes with different weights, and improves the accuracy of the final searched network architecture.

3) We propose an architecture search method based on a dual attention mechanism (DAM-DARTS), which reduces the time required to search for the best architecture while improving the accuracy and scalability of the searched network architecture.

4) Through abundant comparative experiments on public datasets, the results show that our proposed search strategy is reliable and highly competitive with other state-of-the-art methods.

2. Related work

NAS aims to automatically search for a proper neural network architecture to replace the tedious manual design process, and it has received a lot of focus in recent years. In terms of search strategies, NAS can be divided into three types: RL, EAs and differentiable weight-sharing strategies. At an early stage, Zoph and Le [35] proposed to describe neural networks with recurrent neural networks (RNNs) and then trained the RNNs with RL and utilized 800 GPUs to search for 28 days to generate the final model. Zoph et al. [34] designed a NASNet search space and proposed to first search the cells for constructing the network architecture on a small dataset; they classified the cells into two classes: normal cells and reduction cells. These two types of cells were then stacked in a certain order and scaled to a larger dataset to train them from scratch, and the final model was searched on 500 GPUs for 4 days. The overall flowchart of the automatic search neural network architecture is shown in Figure 1. The MnasNet in [41] employs a hierarchical search strategy and considers the problem as a multi-objective optimization problem, using a proximal policy optimization (PPO) algorithm that traded off speed and accuracy and took 4.5 days to search for the final model on 64 TPUs; it further

improved the network performance. The SGBAN in [42] is a deep learning model for dynamic growth architectures based on a fully connected network (FCN); it progressively extends the design of FCNs to not only learn new data during the growth of the architecture, but it also retains the information obtained from previous datasets, solving the problem of catastrophic forgetting for NAS.

EAs mimic the process of biological evolution and are often used to solve multi-objective optimization problems [43,44]; many researchers use them for NAS in various fields, such as remote sensing [45,46], to reduce computational costs. The developers of AmoebaNet in [30] modified the EA by adding an age property to enhance the network's selection of younger candidate architectures, avoiding the premature scaling of candidates with high accuracy. They spent 7 days searching for the ideal architecture on 450 GPUs. The PNAS in [31] learned the performance of an agent model to predict a structure simultaneously based on a sequential search structure with increasing model complexity; it was slightly less accurate but saved 63 times the computational cost relative to AmoebaNet. A complex topology was introduced in [32] to represent the network structure search process in layers, and the final model was searched in 1.5 days using 200 GPUs in combination with an EA. A crow search algorithm combined with a binary representation was used in [47] to successfully reduce the number of GPUs to 10. Lu et al. [33] proposed to gradually reorganize and modify the convolution operations in the architecture using an EA method, optimized the search target by using the Pareto algorithm and gradually shrunk the network architecture by changing both the classification accuracy and FLOPS during the search process. They completed their experiments using 3.5 days on eight GPUs to further improve the network search efficiency.

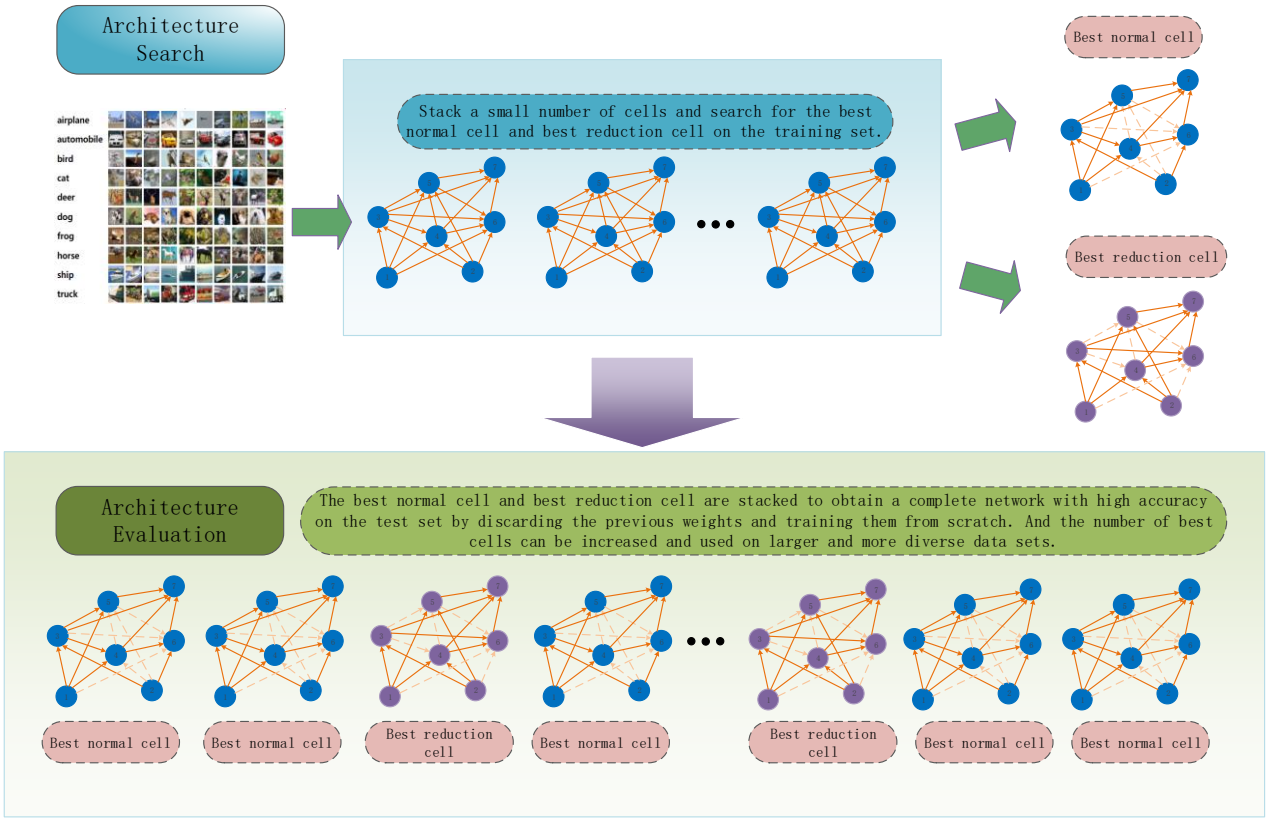


Figure 1. Overall flowchart of NAS.

Gradient-based differentiable weight-sharing search strategies further improve the network search efficiency compared to RL and EAs. The developers of the ENAS in [48] proposed the concept of weight sharing by representing the search space as a DAG and represented the operations in the search space as nodes in the DAG and the edges in the DAG as weights, thus significantly reducing the search time through weight sharing and searching for the best cell on a GPU in only 11.5 hours. In [37], a similar concept to DARTS was utilized, but the authors used edges in the DAG to represent all operations in the search space, while the nodes were the input and output feature images; they approximated the discrete operation space as a continuous differentiable operation space and surpassed the historical accuracy limit. The PC-DARTS proposed in [38] reduced the computational consumption by sampling 1/4 channels of the input image. The P-DARTS in [39] gradually reduced the search space during the search process and increased the number of basic unit stacks in stages to reduce the accuracy difference between training and evaluation when scaling to large datasets. The developers of FairDARTS in [49] addressed the problem of the overselection of skip connect operations by proposing the use of a sigmoid function instead of Softmax to allow multiple operations to be selected; they transformed each operation in the search space from a competitive to a cooperative relationship so that each operation was selected more fairly. The ProxylessNAS in [50] performs an architectural search directly on big data by path pruning, and the same idea of pruning was applied to speed up the search in [51,52].

In summary, NAS methods all involve a large amount of computation and thus have higher requirements for hardware devices. Many RL-based and EA-based methods have achieved desirable results, but the high number of GPUs required has resulted in the methods not being reproducible, and most scholars are unable to conduct further research on them. Although DARTS lowers the threshold for studying NAS compared to other search strategies, there is still a performance gap between the discretized sub-network and the super-network. The sub-network needs to be trained from scratch, and the search process and results may be unstable. On the contrary, RL- and EA-based search strategies are a bit more stable, and multi-objective EA-based search strategies are commonly used to handle tasks on mobile devices, which may better solve the latency problem. For now, the comparison between NAS algorithms cannot be done fairly because the search space and evaluation strategies of different search strategies may differ, so there is no more objective way to compare the performance of each search strategy. We chose the gradient-based approach for our study because of our limited experimental equipment.

Global attention mechanisms and local attention mechanisms were proposed in [53]. The channel attention mechanism was introduced in the SENet in [22], and then the spatial attention mechanism and the channel attention mechanism were combined in the literature [28,29]. The developers of the ATT-DARTS method in [54] proposed a new attentional search space in which the searched attentional operations are concatenated with the operations searched in the original operation space in each edge; it improved the accuracy but also consumed exponentially more time. The developers of the ASM-NAS in [55] proposed an architectural self-attention module, which they added before the output node to select the operations in the search space according to the different weights. In the literature [56], CNAS was added to the squeeze-and-excitation (SE) module, and a channel shuffle convolution module was added to the operation space. A new regularization technique called batch contrastive regularization has been proposed in [57] to improve generalization performance; it alleviates the overfitting problem in deep neural networks. In addition, many network architectures can be worth exploring in various fields, such as the medical field. However, medical images may require image

preprocessing, such as wavelet denoising as in [58] and the method of compressed sensing in [59] for image denoising and image recovery. The processed images may be more conducive to searching for network architectures with higher performance, and these studies are well worth exploring in the future.

