IEICE TRANS. INF. & SYST., VOL.E104–D, NO.8 AUGUST 2021

PAPER Special Section on Computational Intelligence and Big Data for Scientific and Technological Resources and Services

SP-DARTS: Synchronous Progressive Differentiable Neural Architecture Search for Image Classification

Zimin ZHAO[†a)], ***Nonmember***, **Ying KANG**[†], ***Student Member***, **Aiqin HOU**[†b)], ***and*** **Daguang GAN**[††], ***Nonmembers***

SUMMARY Differentiable neural architecture search (DARTS) is now a widely disseminated weight-sharing neural architecture search method and it consists of two stages: search and evaluation. However, the original DARTS suffers from some well-known shortcomings. Firstly, the width and depth of the network, as well as the operation of two stages are discontinuous, which causes a performance collapse. Secondly, DARTS has a high computational overhead. In this paper, we propose a synchronous progressive approach to solve the discontinuity problem for network depth and width and we use the 0-1 loss function to alleviate the discontinuity problem caused by the discretization of operation. The computational overhead is reduced by using the partial channel connection. Besides, we also discuss and propose a solution to the aggregation of skip operations during the search process of DARTS. We conduct extensive experiments on CIFAR-10 and WANFANG datasets, specifically, our approach reduces search time significantly (from 1.5 to 0.1 GPU days) and improves the accuracy of image recognition.

key words: *NAS, DARTS, skip operation, synchronous progressive, image classification*

1. Introduction

Image classification is one of the basic tasks in the field of computer vision. In 2012, AlexNet [1] improved the accuracy of image classification tremendously using an eight-layer convolutional network and won the ImageNet competition championship. Since then, the convolutional neural network has become the mainstream approach for image classification, and various networks have been designed to improve the performance [2]–[5]. In the past, most researchers designed neural networks manually. However, this situation has changed recently due to the development of neural architecture search (NAS), which achieved great success in image classification. For image classification, NAS is expected to design the search space, and learn to discover neural network architectures with good performance automatically. The development of NAS mainly goes through two stages according to the search strategy. In the first stage, although the search strategy has achieved good performance by applying reinforcement learning or evolutionary algorithms, the computational overhead is still considerably high as it often takes hundreds or even thousands of GPU days, which is unaffordable for lots of researchers. Many efforts explore ways to reduce search time. DARTS [6] brings NAS into the second stage: differentiable search, which deviates completely from reinforcement learning and evolutionary algorithms. DARTS uses multiple cell stacks, each of which is represented by a directed acyclic graph (DAG) consisting of multiple nodes. A relaxation method is used to train the mixed operations among nodes by a gradient descent approach, thus reducing the computational overhead to single-digit GPU days. NAS aims at searching for the model of the highest accuracy with the least search time, and the model size should be as small as possible. DARTS overcomes the issue of high computational cost in traditional methods and greatly reduces search time while ensuring model accuracy. It has become a new focus of research in NAS after reinforcement learning and evolutionary algorithms. DARTS can search for a well-performing model automatically by applying a search strategy, which costs less computation and less time. However, DARTS still suffers from many drawbacks, such as discontinuous network configuration, skip operation aggregation, etc., which affect the performance of the search algorithm. Therefore, it is important to investigate and solve these problems to improve accuracy and performance.

The rest of the paper is organized as follows. We first review the related work on NAS in Sect. 2. Then, we propose SP-DARTS in Sect. 3. We evaluate the performance of the proposed approaches through extensive experiments on the CIFAR-10 dataset in Sect. 4. We conclude our work in Sect. 5.

The main contributions of our work are summarized as follows:

- We propose an efficient "synchronous progressive method" for DARTS to bridge the depth and width gap between search and evaluation, and demonstrate its superiority over standard DARTS in terms of accuracy and search time.
- We apply partial channel connections (PC) and edge normalization (EN) to reduce computing overhead caused by the synchronous progressive method. The search stage in DARTS typically costs 1.5 GPU-days, and we reduce this time to 0.1 GPU-days without performance loss.
- We explore the impact of channels of the network and the sampling rate K of the PC method on skip aggregation phenomenon and provide justifications for such

Manuscript received November 11, 2020.
Manuscript revised January 13, 2021.
Manuscript publicized April 23, 2021.