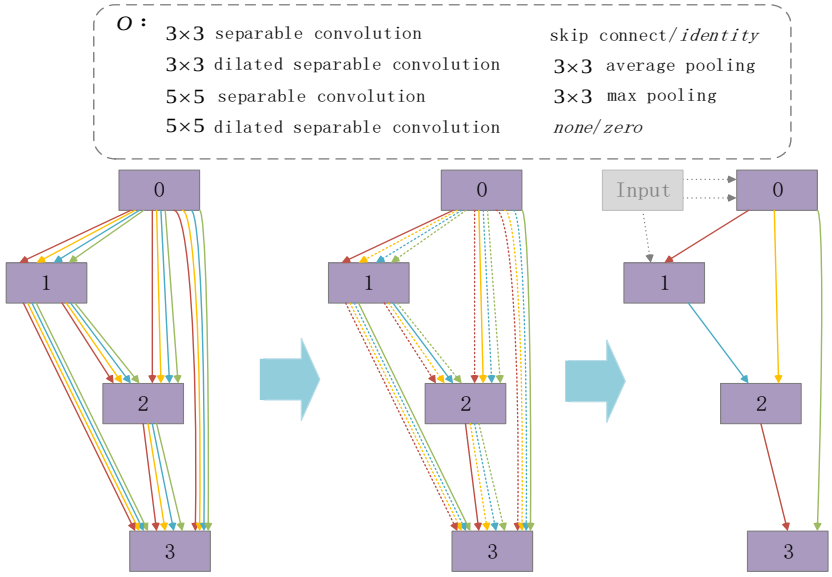
3. Materials and methods

3.1. Preliminary: DARTS

DARTS uses a cell as the basic unit and stacks the best cells searched to form the final convolutional neural network architecture [37], the specific operating space and search process of which is shown in Figure 2. Each cell is represented by a DAG. Node $x^{(i)}$ represents the feature maps, and the directed edge (i,j) represents some transformation operations $o^{(i,j)}$ on $x^{(i)}$. Each cell contains two input nodes and one output node. The two input nodes are the output nodes of the previous two cells, respectively. The two inputs are the outputs of the previous cell and the previous cell of the previous one, respectively. The output is the series output of all intermediate nodes. Each intermediate node is connected to two input nodes and all intermediate nodes of the previous sequence; the expression is defined as follows:

$$x^{(j)} = \sum_{i<j} o^{(i,j)} x^{(i)} \tag{1}$$

where $o^{(i,j)} \in O$, O is defined as the search space and contains eight search operations.



(a) (b) (c)

Figure 2. Demonstration of network architecture search process in DARTS. The optional operations in the search space are shown in the dashed box in the figure, and the operation selection process in each cell is shown below the dashed box. The dashed line indicates the discarded operations, and finally, only one operation is kept between every two intermediate nodes, which is represented by a solid line in the figure.

The DARTS algorithm relaxes all operations to the architectural parameters by using the softmax function, thus making the discrete search space continuous; the transformed Eq (1) is expressed as follows:

$$\overline{o}^{(i,j)}(x) = \sum_{o \in O} \frac{exp(\alpha_o{}^{(i,j)})}{\sum_{o' \in O} exp(\alpha_{o'}{}^{(i,j)})} o(x) \tag{2}$$

The mission of search is then transformed into learning to have a continuous set of variables $\alpha = \{\alpha^{(i,j)}\}$. After the search, the discrete architectures are replaced by the operations with the highest weight in $\overline{o}^{(i,j)}$, i.e., $o^{(i,j)} = argmax_{o \in O} \alpha_o{}^{(i,j)}$.

Then, the architecture parameters α and network parameters ω are learned jointly and the parameters are updated using the loss function, which is used in this paper as the cross-entropy loss function. The training loss and validation loss are respectively denoted as L_{train} and L_{val}. Both loss functions are jointly dictated by the two parameters α and ω. The goal of the architectural search is to find α to minimize the validation loss $L_{val}(\omega\ (\alpha), \alpha)$, where ω is found by minimizing the training loss, i.e., $\omega\ (\alpha) = argmin_{\omega} L_{train}(\omega, \alpha)$. The bilevel optimization problem is expressed as

$$\begin{aligned} &\min_{\alpha} L_{val}(\omega\ (\alpha), \alpha) \\ &s.t.\ \omega\ (\alpha) = argmin_{\omega} L_{train}(\omega, \alpha) \end{aligned} \tag{3}$$

The final model is determined by the learned α parameters, where the operation with the largest weight is chosen between every two nodes (except the none operation) and the two largest edges are chosen to be connected at each intermediate node. In addition, a second-order approximation is proposed in DARTS to solve the above bilevel optimization problem by approximating the optimization problem as follows:

$$\nabla_{\alpha} L_{val}(\omega\ (\alpha), \alpha) \approx \nabla_{\alpha} L_{val}(\omega', \alpha) - \xi \cdot \nabla_{\alpha} L_{train} \tag{4}$$

where $\omega' = \omega - \xi \nabla_{\omega} L_{train}(\omega, \alpha)$, $\omega^{+\omega + \varepsilon \nabla_{\omega'} L_{val}(\omega', \alpha)}$ and $\omega^{-=\omega - \varepsilon \nabla_{\omega'} L_{val}(\omega', \alpha)}$. When $\xi = 0$, it is transformed into a first-order approximation problem.

3.2. Attention search space

Here, we modify the search space in DARTS by adding two attention operations, CBAM and BAM. To increase search efficiency by reducing the number of operations in the search space, we analyze the effectiveness of the two pooling operations and select only one pooling operation to be added to the final search space. To compare with a more complex pooling operation, a new pooling operation module referred to as double pooling is added to analyze whether increasing the complexity of the pooling operation can improve the accuracy of the search network architecture. Furthermore, we analyze whether adding or removing the none operation after adding two attention modules and selecting the pooling operation has an impact on the network accuracy. Three operations, CBAM, BAM and double pooling, are shown in Figure 3, respectively.

In the double pooling operation module, since max pooling and average pooling provide two different important cues about feature extraction, the input image is subjected to max pooling and average pooling, respectively; then, the two outputs are summed by element to obtain the output image, which results in a more fine-grained channel direction of the desired emphasized or suppressed features

and is more conducive to feature extraction.

The BAM adds channel and spatial attention in parallel and then multiplies them with the input image; it then adds with the input image to acquire the output image. In the channel attention branch, the input image is first transformed to $R^{C\times1\times1}$ by averaging pooling, to $R^{\frac{C}{r}\times1\times1}$ by using a multi-layer perceptron (MLP), to $R^{C\times1\times1}$ to reduce the parameters and, finally, to $R^{C\times H\times W}$ by using batch normalization (BN). In the spatial attention branch, the input image is reduced by 1×1 convolution to reduce the number of channels, and then two 3×3 dilated convolutions are performed to expand the field of perception to extract features; then, it is transformed by 1×1 convolution to $R^{1\times H\times W}$. Eventually, the spatial attention branch is processed by BN and then expanded to $R^{C\times H\times W}$ to be able to combine with the channel attention branch.

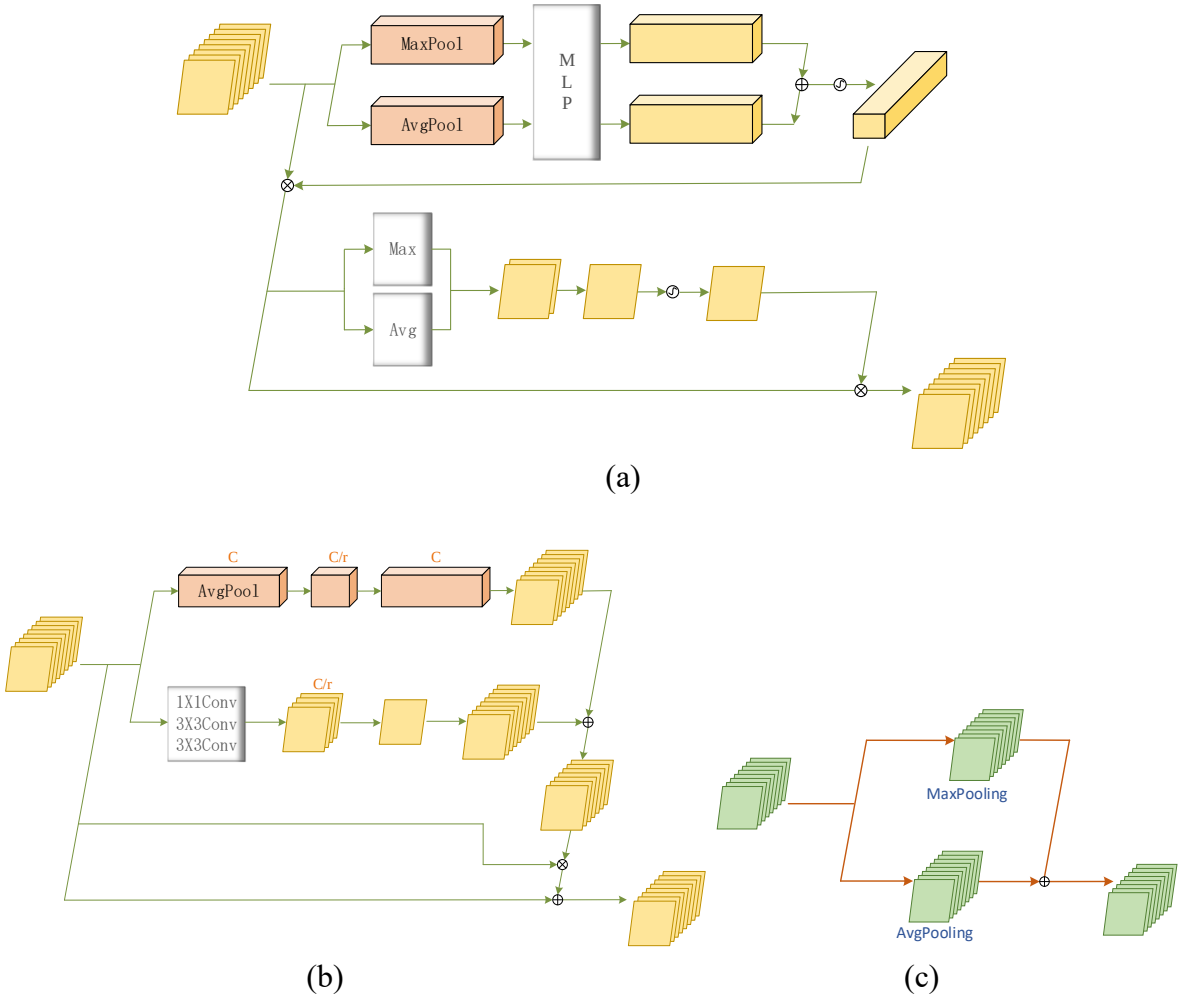


Figure 3. Attention to the new modules added in the search space; (a) shows the CBAM, (b) shows the BAM and (c) shows the double pooling module. C in the figure indicates the number of channels.