[†]The authors are with School of Information Science and Technology, Northwest University, Xi'an, Shaanxi 710127, China.
[††]The author is with Wanfang Data Co., Beijing, 100038, China.
a) E-mail: zhaozimin@stumail.nwu.edu.cn
b) E-mail: houaiqin@nwu.edu.cn (Corresponding author)

DOI: 10.1587/transinf.2020BDP0009

impact.
- We conduct experiments to show the skip operation aggregation phenomenon caused by DARTS and employ the Sigmoid activation function to eliminate this phenomenon.

2. Related Work

Some early studies proposed pioneering search architectures based on reinforcement learning (RL) [7]–[9]. NASNet can generate strings that represent the structure of neural networks through RNN [7], and continuously update the gradient of strategy to generate a network architecture with better performance. However, the procedure is computationally very intensive and can only process small data sets. NASNet is among the first that proposed a cell-based search space, where a cell is a DAG consisting of an orderly sequence of nodes. NASNet searches for operation types and topological connections in the cell and repeats the cell to form the entire network architecture. The depth, width, and sampling operation of the architecture are all set manually, which greatly improves the search efficiency of the algorithm compared with the work in [7]. The approaches based on evolutionary algorithms (EA) [10]–[12] have also achieved great performance. Some research efforts applied EA to the search field to initialize a neural network model population, from which each individual model is evaluated for its adaptability on the validation set. The algorithm proceeds to reproduce and mutate the superior architectures while eliminating the inferior ones. RL- and EA-based methods generally incur very high computational overhead. DARTS is the first work that applies a gradient-based method to the field of NAS and reduces the search time to 1.5 GPU days. Liu et al. initialized all operations between nodes in each cell and generated weights using multi-cell stacking. They also relaxed the search space to be continuous, so that the architecture can be optimized by gradient descent and reduce the computational cost in the procedure of search [6]. A significant number of research efforts are based on [13] to improve differentiable search. Xin et al. [14] pointed out that the depth discontinuity of the search stage and evaluation stage in [6] causes the low accuracy of the search model. They also proposed to divide the search process into multiple stages, increase the network depth continuously to address the discontinuity of depth, and discard some of the operations when switching the search stage to adapt GPU memory. Their result shows that the accuracy is improved and the search time is reduced to 0.3 GPU days. GDSA [7] is proposed with a partial optimization subgraph method and a new loss function, which reduced the search time to 0.21 GPU days. In addition, the work in [7] also mentioned that many architecture search techniques of reduction cell are very similar, and designed reduction cell manually, which can reduce search time and improve search accuracy in the search stage. Xu et al. proposed PC-DARTS to reduce the search time by partial channel sampling, solved the problem of skip operation aggregation, and utilized edge regularization to solve the problem of search instability [15]. Chu et al. pointed out the reason for skip operation aggregation is that the softmax relaxation method of discrete operations causes unfair competition in the processing of gradient descent of weight [16]. Therefore, they used a sigmoid relaxation function to solve this problem, and proposed a one-zero loss function to accelerate the convergence speed of mixed operation weights.

3. The Proposed Approach

This section introduces a DARTS-based framework of neural architecture search to address the drawbacks of the original DARTS.

3.1 Preliminary: Differentiable Architecture Search (DARTS)

Our proposed algorithm is based on DARTS [6], which has two stages for search and evaluation. During the search stage, a model stacked with L cells is used to search, and the input of each cell is two previous cells. A cell is defined as a directed acyclic graph (DAG) consisting of a sequence of N nodes denoted as $\{x_0, x_1, \cdots, x_{N-1}\}$. The number of input and output channels for each cell except for input node and output node is denoted as C. There are 8 operations defined in search space (O), including the operations of 3×3 and 5×5 separable convolutions, 3×3 and 5×5 dilated separable convolutions, 3×3 max pooling, 3×3 average pooling, identity, and zero. Each operation represents some function $o(\cdot)$, and each edge $E(i, j)$ represents the mixed weighted operation from $node^{(i)}$ to $node^{(j)}$, where the weights are parameterized by vector $\alpha^{(i,j)}$. We make the search space continuous using softmax over all operations as follows:

$$o^{(i,j)}(x) = \sum_{o \epsilon O} \frac{exp(\alpha_o^{(i,j)})}{\sum_{o' \epsilon O} exp(\alpha_{o'}^{(i,j)})} o(x). \tag{1}$$