The main difference between the CBAM and the BAM is that the CBAM changes the two modules of channel and spatial attention from parallel to series. The channel attention precedes the spatial attention. We start by introducing the channel attention branch; first, the input image is transformed into $R^{C\times1\times1}$ by max pooling and average pooling before being passed through the MLP

containing a hidden layer. And, the two branches are summed and multiplied with the input image by rectified linear unit (ReLU) activation as the input of the next spatial attention branch. In the spatial attention branch, the input image of this branch is transformed into $R^{1\times H\times W}$ by max pooling and average pooling; the two branches are concatenated by channel and then passed through a 3×3 dilated convolution before finally being multiplied with the input image of the spatial attention branch after the ReLU activation to obtain the final output image.

3.3. Attention mechanism module

In the network architecture search process, DARTS does not select different nodes according to the importance of each node in the cell of the network architecture, ignoring the attention relationship between each intermediate node; so, we introduce an attention mechanism module that strengthens the attention relationship between all input nodes of each intermediate node in the cell. The introduced attention mechanism module is shown in Figure 4.

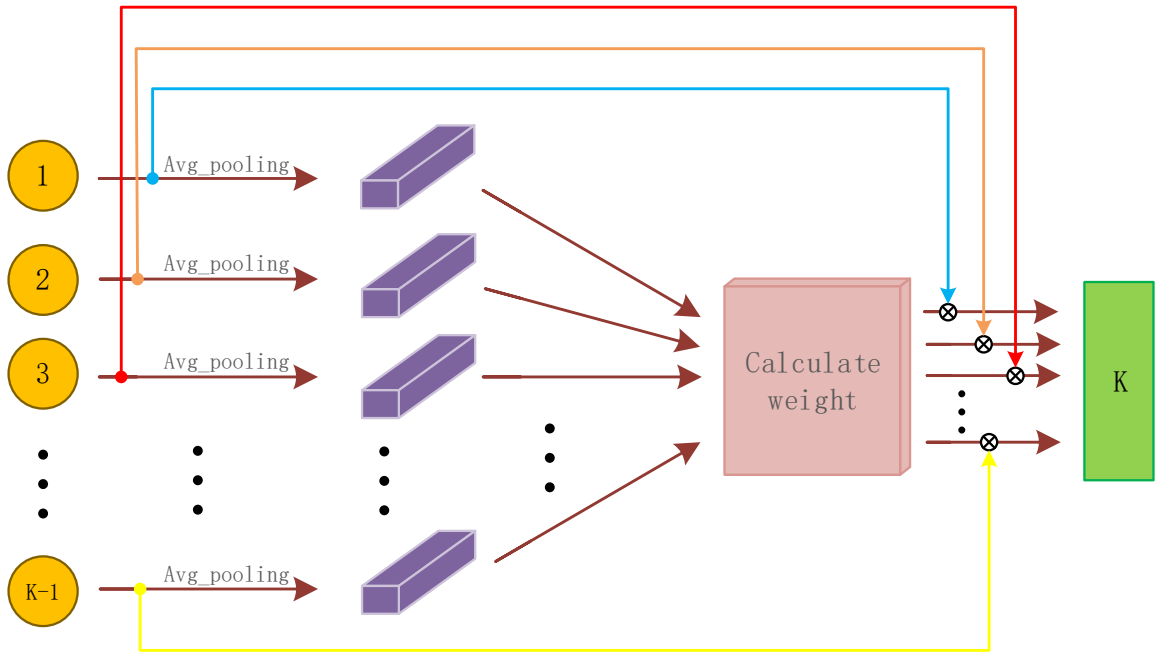


Figure 4. Attention mechanism module for node K, where $1 \cdots K-1$ is the intermediate node and K is the output node.

From Eqs (1) and (2), we have

$$x^{(j)} = \sum_{i<j} \overline{o}^{(i,j)} x^{(i)} \tag{5}$$

Let $\overline{o}^{(i,j)} x^{(i)} = y^{(i)}$; then, $y^{(i)}$ is fed into the attention mechanism module as an input. We obtain $a^{(i)} = \Phi\left(y|(i)\right)$, where Φ is the average pooling operation, and then convert the output feature image to $R^{C\times 1\times 1}$. A new feature map $A^{(j)}$ is generated for $a^{(i)}$ in series by channel, expressed as

$$A^{(j)} = \left[a^{(1)}, a^{(2)}, \cdots, a^{(j-1)}\right] \tag{6}$$

$A^{(j)}$ is then fed into an MLP containing two hidden layers, expressed as

$$M^{(j)} = \delta\left(MLP\left(A^{(j)}\right)\right) = \delta\left(W_1\left(W_0\left(A^{(j)}\right)\right)\right) \tag{7}$$

where δ is the sigmoid activation function, $W_0 \in R^{\frac{C'}{r}\times C'}$, $W_1 \in R^{C'\times\frac{C'}{r}}$ and C' is the sum of all channels of $A^{(j)}$, expressed as

$$C' = C_{a^{(1)}} + C_{a^{(2)}} + \cdots + C_{a^{(j-1)}} \tag{8}$$

Then, $M^{(j)}$ will be grouped again according to the number of channels of $a^{(i)}$. After grouping, $\kappa^{(i)}$ and $a^{(i)}$ will have the same amounts of channels, expressed as

$$\kappa^{(1)}, \kappa^{(2)}, \cdots, \kappa^{(i)}, \cdots, \kappa^{(j-1)} \tag{9}$$

Finally, it is multiplied as a weight with the input $x^{(i)}$ and expressed as

$$x^{(j)} = \sum_{i<j} \overline{o}^{(i,j)}\left(x|(i)\cdot\kappa^{(i)}\right) \tag{10}$$

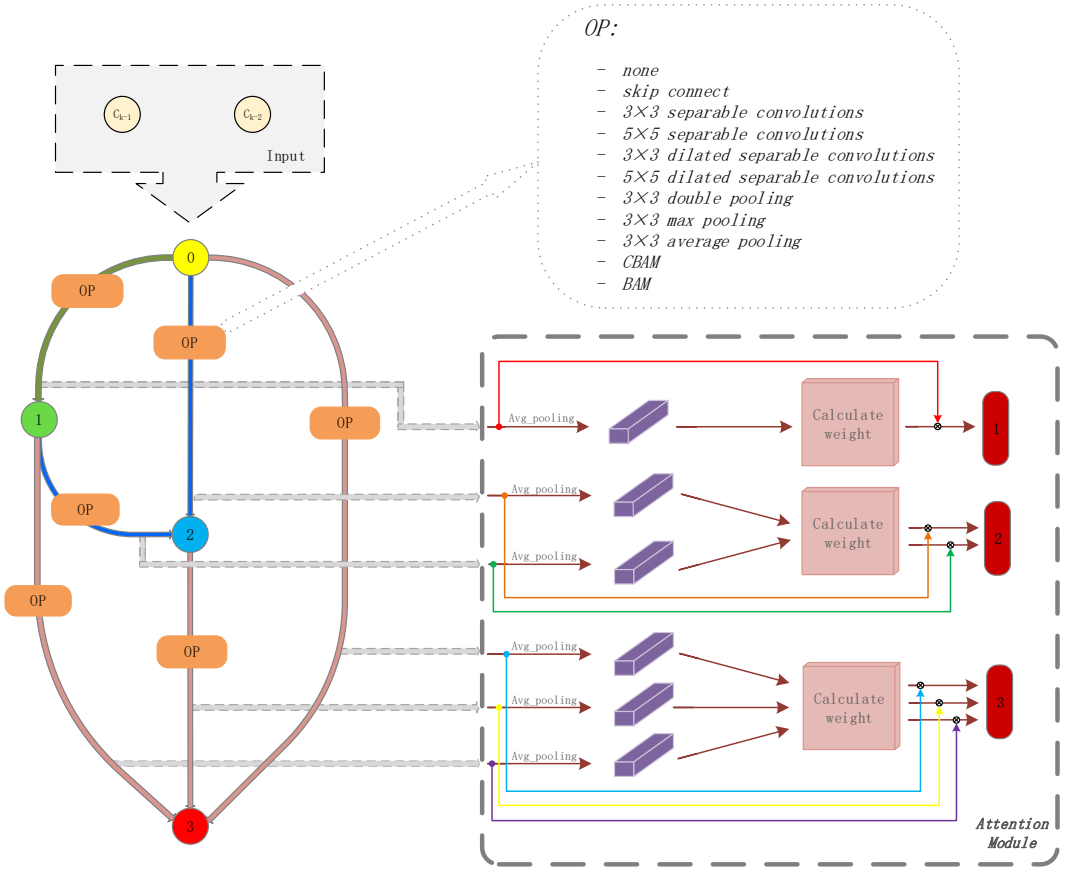


Figure 5. General diagram of network structure search by the DAM-DARTS method (the number of intermediate nodes is four in the diagram as an example).