The output of each intermediate node is expressed as:

$$x^j = \sum_{i<j} o^{(i,j)}(x^{(i)}). \tag{2}$$

The output node of a cell is calculated as:

$$x_{N-1} = concat(x_2, x_3, \ldots, x_{N-2}). \tag{3}$$

The training and valid loss are determined by the architecture α and weights w of network, denoted by L_{train} and L_{val}. The optimization approach is as follows:

$$\begin{aligned} &\min_{\alpha} L_{val}(w^*(\alpha), \alpha) \\ &s.t.\ \ w^*(\alpha) = argmin_w L_{train}(w, \alpha), \end{aligned} \tag{4}$$

where the weights w^* is obtained by minimizing the training loss L_{train}. In the search stage, the network depth $L = 8$, and the initial number of channels $C = 16$. There are $N = 7$ nodes in each cell, including two input nodes, four intermediate nodes, and one output node. In the evaluation stage, the structure of each cell remains the same as the search

stage, but the depth L is increased from 8 to 20 and the number of channels C is increased from 16 to 36. The optimal model selected from the search stage is determined after training on the training dataset and evaluation on the valid dataset.

3.2 Synchronous Progressive DARTS

The network depth and width are not continuous between the search and evaluation stages of DARTS: the depth is from 8 cells to 20 cells, and the width (i.e., the initial number of channels) is from 16 to 36. It is well known in the neural architecture search algorithm that if the model performs well in the search stage, it also performs well in the evaluation stage. In other words, the model with good performance in the search stage performs well with probability P in the evaluation stage. Therefore, the higher the probability P is, the better performance of the search algorithm has. However, the discontinuity of depth and width may reduce the probability P, as the continuity worsens, and the difference of performances between the search and evaluation stages is greater. To address this drawback, we propose an approach based on DARTS to change the depth and width in a synchronous progressive way during the search stage, referred to as SP-DARTS (Synchronous Progressive DARTS), which divides the search process into three stages. The steps of SP-DARTS are shown in Algorithm 1. The search process in the original DARTS and our SP-DARTS are shown in Fig. 1 (a) and (b), respectively, for comparison.

In SP-DARTS, we adopt a staged strategy that splits the search process into three stages where the depth and width are gradually increased from 5 to 20, and from 16 to 36, respectively. At the end of each stage, the depth and width of the model are increased for the next stage. The procedure is provided in Fig. 2. The algorithm divides the search process into three stages, each of which performs the same process with different network settings. In the first stage, 3 normal cells and 2 reduction cells are stacked, so the depth of the network is 5. The number of input and output channels of each cell is set to 16, so the width of the network is 16. The second stage initializes the network to a depth of 11 and a width of 24. The third stage initializes the network to a depth of 17 and a width of 32. The same process is carried out in each search stage: the model parameters and architecture parameters of the network are optimized by gradient descent according to Eq. (8). After search, the evaluation stage initializes the network to a depth of 20 and a width of 36. This way, the network's depth and width are gradually increased, and the discontinuity between the search and evaluation stages is addressed. However, the computational cost during the process increases linearly with respect to the depth of the model, and increases by power of two with respect to the width. Therefore, SP-DARTS cannot run on a single GPU directly. In order to address this problem, we use the "search space approximation" method proposed in [14], which progressively reduces the operations in different stages to fit into a GPU. As shown in Fig. 2, while 0, 1, 2 and 3 are four intermediate nodes in the cell, the lines of different colors represent different operations. After the first stage, four operations with the maximum weight are retained according to the operation weight, while other operations are discarded. After the second stage, two operations are retained in the same way, and one operation is retained after the third stage.

The progressive method in SP-DARTS can learn more complex representations with the increasing depth of the network, whereas each layer becomes simpler [17], [18]. Similarly, the network can learn more abundant features with an increasing width [19]. Because the network depth and width are different, we propose a synchronous progressive method in the search stage to make the depth and width approximate to the corresponding values in the evaluation stage. Therefore, the operations of the final network architecture are different with a high probability.