Combining Subsections 3.2 and 3.3, we obtain a dual attention mechanism-based neural network architecture search method called DAM-DARTS, and the network architecture diagram of the DAM-DARTS method is shown in Figure 5.

4. Results

We validate the reliability of the proposed network search method on several datasets. The datasets are first introduced in Subsection 4.1, and the details of the specific implementation experiments are presented in Subsection 4.2. The feasibility of the proposed attention search space is verified by ablation experiments in Subsection 4.3, and the performance of the best structure searched with the addition of the attention mechanism module is verified. Finally, the validity of our method is verified by comparative experiments in Subsection 4.4.

4.1. Datasets

We performed experiments on three representative public datasets for image classification [6,42,57], namely, Fashion-MNIST, CIFAR10 and CIFAR100. The whole experimental process is divided into two phases, i.e., the architecture search phase and the architecture evaluation phase. These two phases were conducted separately. In the network architecture search phase, a good network architecture cell was searched on an architecture with a small number of cells stacked by using the search strategy proposed in this paper. In the architecture evaluation phase, stacking a larger number of cells discards the weights from the previous phase to train and evaluate the stacked network from scratch. We performed architecture search on the CIFAR10 dataset and then performed architecture evaluation on each of the three datasets.

CIFAR10 has a total of 60,000 RGB images, of which one-sixth comprised the test set and the remaining was the training set. The dataset was divided into 10 classes, where each class had 6000 images and one-sixth of which were test set images and the rest are training set images; each image had a resolution size of 32×32. The 10 classes were completely mutually exclusive without any overlap. The number of images in the CIFAR10 dataset is not very large, but it has a rich training set, and the architecture search phase of the experiment on this dataset is beneficial to get the best cell. CIFAR100 and CIFAR10 are similar in composition, as both contain 60,000 color images; the difference is that CIFAR100 contains 20 super-classes, each of which contain five subclasses, totaling 100 subclasses; each subclass contains 600 images, including 500 images of the training set and 100 images of the test set. The Fashion-MNIST dataset is similar to the MNIST dataset and contains a total of 70,000 grayscale images from 10 categories, where one-seventh comprised the test set images and the remaining were used as the training set images with a resolution size of 28×28; where the categories are different, we replaced abstract handwritten numeric symbols with more practical human clothing.

4.2. Implementation details

In the network architecture search phase, which has similar parameter settings in DARTS, the number of cell stacks was set to eight, so it contained six normal cells and the rest were reduction cells; the number of intermediate nodes of each cell was used for comparison experiments, applying four

and six, respectively. First-order approximation and second-order approximation were used to verify the objective function separately. In addition, in order to reduce the search time, part of the search strategy in PC-DARTS was also used, and we set K as 4 and the batch size as 256. A total of 50 epochs were trained in the architecture search phase, and the initial amounts of channels was 16. A warm-up strategy was also used to train only the network parameters in the first 15 epochs and both network parameters and architecture parameters in the last 35 epochs. We performed experiments using two NVIDIA GeForce RTX 2080Ti GPUs.

In the network architecture evaluation phase, the whole network consisted of a stack of 20 cells, where there were two reduction cells and the rest were normal cells. And, all 50,000 training set images were used to train the whole network from scratch by discarding the network parameters trained in the previous phase. The initial amounts of channels were set to 36. To save time, only the first 200 epochs of network architecture accuracy were observed in the exploration of the network architecture search space phase, and then the architecture was evaluated on the test set (the test set was not used in the network architecture search phase). For all other experiments, the number of epochs was set to 600 in order to compare with other methods more fairly. The batch size was fixed at 96. The probability of the drop path was fixed at 0.3 and normalized by using cutout.

Network evaluation uses a performance evaluation metric commonly used in image classification: top-1 accuracy. Top-1 accuracy is the final prediction of the class with the highest probability among the labels finally predicted on the test set, and if it is correct, the prediction is correct, and if not, the prediction is incorrect. In addition, the amounts of parameters in the network architecture search phase and the architecture evaluation phase, as well as the time spent in the search phase, are compared in the tables in the later sections. Other specific settings were the same as those for the DARTS-based approach; see the literature [37–39].

4.3. Architecture search

First, in the defined architecture search space O, the best cell with the number of intermediate nodes N of 4 was searched; the operations in the architecture space are shown in Figure 5. Second, we compared the effects of max pooling, average pooling and the more complex double pooling on the accuracy of the architecture searched from the search space, respectively. We analyzed the effect of having the none operation and not having the none operation on the accuracy of the architecture searched from the search space. And, we analyzed whether increasing N to 6, i.e., increasing the depth of cells, could improve the accuracy of the searched network architectures. We verify the effect of using the first-order approximation and second-order approximation in DARTS on the accuracy of the searched network architecture.

To explore the problem of the impact of the size of N on the accuracy of the network architecture and minimize the difference in the amounts of parameters between the network architecture with N of 4, we reduced the amounts of operations in the search space with N of 6, removed the large kernel convolution operation with two convolutional kernels of size 5×5 and redefined an architecture search space O'. The architecture search findings are shown in Table 1.

According to Table 1, the network architecture with N of 4 that was searched using the first-order approximation and max pooling operation had better network performance. Although the network architecture was searched with a number of intermediate nodes of six and the second-order approximation and max pooling operation yielded slightly higher accuracy, the number of parameters

in its architecture evaluation phase grew nearly twice. One can also find that increasing the complexity of the pooling operation does not improve the precision of the searched network architecture, but instead, it also increases the amounts of parameters of the network architecture.

It can also be observed that increasing the size of N slightly improves the accuracy of the searched network architecture, but the increased exponential number of parameters also consumes a large number of computational resources. Even though the large kernel convolution operation has been removed to reduce the amounts of parameters partially, we believe that it is the increased attention operation in the search space that is sufficient to enhance the network architecture performance. In addition, we found that the first-order approximation and second-order approximation in the DARTS method have little effect on the experiment; the network architecture with higher accuracy was searched under the first-order approximation setting, so we finally used the first-order approximation.

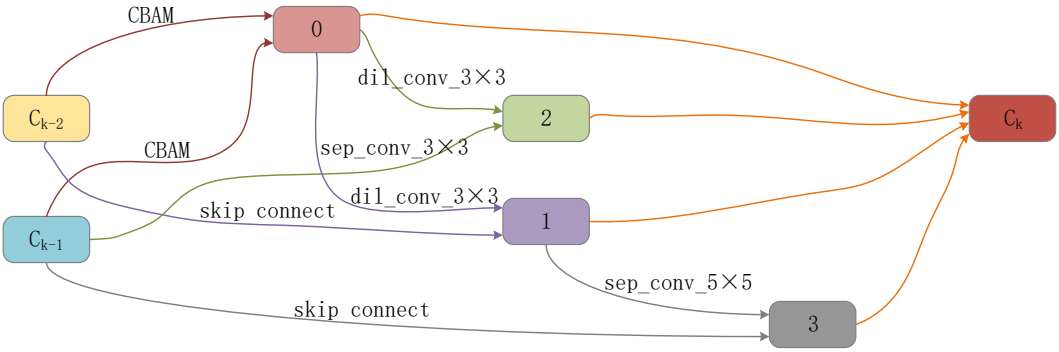
Table 1. Architectural accuracy of different search spaces**.**

Method	First order	Second order	None	Double pooling	Average pooling	Max pooling	Top-1 Accuracy (%)	Search parameters (MB)	Evaluation parameters (MB)
Ours-4	√		√	√			96.12	0.38	4.18
	√			√			95.77	0.38	4.74
	√				√		95.49	0.38	3.81
	√					√	**96.77**	**0.38**	**3.24**
		√	√			√	95.88	0.38	4.83
		√		√			95.46	0.38	4.21
		√			√		95.93	0.38	4.39
		√				√	96.16	0.38	3.78
Ours-6	√			√			95.24	0.44	6.48
		√		√			95.74	0.44	5.95
	√		√		√		94.46	0.44	5.91
	√				√		96.42	0.44	5.79
		√				√	96.80	0.44	6.09
DARTS		√	√		√	√	96.72	1.93	3.35

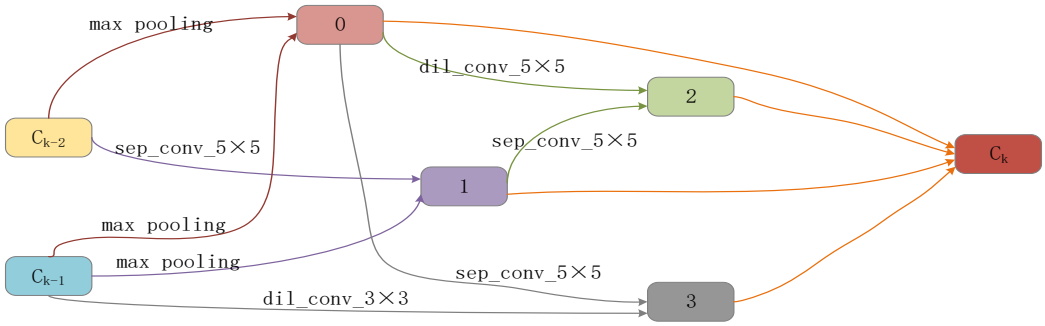
**Note*: Ours-4 denotes a network architecture with four intermediate nodes, and Ours-6 denotes a network architecture with six intermediate nodes.