Algorithm 1 SP-DARTS

Input: Number of search stages T; The network depth, channels for stage t: $Layer_t$, $Channel_t$;
The proportion of PC selected channels for stage t: K_t;
Initial architecture α, model weights ω, edge normalization coefficient β;
Number of saved operations for stage t: O_t;
for t = 1 to T **do**
  Update network configuration to $Layer_t$, $Channel_t$, K_t, O_t;
  Reinitialize architecture α, model weights ω, and EN coefficient β based on O_t;
  while not converged **do**
    Sample a batch of training data X_{train} and a batch of validation data X_{val};
    Update α and β by descending $\nabla_\alpha(L_{val}(\omega,\alpha,\beta)+\omega_{0-1}L_{0-1})$, $\nabla_\beta(L_{val}(\omega,\alpha,\beta)+\omega_{0-1}L_{0-1})$;
    Update ω by descending $\nabla_\omega L_{train}(\omega,\alpha,\beta)$;
  end while
end for
Derive the final architecture based on the learned α and β;

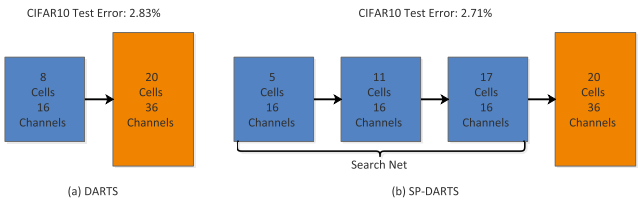


Fig. 1 Difference between DARTS and SP-DARTS.

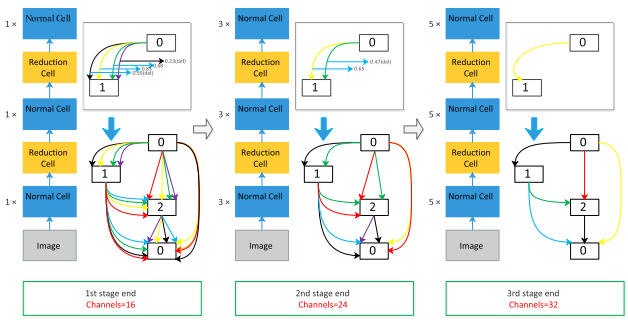


Fig. 2 Search space approximation.

3.3 Problems of Memory Inefficiency and Skip Operations Aggregation

A drawback of DARTS lies in memory inefficiency [15]. In the search stage, the network width is large so that the storage of convolution template parameters of intermediate nodes and the storage of output results take up a large proportion. The batch size of training needs to be limited to complete the search on a single GPU, which reduces the search speed and accuracy. Besides, the full channel connections restrict the network's depth and width from setting a large number potentially in the search stage, which also affects the search's accuracy. Following the partial channel connections method proposed in [15], we propose the information flow between nodes in the cell as the following expression:

$$f_{i,j}(x_i, s_{i,j}) = \sum_{o \epsilon O} \sigma(\alpha^o_{i,j}) \cdot o(S_{i,j} * x_i) + (1 - S_{i,j} * x_i), \quad (5)$$

where $\sigma()$ represents the sigmoid activation on $\sigma^o_{i,j}$, and $S_{i,j} * x_i$ and $(1 - S_{i,j} * x_i)$ denote the selected and masked channels, respectively. The hyper-parameter K is set to express the proportion of selected channels by $1/K$. The partial channel connections also mitigate the skip operation aggregation problems in DARTS. In the early studies, architecture search algorithms generally preferred no parameter or weak parameter operations because they had few parameters to train, and thus the output was more continuous. By contrast, for those operations with more parameters, each epoch's update transmits discontinuous information before all parameters are well optimized. As a result, weak parameter operations tend to accumulate an enormous weight. It is difficult for other operations to "defeat" them, even if they have been trained to a certain extent. According to the statistics of the number of skip operations through running DARTS for 3 times, as can be seen from the Fig. 3, with the increase of training epochs, DARTS has a severe skip operation aggregation phenomenon. Partial channel connections limit the number of skip operations because the operation between nodes only passes through the partial channel. It can be more effective by sampling a part of super-net to reduce the redundant space without any performance loss.

In this paper, we not only adopt partial channel connections, but also use a sigmoid function to activate the mixed operation, which addresses unfair competition between operations in the search stage. It has been pointed out in the literature that the root cause of the skip operations aggregation and weak parameter operations is the unfair competition between operations in the search stage [16]. Specifically, softmax relaxation was used for mixing operations between nodes within a cell in the DARTS. The nature of the softmax function determines that an increase of one operation weight results in a decrease of the others, affecting competition between operations in mixed operation. The increase of weight in the early stage of weak parameter operation greatly weakens the weight of other operations, leading

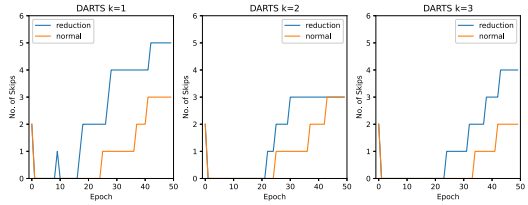


Fig. 3 The number of skip connections when searching with DARTS.