We finalized the search space O as 3×3 and 5×5 separable convolution, CBAM, BAM, 3×3 and 5×5 dilated separable convolution, skip connect and 3×3 max pooling. There were eight operations in total, and the size of N was 4.

According to the experimental setup described in Subsection 4.2, the best normal cell and reduction cell searched by replacing the finalized network architecture search space were obtained as shown in Figure 6, where Figure 6(a) shows the normal cell and Figure 6(b) shows the reduction cell. The best network architecture searched by using the second-order approximation and max pooling operation with 96.80% accuracy is also shown in Figure 8 for N = 6. This architecture is not considered in this paper due to the large number of parameters. In Figures 7 and 8, both are normal cells (a) or both are reductive cells (b).



(a)



(b)

Figure 6. Best cells searched in Ours-4.

We added the attention module proposed in Subsection 3.3 to the search strategy and researched the best cell on CIFAR10; the best cell searched is shown in Figure 7. Then, we evaluated the architecture of the searched best architecture on three datasets, i.e., CIFAR10, CIFAR100 and Fashion-MNIST, for which 600 epochs were trained from scratch. Then, the best network architectures searched in DARTS were also re-performed in an architecture evaluation to compare with our searched architectures; the outcomes are shown in Table 2. According to Table 2, the DAM-DARTS method is 0.1% less accurate than the DARTS method on the CIFAR10 dataset, but it significantly reduces the time spent to search for the best architecture, and the number of architecture parameters in the network architecture evaluation phase is also reduced. According to Table 2, we can also see that the architectural accuracy on the CIFAR100 and Fashion-MNIST datasets is higher than that of the DARTS method, which indicates that the DAM-DARTS search strategy we proposed is more extensive and can be better extended to other different and larger datasets.

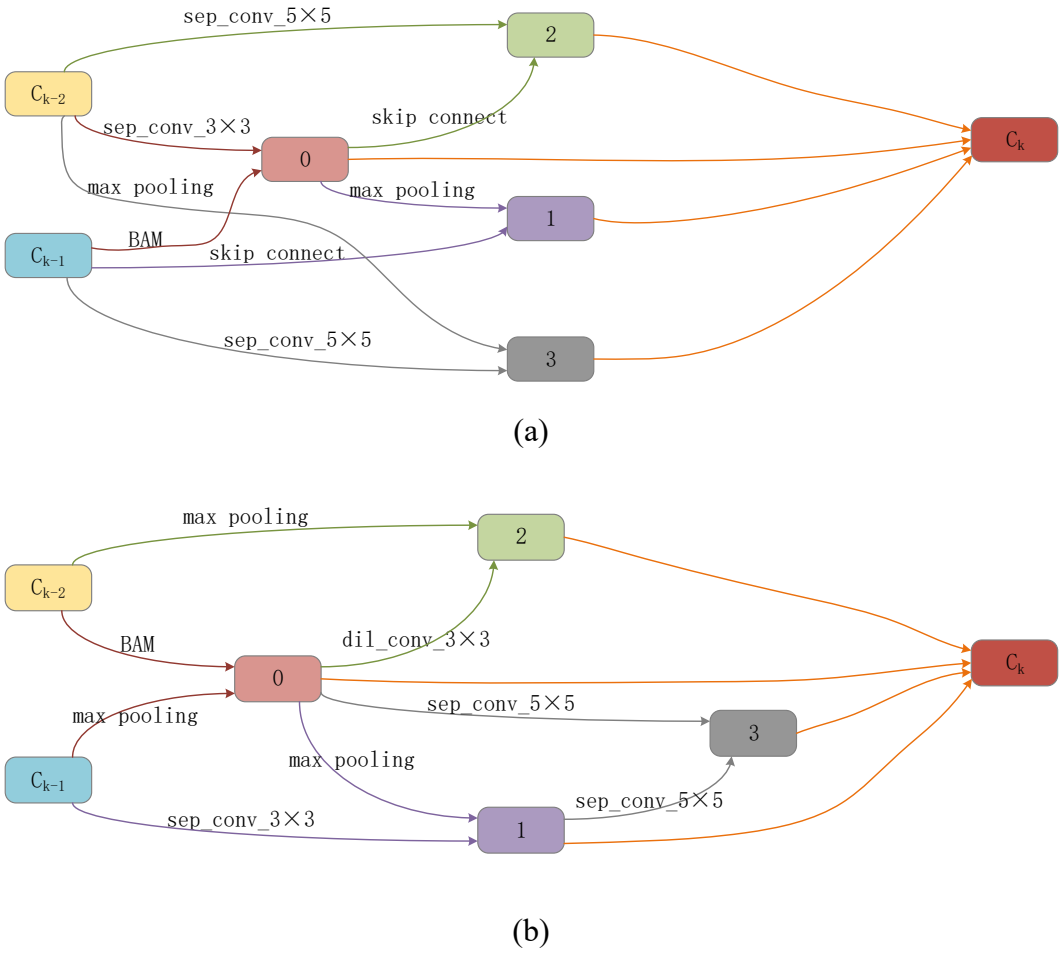


Figure 7. Best cells searched in DAM-DARTS.

We also evaluated the architecture of the search-out best network architecture that only changes the search space, i.e., the network architecture in Figure 6; the evaluation results are shown together in Table 2, called attention search space DARTS (ATTSS-DARTS). Since the amounts of parameters of the network architecture in Figure 8 was too large, the network architecture in Figure 8 was not considered in this study and the network architecture in Figure 6 was selected as the final best network architecture. According to Table 2, it can be seen that the ATTSS-DARTS network architecture performed better than DARTS on all three datasets for each performance metric, which also proves the effectiveness of our proposed architecture search space. In addition, although the architectural accuracy of ATTSS-DARTS was lower than that of DAM-DARTS and the time spent on searching the architecture was longer than that of DAM-DARTS, the size of parameters in the architecture search phase and architecture evaluation phase was smaller than that of DAM-DARTS, so it is also valuable to search the optimal network architecture only in the search space, which we designed without adding the attention module in Subsection 3.3. It also illustrates the effectiveness of the attention search space proposed in this paper.

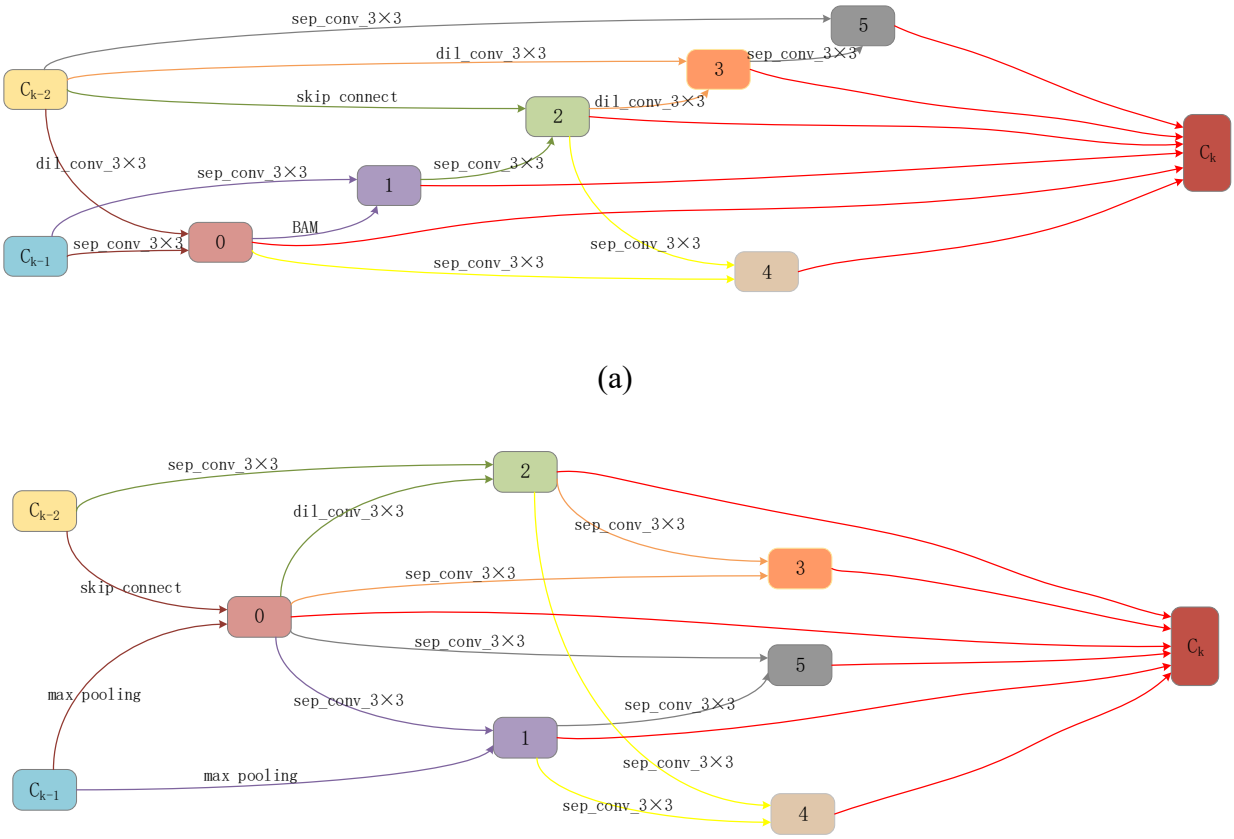


Figure 8. Best cells searched in Ours-6.

Table 2. Ablation experiments on the CIFAR10, CIFAR100 and Fashion-MINIST datasets.