to the aggregation of weak parameter operations. In addition, such unfairness affects the operation competition of all precursor nodes of each node, which greatly inhibits the increase of operation weight, as it might be better in the future during the search stage, and affects the performance of the search model. If sigmoid activation function is applied according to Eq. (5), the relation between operations is free, and fair competition is thus established. Even if the activation value of weight reaches the upper limit of 1, weights of other operations can continue to update well, so as to mitigate the weak parameter operation aggregation. Also, it is proved in [15] that the edge normalization method can solve the search instability problem caused by partial channel sampling and improves the search quality. Therefore, the intermediate node is calculated as:

$$x_j = \sum_{i<j} \frac{exp(\beta_{i,j})}{\sum_{k<j} exp(\beta_{k,j})} \cdot f_{i,j}(x_i), \quad (6)$$

where $\beta_{i,j}$ denotes the weight from $node(i)$ to $node(j)$, and $f_{i,j}(x_i)$ denotes the value of forward propagation. The rule that the input weights of precursor nodes of each intermediate node are equal is broken by using edge normalization.

3.4 Problem of Mixed Operation Discretization

As mentioned before, in the evaluation stage of DARTS, there are problems of discontinuities of network depth and width, but there still exists a problem: discretization of mixed operations. According to the method proposed in [6], the operation with the largest weight in mixed operation of all precursor nodes for each intermediate node is selected first after the search stage, so that there is only one operation for the edge from each precursor node to this node. Then two precursor nodes with the maximum weight are selected to determine the cell structure. However, we found that these weights are very close to each other through running DARTS. At the end, it is hard to tell which operations are good and which ones are bad. It is expected that one-hot architecture will be generated at the end of the search stage of DARTS, where the selected operation has a weight of 1 and the unselected operation has a weight of 0. Therefore, we use the sigmoid function and set the activation value to be either 0 or 1. This way, the difference between the model after searching and the model after discretization is smaller, and the performance loss also becomes smaller. The additional loss function [16] is as follows:

$$L_{0-1} = -\frac{1}{N} \sum_{i}^{N} (\sigma(\alpha_i) - 0.5)^2, \quad (7)$$

where N is the total number of operations in a cell. The loss of additional function is minimum when $\sigma(\alpha_i)$ is 0 or 1, and the loss is maximum when $\sigma(\alpha_i)$ is 0.5. The weight $\sigma(\alpha_i)$ converges faster in the direction of 0 or 1 using this additional loss function, and the muti-hot architecture will be formed at the end of search. It also expands the solution space compared to one-hot. In this paper, the optimization method is as follows:

$$\begin{aligned} &\min_{\alpha} L_{val}(w^*(\alpha)) + w_{0-1}L_{0-1} \\ &s.t. w^*(\alpha) = argmin_w L_{train}(w, \alpha), \end{aligned} \tag{8}$$

where w_{0-1} is the weight of loss function L_{0-1}, and as in Eq. (4), W represents model parameters, and α represents architecture parameters. The search algorithm optimizes model parameters with training dataset, and optimizes the architecture parameters with validation dataset. The discretization method after searching is to multiply the weight activation value $\sigma(\alpha_{i,j})$ and the edge normalization coefficient $\frac{exp(\beta_{i,j})}{\sum_{k<j} exp(\beta_{k,j})}$, and then generate the architecture according to the discretization method in DARTS. Therefore, the combined flow of Eq. (3), Eq. (5), Eq. (6) and Eq. (8) constitute our method in this paper; Eq. (5) and Eq. (6) are the calculation methods of intermediate nodes within a cell; Eq. (3) is the calculation method of the output of each cell, and Eq. (8) is the optimization goal of the overall algorithm. The rest of the algorithm is the same as DARTS.

4. Experiments and Results

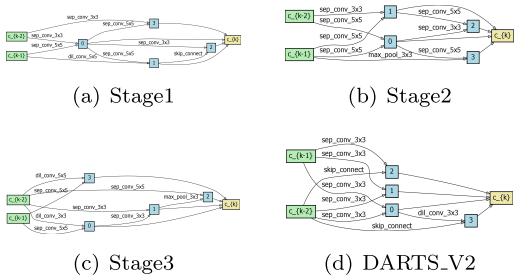
We conduct experiments on image classification datasets of CIFAR-10, which has 50k/10k training/testing RGB images with a spatial resolution of 32×32. These images are equally distributed over 10 classes. The training dataset is equally split into two subsets in the architecture search stage, one for tuning network parameters, the other for tuning the architecture. In the evaluation stage, the standard training/testing is used.

4.1 Architecture Search

The search process consists of 3 stages. The network configurations are the same as DARTS, except that the zero operation is dropped. Since zero operation has no gradient, it is useless without a softmax function. In the initial stage (stage 1), the network depth $L = 5$, the initial channel number $C = 16$, partial channel sampling $K = 4$. In the intermediate stage (stage 2), $L = 11$, $C = 24$, $K = 4$. In the final stage (stage 3), $L = 17$, $C = 32$, $K = 4$. Meanwhile, the operation space's size in three stages is 7, 4, and 2, respectively. After the final search stage is completed, only one operation left to generate the model structure. For each stage, we train the network using a batch size of 256 for 25 epochs. Since the partial channel sampling is applied, the larger batch size is allowed for acceleration.In the first 10 epochs, only the network parameters are tuned at each stage, while the network and architecture parameters are tuned in the remaining 15 epochs. The architecture parameters are optimized by Adam optimizer, where the learning rate $\eta = 0.0006$, weight decay $wd = 0.001$ and momentum $\beta = (0.5, 0.999)$. The model parameter optimizer adopts momentum SGD optimization. The initial learning rate $\eta = 0.025$, which is decayed to 0 following the cosine rule, and the momentum $\beta = 0.9$, weight decay $wd = 0.0003$, the weight of loss function L_{0-1} $w_{0-1} = 10$. The search results of the model are shown in Table 1. SP-DARTS(a) results from using loss function L_{0-1}, and SP-DARTS(b) is the result without loss function L_{0-1}. The experiments are performed on a single Tesla V100 GPU and cost 0.11 GPU days. Compared with DARTS, it is clear that the search time is reduced and the model accuracy is improved. Figure 4 shows the searched model, where the search depths of (a), (b) and (c) are 5, 11 and 17, respectively, and the initial number of channels is 16, 24 and 32, respectively. We observe that the cell becomes more hierarchical as the search progresses. (d) is the normal cell searched by DARTS_v2.

Table 1 Comparison of NAS algorithms on CIFAR-10.

Models	Params (M)	Test Error (%)	Search Cost (GPU-days)	Type
DenseNet-BC[25]	25.6	3.46	-	manual
NASNET-A[2]	3.3	2.65	2000	RL
BlockQNN[26]	39.8	3.54	96	RL
AmoebaNet-B[23]	2.8	2.55	3150	evolution
Hireachical Evolution[12]	15.7	3.75	300	evolution
PNAS[28]	3.2	3.41	225	SMBO
ENAS[27]	4.6	2.89	0.5	RL
DARTS(first order)[14]	3.3	3.00	1.5	gradient-based
DARTS(second order)[14]	3.3	2.76	4	gradient-based
SNAS[29]	2.8	2.85	1.5	gradient-based
GDAS[18]	3.4	2.93	0.21	gradient-based
Random search baseline	3.2	3.29	4	random
SP-DARTS(a)	3.64	2.71	0.11	gradient-based
SP-DARTS(b)	3.66	2.78	0.11	gradient-based



(a) Stage1 (b) Stage2 (c) Stage3 (d) DARTS_V2

Fig. 4 Search result

4.2 Architecture Evaluation

The same as DARTS, in the evaluation stage, the model is stacked with 20 cells and 36 initial channels, and model parameters are initialized randomly and trained with a batch size of 128 for 600 epochs. We also use the cutout regularization [20] with the length of 16, Drop-path [21] with a rate of 0.3 and auxiliary towers [5] with a weight of 0.4. The standard SGD optimizer and momentum gradient descent method are adopted, where the weight decay $wd = 0.0003$, momentum $\beta = 0.9$, and initial learning rate $\eta = 0.025$ and is decayed to 0 using cosine annealing.