CIFAR10				
Model	Accuracy	Search parameters	Evaluation parameters	Search cost
DARTS	97.31	1.93 MB	3.35 MB	23.5 hours
ATTSS-DARTS	97.34	0.38 MB	3.25 MB	16.67 hours
DAM-DARTS	97.21	0.56 MB	3.31 MB	6.5 hours
CIFAR100				
Model	Accuracy	Search parameters	Evaluation parameters	Search cost
DARTS	82.95	1.93 MB	3.40 MB	23.5 hours
ATTSS-DARTS	83.17	0.38 MB	3.30 MB	16.67 hours
DAM-DARTS	83.36	0.56 MB	3.36 MB	6.5 hours
Fashion-MNIST				
Model	Accuracy	Search parameters	Evaluation parameters	Search cost
DARTS	96.20	1.93 MB	3.35 MB	23.5 hours
ATTSS-DARTS	96.28	0.38 MB	3.24 MB	16.67 hours
DAM-DARTS	96.30	0.56 MB	3.31 MB	6.5 hours

4.4. Diagnostic experiments

We used the proposed DAM-DARTS method to train the searched best network architecture from scratch for 600 epochs, and then we conducted comparison experiments with other existing networks on the three datasets CIFAR10, CIFAR100 and Fashion-MNIST; the outcomes are shown in Tables 3–5, respectively. The test error of DAM-DARTS on the CIFAR10 dataset reached 2.79%, the size of the parameters of the architecture evaluation phase was 3.3 MB and the time taken to search for the best architecture was reduced to 0.27 GPU-days. The test error of DAM-DARTS on the CIFAR100 dataset reached 16.64%, and the size of parameters in the architecture evaluation phase was 3.36 MB. On the Fashion-MNIST dataset, the test error reached 3.7% and the size of parameters of the architecture evaluation phase was 3.31 MB.

Table 3. Results of comparison with other methods on CIFAR10.

Architecture	Test error (%)	Evaluation parameters (M)	Search cost (GPU-days)	Search method
DenseNet-BC [14]	3.46	25.6	-	MN
ResNet [13]	4.61	1.7	-	MN
SENet [22]	4.05	11.2	-	MN
NASNet-A + cutout [34]	2.65	3.3	2000	RL
AmoebaNet-A + cutout [30]	3.34	3.2	3150	EL
AmoebaNet-B + cutout [30]	2.55	2.8	3150	EL
PNAS [31]	3.41	3.2	225	SMBO
Hierarchical evolution [32]	3.75	15.7	300	EL
BCAS [47]	3.48	8	3	EL
NSGANetV1-A1 [33]	3.49	0.9	27	EL
ENAS [48]	3.54	4.6	0.5	RL
ENAS + cutout [48]	2.89	4.6	0.5	RL
DARTS + cutout [37]	2.76	3.3	1	GD
PC-DARTS + cutout [38]	2.57	3.6	0.1	GD
P-DARTS + cutout [39]	2.50	3.4	0.3	GD
ProxylessNAS + cutout [50]	2.08	5.7	4.0	GD
Att-DARTS + cutout [54]	2.62	3.2	10*	GD
ASM-NAS + cutout [55]	2.59	3.1	0.5	GD
FairDARTS-a [49]	2.54	2.8	0.42	GD
DAM-DARTS + cutout	2.79	3.3	0.27	GD

**Note*: Because the Att-DARTS network in the original article does not specify the time spent on the search, we reproduced the original code on our device; the * symbol indicates the time spent on our device to reproduce the code in the original article. MN denotes manual design method, EL denotes evolution method and GD denotes gradient method. The latter table is also represented as such.

In Table 3, we compare the architecture accuracy (test error), the size of parameters in the network architecture evaluation phase and the time spent in the search phase, respectively. The first block of the table shows the manually designed neural network architecture method and its searched network

architecture performance; the second block shows the automatic NAS method based on RL and evolution, as well as its searched network architecture performance; the third block shows the automatic NAS method based on gradient searching, i.e., based on the DARTS method, as well as its searched network architecture performance.

As we can see in Table 3, the DAM-DARTS method first outperformed the manually designed neural network architecture in terms of architectural accuracy, and the network architecture searched by the method which we proposed outperformed the network architecture searched by the RL and EA-based methods. Comparison with the DARTS-based method shows that the DAM-DARTS method performed slightly worse in terms of architectural accuracy, but the method with higher accuracy than this method takes more time than our method to search for the best network architecture. Moreover, the best network architecture searched by the DAM-DARTS search strategy scales better than the other methods and can be better applied directly to other larger datasets of different categories without the need to search the best network architecture again on a new dataset, as verified in Tables 4 and 5. In addition, for the ATT-DARTS network, which also contains attention in Table 3, because the time taken to search for the best network architecture is not stated in the original paper, we applied the code disclosed in the original paper using our experimental equipment, and the search took 10 GPU-days.

Table 4. Results of comparison with other methods on CIFAR100.

Architecture	Test error (%)	Evaluation parameters (M)	Search cost (GPU-days)	Search method
ResNet-101 [13]	22.22	25.3	-	MN
DenseNet-161 [14]	21.56	26.0	-	MN
SENet-50 [22]	21.42	26.5	-	MN
NASNet-A + cutout [34]	18.34	3.3	1800	RL
AmoebaNet-A + cutout [30]	18.38	3.1	3150	EL
PNAS [31]	19.53	3.2	225	SMBO
ENAS + cutout [48]	17.92	3.4	0.5	RL
DARTS + cutout [37]	17.76	3.3	1.5	GD
PC-DARTS + cutout [38]	17.01	4.0	0.1	GD
P-DARTS + cutout [39]	16.55	3.4	0.3	GD
GDAS + cutout [60]	18.38	3.4	0.2	GD
DARTS- [61]	17.51	3.3	0.4	GD
Att-DARTS + cutout [54]	16.54	3.2	10*	GD
DARTS + [62]	16.28	3.7	0.4	GD
DAM-DARTS + cutout	16.64	3.36	0.27	GD

**Note*: Since the Att-DARTS network in the original article does not specify the time taken for the search, we reproduced the original code on our device; the * symbol indicates the time taken to reproduce the code disclosed in the original article on our device.

We extended the searched best network architectures to the CIFAR100 dataset for architecture evaluation, trained 600 epochs from scratch and compared them with other methods. As shown in Table 4, the best architectures obtained by our method outperformed most existing NAS methods. The

three methods P-DARTS, ATT-DARTS and DARTS+, although the architecture accuracy was slightly higher than DAM-DARTS, took more time to search for the best architecture than the DAM-DARTS method; the size of parameters in the architecture evaluation phase was also larger for the P-DARTS and DARTS+ methods than the DAM-DARTS method. The accuracies of both the DARTS and PC-DARTS methods in Table 3 were slightly higher than that of the DAM-DARTS, but the architectural accuracies exhibited on the CIFAR100 dataset in Table 4 and the Fashion-MNIST dataset in Table 5 were below those of the DAM-DARTS method. It shows that the proposed method in this paper has better migration capability, as it can be better migrated to CIFAR100 and Fashion-MNIST datasets. The advantage of the method in this paper is that it can reduce the computational time consumption without reducing the accuracy. It can be seen in Tables 3–5 that the DAM-DARTS method has a significant effect on reducing the search cost, and that, although the PC-DARTS method has a lower cost than the proposed method on CIFAR10 and CIFAR100, it does not have an advantage in either architectural accuracy or scalability on CIFAR100. Therefore, weighing the magnitude of each performance metric, our DAM-DARTS method outperforms and is highly competitive with most existing methods.

We reproduced two representative methods, namely, the DARTS method and the PC-DARTS method, by using our experimental equipment, researched the best network architecture using the source code and performed the validation evaluation; the findings are shown in Table 6. Through Table 6, we can find that the accuracy of network architectures searched from different devices varies, and we did not search for a network architecture with the same accuracy as the best network architecture searched in the original article. We guess that, perhaps, the searched network architecture was reduced because the number of searches was less, or, perhaps, the searched network architecture was not unique. In addition, we also reproduced the P-DARTS method, but the test error only reached 3.43%, not 2.5% as in the original. The time required to search for the best architecture varied from device to device, and we have searched for a network architecture that can compete with existing advanced methods with limited devices and various uncertainties, which also illustrates the feasibility of the DAM-DARTS proposed in this paper. It can be inferred that the DAM-DARTS method can search neural network architectures with higher accuracy at a faster rate and fewer parameters under better experimental equipment conditions.

Table 5. Results of comparison with other methods on Fashion-MNIST.

Architecture	Test error (%)	Evaluation parameters (M)	Search cost (GPU-days)	Search method
ResNet [13]	5.10	11.1	-	MN
DenseNet [14]	4.61	25.6	-	MN
NASNet-A + cutout [34]	3.66	2.5	1800	RL
AmoebaNet-A + cutout [30]	3.67	2.3	3150	EL
PNAS [31]	3.89	2.5	225	SMBO
ENAS + cutout [48]	3.79	2.6	0.5	RL
DARTS + cutout [37]	3.77	3.35	1.6	GD
Random Sample [37]	3.95	2.5	-	-
GDAS + cutout [60]	3.76	2.4	0.21	GD
DAM-DARTS + cutout	3.70	3.31	0.27	GD

Table 6. Comparison of the results in the original paper with our reproduction of the original code on CIFAR10.