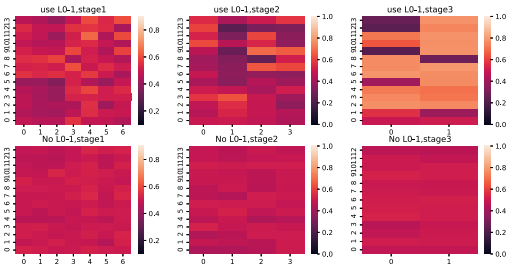


Fig. 5 Weight thermal diagram

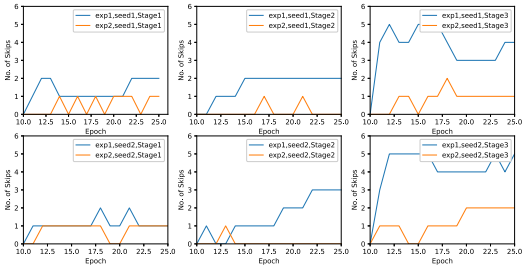


Fig. 6 Comparative experiment on K.

4.3 Diagnostic Experiments

4.3.1 Utility of Loss Function

In Eqs. (1), the softmax function is adopted in DARTS to activate the mixed operation between nodes in the cell, and the result of weight is in (0,1), which makes the weights of mixed operations very similar after searching. In this paper, comparative experiments on the activation for operation weight and whether to add the loss function L_{0-1} are conducted additionally. The results are provided in Fig. 5. The combination of sigmoid activation and L_{0-1} makes the difference in operations weights larger. It is easier to distinguish the quality of the operation and the architecture structure is more reasonable.

4.3.2 Discussion of Skip Operation Aggregation

We conduct comparative experiments on partial channel sampling proportion K. As shown in Fig. 6, the x-axis is the number of epochs in each training stage. Since the first 10 epochs are only used to tune the model parameters, which do not affect the architecture parameters, we consider the epochs starting from 10 to 25 in the experiment. The parameters of exp1 in the three stages include the network depth $L = 8-8-8$, the initial number of channel $C = 16-32-64$, and partial channel sampling proportion $K = 4-4-4$. The parameters of exp2 in the three stages are the network depth $L = 8-8-8$, the initial number of channel $C = 16-32-64$, and partial channel sampling proportion $K = 4-8-8$. Each experiment is executed twice with a different random seed. Two conclusions drawn from these two experiments: 1) The skip operations aggregation in DARTS is aggravated with the increase of the initial number of channels; 2) Partial channel connection inhibits skip operation, and the inhibitory effect increases with the increase of K.

The reasons for such conclusions are discussed as follows: For conclusion 1), with the increasing number of initial channels, the features that can be extracted are richer, so the model can achieve the performance with fewer channels in the case of more skip operations. On the other hand, the larger the model is, the more discontinuous the output gradient of loss for multi-parameter operations will be. Therefore, the model will prefer the skip operations that make the loss function decay rapidly. For conclusion 2), partial channel sampling is adopted to transmit forward through the mixed operation. A concatenation operation of other channels not processed is adopted when reaching the front node, which is essentially equivalent to skip operation, and more original information of nodes is retained. Therefore, additional skip operations are not very necessary in the search process. However, having no skip operation in the search result is also one of our approach's drawbacks. The removal of the skip operation will affect the capability of the model. For the architecture search methods such as DARTS, it is essential to find K, the depth L, and initial channel number C to balance if partial channel connection is adopted. If K is too large, the aggregation of weak parameters operation will be severe. While K is too small, the inhibitory of weak parameter operation will be strong.

We further apply SP-DARTS to the image resource of WAN FANG datasets to search for a required image. The experimental results show that our method performs well on other datasets and has a good generalization capability.

5. Conclusion

This paper proposed a synchronization progressive approach based on DARTS for network depth and width and adopted several effective optimization methods in the latest studies. The experiments are conducted on the CIFAR-10 and WANFANG datasets, and the results show that the search time is reduced to 0.1 GPU days with high accuracy.

Meanwhile, we also found some cases in the experiments where the comparative experiments were designed to make it clear and provide a preliminary explanation. There are some drawbacks of the proposed method, such as excessive inhibition of skip operation. The neural architecture search (NAS) algorithm still has no unified standard. Various training processes and various parameters make it difficult to know each step's exact contribution to the results. Although the differential scheme greatly reduces the search time, the training process is still very time-consuming. In the future, the evaluation stage's time cost will be greatly reduced with the development of computing hardware and the further study of model evaluation. For the improvement of DARTS, a feasible direction is to break the sharing cell scheme and the fixed structure of the cell, which will enrich the search space and be beneficial to searching for a better model. In conclusion, it is hoped that deep learning will develop towards automation with the development of NAS, which can achieve a better model architecture.

Acknowledgments

This research is sponsored by National Key Research and Development Plan of China under Grant No. 2017YFB1400301. Thanks for the collaboration of Dr. Liqiong Chang.