	DARTS	DARTS (our search)	PC-DARTS	PC-DARTS (our search)
Accuracy	97.31	97.11	97.45	97.10
Search parameters (MB)	1.93	1.93	0.30	0.30
Evaluation parameters (MB)	3.35	3.19	3.63	3.71

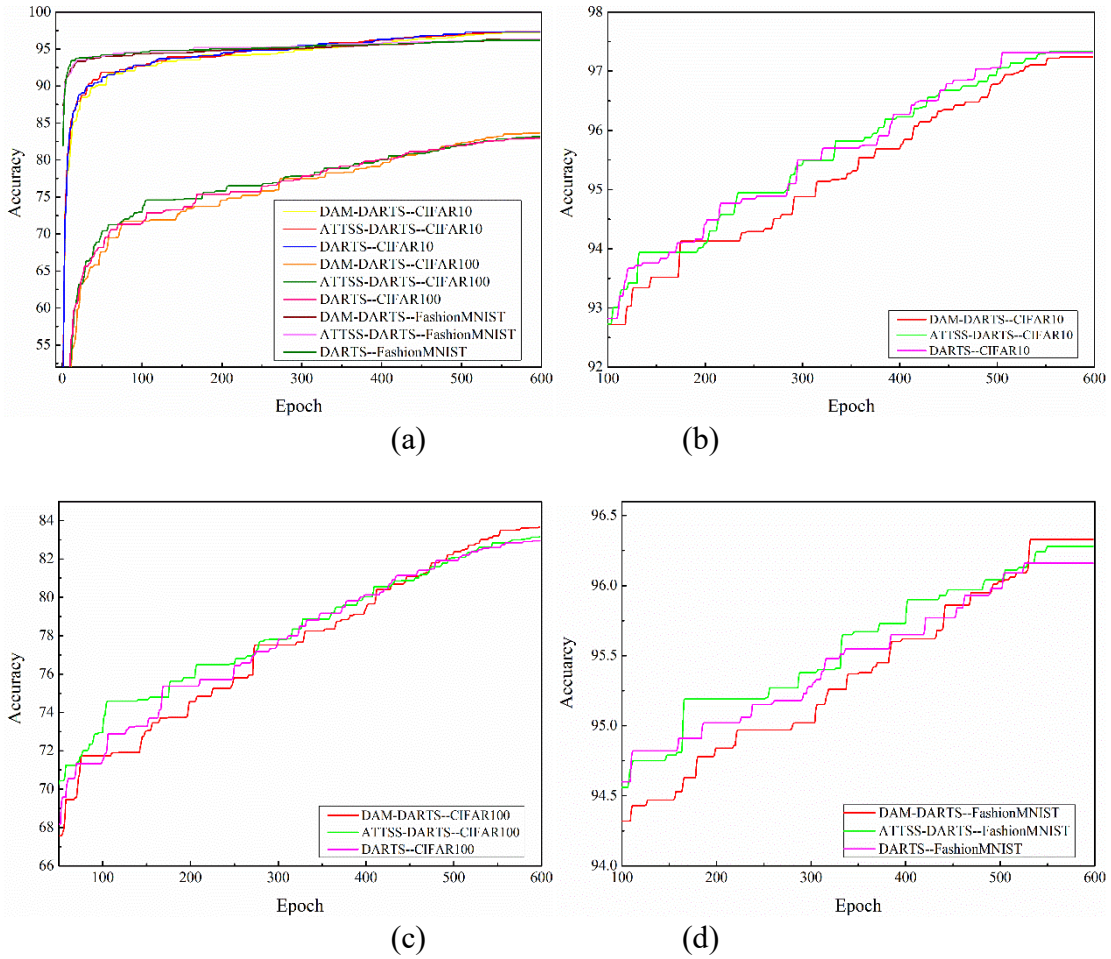


Figure 9. Comparison of accuracy changes during architecture evaluation phase.

We show the accuracy variation of 600 epochs in the architecture evaluation phase in Figure 9, where Figure 9(a) shows the comparison of the accuracy variation curves of the three networks DAM-DARTS, ATTSS-DARTS and DARTS on the three datasets CIFAR10, CIFAR100 and Fashion-MNIST. To show the curve variation more clearly, the accuracy of some epochs is shown in Figure 9(b)–(d), respectively. We can see in Figure 9 that DAM-DARTS performed better on the CIFAR100 and Fashion-MNIST datasets. In addition, the best network searched by the DAM-DARTS method performed better on the datasets with more categories, i.e., the CIFAR100 dataset, which also indicates that the proposed search strategy will be more suitable for the datasets with more categories and a larger number of images, and it is more significant for application. The network performance of ATTSS-DARTS is also exhibited in Figure 9, showing that it also performed well. Regarding the

CIFAR10 dataset, the final accuracy of the DAM-DARTS network architecture was slightly lower than that of DARTS, but the curve shows that the DAM-DARTS network still had an increasing trend at the end of the curve, while the DARTS network was almost constant. This also shows that, if both networks are trained for more time (more epochs), the architectural accuracy of the DAM-DARTS network proposed in this paper will likely exceed that of DARTS, further demonstrating the long-term nature of our proposed network architecture search method based on the dual attention mechanism.

5. Conclusions

Automatic neural network architecture search reduces the efforts one has to spend on designing a suitable neural network architecture as compared to the traditional methods based on the manual design of neural networks. We have proposed an automatic neural network architecture search method based on a dual-attention mechanism (DAM-DARTS). First, we proposed a more efficient architecture search space, adding two attention operations. And, we also analyzed the effects on the accuracy of the searched architectures, such as the change of other operations in the search space and the change of the number of intermediate nodes in the architecture basic unit. Second, we designed and added an improved attention mechanism module to enhance the attention between the nodes in the cell to further improve the accuracy of the searched network architecture. Through extensive comparison experiments on three datasets, CIFAR10, CIFAR100 and Fashion-MINIST, we demonstrated that the DAM-DARTS method proposed in this paper improves the performance of the searched best network architecture. The architectural accuracies of the searched DAM-DARTS network and ATTSS-DARTS network were higher than those of most other DARTS-based methods, and the time required to search for the best network architecture was reduced. Moreover, the best network architecture that was searched is more scalable and can be better applied to different datasets, so it has greater application value. In the future, we will verify the effectiveness of our search strategy on more and larger datasets and integrate the proposed method into more advanced methods to continue to improve the proposed method.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61305001, and the Natural Science Foundation of Heilongjiang Province of China under Grant F201222.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. A. Krizhevsky, I. Sutskever, E. G. Hinton, Imagenet classification with deep convolutional neural networks, *Commun. ACM*, **60** (2017), 84–90. https://doi.org/10.1145/3065386
2. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, et al., Generative adversarial networks, *Commun. ACM*, **63** (2020), 139–144. https://doi.org/10.1145/3422622

3. S. Xie, R. Girshick, P. Dollar, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Honolulu, HI, (2017), 1492–1500. https://doi.org/10.1109/CVPR.2017.634
4. X. Zhang, X. Zhou, M. Lin, R. Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Salt Lake City, UT, (2018), 6848–6856. https://doi.org/10.1109/CVPR.2018.00716
5. N. Ma, X. Zhang, T. H. Zheng, J. Sun, Shufflenet v2: Practical guidelines for efficient CNN architecture design, in *Proceedings of the European conference on computer vision (ECCV)*, Munich, GERMANY, **11218** (2018), 122–138. https://doi.org/10.1007/978-3-030-01264-9_8
6. M. Zhu, Q. Chen, Big data image classification based on distributed deep representation learning model, *IEEE Access*, **8** (2020), 133890–133904. https://doi.org/10.1109/ACCESS.2020.3011127
7. Y. Chen, D. Zhao, L. Lv, Q. Zhang, Multi-task learning for dangerous object detection in autonomous driving, *Inf. Sci.*, **432** (2018), 559–571. https://doi.org/10.1016/j.ins.2017.08.035
8. H. Zhao, Y. Zhang, S. Liu, J. Shi, C. Loy, D. Lin, et al., Psanet: Point-wise spatial attention network for scene parsing, in *Proceedings of the European Conference on Computer Vision (ECCV)*, (2018), 267–283. https://doi.org/10.1007/978-3-030-01240-3_17
9. J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, et al., Deformable convolutional networks, in *Proceedings of the IEEE international conference on computer vision*, (2017), 764–773. https://doi.org/10.1109/ICCV.2017.89
10. O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in *International Conference on Medical Image Computing and Computer-assisted Intervention*, Springer, Cham, (2015), 234–241. https://doi.org/10.1007/978-3-319-24574-4_28
11. F. Jia, J. Liu, C. X. Tai, A regularized convolutional neural network for semantic image segmentation, *Anal. Appl.*, **19** (2021), 147–165. https://doi.org/10.1142/S0219530519410148
12. P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, et al., Understanding convolution for semantic segmentation, in *2018 IEEE winter conference on applications of computer vision (WACV)*, NV, (2018), 1451–1460. https://doi.org/10.1109/WACV.2018.00163
13. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, (2016), 770–778. https://doi.org/10.1109/CVPR.2016.90
14. G. Huang, Z. Liu, L. V. D. Maaten, K. Q. Weinberger, Densely connected convolutional networks, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2017), 4700–4708. https://doi.org/10.1109/CVPR.2017.243
15. C. L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Semantic image segmentation with deep convolutional nets and fully connected crfs, preprint, arXiv:1412.7062. https://doi.org/10.48550/arXiv.1412.7062
16. C. L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, *IEEE Trans. Pattern Anal. Mach. Intell.*, **40** (2017), 834–848. https://doi.org/10.1109/TPAMI.2017.2699184
17. C. L. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-decoder with atrous separable convolution for semantic image segmentation, in *Proceedings of the European conference on computer vision (ECCV)*, **11211** (2018), 833–851. https://doi.org/10.1007/978-3-030-01234-2_49

18. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., Going deeper with convolutions, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Boston, MA, (2015), 1–9. https://doi.org/10.1109/cvpr.2015.7298594
19. S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in *International conference on machine learing*, PMLR, **37** (2015), 448–456.
20. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Seattle, WA, (2016), 2818–2826. https://doi.org/10.1109/CVPR.2016.308
21. C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in *Thirty-first AAAI conference on artificial intelligence*, 2017.
22. J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, **42** (2020), 2011–2023. https://doi.org/10.1109/TPAMI.2019.2913372
23. P. Ren, Y. Xiao, X. Chang, P. Huang, Z. Li, X. Chen, et al., A comprehensive survey of neural architecture search: Challenges and solutions, *ACM Comput. Surv.*, **54** (2021), 1–34. https://doi.org/10.1145/3447582
24. H. Cai, C. Gan, T. Wang, Z. Zhang, S. Han, Once-for-all: Train one network and specialize it for efficient deployment, preprint, arXiv:1908.09791. https://doi.org/10.48550/arXiv.1908.09791
25. Z. Ding, Y. Chen, N. Li, D. Zhao, Z. Sun, C. Chen, BNAS: Efficient neural architecture search using broad scalable architecture, *IEEE Trans. Neural Networks Learn. Syst.*, **33** (2021), 5004–5018. https://doi.org/10.1109/TNNLS.2021.3067028
26. J. Zhao, R. Zhang, Z. Zhou, S. Chen, J. Jin, Q. Liu, A neural architecture search method based on gradient descent for remaining useful life estimation, *Neurocomputing*, **438** (2021), 184–194. https://doi.org/10.1016/j.neucom.2021.01.072
27. H. Zhang, I. Goodfellow, D. Metaxas, A. Odena, Self-attention generative adversarial networks, in *International conference on machine learning*, **97** (2019), 7354–7363.
28. J. Park, S. Woo, Y. J. Lee, I. Kweon, Bam: Bottleneck attention module, preprint, arXiv:1807.06514. https://doi.org/10.48550/arXiv.1807.0651
29. S. Woo, J. Park, Y. J. Lee, I. Kweon, Cbam: Convolutional block attention module, in *Proceedings of the European conference on computer vision (ECCV)*, (2018), 3–19. https://doi.org/10.1007/978-3-030-01234-2_1
30. E. Real, A. Aggarwal, Y. Huang, Q. Le, Regularized evolution for image classifier architecture search, in *Proceedings of the aaai conference on artificial intelligence*, **33** (2019), 4780–4789. https://doi.org/10.1609/aaai.v33i01.33014780
31. C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, J. L. Li, et al., Progressive neural architecture search, in *Proceedings of the European conference on computer vision (ECCV)*, (2018), 19–34. https://doi.org/10.1007/978-3-030-01246-5_2
32. H. Liu, K. Simonyan, O. Vinyals, C. Fernando, K. Kavukcuoglu, Hierarchical representations for efficient architecture search, preprint, arXiv:1711.00436. https://doi.org/10.48550/arXiv.1711.00436
33. Z. Lu, I. Whalen, Y. Dhebar, K. Deb, E. D. Goodman, W. Banzhaf, et al., Multiobjective evolutionary design of deep convolutional neural networks for image classification, *IEEE Trans. Evol. Comput.*, **25** (2020), 277–291. https://doi.org/10.1109/TEVC.2020.3024708

34. B. Zoph, V. Vasudevan, J. Shlens, Q. Le, Learning transferable architectures for scalable image recognition, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2018), 8697–8710. https://doi.org/10.1109/CVPR.2018.00907
35. B. Zoph, V. Q. Le, Neural architecture search with reinforcement learning, preprint, arXiv:1611.01578. https://doi.org/10.48550/arXiv.1611.01578
36. M. Wistuba, Practical deep learning architecture optimization, in *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, (2018), 263–272. https://doi.org/10.1109/DSAA.2018.00037
37. H. Liu, K. Simonyan, Y. Yang, Darts: Differentiable architecture search, preprint, arXiv:1806.09055. https://doi.org/10.48550/arXiv.1806.09055
38. Y. Xu, L. Xie, X. Zhang, X. Chen, G. Qi, Q. Tian, et al., Pc-darts: Partial channel connections for memory-efficient differentiable architecture search, preprint, arXiv:1907.05737. https://doi.org/10.48550/arXiv.1907.05737
39. X. Chen, L. Xie, J. Wu, Q. Tian, Progressive differentiable architecture search: Bridging the depth gap between search and evaluation, in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, (2019), 1294–1303.
40. H. Cai, C. Gan, T. Wang, Z. Zhang, S. Han, Once-for-All: Train one network and specialize it for efficient deployment, preprint, arXiv: 1908.09791. https://doi.org/10.48550/arXiv.1908.09791
41. M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, et al., Mnasnet: Platform-aware neural architecture search for mobile, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2019), 2820–2828.
42. Z. Zhang, Y. Chen, C. Zhou, Self-growing binary activation network: A novel deep learning model with dynamic architecture, *IEEE Trans. Neural Networks Learn. Syst.*, 2022. https://doi.org/10.1109/TNNLS.2022.3176027
43. Q. M. Phan, H. N. Luong, Enhancing multi-objective evolutionary neural architecture search with training-free Pareto local search, *Appl. Intell.*, **2022** (2022), 1–19. https://doi.org/10.1007/s10489-022-04032-y
44. Q. M. Phan, H. N. Luong, Enhancing multi-objective evolutionary neural architecture search with surrogate models and potential point-guided local searches, in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, Springer, Cham, **12798** (2021), 460–472. https://doi.org/10.1007/978-3-030-79457-639
45. A. Ma, Y. Wan, Y. Zhong, J. Wang, L. Zhang, SceneNet: Remote sensing scene classification deep learning network using multi-objective neural evolution architecture search, *ISPRS J. Photogramm. Remote Sens.*, **172** (2021), 171–188. https://doi.org/10.1016/j.isprsjprs.2020.11.025
46. M. Song, Y. Zhong, A. Ma, R. Feng, Multiobjective sparse subpixel mapping for remote sensing imagery, *IEEE Trans. Geosci. Remote Sens.*, **57** (2019), 4490–4508. https://doi.org/10.1109/TGRS.2019.2891354
47. M. Ahmad, M. Abdullah, H. Moon, S. Yoo, D. Han, Image classification based on automatic neural architecture search using binary crow search algorithm, *IEEE Access*, **8** (2020), 189891–189912. https://doi.org/10.1109/ACCESS.2020.3031599
48. H. Pham, M. Guan, B. Zoph, Q. Le, J. Dean, Efficient neural architecture search via parameters sharing, in *International Conference on Machine Learning*, PMLR, **80** (2018), 4095–4104.

49. X. Chu, T. Zhou, B. Zhang, J. Li, Fair darts: Eliminating unfair advantages in differentiable architecture search, in *European conference on computer vision*, Springer, Cham, **12360** (2020), 465–480. https://doi.org/10.1007/978-3-030-58555-6_28
50. H. Cai, L. Zhu, S. Han, Proxylessnas: Direct neural architecture search on target task and hardware, preprint, arXiv:1812.00332. https://doi.org/10.48550/arXiv.1812.00332
51. Y. Bian, Q. Song, M. Du, J. Yao, H. Chen, Subarchitecture ensemble pruning in neural architecture search, *IEEE Trans. Neural Networks Learn. Syst.*, 2021. https://doi.org/10.1109/TNNLS.2021.3085299
52. J. Zhang, D. Li, L. Wang, L. Zhang, One-shot neural architecture search by dynamically pruning supernet in hierarchical order, *Int. J. Neural Syst.*, **31** (2021), 2150029. https://doi.org/10.1142/S0129065721500295
53. T. M. Luong, H. Pham, D. C. Manning, Effective approaches to attention-based neural machine translation, preprint, arXiv:1508.04025. https://doi.org/10.48550/arXiv.1508.04025
54. K. Nakai, T. Matsubara, K. Uehara, Neural architecture search for convolutional neural networks with attention, *IEICE Trans. Inf. Syst.*, **104** (2021), 312–321. https://doi.org/10.1587/transinf.2020EDP7111
55. J. Hao, W. Zhu, Architecture self-attention mechanism: nonliner optimization for neural architecture search, *J. Nonlinear Var. Anal.*, **5** (2021), 119–140. https://doi.org/10.23952/jnva.5.2021.1.08
56. Y. Weng, T. Zhou, L. Liu, C. Xia, Automatic convolutional neural architecture search for image classification under different scenes, *IEEE Access*, **7** (2019), 38495–38506. https://doi.org/10.1109/ACCESS.2019.2906369
57. M. Tanveer, K. H. Tan, F. H. Ng, K. M. Leung, H. J. Chuah, Regularization of deep neural network with batch contrastive loss, *IEEE Access*, **9** (2021), 124409–124418. https://doi.org/10.1109/ACCESS.2021.3110286
58. A. Ouahabi, A review of wavelet denoising in medical imaging, in *2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA)*, (2013), 19–26. https://doi.org/10.1109/WoSSPA.2013.6602330
59. A. E. Mahdaoui, A. Ouahabi, M. S. Moulay, Image denoising using a compressive sensing approach based on regularization constraints, *Sensors*, **22** (2022), 2199. https://doi.org/10.3390/s22062199
60. X. Dong, Y. Yang, Searching for a robust neural architecture in four GPU hours, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2019), 1761–1770.
61. X. Chu, X. Wang, B. Zhang, S. Lu, X. Wei, J. Yan, Darts-: robustly stepping out of performance collapse without indicators, preprint, arXiv:2009.01027. https://doi.org/10.48550/arXiv.2009.01027
62. H. Liang, S. Zhang, J. Sun, X. He, W. Huang, K. Zhuang, et al., Darts+: Improved differentiable architecture search with early stopping, preprint, arXiv:1909.06035. https://doi.org/10.48550/arXiv. 1909.06035