References

[1] A. Krizhevsky, I. Sutskever, and G.E. Hinton, "Imagenet classification with deep convolutional neural networks," Communications of The ACM, vol.60, no.6, pp.84–90, June 2017.
[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," ICLR 2015: Int. Conf. Learning Representations 2015, 2015.
[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2016 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.770–778, 2016.
[4] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.
[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2015 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.1–9, 2015.
[6] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," ICLR 2019: 7th Int. Conf. Learning Representations, 2019.
[7] B. Zoph and Q.V. Le, "Neural architecture search with reinforcement learning," CoRR, vol.abs/1611.01578, 2016.
[8] B. Zoph, V. Vasudevan, J. Shlens, and Q.V. Le, "Learning transferable architectures for scalable image recognition," CoRR, vol.abs/1707.07012, 2017.
[9] Z. Zhong, J. Yan, W. Wu, J. Shao, and C.L. Liu, "Practical block-wise neural network architecture generation," 2018 IEEE/CVF Conf. Comput. Vis. Pattern Recognit., pp.2423–2432, 2018.
[10] J.D. Dong, A.C. Cheng, D.C. Juan, W. Wei, and M. Sun, "Dpp-net: Device-aware progressive search for pareto-optimal neural architectures," Proc. European Conf. Comput. Vis. (ECCV), LNCS, vol.11215, pp.540–555, 2018.
[11] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," ICLR 2018: Int. Conf. Learning Representations 2018, 2018.
[12] E. Real, A. Aggarwal, Y. Huang, and Q.V. Le, "Regularized evolution for image classifier architecture search," Proc. AAAI Conf. Artifi. Intelli., vol.33, no.1, pp.4780–4789, 2019.
[13] S. Xie, H. Zheng, C. Liu, and L. Lin, "SNAS: Stochastic neural architecture search," CoRR, vol.abs/1812.09926, 2018.
[14] X. Chen, L. Xie, J. Wu, and Q. Tian, "Progressive differentiable architecture search: Bridging the depth gap between search and evaluation," 2019 IEEE/CVF Int. Conf. Comput. Vis. (ICCV), pp.1294–1303, 2019.
[15] Y. Xu, L. Xie, X. Zhang, X. Chen, G.J. Qi, Q. Tian, and H. Xiong, "Pc-darts: Partial channel connections for memory-efficient architecture search," ICLR 2020: Eighth Int. Conf. Learning Representations, 2020.
[16] X. Chu, T. Zhou, B. Zhang, and J. Li, "Fair darts: Eliminating unfair advantages in differentiable architecture search," Proc. European Conf. Comput. Vis. (ECCV), LNCS, vol.12360, pp.465–480, arXiv preprint arXiv:1911.12126, 2019.
[17] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: a comparison between shallow and deep architectures," IEEE Trans. Neural Netw. Learn. Syst., vol.25, no.8, pp.1553–1565, Aug. 2014.
[18] M.D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," Proc. 13th European Conf. Comput. Vis., (ECCV), LNCS, vol.8689, 2014, pp.818–833, 2014.
[19] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, "The expressive power of neural networks: a view from the width," NIPS'17 Proc. 31st Int. Conf. Neural Information Processing Systems, pp.6232–6240, 2017.
[20] T. DeVries and G.W. Taylor, "Improved regularization of convolutional neural networks with cutout," arXiv preprint arXiv:1708.04552, 2017.
[21] G. Larsson, M. Maire, and G. Shakhnarovich, "Fractalnet: Ultra-deep neural networks without residuals," ICLR 2017: Int. Conf. Learning Representations 2017, 2017.

Zimin Zhao received the B.S. degree in Electronic Information Science from Northwest University in 2020. His researches focus on automated machine learning and automated deep learning.

Ying Kang received the B.S. degree in School of Software Engineering from Northwest University. Her researches focus on recommendation system and distributed system. She is now a graduate student in School of Information Science and Technology from Northwestern University.

Aiqin Hou received the Ph.D. degree in school of information science and technology from Northwest University in 2018. She is currently an Associate Professor with the School of Information Science and Technology, Northwest University of China. Her research interests include big data, high performance network, and resource scheduling.

Daguang Gan received the M.S. degree from the China institute of science and technology in 2008. He is an assistant to the general manager of Beijing wanfang software co., LTD., mainly engaged in knowledge organization, information retrieval and information analysis.